GRAPH CLUSTERING

by

FLOW SIMULATION

Graaf clusteren door simulatie van stroming

met een samenvatting in het Nederlands

Proefschrift ter verkrijging van de graad van doctor aan de Universiteit Utrecht op gezag van de Rector Magnificus, Prof. Dr. H.O. Voorma, ingevolge het besluit van het College voor Promoties in het openbaar te verdedigen op maandag 29 mei 2000 des middags te 12.45 uur

door

Stijn Marinus van Dongen

geboren op 4 september 1969
te Smallingerland.

promotoren: Prof. M. Hazewinkel
Faculteit Wiskunde en Informatica, Universiteit Utrecht
Centrum voor Wiskunde en Informatica, Amsterdam

Prof. J. van Eijck
Faculteit Letteren, Universiteit Utrecht
Utrecht Institute of Linguistics OTS
Centrum voor Wiskunde en Informatica, Amsterdam

The research for this thesis was carried out at the Centre for Mathematics and Computer Science (CWI) in Amsterdam and partially funded by Stichting Physica.

Contents

1

Introduction

Cluster Analysis is the mathematical study of methods for recognizing natural groups within a class of entities.

This is a common type of definition for the discipline of cluster analysis, and it is unsatisfactory in at least one respect. However, this cause for dissatisfaction must be welcomed, since it points directly to some of the essential questions and problems in cluster analysis. These essential aspects will be briefly discussed in this introduction, followed by an account of the main contribution of the thesis, the *Markov Cluster Process*, a cluster process designed within the setting of graphs. This setting defines a relatively young area in cluster analysis referred to as *graph clustering*, which has connections to the clearly scoped field of *graph partitioning*. Clustering and graph-clustering methods are also studied in the large research area labelled pattern recognition. These disciplines and the applications studied therein form the natural habitat for the Markov Cluster Algorithm. Their relative positions are sketched, anticipating a more thorough topography in the first part of this thesis.

The Markov Cluster Process (abbreviated *MCL* process) defines a sequence of stochastic matrices by alternation of two operators on a generating matrix. It is basically all that is needed for the clustering of graphs, but it is useful to distinguish between the *algorithm* and the *algebraic process* employed by the algorithm. The study of mathematical properties thus belongs to the process proper, and aspects such as interpretation and scaling belong to the algorithm or a particular implementation of the algorithm. The second part of this thesis covers the conceptual and theoretical foundation of the Markov Cluster (abbreviated *MCL*) Algorithm and Process. Other proposals for graph clustering are discussed, and several classic mathematical results are presented which are relevant in the broad perspective of graph clustering and graph partitioning, in particular the relationship between graph spectra and connectivity properties. The third part of the thesis is concerned with algorithmic issues such as complexity, scalability, and implementation, and various types of experiments and benchmarks establishing the particular strengths and weaknesses of the algorithm.

The thesis aims at an audience with a mathematical background. A marked exception is the appendix *A cluster miscellany* on page 149, which gives a bird's eye view of various aspects of cluster analysis. Among them are the history of the discipline, the role of the computer, and the etymology of some of the words naming concepts central to cluster analysis and the Markov Cluster Algorithm. The organization of this introduction largely mirrors the structure of the remainder of the thesis, consisting of the three parts *Cluster*

Analysis and Graph Clustering, *The Markov Cluster Process*, and *Markov Cluster Experiments* respectively. The introduction concludes with a detailed account of the structure and contents of the thesis.

1.1 Cluster analysis and graph clustering

The definition given above is a common denominator of the definitions found in cluster monographs. It is rather dry, but that is a shortcoming inherent in definitions in general. More importantly, it is vague. There is nothing wrong with the highly indeterminate *entities*, as this indicates that methods are studied in abstracto, without studying a specific type of entity like for example *man* or *meteorite*. The problem lies with the word 'natural', as anyone who has ever attempted to use the word argumentatively may confirm. What a person judges to be natural is in general heavily influenced by personality, world context, a priori knowledge, and implicit or explicit expectations. If attention is restricted to the grouping of visual stimuli into patterns by the eye and mind, it is seen that this process is really a complex interaction of low-level cognitive functions (which can be situated in the eye) and high-level functions (representing contextual and emotional stimuli). Witness for example Figure 1, adapted from [**151**]. What are the natural groups? It depends. There are several clouds of data points. On the left they may be seen to form a ring with a sphere in its centre, or the two may be seen as one big sphere, or each cloud may be seen as a sphere in its own right. In general, if a ring is seen, then it is 'natural' to see one long stretched shape on the right, as this needs the same kind of linking required for forming the ring. In this instance, there is a tendency to scale the perceived objects such that they are balanced. The way in which cluster structure is perceived locally is affected by the overall perceived cluster structure. High-level and low-level cognitive functions interact in this process, and aspects such as balancedness and regularity compete with each other (see the cluster miscellany on page 149 for examples). In the automated recognition of cluster structure, this interaction of high-level objectives with low-level data cues implies that it is unlikely to find good algorithms that work from data to solution in a one-way fashion, whereas 'looping' algorithms are difficult to analyse and understand. Dimensionality is an important and inherent part of the problem, effectively ruling out optimization approaches.

There are many problems and desiderata in cluster analysis that have a cognitive flavour. Examples are the extent to which different scales are perceived or permitted, the extent to which symmetry and regularity influence the grouping, the range of shapes and combinations of shapes that is recognized (e.g. thin and stretched, compact sphere-like, curved), whether certain shapes are preferable to others, and whether there is the possibility that data points are classified as noise or outlier. The overall desideratum is to obtain balanced clustering solutions with respect to these dimensions, and the overall problem is how to achieve this. The significance of this cognitive flavour is that it is too much to expect a single method to produce a best or most natural grouping in each instance, due to the adaptive nature of the word 'natural'. Furthermore, if the recognition process needs to take high-level a priori knowledge into account, then the task of recognition will have to be split up among different units and methods, modelling different levels of cognition. Cluster analysis comes down to *bootstrapping* in a setting with no

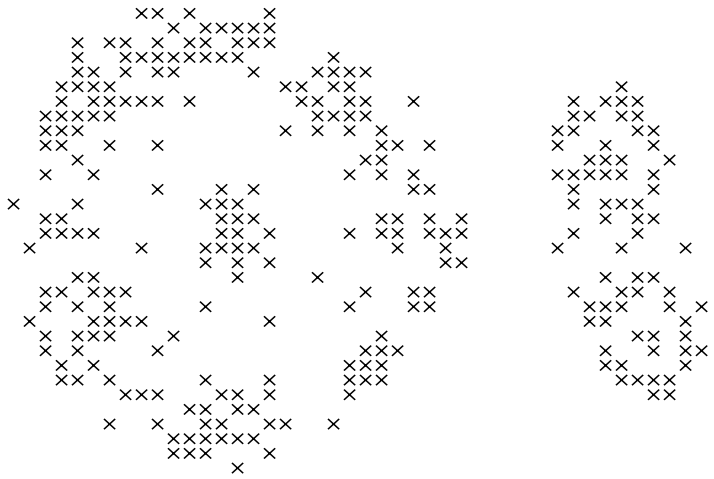


Figure 1. How many groups?

a priori knowledge at all except for the raw data. From a mathematical point of view, cluster analysis is the last step in a progression of category sorting problems, preceded by e.g. classification (where the classes are known) and discriminant analysis.

1.1.1 Vector clustering and graph clustering. Applications are numerous in cluster analysis, and these have led to a bewildering multitude of methods. However, classic applications generally have one thing in common: They assume that entities are represented by vectors, describing how each entity scores on a set of characteristics or features. The dissimilarity between two entities is calculated as a distance between the respective vectors describing them. This implies that the dissimilarity corresponds with a distance in a Euclidean geometry and that it is very explicit; it is immediately available. Geometric notions such as *centre of gravity*, *convex hull*, and *density* naturally come into play. Graphs are objects that have a much more combinatorial nature, as witnessed by notions such as *degree* (number of neighbours), *path*, *cycle*, *connectedness*, et cetera. Edge weights usually do not correspond with a distance or proximity embeddable in a Euclidean geometry, nor need they resemble a metric. I refer to the two settings respectively as *vector clustering* and *graph clustering*.

The graph model and the vector model do not exclude one another, but one model may inspire methods which are hard to conceive in the other model, and less fit to apply there. The *MCL* process was designed to meet the specific challenge of finding cluster structure in simple graphs, where dissimilarity between vertices is implicitly defined by the connectivity characteristics of the graph. Vector methods have little to offer for the graph model, except that a generic paradigm such as single link clustering can be given meaning in the graph model as well. Conversely, many methods in the vector model construct objects such as *proximity graphs* and *neighbourhood graphs*, mostly as a notational convenience. For these it is interesting to investigate the conditions under

which graph methods can be sensibly applied to such derived graphs. This issue is discussed in Section 3.3 and Chapter 10.

1.1.2 Applications in graph clustering. The current need for and development of cluster methods in the setting of graphs is mostly found within the field of *graph partitioning*, where clustering is also referred to as *coarsening*, and in the area of pattern recognition. Clustering serves as a preprocessing step in some partitioning methods; this issue is introduced below and expanded in Chapter 3. Pattern recognition sciences can be viewed as the natural surroundings for the whole of cluster analysis from an application point of view. Due to its fundamental and generic cognitive nature the clustering problem occurs in various solutions for pattern recognition tasks, often as an intermediate processing step in layered methods. Another reason why the pattern recognition sciences are interesting is that many problems and methods are formulated in terms of graphs. This is natural in view of the fact that higher-level conceptual information is often stored in the form of graphs, possibly with typed arcs. There are two stochastic/graph-based models which suggest some kinship with the *MCL* algorithm: Hidden Markov models and Markov Random Fields (discussed in Chapter 2). The kinship is rather distant; the foremost significance is that it demonstrates the versatility and power of stochastic graph theory applied to recognition problems. With the advent of the wired[1] and hyperlinked era the graph model is likely to gain further in significance.

1.1.3 Graph clustering and graph partitioning. Cluster analysis in the setting of graphs is closely related to the field of *graph partitioning*, where methods are studied to find the optimal partition of a graph given certain restrictions. The generic meaning of 'clustering' is in principle exactly that of 'partition'. The difference is that the semantics of 'clustering' change when combined with adjectives such as 'overlapping' and 'hierarchic'. A partition is strictly defined as a division of some set S into subsets satisfying a) all pairs of subsets are disjoint and b) the union of all subsets yields S (these and other definitions and naming conventions are introduced in Chapter 4).

In graph partitioning the required partition sizes are specified in advance. The objective is to minimize some cost function associated with links connecting different partition elements. Thus the burden of finding natural groups disappears. The graphs which are to be partitioned are sometimes extremely homogeneous, like for example meshes and grids. The concept of 'natural groups' is hard to apply to such graphs. However, graphs possessing natural groups do occur in graph partitioning: The graph in Figure 2 is taken from the article *A computational study of graph partitioning* [55]. If natural groups are present then a clustering of the graph may aid in finding a good partition, if the granularity of the clustering is fine compared with the granularity of the required partition sizes. There is a small and rather isolated stream of publications on graph clustering in the setting of circuit lay-out. One cause for this isolation is that the primary data-model is that of a *hypergraph* with nodes that can be of different type, another is the cohesive nature of the research in this area. The relationship between graph clustering and graph partitioning is further discussed in Chapter 2.

[1] Wired in the sense of connected; the connections may well be wireless.

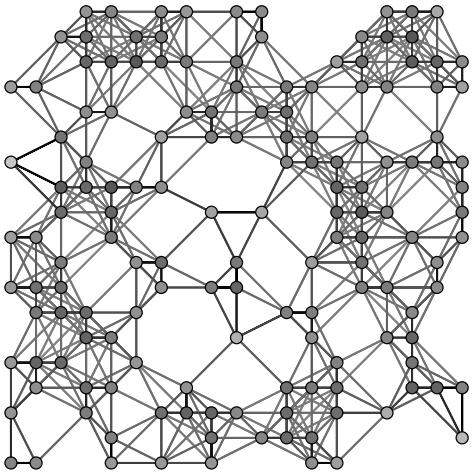


Figure 2. A geometric graph.

1.2 The Markov Cluster Algorithm

The main contribution of this thesis is a powerful new cluster algorithm with many good properties, called the *Markov Cluster Algorithm* or *MCL algorithm*. It was designed within the mathematical setting of graphs and inspired by a simple paradigm (Section 1.2.1 below), naturally leading to the formulation of an algebraic process for stochastic matrices called the *MCL* process. This process forms the engine of the *MCL* algorithm; one of the algorithm's particular assets is that it does not contain high-level procedural rules for the assembling, splitting, or joining of groups.

The problems and aims described in the previous section apply as much to graph clustering as they do to the field of clustering at large. Thus far few methods have been formulated which make distinct use of the topology offered by graphs. The methods that *have* been formulated have not yet found their way into the cluster analysis monographs, the main reason being that the graph model is much of a new kid on the block. A detailed account is given in Chapter 2. It should be mentioned that many classical methods in cluster analysis are formulated in such an abstract manner that it is easy to apply them in the setting of graphs. Peculiarly, methods such as found in the single-linkage family even allow a graph-theoretical formulation, but the graph-theoretical concepts that are used are neither very sophisticated nor powerful. The experiments in Chapter 10 indicate that this is unavoidable.

The graph in Figure 2 with 150 nodes, also used in [55], will be used throughout this chapter as the running example. The article [55] is about graph partitioning, and this type of *geometric* graph is often used in the field. Two nodes are connected if the distance between them is at most $\sqrt{8}$ units. The strength of a bond is inversely related to the distance, and corresponds with its grey level. Note that this is a peculiar type of

graph because the similarity function on the nodes is derived from their depicted location (which means that a simple transformation step makes the graph embeddable in Euclidean space). Most example graphs used throughout this thesis have the property that they allow a somewhat pleasing pictorial representation, for obvious reasons. One should be aware — and convince oneself — that the working of the *MCL* algorithm does not depend on the availability of such a representation.

1.2.1 The graph clustering paradigm. The *graph clustering paradigm* postulates that 'natural' groups in graphs, the groups that are sought and not known, have the following property:

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

Variants of this idea also inspired two cluster algorithms proposed in the setting of graph partitioning [**75, 174**]. Interestingly, these algorithms take a completely different turn compared with the approach described here — an issue further discussed in Chapter 5.

At the heart of the *MCL* algorithm lies the idea to simulate flow within a graph, to promote flow where the current is strong, and to demote flow where the current is weak. If natural groups are present in the graph, then according to the paradigm current across borders between different groups will wither away, thus revealing cluster structure in the graph.

Simulating flow through a graph is easily done by transforming it into a Markov graph[2], i.e. a graph where for all nodes the weights of outgoing arcs sum to one. Flow can be expanded by computing powers of the associated stochastic (Markov) matrix, which amounts to the usual discrete Markov process. This is not sufficient in itself, as the Markov process does not exhibit cluster structure in its underlying graph.

The paradigm is empowered by inserting a new operator into the Markov process, called inflation. Whereas flow expansion is represented by the usual matrix product, flow inflation is represented by the entry-wise Hadamard–Schur product combined with a diagonal scaling. The inflation operator is responsible for both strengthening and weakening of current. The expansion operator is responsible for allowing flow to connect different regions of the graph.

The resulting algebraic process is called the *Markov Cluster Process* or *MCL* process. The process converges quadratically in the neighbourhood of so called doubly idempotent matrices (idempotent under both expansion and inflation).

1.2.2 Putting the paradigm to work. It turns out that the *MCL* process is appealing both in theory and in practice. In terms of flow the process brings about exactly the behaviour as expected according to the graph cluster paradigm. In the limit matrices, which are doubly idempotent, flow has fully stabilized. The indices of a doubly idempotent matrix

[2] Using the standard definition of a random walk on a graph.

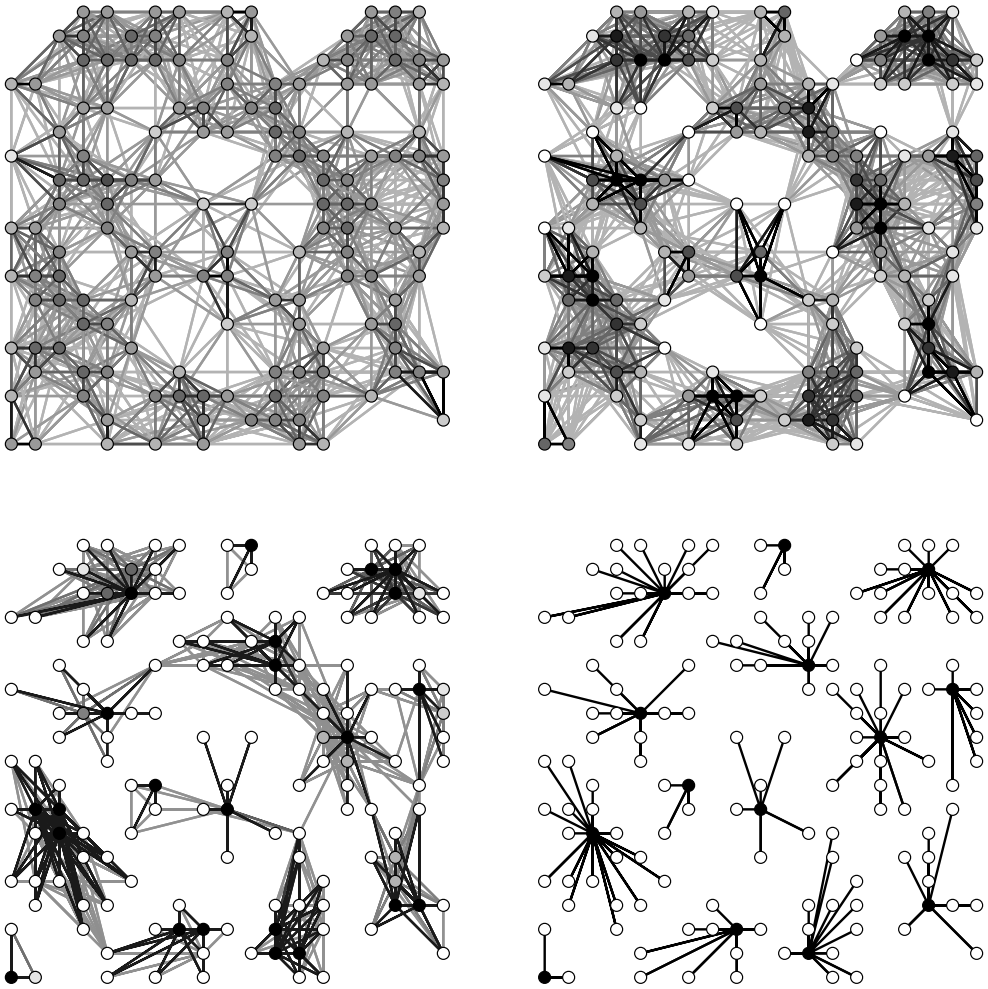


Figure 3. Successive stages of flow simulation by the *MCL* process.

(nodes of the associated graph) can either be classified as attractors or as nodes which are being attracted by attractors (Theorem 1 on page 57). The attractor systems of the matrix (corresponding with a complete subgraph of the associated graph) each induce a cluster which consists of the attractor system and all nodes that it attracts (Definition 8 on page 58). In this thesis a *weakly connected component* of a directed graph G is defined as a maximal subgraph of G which contains at least one strongly connected component C of G (i.e. a subgraph in which there is a path between every ordered pair of nodes), together with all nodes x in G for which there is a path in G going from x to an element of C. The clustering associated with a doubly idempotent matrix thus corresponds with all weakly connected components of the associated graph. Overlap can occur if nodes

are attracted to more than one attractor system, but this phenomenon has only been observed so far if the overlapping part of clusters is left invariant by an automorphism of the input graph (Section 10.2).

The *MCL* process is illustrated in Figure 3 for the geometric graph in Figure 2. Four iterands are depicted from left to right and top to bottom. For each node, at most sixteen neighbours are shown. The bottom right graph corresponds with the limit from this particular *MCL* process. The degree of shading of a bond between two nodes indicates the maximum value of flow, taken over the two directions: the darker the bond the larger the maximum. The degree of shading of a node indicates the total amount of incoming flow. Thus, a dark bond between a white node and a black node indicates that the maximum flow value is found in the direction of the dark node, and that hardly any flow is going in the other direction. The bottom right graph represents a situation where flow is constant; the dark nodes are attractors and the bonds indicate which nodes are attracted to them. The corresponding matrix is idempotent. The picture contains all necessary information needed to reconstruct the matrix — the limit of an *MCL* process is in general highly structured. The bottom right graph generically induces a clustering of the input graph by taking as clusters all the weakly connected components (Definition 8 in Chapter 6).

1.2.3 Mathematics behind the *MCL* process. The *MCL* process consists of alternation of expansion and inflation, which are both operators converting one stochastic matrix into another. The limits which may result from an *MCL* process are nonnegative idempotent stochastic matrices. It is proven in Chapter 6 that the process converges quadratically around these limit points.

Expansion, which is just normal matrix multiplication, belongs to the language of linear algebra. It has been studied specifically in the setting of nonnegative matrices, and even more focused research exists in the setting of Markov matrices. The image of a stochastic matrix M under inflation with parameter r is the product $M^{\circ r} d_t$, where $M^{\circ r}$ denotes the Hadamard power and d_t is a diagonal matrix such that the product is stochastic again. Under a certain (weak) condition there is a basis in which inflation is represented by Hadamard–Schur products only (Chapter 7). Inflation is a highly nonlinear operator, which severely impedes the development of mathematical tools for describing its interaction with expansion. Unfortunately, the many results in Hadamard–Schur theory are of little use even for solely describing inflation, because the relevant spectral inequalities go the wrong way. In this thesis I develop structure theory for the class of so called *diagonally positive semi-definite* matrices, which is preserved by both inflation and expansion. These matrices are shown to posses structure generalizing the structure present in the limits of the *MCL* process. The spectral working of the inflation parameter on these matrices can be described qualitatively in terms of their associated structure. In general, the iterands of any *MCL* process with a symmetric generating matrix are guaranteed to be *diagonally symmetric*, and for the standard parametrization it is guaranteed that all even iterands are also diagonally positive-semidefinite.

There is a clear relationship between the working of the *MCL* process and the spectral properties of respectively the generating matrix, the iterands, and the limit (Section 7.2). The process is well described as a *stochastic uncoupling* process, and it has long been known that large subdominant eigenvalues of nonnegative matrices correspond with the phenomenon of uncoupling. In some cases this uncoupling is truly annoying, as it impedes for example the computation of the eigenvector corresponding with the dominant eigenvalue. A remedy for this is found in the beautiful theory of stochastic uncoupling and Perron uncoupling. Another example comes from the theory of rapidly mixing Markov chains, where chains are rapidly mixing iff the subdominant eigenvalues are well separated from the dominant eigenvalue (Chapter 8). Most relevant to the material presented here though are the cases where uncoupling is actually desirable. In Chapter 8 the basic results underlying spectral methods in graph partitioning are described, and some of this material is applied to an example used in the exposition of the *MCL* process. A well-known theorem of Fiedler for nonnegative symmetric matrices, which is part of the theoretical foundation of graph partitioning, is shown to generalize very easily towards nonnegative diagonally symmetric matrices. A less well known[3] lemma by Powers for irreducible symmetric matrices, which satisfactorily explains the working of spectral methods, is shown to generalize[4] towards nonnegative irreducible matrices, and is also applied to a running example used throughout the thesis.

There are two areas of research which may provide additional insights into (special cases of) the *MCL* process. In both areas concepts are studied which can be linked to both expansion and inflation. These are respectively the theory of symmetric circulants, in particular with respect to majorization aspects (Section 6.4), and the theory associated with Hilbert's distance for nonnegative vectors (Section 7.4). In this thesis the basic relationships are established, and ideas for further research are sketched.

1.2.4 Benefits. In addition to the fact that the limit of the *MCL* process allows a generic cluster interpretation (Theorem 1, page 57, and Definition 8, page 58), one of the main theoretical results of this thesis is that the iterands of the *MCL* process allow such interpretation as well (Theorem 9, page 81). Further benefits of the Markov Cluster process and algorithm are listed below.

- The *MCL* process forms the entire chassis and engine of the *MCL* algorithm. This means that the formulation of the algorithm is highly simple and elegant, as the algorithm basically consists of alternation of two different operators on matrices, followed by interpretation of the resulting limit. There are no high-level procedural instructions for assembling, joining, or splitting groups.
- In cluster analysis heuristic and optimization approaches prevail, and there is in general a lack of (computationally feasible) mathematical concepts relating to cluster structure. It is a valuable result that the matrix iterands and limits of a simple algebraic process, the *MCL* process, allow interpretation in terms of cluster structure.

[3] But probably part of the collective graph partitioning subconscious.

[4] In an entirely straightforward manner.

- By varying parameters in the *MCL* process, clusterings of different granularity are found. In terms of the previous analogy, the steering controls come for free with the engine. The number of groups can not and need not be specified in advance, but the algorithm can be tuned to different contexts.
- The issue 'how many clusters?' is not dealt with in an arbitrary manner, but rather by strong internal logic. Cluster structure leaves its marks on the cluster process, and the flow parameters control the granularity of the cluster imprint.
- The rate of convergence of the *MCL* process, and projection of the iterands afterwards onto the resulting clustering, give hooks for unsupervised parameter adjustment.
- The limit of the *MCL* process is very sparse, and its iterands are sparse in a weighted sense. This implies that the algorithm scales very well. Given certain sparseness conditions, pruning can be incorporated into the algorithm resulting in a complexity of $\mathcal{O}(Nk^2)$, where N is the number of nodes, and where k is the average or maximum number of neighbours that nodes are allowed to have.

1.2.5 Limitations. Of course, the *MCL* algorithm is not a panacea, and has limitations as well. Problem instances in which the diameters of the clusters are not too large allow a regime of pruning while maintaining the quality of the clusterings retrieved. If the diameter grows large, this approach becomes infeasible (Chapter 11). For one thing, this limits the applicability of the *MCL* algorithm to derived graphs in the vector model (e.g. neighbourhood graphs).

1.3 *MCL* experiments and benchmarking

The standardized comparison of cluster methods on common series of problems is altogether non-existent in cluster analysis. This is a deplorable situation, even though such standardized comparison (i.e. benchmarking) invites thorny problems. An obstacle might be that benchmarking more or less requires that the quality of a solution is measurable in terms of a cost function on clusterings. It is non-trivial to devise a cost function for clusterings of arbitrary sizes and distribution, but still doable. A bigger problem is that cost functions inevitably favour the detection of certain shapes above others, e.g. spherical above stretched or vice versa. The main implication is that clustering by optimization of a cost function has severe drawbacks if clusters can be of mixed type, even disregarding the size of the search space. These issues have no impact on benchmarking, as the combination of the type of problem instance and the cost function used can be tuned in advance.

Most evidence presented in cluster analysis research is expository or occasional. Common examples do exist, and are sometimes used to support a claim of improved or superior behaviour. Most often the measurement is by visual judgement or by reasoning. The situation for graph clustering is not much better. Benchmark graphs do exist in graph partitioning, but these are in the form of *netlists* rather than normal graphs. Netlists are converted to normal graphs in order to prepare them for both clustering (or equivalently coarsening) and partitioning, but the conversion step is not standardized. Benchmarking is common practice for the resulting partition, but not so for the intermediate clustering step.

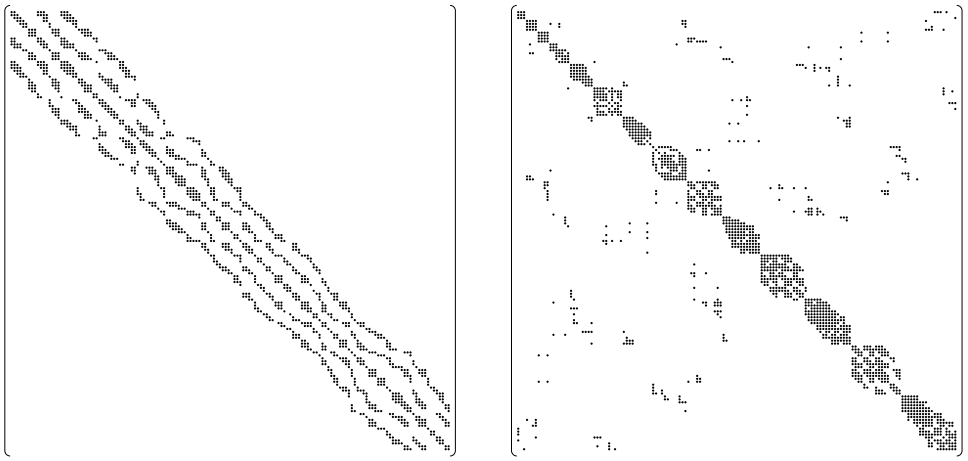


Figure 4. The visual impact of permuting an incidence matrix according to cluster structure.

This thesis also contains several examples of the occasional kind. Additionally, in Chapter 12 a fairly generic mechanism is proposed for random generation of test-graphs with hooks for controlling the density characteristics. This ensures that a good a priori clustering is known. The *MCL* algorithm is applied to these test graphs, and the resulting clusterings are compared to the a priori clustering by means of a performance criterion and a metric defined on the space of partitions of a set of fixed cardinality. A generic performance criterion for weighted graphs is derived in Chapter 9, by stepwise refinement of a particularly appealing though naive criterion for simple graphs. The metric defined on the space of partitions is useful in judging continuity properties by comparing clusterings at different levels of granularity, and for comparing retrieved clusters with the a priori constructed clusterings.

1.3.1 Graph clustering and incidence matrix block structure. In general, it is difficult to visualize a graph and a corresponding clustering, because a graph is rather special if it can be conveniently laid out in the plane. However, the incidence matrix of a graph allows to some extent visualization of a clustering. This is done by relabelling the nodes of the graph such that all nodes which are in the same cluster have consecutive labels. The relabelling corresponds with a permutation of the original incidence matrix. If the clustering is good, that is, if there are many edges within each cluster and few going out, this will result in a permuted matrix appearing to have blocks on the diagonal, with a few nonzero entries scattered throughout the remaining parts of the matrix. Figure 4 shows the incidence matrix of the graph in Figure 2 on the left, with the nodes ordered coordinate-wise, left to right and top to bottom. The matrix on the right hand side is the incidence matrix where nodes are labelled such that the labelling is aligned with the clustering corresponding with the bottom right picture in Figure 3. The contrast between the two representations is remarkable. The black/white structure of some of the blocks shows that some clusters still allow a natural subdivision, which is confirmed by a closer inspection of the clustering in Figure 3 and the input graph in Figure 2.

1.4 Organization

The remainder of the thesis is organized in much the same way as this introduction. Part I contains two chapters. Chapter 2 is concerned with the position of cluster analysis as a data analysis and a pattern recognition science. The position of graph clustering within cluster analysis is sketched in Chapter 3, and related graph problems are discussed, in particular the relationship between graph clustering and graph partitioning. Some of the ideas in these chapters were first formulated in [**44**].

The second part is entirely devoted to the formulation and study of the *MCL* process and the *MCL* algorithm. Notations and definitions are covered in Chapter 4. The chapter after that introduces various graph clustering proposals previously made, which can roughly be categorized as respectively combinatorial and probabilistic approaches. It is shown that these approaches rely on the same abstract principle. The material in Chapters 5 and 6 was published in the technical report [**45**]. It covers the basic mathematics needed to connect the limits of the *MCL* process to clusterings of graphs. It is shown that the *MCL* process distributes over the Kronecker product, and a categorization of equilibrium states is given. Convergence of the process towards these states is treated in depth, and it is shown that the clusterings associated with certain instable equilibrium states are stable under perturbations of the states[5]. Special graphs on which expansion and inflation act as each others inverse are derived, which share the automorphism group of a ring graph. The corresponding matrices are symmetric circulants.

A large part of Chapter 7 is contained in an article submitted for publication [**46**]. It is shown that the inflation operator Γ_r maps the class of diagonally symmetric matrices onto itself for $r > 0$, and that Γ_r maps the class of diagonally positive semi-definite matrices onto itself for $r \in \mathbb{N}$ (Section 7.1). Diagonally positive semi-definite (*dpsd*) matrices are then shown to posses structural properties which generalize the structural properties of nonnegative idempotent matrices, which are the generic limits of the *MCL* process, and which need not be *dpsd* (Section 7.2). A clear connection between the two is provided by the extreme parametrization of Γ_r, by setting $r = \infty$. For M stochastic and *dpsd*, it is true that $\Gamma_\infty M$ is a matrix for which some finite power is nonnegative idempotent. Further properties of *dpsd* matrices are derived in terms of two decompositions into rank 1 idempotents with specific properties (Section 7.3). The theory of Hilbert's distance defined for nonnegative vectors is introduced, and it is shown that in this framework expansion and inflation can be linked to each other. This yields a simplified proof for one of the theorems in Chapter 6. Discussions and conjectures make up the last part of Chapter 7.

The relationship between spectrum and cluster structure is further investigated in Chapter 8, where classic (spectral) results basic to the field of graph partitioning are presented. Two simple generalizations of these results towards *dpsd* matrices are given and applied to examples used in Chapter 6. The chapter also contains a section describing the role of spectral techniques in the field of *rapidly mixing Markov chains*.

[5] Except for the phenomenon of overlap, which may occur in clusterings associated with the *MCL* process.

The third part contains material relevant to testing and benchmarking the *MCL* algorithm, which has not been published yet. A generic performance criterion for clusterings of weighted graphs is derived in Chapter 9, by a stepwise refinement of a particularly simple and appealing criterion for simple graphs. The most refined criterion uses a particular Schur convex function of which several properties are established. The chapter also introduces a metric defined on the space of partitions, which is useful for comparing different clusterings of the same graph. In Chapter 10 the qualitative properties of the *MCL* algorithm are studied, using moderately small graphs. The phenomena of overlap, attractor systems, and the effect of loops and inflation on cluster granularity are discussed. The *MCL* algorithm is shown to have strong separating power by applying it to torus grid graphs (Section 10.5). This is mainly of theoretical interest though, because subsequently it is demonstrated that clustering of normal grid graphs is very sensitive to perturbations in the input, in case the diameter of the natural clusters is large (Section 10.6). This implies that applying the *MCL* algorithm to neighbourhood graphs derived from the vector model has its limitations, though the *MCL* algorithm works well for random geometric graphs such as in Figure 2. The considerations in this section also inspire a tentative proposal for border detection in image bitmaps using information from early iterands of the *MCL* process (Section 10.7).

Chapter 11 is concerned with scaling the algorithm by incorporation of a pruning step. Various pruning schemes are considered, and conditions are postulated under which pruning will not affect the quality of the clusterings retrieved. The *C*–implementation and the data structures (implementing sparse matrix operations) used in conducting the experiments in this thesis are briefly described. In Chapter 12 I report on putting the *MCL* algorithm to the test against randomly generated cluster test graphs. These are constructed in such a way that an a priori clustering is known for which it is guaranteed in a probabilistic sense that there is no clustering that is evidently better. The *MCL* algorithm performs well and is very robust under perturbations of the input for this class of test graphs.

Part I

Cluster Analysis and Graph Clustering

2

Cluster analysis

Cluster analysis came into being as a bundling of exploratory data techniques which were scattered over many sciences. This embodied an effort to unify and distinguish different frameworks, to separate method from data, and to separate implementation from method. Methods still exist in multitudes, but the reasons for this are well understood. Contrasting this numerosity is the prevalence of a single data model in the cluster analysis monographs, namely that where entities are represented by vectors. Special attention is paid to this issue throughout this chapter and the next, as this thesis is concerned with a cluster algorithm in the setting of graphs.

In Section 2.1 the position of cluster analysis as an exploratory data analysis science is summarized. This section is concise, because it is of limited interest for the subject of graph clustering. A short history of cluster analysis, tied to the perspective of exploratory data analysis, is found in Section 4 of the appendix *A cluster miscellany* (page 154). In Section 2.2 problems and methods from the pattern recognition sciences are introduced. These are interesting because clustering methods have found employment there, and because (intermediate) problems and results in pattern recognition are often phrased in terms of graphs. Two graph based stochastic methods, namely Hidden Markov Models and Markov Random Fields, are discussed in order to illustrate the versatility of stochastic graph concepts and to exemplify the significant conceptual and mathematical differences between these methods and the *MCL* process. Section 2.3 is concerned with the history and characteristics of the research that is collectively labelled as cluster analysis, and the extent to which this label corresponds with a coherent discipline from a mathematical point of view.

2.1 Exploratory data analysis

Cluster analysis is usually seen as the result of cross-fertilization between mathematics and sciences such as biology, chemistry, medicine, and psychology. The latter provide the practical applications that yield the problem formulation, while the study of those problems in an abstract setting belongs to mathematics proper. In this classic setting (more than a century old now) the nature of the problem formulation is that of *exploring* data to see if cohesive structure is present; cluster analysis is then ranged under the flag of *exploratory data analysis*. Other mathematical disciplines of this type are *discriminant analysis:* assigning objects to (a priori known) classes given a number of possibly incomplete observations, *factor analysis:* uncovering correlations between variables by their observed behaviour, *mixture resolving:* estimating parameters for mixtures of distributions (e.g. via the Maximum–Likelihood method), and *dispersion analysis:* methods

for detecting the effect of individual factors on the results of an experiment. Among this list cluster analysis is the most vague in the sense that neither problem nor aim allow a satisfactory description. The least is known — observations only — and the most is wanted: a classification of these, resulting in a bootstrapping problem in optima forma. Usually, the more constrained a problem is, the more this suggests a particular way of solving it. The reverse is true as well; Cluster analysis is pervaded by a host of different mathematical techniques (Section 2.3). The most prominent data model in exploratory data analysis is that were entities are represented by vectors (representing scores on sets of attributes). I refer to this setting as *the vector model*. It is also discussed in greater detail in Section 2.3.

2.2 Pattern recognition sciences

New employment for exploratory techniques has been found in the young but vibrant field of *pattern recognition*, along with a host of other techniques from for example statistical decision theory, signal processing, and linear algebra. In the pattern recognition sciences methods are studied for emulation of cognitive skills, i.e. for detection or recognition of structure in data. Very often these data correspond with auditory or visual signals, or with variables which are measured along the coordinates of a two or three-dimensional space. Examples of this kind are speech recognition, food screening, matching of fingerprints, machine vision, satellite image processing, and more generally the processing of geographic, atmospheric, oceanic, or astronomic data. The hardness of the problems varies widely: the more is known a priori, the easier the problem. The difficulty of matching problems, where it has to be decided whether a new object is the same as one out of a collection of known objects, depends highly on the variability of the object appearances. Iris or fingerprint matching is thus relatively easy, whereas speech recognition or recognition of 3-D objects is much more difficult. Utterances may vary in tone, pronunciation, emphasis, colour, and duration, even for a single person. The appearance of objects may vary according to the angle of view, the orientation of the object, the viewing distance, the background, the location and intensity of the light source(s), and the overlap by other objects.

2.2.1 Cluster analysis applications. Some of the applications of cluster analysis in pattern recognition (usually corresponding with an intermediate processing stage) as listed by Jain and Dubes in [**94**] are: grammatical inference, speech and speaker recognition, image segmentation, and image matching. In general, the role of cluster analysis is the joining of primitive data elements in regions or time-frames, which do not yet need to correspond with high-level objects. Clustering is thus a base method for diminishing dimensionality. Other applications of clustering in pattern recognition are reducing feature dimensionality and reducing the dimensionality of a search space (e.g. the set of all Chinese characters).

2.2.2 The data model in spatial pattern recognition. At first sight image segmentation and spatial data segmentation seem to fit well within the vector model, as they concern measurements in Euclidean spaces. However, the data model differs considerably from

that in exploratory data analysis. In the latter setting, the distribution of the vectors themselves over the geometry is of interest. In spatial data segmentation the vectors are just the (x, y) coordinates of pixels or the (x, y, z) coordinates of voxels (the 3-D equivalent of a pixel). The vectors sample some area, which is usually box-shaped. The proximity between two vectors is not related to their distance, but defined in terms of the similarity between the measurements on the corresponding units of the sample space. Typically, only neighbouring pixels or voxels are considered for the proximity relationship, as one is interested in finding regions of contiguous units, which are homogeneous with respect to the measurements. This localization of the neighbour relationship induces a lattice or grid-like graph. Consequently, graph-theoretical concepts and techniques play an important role in spatial data segmentation.

This is in fact true for the field of pattern recognition at large. Data may be split up in collections of primitive patterns. The primitive patterns are to be matched with a catalogue of generic primitives, while at the same time the collection has to be split into higher-level patterns representing objects. Examples of primitive patterns are phonemes in speech recognition and stroke primitives in optical character recognition. In the latter case, characters can also be viewed as primitives for the word-level, and words can be viewed as primitives for the sentence level. The important thing is that the interaction or succession of such primitives is governed by constraints. This induces graphs where the primitives are nodes and the constraints induce and exclude neighbour relations. The fertile combination of graphs and stochastic models in pattern recognition is the subject of the following section.

2.2.3 Graph–based stochastic methods. Two stochastic methods in pattern recognition deserve special mention, namely Hidden Markov Models (HMM) and Markov Random Fields (MRF). These techniques draw upon the same mathematical apparatus as the *MCL* process — which is where the resemblance more or less stops. The common denominator of HMM, MRF, and MCL is that they all use stochastic concepts in the setting of graphs.

A hidden Markov model is a probabilistic model for data observed in a sequential fashion, based on two primary assumptions. The first assumption is that the observed data arise from a mixture of K probability distributions corresponding with K states. The second assumption is that there is a discrete Markov chain generating the observed data by visiting the K states according to the Markov model. The hidden aspect of the model arises from the fact that the state sequence is not directly observed. Instead, the state sequence must be inferred from a sequence of observed data using the probability model (adapted from [**170**], page 373). The most noteworthy application of the hidden Markov model is found in speech recognition, where it is used both for the recognition of words in terms of phonemes (which are the states), and the construction of sentences from words. In both cases the sequence of primitives (phonemes or words) is governed by constraints, in the sense that the true state of the previously observed primitive induces a probability distribution on the most likely state of the currently observed primitive. This is modelled by a Markov chain of transition probabilities. In [**170**] computational

biology is listed as another area where HMMs have found employment, and [**157**] names the applications *lip reading* and *face tracking*.

A Markov Random Field is the counterpart of a one-dimensional Markov chain, where the bidirectionality of past–present–future is superseded by the spatial concept of neighbouring sites. The origin of this concept is found in the Ising model for ferromagnetism, and in physics it is applied to statistical mechanical systems exhibiting phase transitions ([**100**], page 120). In pattern recognition, MRFs are applied to fundamental tasks such as segmentation and restoration of images, edge detection, and texture analysis and synthesis.

As an example, consider a rectangular array of pixels, where each pixel may assume one of G grey levels. Typically an MRF is used to model the fact that the grey levels of neighbouring pixels are correlated. This correlation reflects higher-level properties of an image, such as the presence of regions, boundaries, and texture. An MRF yields the tools to infer such high-level properties from the low-level pixel information by stochastic techniques. It is assumed that the overall shading of the array is the result of a stochastic process indexed by the pixels. An MRF requires a symmetric neighbourhood structure defined for the pixels; the simplest case is that where only adjacent pixels are neighbours. The property that defines an MRF is that *The conditional probability that a pixel assumes a grey level, given the grey levels of all other pixels, is equal to the conditional probability that it assumes the grey level given only the grey levels of its neighbouring pixels.*

The state space of the MRF is thus enormous; it amounts to all possible grey colourings of the array. An MRF can also be used to model the presence of edges or boundaries in a picture, by creating an edge array similar to the pixel array[1]. The edge array can be observed only indirectly by looking at the pixel array. The equivalence of MRFs with Gibbs distributions allows the modelling of expectations regarding boundaries — smoothness, continuity — in the MRF via so called clique potentials. Combining the intensity and boundary processes yields a compound MRF which can be used to restore noisy images, detect edges, detect regions, or a combination of these [**33**]. This requires specification of a measurement model for the observed image (i.e. choice of the stochastic model, parameter estimation). A typical approach for image restoration is: *A maximum a posteriori probability estimate of the image based on the noisy observations then [after specification of the measurement model] is found by minimizing the posterior Gibbs energy via simulated annealing* ([**63**], page 499). The MRF model is very powerful in its flexibility — diverse types of interaction (corresponding e.g. with texture, regions, and edges) between pixels can be modelled, and it is backed up by a sound mathematical apparatus. However, its computational requirements are considerable, and the issues of supervised and unsupervised parameter learning are no less difficult than they are in general.

The Hidden Markov Model and the theory of Markov Random Fields illustrate the versatility of the graph model in pattern recognition, and its use in inferencing higher-level structure by stochastic techniques. The stochastic concepts used in the HMM and MRF

[1] A horizontal respectively vertical edge is thought to separate two vertically respectively horizontally adjacent pixels.

and their applications are closely tied to a stochastic view on the realization and structure of patterns in real life. This situation is somewhat different for the *MCL* algorithm: its formulation uses the notion of *flow*, utilizing the presence of cohesive structure, rather than drawing upon a model for the realization of this structure and retrieving it via a posteriori optimization.

2.2.4 The position of cluster analysis. Publications on cluster analysis in the setting of pattern recognition are usually rather theoretical, and blend in with publications in the more traditional line of exploratory data research. Most publications which are solely concerned with clustering contain either a new cluster algorithm or suggestions for improvement of existing algorithms. A few publications focusing on the graph model do exist, but these are mostly concerned with neighbourhood graphs (see Section 3.3 in the next chapter).

Clustering methods are occasionally mentioned as a tool for segmentation or as a necessary intermediate processing step in various applications, but I have not found any systematic comparison of different methods plugged into the same hole. The most likely reason for this is that the role of clustering is not sufficiently essential, as it is not a dedicated solution for a specific problem. Benchmarking is in principle very well possible and Chapter 12 is devoted to this issue.

2.3 Methods and concepts in multitudes

Cluster analysis is a field that has always been driven by a demand from various disciplines engaged in exploratory data analysis, like for example taxonomy, chemistry, medicine, psychiatry, market research, et cetera. The monographs on cluster analysis give long listings of applications, see e.g. Anderberg [**10**], Everitt [**54**], and Mirkin [**132**]. This wide range of interest groups, most of which do not have common channels of communication, has resulted in an enormous variety of clustering methods. Adding to this is the elusive nature of the problem. There is no obvious step immediately transforming the loose formulation of the problem into a strategy for solving it. Rather, ten different people will come up with ten different methods, and they may well come up with twenty, because the problem is attractive in that it seems intuitively so simple, yet in practice so hard. On a related note, Blashfield et al report in [**22**] on a questionnaire regarding the use of cluster software, with fifty-three respondents yielding fifty different programs and packages. It is sometimes said that there are as many cluster methods as there are cluster analysis users. This thesis indeed offers Yet Another Cluster Method[2]. The method operates in a setting that is relatively new to cluster analysis though (see the next chapter), it depends on an algebraic process that has not been studied before, and this process has properties that make it particularly suitable as a means for detecting cluster structure (Parts II and III of this thesis).

[2]This is yet another Yet Another qualification; see page 160.

2.3.1 Conceptual confusion. Different disciplines use different concepts and different wordings, so cluster–related research has yielded a plethora of methods and concepts. Kaufman and Rousseeuw put it this way in [**103**], page vii: *Rather than giving an extensive survey of clustering methods, leaving the user with a bewildering multitude of methods to choose from* (...). They have a positive attitude, as they write on page 3:

> (...) automatic classification is a very young scientific discipline in vigorous development, as can be seen from the thousands of articles scattered over many periodicals (mostly journals of statistics, biology, psychometrics, computer science, and marketing). Nowadays, automatic classification is establishing itself as an independent scientific discipline (...)

The qualification 'very young' is debatable though, as the first survey articles and monographs in cluster analysis began to appear in the sixties and early seventies. The purpose of Kaufman and Rousseeuw was to write an applied book for the general user. In the field of classification, the book by Jardine and Sibson [**95**] is a long standing reference which aims to give a mathematical account of the methods employed in taxonomy (the theory of classification of living entities). The significance of this book is enlarged by its emphasis on mathematics rather than limited by its focus on taxonomy. The tone of their introduction is somewhat more dejected ([**95**], page page ix):

> Terminological confusion and the conceptual confusions which they conceal have loomed large in the development of methods of automatic classification. Partly this has arisen from the parallel development of related methods under different names in biology, pattern recognition, psychology, linguistics, archaeology, and sociology. Partly it has arisen from failure by mathematicians working in the various fields to realize how diverse are the problems included under the headings 'classification', 'taxonomy', and 'data analysis'.

In mathematics, the elusive nature of the vector cluster problem is reflected in the fact that there is no particular piece of mathematical machinery just right for the job. The result is that many mathematical tools and disciplines are used in modelling of the problem domain and formulation of cluster strategies. The list includes matrix algebra, geometry, metric spaces, statistics, set theory, graph theory, information theory, and several combinations out of these. Witness Mirkin in [**132**], page xiv: (...) *the reader is assumed to have an introductory background in calculus, linear algebra, graph theory, combinatorial optimization, elementary set theory and logic, and statistics and multivariate statistics.* The words *introductory* and *elementary* should be noted though. Cluster analysis draws upon the different disciplines mainly for formulation of both problems and strategies sought to apply to them. With few exceptions, not much new material is being developed nor is there much need to apply existing theorems.

Clustering has also been presented in the setting of computing paradigms such as simulated annealing [**97, 106**], genetic algorithms, and neural networks. In these instances however, the computing paradigms embraced the clustering problem rather than vice versa, and the approaches have not yet swept the area.

2.3.2 The prevalent data model in cluster analysis. *Cluster Analysis is the mathematical study of methods for recognizing natural groups within a class of entities.* Consider this definition once more. It speaks of 'natural groups', indicating that entities are somehow related to each other. Apparently, there is a notion of greater or lesser distance or similarity between entities, or it would be impossible to discriminate between different pairs and constellations of entities. It is actually difficult to imagine a class of entities where such a notion is lacking; the art of discrimination is that essential to sensory systems, intelligence, and awareness. Whereas this digression may seem too aspiring, it appears that attempts to formulate the essence of cluster analysis readily lead to such cognitive and philosophical issues. For example, Sokal writes in [**156**], page 1:

> But since classification is the ordering of objects by their similarities (...) and objects can be conceived of in the widest sense including processes and activities — anything to which a vector of descriptors can be attached, we recognize that classification transcends human intellectual endeavor and is indeed a fundamental property of living organisms. Unless they are able to group stimuli into like kinds so as to establish classes to which favorable or avoidance reactions can be made, organisms would be ill-adapted for survival.

This kind of comment is frequently found in the literature, and it spells out the burdensome idea that methods in cluster analysis have to compete with nothing less than a fundamental property of living organisms. The citation also demonstrates that at the time of writing (1976) cluster analysis was more or less tied to the framework where entities are represented by vectors, an observation still valid today. In the framework each entity is examined with respect to a fixed number of characteristics; this gives a vector of numbers, and this vector represents the entity. For example, if the entities are medical records, then the characteristics might be *length, weight, age, alcohol consumption, blood pressure, liver thickening*, if the entities are meteorite rocks then the characteristics can be the respective fractions of different types of chemical compounds. The distance between two entities is then defined to be a function of the difference of the two corresponding vectors. Usually a scaling of the vectors is involved in order to weigh different characteristics, and some norm is taken of the difference vector.

This setting is prevalent throughout the cluster analysis literature, and it is the de facto standard in monographs on the subject — see the list of monographs compiled in the next section. On the one hand, this persistence of the same data model is hardly surprising, as it is a highly generic model that apparently suits the existing needs. On the other hand, there is an interest in clustering algorithms in the graph partitioning community and occasional other areas, feeding a small but steady stream of publications which is isolated from the cluster analysis literature at large in terms of references and surveys. This is discussed in the next chapter.

2.3.3 The coherence of cluster analysis as a discipline. As evidenced in the beginning of this section, the field of cluster analysis is of fragmented heritage. This situation began to change somewhat with the advent of dedicated conferences and publication of several monographs on the subject, beginning with the books by Sokal and Sneath [**155**] (1963), Jardine and Sibson [**95**] (1971), Anderberg [**10**] (1973), Duda and

Hart [**50**] (1973), Everitt [**54**] (1974), and Hartigan [**79**] (1975). Other often cited references are Ryzin [**166**] (conference proceedings, 1977), Kaufman and Rousseeuw [**103**] (1983), Jain and Dubes [**94**] (1988), Massart and Kaufman [**119**] (1990), and Mirkin [**132**] (1996). These monographs often contain huge collections of references to articles from different disciplines, and they embody considerable efforts to create a unified approach to cluster analysis. The citation of Good on page 157 reflects a certain longing for the field to become whole. The same feeling is occasionally expressed by other researchers. According to Kaufman and Rousseeuw the era of cluster analysis as an independent scientific discipline has already begun (see the citation on page 22). However, much can be brought to stand against this point of view, without disputing the viability of the label 'cluster analysis' as a flag under which a motley crew sails.

The most prominent reason why cluster analysis is not the home ground of a coherent scientific community is simply the wide variety of mathematical concepts that can be employed. No single branch of mathematics is particularly apt for the job, and problems in cluster analysis defy a clinching formal description. Such a description may emerge, but the general opinion is that the conditions are unfavourable, as the generic clustering problem formulation assumes little and aspires much.

A second reason is found in the lack of an application area with a very definite and urgent need for good clustering algorithms, with common cost functions and benchmarks, and preferably with problem instances that obey some variant of Moore's law — a doubling of the typical problem size every three years or so. Graph partitioning, with its current emphasis on multilevel approaches, may turn out to provide such pull from the application side for cluster methods in the setting of graphs. Otherwise, this is a phenomenon unfamiliar to cluster analysis, simply because the large majority of applications either has an exploratory nature, or concerns the embedding of clustering methods in a compound (e.g. recognition) method that is not widely recognized as a fundamental step in the solution of a problem of general importance and urgency.

All this can be summed up in the observation that researchers in cluster analysis share neither a common mathematical framework nor a common application area generating sufficient pull. This does not imply that there is no use for the gathering forces of monographs and conferences dedicated to the subject. The opposite is true, as there is without doubt a general interest in clustering methods, and a real danger of wasted efforts and reinvention of the wheel over and over again. However, the state of affairs in cluster analysis is that the gathering of efforts does not have a strong synergetic effect.

3

Graph clustering

The *graph* concept is extremely general, and allows a rough categorization in *complete weighted* graphs, *weighted structured* graphs, and *simple* graphs. These three categories can be viewed as constituting the two ends and the middle of a spectrum of possibilities. For convenience I shall assume that weights, if present, are nonnegative. The clustering problem is one of the few problems which makes sense for all three categories. Until now it has mainly been studied in the setting of weighted complete graphs, which are often also metric — in which case it is really better to use the label metric space. I adhere to the custom that *graphs* — without further qualification — are simple graphs, which is in line with common usage. In this chapter I discuss the clustering problem in the setting of simple graphs, which is in a sense the natural habitat of the *MCL* algorithm, and the resemblances and differences with clustering in the setting of complete weighted graphs. To this end, a brief sketch is given of different types of problems in graph theory.

The research area *combinatorial optimization* involves graphs from all three categories, but individual problems (e.g. the Travelling Salesman Problem, the clique problem) are generally attached to a specific category. Clustering is a sibling of the optimization problem known as *graph partitioning* (Section 3.2), which is a well-defined problem in terms of cost functions. The field of combinatorial optimization appears in the course of a brief non-exhaustive survey of different branches in graph theory (Section 3.1), which is interesting because it naturally brings along the issue of tractability. The chapter concludes with a discussion of the implications of the differences between the vector model and the graph model for clustering.

3.1 Graphs, structure, and optimization

The concept of a *simple graph* is tightly associated with the study of its structural properties. Examples are the study of regular objects such as *distance regular graphs*, the study of *graph spectra* and *graph invariants*, and the theory of *randomly generated graphs*. The latter concept is introduced in some more detail, because it is used in defining a benchmark model for graph clustering in Chapter 12. A generic model via which a simple graph on N edges is randomly generated is to independently realize each edge with some fixed probability p (a parameter in this model). Typical questions in this branch of random graph theory concern the behaviour of quantities such as the number of components, the girth, the chromatic number, the maximum clique size, and the diameter, as n goes to infinity. One direction is to hold p fixed and look at the behaviour of the quantity under consideration; another is to find bounds for p as a function of n

such that almost every random graph generated by the model has a prescribed property (e.g. diameter k, k fixed).

In the theory of nonnegative matrices, structural properties of simple graphs are taken into account, especially regarding the $0/1$ structure of matrix powers. Concepts such as cyclicity and primitivity are needed in order to overcome the complications arising from this $0/1$ structure. A prime application in the theory of Markov matrices is the classification of states in terms of their graph-theoretical properties, and the complete understanding of limit matrices in terms of the analytic and structural (with respect to the underlying graph) properties of the generating matrix.

A different perspective is found in the retrieval or recognition of combinatorial notions such as above. In this line of research, algorithms are sought to find the diameter of a graph, the largest clique, a maximum matching, whether a second graph is embeddable in the first, or whether a simple graph has a Hamiltonian circuit or not. Some of these problems are doable; they are solvable in polynomial time, like the diameter and the matching problems. Other problems are extremely hard, like the clique problem and the problem of telling whether a given graph has a Hamiltonian cycle or not. For these problems, no algorithm is known that works in polynomial time, and the existence of such an algorithm would imply that a very large class of hard problems admits polynomial time[1] algorithms for their solution. Other problems in this class are (the recognition version of) the Travelling Salesman Problem, many network flow and scheduling problems, the satisfiability problem for logic formulas, *and* (a combinatorial version of) the clustering problem. Technically, these problems are all in both the class known as NP, and the subclass of NP–complete problems. Roughly speaking, the class NP consists of problems for which it can be verified in polynomial time that a solution to the problem is indeed what it claims to be. The NP–complete problems are a close-knit party from a complexity point of view, in the sense that *if any NP–complete problem admits a polynomial–time solution, then so do all NP–complete problems* — see [**134**] for an extensive presentation of this subject. The subset of problems in NP for which a polynomial algorithm exists is known as P. The abbreviations NP and P stand for *nondeterministic polynomial* and *polynomial* respectively. Historically, the class NP was first introduced in terms of nondeterministic Turing machines. The study of this type of problems is generally listed under the denominator of *combinatorial optimization*.

3.2 Graph partitioning and clustering

In graph partitioning well–defined problems are studied which closely resemble the problem of finding good clusterings, in the settings of both simple and weighted graphs respectively. Restricting attention first to the bi-partitioning of a graph, let (S, S^c) denote a partition of the vertex set V of a graph (V, E) and let $\Sigma(S, S^c)$ be the total sum of weights of edges between S and S^c, which is the cost associated with (S, S^c). The maximum cut problem is to find a partition (S, S^c) which maximizes $\Sigma(S, S^c)$, and the minimum quotient cut problem is to find a partition which minimizes $\Sigma(S, S^C) / \min(|S|, |S^c|)$. The

[1] Polynomial in the size of the problem instance; for a graph this is typically the number of vertices.

recognition versions of these two problems (i.e. given a number L, is there a partition with a cost not exceeding L?) are NP–complete; the optimization problems themselves are NP–hard [**62**], which means that there is essentially only one way known to prove that a given solution is optimal; list all other solutions with their associated cost or yield. In the standard graph partitioning problem, the sizes of the partition elements are prescribed, and one is asked to minimize the total weight of the edges connecting nodes in distinct subsets in the partition. Thus sizes $m_i > 0$, $i = 1, \ldots, k$, satisfying $k < n$ and $\sum m_i = n$ are specified, where n is the size of V, and a partition of V in subsets S_i of size m_i is asked which minimizes the given cost function. The fact that this problem is NP–hard (even for simple graphs and $k = 2$) implies that it is inadvisable to seek exact (best) solutions. Instead, the general policy is to devise good heuristics and approximation algorithms, and to find bounds on the optimal solution. The role of a cost/yield function is then mainly useful for comparison of different strategies with respect to common benchmarks. The clustering problem in the setting of simple graphs and weighted structured graphs is a kind of mixture of the standard graph partitioning problem with the minimum quotient cut problem, where the fact that the partition may freely vary is an enormously complicating factor.

3.2.1 Graph partitioning applications. The first large application area of graph partitioning (GP) is found in the setting of sparse matrix multiplication, parallel computation, and the numerical solving of PDE's (partial differential equations). Parallel computation of *sparse matrix multiplication* requires the distribution of rows to different processors. Minimizing the amount of communication between processors is a graph partitioning problem. The computation of so called *fill-reducing orderings of sparse matrices* for solving large systems of linear equations can also be formulated as a GP–problem [**74**]. PDE solvers act on a mesh or grid, and parallel computation of such a solver requires a *partitioning of the mesh*, again in order to minimize communication between processors [**53**].

The second large application area of graph partitioning is found in 'VLSI CAD', or Very Large Scale Integration (in) Computer Aided Design. In the survey article [**8**] Alpert and Kahng list several subfields and application areas (citations below are from [**8**]):

- *Design packaging*, which is the partitioning of logic arrays into clustering. "This problem (...) is still the canonical partitioning application; it arises not only in chip floorplanning and placement, but also at all other levels of the system design."
- Partitioning in order to improve mappings from HDL[2] descriptions to gate– or cell–level netlists.
- Estimation of wiring requirements and system performance in high–level synthesis and floorplanning.
- Partitioning supports "the mapping of complex circuit designs onto hundreds or even thousands of interconnected FPGA's" (Field–Programmable Gate Array).
- A good partition will "minimize the number of inter-block signals that must be multiplexed onto the bus architecture of a hardware simulator or mapped to the global interconnect architecture of a hardware emulator".

[2]Hardware Description Language.

Summing up, partitioning is used to model the distribution of tasks in groups, or the division of designs in modules, such that there is a minimal degree of interdependency or connectivity. Other application areas mentioned by Elsner in [**53**] are hypertext browsing, geographic information services, and mapping of DNA sequences.

3.2.2 Clustering as a preprocessing step in graph partitioning. There is an important dichotomy in graph partitioning with respect to the role of clustering. Clustering can be used as a preprocessing step if natural groups are present, in terms of graph connectivity. This is clearly not the case for meshes and grids, but it is the case in the VLSI CAD applications introduced above, where the input is most often described in terms of hypergraphs.

The currently prevailing methods in hypergraph partitioning for VLSI circuits are so-called *multilevel* approaches where hypergraphs are first transformed into normal graphs by some heuristic. Subsequently, the dimensionality of the graph is reduced by clustering; this is usually referred to as *coarsening*. Each cluster is contracted to a single node and edge weights between the new nodes are computed in terms of the edge weights between the old nodes. The reduced graph is partitioned by (a combination of) dedicated techniques such as spectral or move-based partitioning.

The statement that multi-level methods are prevalent in graph partitioning (for VLSI circuits) was communicated by Charles J. Alpert in private correspondence. Graph partitioning research experiences rapid growth and development, and seems primarily US–based. It is a non-trivial task to analyse and classify this research. One reason for this is the high degree of cohesion. The number of articles with any combination of the phrases *multi-level, multi-way, spectral, hypergraph, circuit, netlist, partitioning*, and *clustering* is huge — witness e.g. [**7, 9, 31, 32, 38, 39, 73, 75, 81, 101, 102, 138, 174**]. Spectral techniques are firmly established in graph partitioning, but allow different approaches, e.g. different transformation steps, use of either the Laplacian of a graph or its adjacency matrix, varying numbers of eigenvectors used, and different ways of mapping eigenspaces onto partitions. Moreover, spectral techniques are usually part of a larger process such as the multi-level process described here. Each step in the process has its own merits. Thus there is a large number of variables, giving way to an impressive proliferation of research. The spectral theorems which are fundamental to the field are given in Chapter 8, together with simple generalizations towards the class of matrices central to this thesis, diagonally symmetric matrices.

3.3 Clustering in weighted complete versus simple graphs

The *MCL* algorithm was designed to meet the challenge of finding cluster structure in simple graphs. In the data model thus far prevailing in cluster analysis, here termed a *vector model* (Section 2.3 in the previous chapter), entities are represented by numerical scores on attributes. The models are obviously totally different, since the vector model induces a (metric) Euclidean space, along with geometric notions such as convexity, density, and centres of gravity (i.e. vector averages), notions not existing in the (simple)

graph model. Additionally, in the vector model all dissimilarities are immediately available, but not so in the (simple) graph model. In fact, there is no notion of similarity between disconnected nodes except for third party support, e.g. the number of paths of higher length connecting two nodes.

Simple graphs are in a certain sense much closer to the space of partitions (equivalently, clusterings) than vector–derived data. The class of simple graphs which are disjoint unions of complete graphs are in a natural 1–1 correspondence with the space of partitions. A disjoint union of simple complete graphs allows only one sensible clustering. This clustering is perfect for the graph, and conversely, there is no other simple graph for which the corresponding partition is better justified[3]. Both partitions and simple graphs are generalized in the notion of a hypergraph. However, there is not a best constellation of points in Euclidean space for a given partition. The only candidate is a constellation where vectors in the same partition element are infinitely close, and where vectors from different partition elements are infinitely far away from each other.

On the other hand, a cluster hierarchy resulting from a hierarchical cluster method (most classic methods are of this type) can be interpreted as a tree. If the input data is Euclidean, then it is natural to identify the tree with a tree metric (which satisfies the so called *ultrametric inequality*), and consider a cluster method 'good' if the tree-metric produced by it is close to the metric corresponding with the input data. Hazewinkel [**80**] proved that single link clustering is optimal if the (metric) distance between two metrics is taken to be the Lipshitz distance. It is noteworthy that the tree associated with single link clustering can be derived from the minimal spanning tree of a given dissimilarity space [**88**].

What is the relevance of the differences between the data models for clustering in the respective settings? This is basically answered by considering what it requires to apply a vector method to the (simple) graph model, and vice versa, what it requires to apply a graph method to the vector model. For the former, it is in general an insurmountable obstacle that generalized (dis)similarities are too expensive (in terms of space, and consequently also time) to compute. Dedicated graph methods solve this problem either by randomization (Section 5.4) or by a combination of localizing and pruning generalized similarities (the *MCL* algorithm, Chapter 11).

More can be said about the reverse direction, application of graph methods to vector data. Methods such as single link and complete link clustering are easily formulated in terms of threshold graphs associated with a given dissimilarity space (Chapter 4). This is mostly a notational convenience though, as the methods used for going from graph to clustering are straightforward and geometrically motivated. For intermediate to large threshold levels the corresponding threshold graph simply becomes too large to be represented physically if the dimension of the dissimilarity space is large. Proposals of a combinatorial nature have been made towards clustering of threshold graphs, but these are computationally highly infeasible (Section 4.2).

[3]These considerations form the starting point for the formulation of generic performance criteria for clusterings of simple and weighted graphs in Chapter 9.

Another type of graph encountered in the vector model is that of a *neighbourhood graph* satisfying certain constraints (see [**94**] for a detailed account). Examples are the Minimum Spanning Tree, the Gabriel Graph, the Relative Neighbourhood Graph, and the Delaunay Triangulation. For example, the Gabriel Graph of a constellation of vectors is defined as follows. For each pair of vectors (corresponding with entities) a sphere is drawn which has as centre the geometric mean of the vectors, with radius equal to half the Euclidean distance between the two vectors. Thus, the coordinates of the two vectors identify points in Euclidean space lying diametrically opposed on the surface of the sphere. Now, two entities are connected in the associated Gabriel Graph if no other vector lies in the sphere associated with the entities. The step from neighbourhood graph to clustering is made by formulating heuristics for the removal of edges. The connected components of the resulting graph are then interpreted as clusters. The heuristics are in general local in nature and are based upon knowledge and/or expectations regarding the process via which the graph was constructed.

The difficulties in applying graph methods to threshold graphs and neighbourhood graphs are illustrated in Section 10.6. The *MCL* algorithm is tested for graphs derived by a threshold criterion from a grid-like constellation of points, and is shown to produce undesirable clusterings under certain conditions. The basic problem is that natural clusters with many elements and with relatively large diameter place severe constraints on the parametrization of the *MCL* process needed to retrieve them. This causes the *MCL* process to become expensive in terms of space and time requirements. Moreover, it is easy to find constellations such that the small space of successful parametrizations disappears entirely. This experiment clearly shows that the *MCL* algorithm is not well suited to certain classes of graphs, but it is argued that these limitations apply to a larger class of (tractable) graph clustering algorithms.

Part II

The Markov Cluster Process

4

Notation and definitions

This chapter introduces the terminology needed for graphs, (dis)similarity spaces, clusterings, and matrices. 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.

4.1 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.*
   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 $\mathbb{R}_{\geq 0}$, 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 $\mathbb{R}_{>0} \cup \{\infty\}$, 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 Chapter 5.

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

4.2 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**[1]) **extreme** if respectively $\mathcal{P}=\{V\}$ and $\mathcal{P}=\{\text{singletons}(V)\}=\{\{v_1\},\ldots,\{v_t\}\}$. A partition of the form $\{S,S^c\}$ (where S^c is the complement of the set S in V) is called a **bipartition** of V, it is called **balanced** if $|S|=|S^c|$. For a bipartition the set notation is omitted, and it is sloppily written as (S,S^c). Given a bipartition (S,S^c), the corresponding **characteristic difference vector** x is defined by $x_i=1$ for $v_i\in S$, and $x_i=-1$ for $v_i\in S^c$.

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 $1\le i<j\le n$ 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 $d\le t$.

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 $s(u,v)\ge s_i$. 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[2] on t nodes where there is an edge between u and v iff $s(u,v)\ge s_i$.

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 C_x and C_y of $\mathcal{P}_{k-1}$ for which the threshold level s such that [*the subgraph on $C_x\cup C_y$ in the threshold graph of (V,s) at level s is complete*] is maximal. Equivalently, C_x and C_y are such that

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

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

the maximum of the minimal similarity in the restriction of the similarity space (V, s) to $C_X \cup C_Y$, 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 [**89, 95, 121**], 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.

4.3 Matrices

A **column stochastic** matrix is a nonnegative matrix in which the entries of each column sum to one. A matrix is called **column allowable** if its associated graph is voidfree, that is, it has no zero columns. Note that a column stochastic matrix is by definition column allowable.

Let M be a matrix in $\mathbb{R}^{n\times n}$, let α and β both be sequences of distinct indices in the range $1 \ldots n$. The **submatrix** of M corresponding with row indices from α and column indices from β is written $M[\alpha|\beta]$. The determinant of a square submatrix is called a **minor**. The **principal submatrix** with both row and column indices from α is written $M[\alpha]$. Let α be a sequence of distinct indices in the range $1 \ldots n$, denote by α^c the sequence of indices in $1 \ldots n$ which are not part of α. The matrix M is called **irreducible** if for all α containing at least 1 and at most $n-1$ indices the submatrix $M[\alpha|\alpha^c]$ has at least one nonzero coordinate. Otherwise M is called **reducible**. This can also be stated in terms of graphs. Associate a simple graph $G = (V, w)$ with M (note that it is not assumed that M is nonnegative[3]) by setting $w(v_p, v_q) = 1$ iff $M_{pq} \neq 0$. Then the existence of a sequence α such that $M[\alpha|\alpha^c]$ has only zero entries implies that there are no arcs in G going from the subgraph defined on the nodes corresponding with α^c to the subgraph defined on the nodes corresponding with α. Thus M is reducible if the node set of the associated simple graph G can be split into two (non-empty) parts such that there are no arcs going from the first part to the other, and it is irreducible otherwise.

The eigenvalues of a square matrix M of dimension n are written $\lambda_1(M), \ldots, \lambda_n(M)$. If the eigenvalues are real, then they are written in decreasing order, thus $\lambda_1(M) \geq \lambda_2(M) \geq \cdots \geq \lambda_n(M)$. This is a strict rule, carried through for the **Laplacian** (introduced in Chapter 8) of a graph as well. In general, the modulus of the largest eigenvalue of M is called the **spectral radius** of M and is written $\rho(M)$.

Let S be some subset of the reals. Denote the operator which raises a square matrix A to the t^{th} power, $t \in S$, by Exp_t. Thus, $\mathrm{Exp}_t A = A^t$. This definition is put in such general terms because the class of diagonally *psd* matrices (to be introduced later) allows the introduction of fractional matrix powers in a well-defined way. The entry-wise product

[3]The concepts of reducibility and irreducibility are in fact usually defined in the more general setting of complex matrices, but this is not needed here.

between two matrices A and B of the same dimensions $m \times n$ is called the **Hadamard–Schur product** and is denoted by $A \circ B$. It has dimensions $m \times n$ and is defined by $[A \circ B]_{pq} = A_{pq}B_{pq}$. The Hadamard power (with exponent r) of a matrix A of dimensions $m \times n$ has the same dimensions, is written $A^{\circ r}$, and is defined by $[A^{\circ r}]_{pq} = (A_{pq})^r$.

Let A be square of dimension n, and assume some ordering on the set of k-tuples with distinct indices in $\{1, \ldots, n\}$. The k^{th} **compound** of a square matrix A is the matrix of all minors of order k of A, and is written $\mathrm{Comp}_k(A)$. It has dimension $\binom{n}{k}$. Its pq entry is equal to $\det A[u_p|u_q]$, where u_i is the i^{th} k-tuple of distinct indices in the given ordering.

Diagonal matrices (square matrices for which all off-diagonal entries are zero) are written as d_v, where v is the vector of diagonal entries. A **circulant matrix** is a matrix C such that $C_{kl} = C_{k+1\,l+1}$ for all k and l (counting indices modulo the dimension of the matrix). This implies that the first (or any) column (or row) of a circulant defines the matrix. A circulant is written as C_x, where x is its first column vector.

Given a symmetric (hermitian) matrix A, and a real (complex) vector x of fitting dimension, the scalar x^*Ax is called a **symmetric (hermitian) form**, where x^* denotes the hermitian conjugate of x.

The **Perron root** of a nonnegative matrix (i.e. a matrix for which all elements are nonnegative) is its spectral radius. It is a fundamental result in Perron–Frobenius theory ([**19**], page 26) that the Perron root of a nonnegative matrix A is an eigenvalue of A. The corresponding eigenvector is called the **Perron vector** and is guaranteed to be nonnegative. If A is irreducible then the Perron vector has no zero coordinates and the Perron root is a simple eigenvalue of A. An excellent monograph on this and other subjects in the field of nonnegative matrices is [**19**].

There are many different subjects in matrix analysis and many textbooks and monographs on different collections of subjects. I found the following books especially useful: *Matrix Analysis* by Horn and Johnson [**86**], *Topics in Matrix Analysis* by the same authors [**87**], *Nonnegative Matrices In The Mathematical Sciences* by Berman and Plemmons [**19**], *Nonnegative Matrices* by Minc [**130**], *Non–negative matrices and Markov chains* by Seneta [**149**], *Special matrices and their applications in numerical mathematics* by Fiedler [**57**], and *Matrix Computations* by Golub and Van Loan [**67**].

4.4 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 $e_{(i)} \overset{\mathrm{c}}{=} c$.

5

The graph clustering paradigm

In graph clustering one seeks to divide the node set of a graph into natural groups with respect to the edge relation. The first section of this chapter gives a brief account of three related ideas for accomplishing this task. They are formulated in terms of *path numbers*, *random walks*, and *shortest paths*. In Section 5.2 proposals towards graph clustering that have a combinatorial nature are discussed. A relaxation of one of them is the subject of Section 5.3. 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 5.4 probabilistic cluster algorithms based on the ideas in the first section 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. Section 5.5 begins with an example of (deterministically computed) random walks on an undirected graph possessing (weak) cluster structure. 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 5.3) 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 in the last section of this chapter. The relationship between the *MCL* process and cluster interpretation of graphs is the subject of Chapter 6, together with an analysis of convergence of the *MCL* process and stability of the limits with respect to the cluster interpretation. Chapter 7 gives conditions under which iterands of the process have real c.q. nonnegative spectrum, and which imply the presence of generalized cluster structure in the iterands.

5.1 Paths, walks, and cluster structure in graphs

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[1]; 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 5.4. The following section describes 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.

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

5.2 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. [**120, 121, 122, 150**]) 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 [**121**] 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 [**8**] 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 [**61**]. 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 5 is taken from the article [**121**], 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.

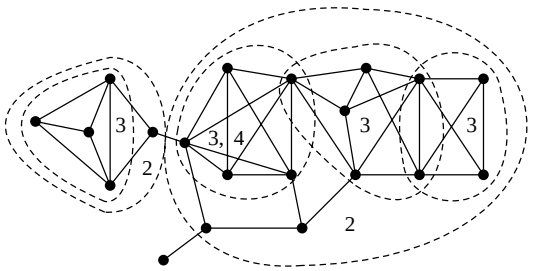


Figure 5. Graph with its k-blocks.

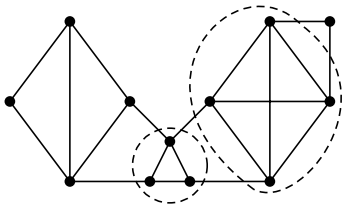


Figure 6. Graph with its 3-blocks.

The lack of balance is even stronger in the graph which is depicted in Figure 6, 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 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 [**121**] and Tomi [**161**] 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.

5.3 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 [**80**] 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 [**160**], who was apparently unaware of the work of Matula.

5.3.1 k-path clustering. 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

$$Z_{k,G}(v_i, v_j) = (M^k)_{ij}, \qquad i \neq j \tag{1}$$

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.

5.3.2 Odd and even. The graph G_1 in Figure 7 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 9.

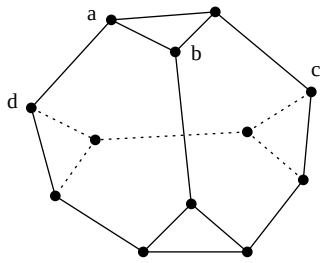


Figure 7. Topped tetraeder G_1.

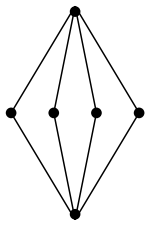


Figure 8. Bipartite graph G_2.

$$M = \mathcal{M}_{G_1}$$

$$\begin{pmatrix}
0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0
\end{pmatrix}$$

$$M^2$$

$$\begin{pmatrix}
3 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
1 & 3 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\
1 & 1 & 3 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\
1 & 1 & 0 & 3 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 1 & 3 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 1 & 1 & 3 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 & 1 & 0 & 3 & 1 & 1 & 1 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 1 & 1 & 3 & 1 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 3 & 0 & 1 & 1 \\
1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 3 & 1 & 1 \\
1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 3 & 1 \\
0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 3
\end{pmatrix}$$

$$M + I$$

$$\begin{pmatrix}
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1
\end{pmatrix}$$

$$(M + I)^2$$

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

Figure 9. Several matrices associated with G_1.

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).

5.3.3 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 9, where the largest off-diagonal matrix entries (diagonal entries are disregarded) are printed in boldface. Each coefficient now yields

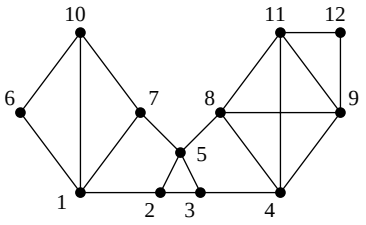


Figure 10. Graph G_3.

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

Figure 11. The matrix $(N+I)^2$, $N = \mathcal{M}_{G_3}$.

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 10, also used on page 42. Its associated matrix after adding loops to each node is given next to it in Figure 11. 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 8 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[2]. By adding multiple loops to each node of G_2 it can be ensured that the resulting clustering corresponds with connected components only

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

(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.

5.3.4 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 10. I remarked that under the coefficient Z_{4,G_3} 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.

5.4 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. An application of the latter is briefly described in Chapter 8, namely polynomial time approximation schemes based on Markov chains. These are interesting because a necessary condition is in general that the Markov chains be *rapidly mixing*, which essentially requires that the subdominant eigenvalue is well separated from the largest eigenvalue one. This relationship between the spectrum of a graph and its connectivity properties plays a role in many applications in graph theory (Chapter 8), and it does so too in the *MCL* process.

5.4.1 Probabilistic cluster algorithms. In the graph partitioning community, several randomized cluster algorithms have been proposed. I follow the survey article [**8**] by Alpert and Kahng which was written in 1995. Karger [**99**] 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 [**29, 159**]. 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 [**75**]; 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 [**174**] 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.

5.4.2 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.[3] This will be the starting point for the *MCL* algorithm. An excellent survey on random graphs is [**114**] 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*

[3] 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.

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

$$\mathcal{T}_G = \mathcal{M}_G d^{-1} \tag{2}$$

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.

LEMMA 1. *Let G be undirected and void-free*[4]*, 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

$$\frac{\lambda_k(M)}{u} \leq \lambda_k(T) \leq \frac{\lambda_k(M)}{l} \qquad \lambda_k(T) > 0 \tag{3}$$

$$\frac{\lambda_k(M)}{l} \leq \lambda_k(T) \leq \frac{\lambda_k(M)}{u} \qquad \lambda_k(T) < 0 \tag{4}$$

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 ([**86**], 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 ([**86**], 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. □

5.4.3 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

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

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 12 the matrix $M = \mathcal{T}_{G_3+I}$ (corresponding with the graph G_3 in Figure 10) 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 10 clearly has this property, and the limit is found in Figure 12, 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 [**19**]. 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.

5.5 An example *MCL* run

Consider Figure 12 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 11, inducing the similarity coefficients $Z_{k,G}$. 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.

5.5.1 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

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

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

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

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^∞

Figure 12. Powers of $M = \mathcal{T}_{G_3+I}$, the Markov matrix associated with the graph G_3 in Figure 10, loops added to G_3

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_1q}/T_{p_2q} 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.

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

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

DEFINITION 3. *Given a matrix $M \in \mathbb{R}^{k\times l}$, $M \geq 0$, 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 $\Gamma_r : \mathbb{R}^{k\times l} \to \mathbb{R}^{k\times l}$ is defined by*

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

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 *m*ajorization. This is discussed in the following chapter, Section 6.2. Perhaps surprisingly, the Γ_r operator maps a rather large class of matrices with real spectrum onto itself, and if $r \in \mathbb{N}$, the subset of this class with nonnegative spectrum is preserved as well. These classes are introduced in Chapter 7.

5.5.2 Iterating expansion and inflation. Figure 13 gives the result of applying Γ_r to the Markov matrix M^2 given in Figure 12. 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[5] of this process (provisionally denoted by M^{∞}_{mcl}) are given in Figure 13 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.

In the next section the *MCL* process is formally described, and the relationship between equilibrium states of the *MCL* process and clusterings is formalized. 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 14, corresponding with a line–graph on 7 nodes, loops added to each node. An *MCL* run with $e_{(i)} \triangleq 2, r_{(i)} \triangleq 2$ 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.

5.6 Formal description of the *MCL* algorithm

The basic design of the *MCL* algorithm is given in Figure 15; 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[6] of expansion powers $e_{(i)}$ and the

[5] Idempotent under both Exp_2 and Γ_2.

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

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

$\Gamma_2 M^2$, M defined in Figure 12

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

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

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

$(\Gamma_2 \circ Squaring)$ 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 13. Iteration of $(\Gamma_2 \circ Squaring)$ with initial iterand M defined in Figure 12.

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 14. *MCL* run on a line-graph on 7 nodes

row of inflation powers $r_{(i)}$ influence the granularity of the resulting partition. The matrices in Figure 13 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 chapter. 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 the following chapter. It is useful to

```
                                        # G is a voidfree graph.
                                        # e_i ∈ ℕ, e_i > 1, i = 1,....
MCL (G, Δ, e_(i), r_(i)) {              # r_i ∈ ℝ, 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 15. The basic *MCL* algorithm. Convergence is discussed in Chapter 6.

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 $e_i \in \mathbb{N}, e_i > 1$, and $r_i \in \mathbb{R}, r_i > 0$, and is written*

$$(\cdot, e_{(i)}, r_{(i)}) \tag{5}$$

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

$$(\cdot^{d\times d}, e_{(i)}, r_{(i)}) \tag{6}$$

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

$$(M, e_{(i)}, r_{(i)}) \tag{7}$$

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 $T_{2i+1} = \Gamma_{r_i}(T_{2i})$, $i = 1, \ldots, \infty$. □

In practice, the algorithm iterands converge nearly always to a doubly idempotent matrix. In the next section 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 Chapter 7 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, in Chapter 7 it is shown that 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. The figure below 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 5 taken from [**121**], loops added to the graph.

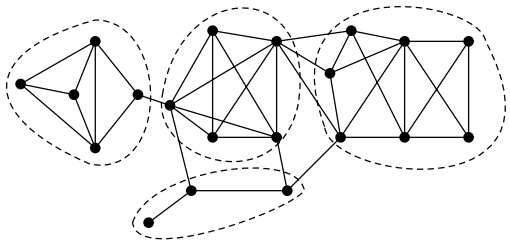


Figure 16. *MCL* Clustering of the graph in Figure 5.

6

Basic *MCL* theory

This chapter 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 6.2 simple properties of the Γ operator are derived. The next two sections deal with equilibrium states which may possibly arise as limits in an *MCL* process. The chapter concludes with a section on convergence towards equilibrium states and the stability of the *MCL* process around them.

6.1 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 [**19**]. 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 59, 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.*

$$(\alpha \to \beta) \wedge (\beta \to \gamma) \implies \alpha \to \gamma \tag{8}$$

$$(\alpha \to \alpha) \wedge (\alpha \to \beta) \implies \beta \to \alpha \tag{9}$$

$$\alpha \to \beta \implies \beta \to \beta \tag{10}$$

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 8. 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 $k \in \mathbb{N}$ such that $(M^k)_{\beta\alpha} = 0$, whereas $M_{\beta\alpha} \neq 0$. The existence of such a circuit implies by 9 and 10 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 59 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 $\to$ on V_x. Define a relation ν on $\mathcal{E} \times V$ by setting $\nu(E, \alpha) = 1$ if $\exists \beta \in E$ with $\alpha \to \beta$, and $\nu(E, \alpha) = 0$ otherwise. The overlapping clustering $C\mathcal{L}_M = \{C_1, \ldots, C_d\}$ associated with M, defined on V, has d elements. The i^{th} cluster $C_i, i = 1, \ldots, d$ is defined by Equation (11).*

$$C_i = \{\, v \in V \mid \nu(E_i, v) = 1 \}. \tag{11}$$

□

Note that the set of clusters is precisely the set of weakly connected components[1] 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 chapter. 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[2]. 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

[1]For the definition of weakly connected components see page 34.

[2]This is suggested by practical evidence. It is conjectured in Chapter 7 that the *MCL* process converges almost always if the input graph is symmetric.

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 Chapter 10.

6.2 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 6.3 to show the existence of certain periodic limits of the *MCL* process.

Following [**118**], 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 (12) and (13) hold. This is denoted by $x \prec y$.*

$$x_{[1]} + \cdots + x_{[k]} \leq y_{[1]} + \cdots + y_{[k]} \qquad k = 1, \ldots, n-1 \tag{12}$$

$$x_{[1]} + \cdots + x_{[n]} = y_{[1]} + \cdots + y_{[n]} \tag{13}$$

□

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 [**118**]).

LEMMA 2. *Let π be a probability vector, let r be a real number, $r > 0$. The two inequalities (14) and (15) are implied by the fact that π and $\Gamma_r \pi$ satisfy the conditions of Lemma 3. The two equalities (16) and (17) are obvious.*

$$\pi \prec \Gamma_r \pi \qquad r > 1 \tag{14}$$

$$\pi \succ \Gamma_r \pi \qquad r < 1 \tag{15}$$

$$\pi = \Gamma_r \pi \qquad r = 1 \tag{16}$$

$$\pi = \Gamma_r \pi \qquad \pi \text{ is homogeneous} \tag{17}$$

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*

$$x_{[1]}/y_{[1]} \leq x_{[2]}/y_{[2]} \leq \cdots \leq x_{[n]}/y_{[n]} \tag{18}$$

□

LEMMA 3. *([**118**], 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

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

□

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 $\pi_i = \max_i$. 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 $\Gamma_s(\pi) \lhd \Gamma_t(\pi)$ 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 Chapter 9 a particular Schur–convex function is one of the main ingredients of a performance criterion for graph clustering (page 104 ff.). A celebrated result in the theory of majorization is that $x \prec y$ iff there is a doubly stochastic matrix D such that $x = Dy$ [**117**].

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 [**117**] 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 Section 7.4). 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 6.4, 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 $r \in \mathbb{R}$ be positive. Denote the Kronecker product by $(\cdot \otimes \cdot)$. Equation (19) holds.*

$$\Gamma_r(A \otimes B) = \Gamma_r A \otimes \Gamma_r B \tag{19}$$

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 (19) by proving that the ratios between two entries in the same column is the same on both sides of Equation (19). 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 (19). 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 (20) holds.*

$$K_j = A_j \otimes B_j \qquad\qquad j = 1,\ldots,\infty \tag{20}$$

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

6.3 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, $c \in \mathbb{R}, c > 1$.
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 8 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.

The implementation with which experiments were carried out so far uses rows $r_{(i)}$ and $e_{(i)}$ which have even much simpler structure. The row $e_{(i)}$ assumes the constant 2 everywhere. The row $r_{(i)}$ is allowed to have a prefix of length N, in which it assumes one constant, and it may assume another constant on the postfix of infinite length starting at position $N+1$. The examples in Chapter 10 use such input rows.

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 66.

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[3] 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 17 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 17 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 $k \in \mathbb{N}$ 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 Chapter 7.

[3] See Definition 25 on page 116 for the precise definition of a ring graph.

6.4 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 17, 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*

$$\alpha_n = \frac{\sqrt[3]{v_n}}{6(n-1)} - \frac{2(3n-4)}{3(n-1)\sqrt[3]{v_n}} - \frac{1}{3(n-1)} \tag{21}$$

$$v_n = 108n^2 - 180n + 64 + 12(n-1)\sqrt{3n(27n-32)} \tag{22}$$

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 17. □

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 non-negative vector satisfying $a + e \prec a$, let f be likewise a vector such that $a + f \lhd a$. Extend the definition of $\prec$ $(\lhd)$ to circulants by setting $C_x \prec C_y$ iff $x \prec (\lhd) y$. Now it is easy to prove that $C_x \prec C_y \implies {C_x}^2 \prec {C_y}^2$, 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.

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

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

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.

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

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

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

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

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

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

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

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

Figure 17. $(\Gamma_2 \circ \mathrm{Exp}_2)$–*id* matrices.

One interesting freak flip–flop state exists in dimension 3, which has the form of a non-symmetric 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[4].

$$\begin{pmatrix} 0.795668870 & 0.004344249 & 0.199986881 \\ 0.199986881 & 0.795668870 & 0.004344249 \\ 0.004344249 & 0.199986881 & 0.795668870 \end{pmatrix} \tag{23}$$

6.5 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 $(M+\epsilon, e_{(i)} \stackrel{c}{=} 2, r_{(i)} \stackrel{c}{=} 2$ 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

[4] Though it is not *diagonally symmetric*; see the next chapter.

all input matrices $M+\epsilon$ with regard to the *MCL* process $(M+\epsilon, e_{(i)} \stackrel{c}{=} 2, r_{(i)} \stackrel{c}{=} 2)$, for ϵ small enough.

THEOREM 2. *The MCL process with standard parameters* $(\cdot, e_{(i)} \stackrel{c}{=} 2, r_{(i)} \stackrel{c}{=} 2)$*, 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* $M \in \mathbb{R}_{\geq 0}{}^{n\times n}$ *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* $\mathbb{R}^{n\times n}$*, 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* $\Gamma((M+\epsilon)^2) = M+\delta$*.*

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

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 $(M + \epsilon_f, e_{(i)} \hat{=} 2, r_{(i)} \hat{=} 2)$ 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 (Chapter 7).

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 C does not have overlap, lead the iterands of the *MCL* process $(M + \epsilon, e_{(i)} \hat{=} 2, r_{(i)} \hat{=} 2)$ to stay within a class of matrices the block structure of which only admits a clustering which is a refinement of 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 $(M, e_{(i)} \hat{=} 2, r_{(i)} \hat{=} 2)$. 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 $(M, e_{(i)} \hat{=} 2, r_{(i)} \hat{=} 2)$, 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 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 $(M+\epsilon, e_{(i)} \overset{c}{=} 2, r_{(i)} \overset{c}{=} 2)$, 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 $(M+\epsilon, e_{(i)} \overset{c}{=} 2, r_{(i)} \overset{c}{=} 2)$ corresponds with a clustering which is identical to or a refinement of 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 $(\frac{1}{k}J_k + \epsilon, e_{(i)} \overset{c}{=} 2, r_{(i)} \overset{c}{=} 2)$, 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 $(M+\epsilon, e_{(i)} \overset{c}{=} 2, r_{(i)} \overset{c}{=} 2)$, 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 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 $(M+\epsilon, e_{(i)} \overset{c}{=} 2, r_{(i)} \overset{c}{=} 2)$, has an associated clustering which is identical to 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 Section 7.4 of the next chapter, 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 Chapter 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 Section 7.2, 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.

7

Structure theory for the MCL process

The *MCL* process is interesting from a mathematical point of view, since it apparently has the power to 'inflate' the spectrum of a stochastic matrix, by pressing large eigenvalues towards 1. This effect is strong enough to overcome the effect of matrix exponentiation, which has the property of exponentiating the associated eigenvalues. The focus in this chapter is on the Γ_r operator. The fundamental property established is that Γ_r maps two nested classes of stochastic matrices with real spectra onto themselves (Theorem 7). The largest class is that of diagonally symmetrizable stochastic matrices, i.e. matrices which are diagonally similar to a symmetric matrix without further constraints. This class is mapped onto itself by Γ_r for arbitrary $r \in \mathbb{R}$. Defining *diagonally positive semi-definite* (*dpsd*) as the property of being diagonally similar to a positive semi-definite matrix, the second class is that of stochastic *dpsd* matrices. This class is mapped onto itself by Γ_r for $r \in \mathbb{N}$.

Using the property that minors of a *dpsd* matrix A are nonnegative, it is shown that the relation $\leadsto$ defined on the nodes of its associated graph (or equivalently on the column, respectively row indices of A) by $q \leadsto p \equiv |A_{pq}| \geq |A_{qq}|$, for $p \neq q$, is a directed acyclic graph (*DAG*) if indices of identical[1] columns, respectively rows are lumped together (Theorem 9). This generalizes the mapping from nonnegative idempotent column allowable matrices onto overlapping clusterings (Definition 8). and it sheds some light on the tendency of the *MCL* process limits to have a larger number of weakly connected components than the input graph. It is then shown that applying Γ_∞ to a stochastic *dpsd* matrix M yields a matrix[2] which has spectrum of the form $\{0^{n-k}, 1^k\}$, where k, the multiplicity of the eigenvalue 1, equals the number of endclasses of the ordering of the columns of M provided by the associated *DAG*. It is not necessarily true that $\Gamma_\infty M$ is idempotent. However, the observation is confirmed that Γ_r tends to inflate the spectrum of M for $r > 1$, as $\Gamma_r M$ may be regarded as a function of varying r for fixed M, and as such is continuous.

In Section 7.1 various lemmas and theorems formalizing the results described above are given. Section 7.2 introduces structure theory for *dpsd* matrices and gives properties of the inflation operator applied to stochastic *dpsd* matrices. Reductions and decompositions of *dpsd* matrices in terms of rank 1 matrices are given in Section 7.3. Hilbert's distance for nonnegative vectors, the contraction ratio of a nonnegative matrix, and their relationship with the *MCL* process comprise Section 7.4. Conclusions, further research, and related research make up the last section.

[1] Modulo multiplication by a scalar on the complex unit circle.

[2] The matrix $\Gamma_\infty M$ is defined as $\lim_{r\to\infty} \Gamma_r M$, which exists for all stochastic M. See Definition 14.

7.1 Properties of inflation and stochastic *dpsd* matrices

At first sight the inflation operator seems hard to get a grasp on mathematically. It is clear that describing its behaviour falls outside the scope of classical linear algebra, as it represents a scaling of matrices that is both non-linear, column-wise defined, and depends on the choice of basis. In general $\Gamma_r M$ can be described in terms of a Hadamard matrix power which is postmultiplied with a diagonal matrix. For a restricted class of matrices there is an even stronger connection with the Hadamard–Schur product. These are the class of stochastic diagonally symmetrizable matrices and a subclass of the latter, the class of stochastic diagonally positive semi-definite matrices.

DEFINITION 11. *A square matrix A is called* ***diagonally hermitian*** *if it is diagonally similar to a hermitian matrix. If A is* real *then A is called* ***diagonally symmetrizable*** *if it is diagonally similar to a symmetric matrix. Given a hermitian matrix A, equivalent formulations for diagonal symmetrizability are:*

i) *There exists a positive vector x such that ${d_x}^{-1} A d_x$ is hermitian, or equivalently, such that $(x_l/x_k)A_{kl} = (x_k/x_l)\overline{A_{lk}}$. If A is real, Identity (24) holds.*

$$d_x{}^{-1} A d_x = [A \circ A^T]^{\circ 1/2} \tag{24}$$

ii) *There exists a positive vector y such that $A d_y$ is hermitian, or equivalently, such that $\overline{A_{kl}}/A_{lk} = y_k/y_l$. The vector y is related to x above via ${d_x}^2 = d_y$ or equivalently $y = x \circ x$.* □

The fact that d_x can always be chosen with a positive real x depends on the following. Let d_u be a diagonal matrix, where each u_i is a complex number on the unit circle. Then the decomposition $A = d_x S {d_x}^{-1}$, where S is hermitian, can be rewritten as

$$A = (d_x d_u)({d_u}^{-1} S d_u)(d_u d_x)^{-1}$$

with $S' = {d_u}^{-1} S d_u$ hermitian. This depends on the fact that on the unit circle the inverse of a complex number equals its conjugate.

DEFINITION 12. *A square matrix is called* ***diagonally positive-semi definite*** *if it is diagonally similar to a positive semi-definite matrix, it is called* ***diagonally positive definite*** *if it is diagonally similar to a positive definite matrix. The phrases are respectively abbreviated as* dpsd *and* dpd. □

REMARK. If M is diagonally symmetrizable stochastic, and y is such that $M d_y$ is symmetric, then $My = y$; thus y represents the equilibrium distribution of M. In the theory of Markov chains, a stochastic diagonally symmetrizable matrix is called *time reversible* or said to satisfy the *detailed balance* condition (See e.g. [**114, 154**]). A slightly more general definition and different terminology was chosen here. The main reason is that the term 'time reversible' is coupled tightly with the idea of studying a stochastic chain via (powers of) its associated stochastic matrix, and is also used for continuous–time Markov chains. The *MCL* process does not have a straightforward stochastic interpretation, and the relationship between an input matrix and the subsequent iterands is much more complex. Moreover, it is natural to introduce the concepts of a matrix being diagonally

similar to a positive (semi–) definite matrix; clinging to 'time reversible' in this abstract setting would be both contrived and unhelpful. The proposed phrases seem appropriate, since several properties of hermitian and *psd* matrices remain valid in the more general setting of diagonally hermitian and *dpsd* matrices. Lemma 13 lists the most important ones, which are easy to verify. Probably all these results are known.

LEMMA 13. *Let A be diagonally hermitian of dimension n, let α be a list of distinct indices in the range $1 \dots n$, let k and l be different indices in the range $1 \dots n$. Let x be such that $S = {d_x}^{-1} A d_x$ is hermitian, and thus $A = d_x S {d_x}^{-1}$. Let $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ be the decreasingly arranged eigenvalues of A (and S), let $a_1 \geq a_2 \geq \cdots \geq a_n$ be the decreasingly arranged diagonal entries of A.*

a) $A[\alpha|\alpha] = d_x[\alpha|\alpha] \;\; S[\alpha|\alpha] \;\; d_x[\alpha|\alpha]^{-1}$, *in particular, the diagonal entries of A equal the diagonal entries of S. This implies that the majorization relationship between eigenvalues and diagonal entries for hermitian matrices carry over to diagonally hermitian matrices: The spectrum of A majorizes the vector of diagonal entries of A:*

$$\sum_{i=1}^{k} \lambda_i \;\; \geq \;\; \sum_{i=1}^{k} a_i \qquad k = 1, \dots, n$$

Together with the first equality this implies that diagonally hermitian matrices satisfy the same interlacing inequalities for bordered matrices as hermitian matrices do.

b) $\mathrm{Comp}_k(A) = \mathrm{Comp}_k(d_x)\ \mathrm{Comp}_k(S)\ \mathrm{Comp}_k({d_x}^{-1})$, *thus the compound*[3] *of a diagonally hermitian matrix is diagonally hermitian. Moreover, the compound of a* dpd *(*dpsd*) matrix is again* dpd *(*dpsd*).*

c) $\det A[\alpha|\alpha] = \det S[\alpha|\alpha]$, *i.e. corresponding principal minors of A and S are equal. If A is* dpsd *then* $\det A[\alpha|\alpha] \geq 0$, *with strict inequality if A is* dpd.

d) *If A is* dpsd *and $A_{kk} = 0$ then the k^{th} row and the k^{th} column of A are zero. If A is* dpsd *and* $\det A[kl|kl] = 0$, *then row k and row l are proportional, and column k and column l are proportional.*

e) *If A is* dpsd, *then for each $k \in \mathbb{N}$, there exists a unique* dpsd *matrix B such that $B^k = A$. This matrix is defined by setting $B = d_x Q \Lambda^{1/k} Q^H {d_x}^{-1}$, where $Q \Lambda Q^H$ is a unitary diagonalization of S, Λ is the diagonal matrix of eigenvalues of S, and $\Lambda^{1/k}$ is the matrix Λ with each diagonal entry replaced by its real nonnegative k^{th} root. This implies that for* dpsd *A, the fractional power A^t, $t \in \mathbb{R}_{\geq 0}$, can be defined in a meaningful way.*

f) *If A, B are both of dimension n and diagonally hermitian,* dpsd, dpd, *then the Hadamard–Schur product $A \circ B$ is diagonally hermitian,* dpsd, dpd.

PROOF. Most statements are easy to verify. For extensive discussion of the majorization relationship between diagonal entries and eigenvalues of hermitian matrices, as well as results on interlacing inequalities see [**86**]. Statement b) follows from the fact that the compound operator distributes over matrix multiplication, and the fact that the compound of a positive (semi–) definite matrix is again positive (semi–) definite.

[3] See page 37 for the definition of the compound of a matrix.

See [**57**] for an overview of results on compounds of matrices. c) follows from the fact that each term contributing to the principal minor in A is a product $\prod_i A_{k_i k_{i+1}}$ where each k_i occurs once as a row index and once as a column index, implying the equalities $\prod_i A_{k_i k_{i+1}} = \prod (x_{k_i}/x_{k_{i+1}}) A_{k_i k_{i+1}} = \prod S_{k_i k_{i+1}}$. Then it is a well known property of positive semi-definite matrices that the principal minors are nonnegative (see e.g. [**86**], page 404). The first statement in d) follows from the fact that principal minors (of dimension 2) are nonnegative. Also, if $\det A[kl|kl] = 0$, then the kl diagonal entry of $\mathrm{Comp}_2(A)$ is zero, and consequently the kl row and the kl column are also zero. Some calculations then confirm the second statement, which will be of use later on. For e) it is sufficient to use the fact that $Q\Lambda^{1/k}Q^H$ is the unique positive semi-definite k^{th} root of S (see [**86**], page 405). □

REMARK. The two most notable properties which do not generalize from hermitian matrices to diagonally hermitian matrices are the absence of an orthogonal basis of eigenvectors for the latter, and the fact that the sum of two diagonally hermitian matrices is in general not diagonally hermitian as well. In the following chapter a well-known theorem by Fiedler, relating the second eigenvector of a symmetric matrix to connectivity properties of its underlying graph, is shown to apply to diagonally symmetric matrices as well.

Statements c) and d) in Lemma 13 are used in associating a *DAG* with each *dpsd* matrix in Theorem 9. First the behaviour of the inflation operator on diagonally symmetrizable and *dpsd* matrices is described.

THEOREM 6. *Let M be a column stochastic diagonally symmetrizable matrix of dimension n, let d_x be the diagonal matrix with positive diagonal such that $S = {d_x}^{-1} M d_x$ is symmetric, and let r be real. Define the positive vector z by setting $z_k = {x_k}^r (\sum_i {M_{ik}}^r)^{1/2}$, and the positive rank 1 symmetric matrix T by setting $T_{kl} = 1/(\sum_i {M_{ik}}^r)^{1/2} (\sum_i {M_{il}}^r)^{1/2}$. The following statement holds.*

$$ {d_z}^{-1} \, (\Gamma_r M) \, d_z \;=\; S^{\circ r} \circ T, \qquad \textit{which is symmetric.} $$

PROOF. Define the vector t by $t_k = \sum_i {M_{ik}}^r$. Then

$$\begin{aligned} \Gamma_r M &= M^{\circ r} \, {d_t}^{-1} \\ &= (d_x \, S \, {d_x}^{-1})^{\circ r} \, {d_t}^{-1} \\ &= {d_x}^{\circ r} \; S^{\circ r} \; ({d_x}^{\circ r})^{-1} \, {d_t}^{-1} \\ &= {d_t}^{1/2} {d_t}^{-1/2} \, {d_x}^{\circ r} \; S^{\circ r} \; ({d_x}^{\circ r})^{-1} {d_t}^{-1/2} {d_t}^{-1/2} \\ &= ({d_t}^{1/2} {d_x}^{\circ r}) \; ({d_t}^{-1/2} S^{\circ r} {d_t}^{-1/2}) \; ({d_t}^{1/2} {d_x}^{\circ r})^{-1} \end{aligned}$$

Since the matrix ${d_t}^{-1/2} S^{\circ r} {d_t}^{-1/2}$ equals $S^{\circ r} \circ T$, the lemma holds. □

THEOREM 7. *Let M be square column stochastic diagonally symmetrizable, let z, S and T be as in Theorem 6.*

i) *The matrix $\Gamma_r M$ is diagonally symmetrizable for all $r \in \mathbb{R}$.*

ii) *If* M *is* dpsd *then* $\Gamma_r M$ *is* dpsd *for all* $r \in \mathbb{N}$, *if* M *is* dpd *then* $\Gamma_r M$ *is* dpd *for all* $r \in \mathbb{N}$.

PROOF. Statement i) follows immediately from Theorem 6. Statement ii) follows from the fact that the Hadamard–Schur product of matrices is positive (semi-) definite if each of the factors is positive (semi-) definite. Moreover, if at least one of the factors is positive definite, and none of the other factors has a zero diagonal entry, then the product is positive definite (see e.g. [**87**], page 309). These are basic results in the theory of Hadamard–Schur products, an area which is now covered by a vast body of literature. An excellent exposition on the subject is found in [**87**]. It should be noted that $r \in \mathbb{N}$ is in general a necessary condition ([**87**], page 453). The above result is pleasant in the sense that it gives both the inflation operator and the *MCL* process mathematical footing.

THEOREM 8. *Let* M *be diagonally symmetric stochastic, and consider the MCL process* $(M, e_{(i)}, r_{(i)})$.

i) *All iterands of this process have real spectrum.*
ii) *If* $r_i = 2$ *eventually, and* $e_i = 2$ *eventually, then the iterands of the process* $(M, e_{(i)}, r_{(i)})$ *are* dpsd *eventually.*

These statements[4] follow from the fact that Exp_2 maps diagonally symmetric matrices onto *dpsd* matrices and from Theorem 6 ii). □

Theorem 8 represents a qualitative result on the *MCL* process. Under fairly basic assumptions the spectra of the iterands are real and nonnegative. In the previous chapter it was furthermore proven that the *MCL* process converges quadratically in the neighbourhood of nonnegative doubly idempotent matrices These combined facts indicate that the *MCL* process has a sound mathematical foundation. The fact remains however that much less can be said about the connection between successive iterands than in the case of the usual Markov process. Clearly, the process has something to do with mixing properties of different subsets of nodes. If there are relatively few paths between two subsets, or if the combined capacity of all paths is low, then flow tends to evaporate in the long run between the two subsets. It was actually this observation which originally led to the formulation of the *MCL* process as the basic ingredient of a cluster algorithm for graphs.

The question now rises whether the *MCL* process can be further studied aiming at quantitative results. It was seen that $\Gamma_r M$, $r \in \mathbb{N}$, can be described in terms of a Hadamard–Schur product of positive semi-definite matrices relating the symmetric matrices associated with M and $\Gamma_r M$ (in Theorem 7). There are many results on the spectra of such

[4]Clearly the condition under ii) can be weakened; it is only necessary that e_i is at least one time even for an index $i = k$ such that $r_i \in \mathbb{N}$ for $i \geq k$. However, the assumptions under ii) can be viewed as a standard way of enforcing convergence in a setting genuinely differing from the usual Markov process.

products. These are generically of the form

$$\sum_{i=1}^{k} \sigma_i(A \circ B) \;\leq\; \sum_{i=1}^{k} f_i(A)\sigma_i(B), \qquad k = 1, \ldots, n.$$

Here $\sigma_i()$ denotes the i–largest singular value, and $f_i(A)$ may stand (among others) for the i–largest singular value of A, the i–largest diagonal entry of A, the i–largest Euclidean column length, or the i–largest Euclidean row length (see [**87**]). Unfortunately such inequalities go the wrong way in a sense. Since the inflation operator has apparently the ability to press several large eigenvalues towards 1, what is needed are inequalities of the type

$$\sum_{i=1}^{k} \sigma_i(A \circ B) \;\geq\; ??? \quad .$$

However, the number of eigenvalues pressed towards 1 by Γ_r depends on the density characteristics of the argument matrix, and it could be zero (noting that one eigenvalue 1 is always present). Moreover, Γ_r has also the ability to press small eigenvalues towards zero. Clearly, one cannot expect to find inequalities of the '$\geq$' type without assuming anything on the density characteristics of M. It is shown in the next section that the classic majorization relation formulated in Lemma 13 a) between the eigenvalues and diagonal entries of a *dpsd* matrix, plus a classification of the diagonal entries of a *dpsd* matrix, gives useful information on the relationship between eigenvalues of a stochastic *dpsd* matrix and its image under Γ_r.

A second area of related research is found in the field of rapidly mixing Markov chains. A good reference is [**154**]. The focus is also on mixing properties of node subsets of Markov graphs, and the Markov graphs used are generally of the time reversible kind, i.e. correspond with diagonally symmetrizable matrices. Transfer of results is not likely however. The derived theorems depend crucially on the fact that a Markov process is considered which corresponds with the row of powers of a given Markov matrix. Bounds securing a minimal amount of mixing are sought in terms of the second–largest eigenvalue of a Markov matrix, and in terms of the notion of conductance, which depends on the equilibrium distribution of the matrix.

7.2 Structure in *dpsd* matrices

The main objective for this section is to establish structure theory for the class of *dpsd* matrices, and study the behaviour of Γ_∞ using these results. It will be shown that for stochastic *dpsd* M the spectrum of the matrix Γ_∞ is of the form $\{0^{n-k}, 1^k\}$, where k is related to a structural property of M. Throughout this section two symbols are used which are associated with a *dpsd* matrix A, namely the symbol $\nrightarrow$ which denotes an arc relation defined on the indices of A, and the symbol $\sim$ which denotes an equivalence relation on the indices of A. It should be clear from the context which matrix they refer to. All results in this section are stated in terms of columns; the analogous statements in terms of rows hold as well.

DEFINITION 13. *Let A be* dpsd *of dimension n, let k and l be different indices in the range $1 \ldots n$.*

i) *Define the equivalence relation $\sim$ on the set of indices $\{1, \ldots, n\}$ by $k \sim l \equiv$* columns k and l of A are scalar multiples of each other via scalars on the complex unit circle.
ii) *Define the arc relation $\hookrightarrow$ on the set of indices $\{1, \ldots, n\}$, for $p \neq q$, by $q \hookrightarrow p \equiv |A_{pq}| \geq |A_{qq}|$.*
iii) *Let E and F be different equivalence classes in $\{1, \ldots, n\}/\sim$. Extend the definition of $\hookrightarrow$ by setting $F \hookrightarrow E \equiv \exists e \in E, \exists f \in F[f \hookrightarrow e]$. By definition of $\hookrightarrow$ and $\sim$ the latter implies $\forall e' \in E, \forall f' \in F[f' \hookrightarrow e']$.* □

LEMMA 14. *Let A be* dpsd *of dimension n, let k and l be distinct indices in the range $1 \ldots n$. Then*

$$l \hookrightarrow k \wedge k \hookrightarrow l \text{ implies } k \sim l.$$

□

This follows from Lemma 13 d) and the fact that the premise implies $\det A[kl|kl] = 0$. Lemma 14 can be generalized towards the following statement.

THEOREM 9. *Let A be* dpsd *of dimension n.*

The arc $\hookrightarrow$ defines a directed acyclic graph (DAG) on $\{1, \ldots, n\}/\sim$.

Note that the theorem is stated in a column-wise manner. The analogous statement for rows is of course also true. The proof of this theorem follows from Lemma 15. □

LEMMA 15. *Let A be* dpsd *of dimension n, suppose there exist k distinct indices $p_i, i = 1, \ldots, k, k > 1$, such that $p_1 \hookrightarrow p_2 \hookrightarrow \cdots \hookrightarrow p_k \hookrightarrow p_1$. Then $p_1 \sim p_2 \sim \cdots \sim p_k$, and thus all $p_i, i = 1, \ldots, k$ are contained in the same equivalence class in $\{1, \ldots, n\}/\sim$. Furthermore, if A is real nonnegative then each of the subcolumns $A[p_1 \ldots p_k|p_i]$ is a scalar multiple of the all-one vector of length k.*

PROOF. Without loss of generality assume $1 \hookrightarrow 2 \hookrightarrow \cdots \hookrightarrow k \hookrightarrow 1$. The following inequalities hold, where the left–hand side inequalities follow from the inequalities implied by $\det A[i\, i{+}1] \geq 0$ and $i \hookrightarrow i + 1$.

$$\begin{array}{ccccc} |A_{i\,i+1}| & \leq & |A_{i+1\,i+1}| & \leq & |A_{i+2\,i+1}| \\ |A_{k-1\,k}| & \leq & |A_{kk}| & \leq & |A_{1k}| \\ |A_{k1}| & \leq & |A_{11}| & \leq & |A_{21}| \end{array}$$

Now let x be positive such that $x_q A_{pq} = x_p \overline{A_{qp}}$. On the one hand, $|A_{kk}| \leq |A_{1k}|$. On the other hand,

$$\begin{aligned}
|A_{kk}| &\geq |A_{k-1\,k}| \\
&= \tfrac{x_{k-1}}{x_k} |A_{k\,k-1}| \\
&\geq \tfrac{x_{k-1}}{x_k} |A_{k-2\,k-1}| \\
&= \tfrac{x_{k-1}}{x_k} \tfrac{x_{k-2}}{x_{k-1}} |A_{k-1\,k-2}| \\
&\cdots \\
&\geq \tfrac{x_{k-1}}{x_k} \tfrac{x_{k-2}}{x_{k-1}} \cdots \tfrac{x_1}{x_2} |A_{k1}| \\
&= \tfrac{x_1}{x_k} |A_{k1}| \\
&= |A_{1k}|
\end{aligned}$$

This implies that $|A_{k-1\,k}| = |A_{kk}| = |A_{1k}|$ and the identities $|A_{i-1\,i}| = |A_{ii}| = |A_{i+1\,i}|$ are established by abstracting from the index k. From this it follows that $\det A[i, i+1|i, i+1] = 0$, and consequently $i \sim i+1$ for $i = 1, \ldots, k-1$ by Lemma 14. The identities $|A_{i-1\,i}| = |A_{ii}| = |A_{i+1\,i}|$ also imply the last statement of the lemma. □

DEFINITION 14. *Define* Γ_∞ *by* $\Gamma_\infty M = \lim_{r \to \infty} \Gamma_r M$. □

This definition is meaningful, and it is easy to derive the structure of $\Gamma_\infty M$. Each column q of $\Gamma_\infty M$ has k nonzero entries equal to $1/k$, (k depending on q), where k is the number of elements which equal $\max_p M_{pq}$, and the positions of the nonzero entries in $\Gamma_\infty M[1 \ldots n|q]$ correspond with the positions of the maximal entries in $M[1 \ldots n|q]$.

THEOREM 10. *Let M be stochastic* dpsd *of dimension n. Let D_M be the directed graph defined on $\{1, \ldots, n\}/\sim$ according to Definition 13, which is acyclic according to Theorem 9. Let k be the number of nodes in $\{1, \ldots, n\}/\sim$ which do not have an outgoing arc in D_M. These nodes correspond with (groups of) indices p for which M_{pp} is maximal in column p.*

The spectrum of $\Gamma_\infty M$ equals $\{0^{n-k}, 1^k\}$.

PROOF. For the duration of this proof, write S_A for the symmetric matrix to which a diagonally symmetrizable matrix A is similar. Consider the identity

$$S_{(\Gamma_r M)} = [\Gamma_r M \circ (\Gamma_r M)^T]^{\circ 1/2}$$

mentioned in Definition 11 i). The matrices $\Gamma_r M$ and $S_{\Gamma_r M}$ have the same spectrum. Now, let r approach infinity. The identity is in the limit not meaningful, since $\Gamma_\infty M$ is not necessarily diagonalizable, and thus the left–hand side may not exist in the sense that there is no symmetric matrix to which $\Gamma_\infty M$ is similar. However, the identity 'spectrum of $\Gamma_\infty M$ = spectrum of $[\Gamma_\infty M \circ (\Gamma_\infty M)^T]^{\circ 1/2}$' *does* remain true, since the spectrum depends continuously on matrix entries (see e.g. [**86**], page 540), and both limits exist. Thus, it is sufficient to compute the spectrum of S_∞, which is defined as

$$S_\infty = [\Gamma_\infty M \circ (\Gamma_\infty M)^T]^{\circ 1/2}$$

Note that the nonzero entries of $\Gamma_\infty M$ correspond with the entries of M which are maximal in their column. Whenever $[\Gamma_\infty M]_{kl} \neq 0$ *and* $[\Gamma_\infty M]_{lk} \neq 0$, it is true that $k \rightarrowtail l$ and $l \rightarrowtail k$. Now consider a column q in S_∞, and assume that $S_{\infty p_i q} \neq 0$, for $i = 1, \ldots, t$. It follows that $q \rightarrowtail p_i \wedge p_i \rightarrowtail q$ for all i, thus $q \sim p_i$ for all i, and $S_\infty[p_1 \ldots p_t | p_1 \ldots p_t]$ is a positive submatrix equal to $1/tJ_t$, where J_t denotes the all one matrix of dimension t. This implies that S_∞ is block diagonal (after permutation), with each block corresponding with an equivalence class in $\{1, \ldots, n\}/\sim$ which has no outgoing arc in the $\rightarrowtail$ arc relation. Each block contributes an eigenvalue 1 to the spectrum of S_∞. Since the spectrum of S_∞ equals the spectrum of $\Gamma_\infty M$, and there are assumed to be k equivalence classes with the stated properties, this proves the theorem. □

OBSERVATION. It was shown that the inflation operator has a decoupling effect on *dpsd* matrices by considering its most extreme parametrization. This result connects the uncoupling properties of the *MCL* process to the effect of the inflation operator on the spectrum of its operand, and it generalizes the mapping of nonnegative column allowable idempotent matrices onto overlapping clusterings towards a mapping of column allowable *dpsd* matrices onto directed acyclic graphs. This generalization is most elegantly described by considering a *dpsd* stochastic matrix M and the matrix $D = \Gamma_\infty M$. From the proof given above it follows that D is a matrix for which some power D^t is idempotent. The overlapping clustering associated with D by taking as clusters all endclasses and the nodes that reach them, is exactly the overlapping clustering resulting from applying Definition 8 on page 58 to D^t. In both cases the clustering is obtained by taking as clusterings the weakly connected components of the graph. On page 136 a short remark is made about associating clusterings with *MCL* iterands in practice.

7.3 Reductions of *dpsd* matrices

The following simple reductions of *dpsd* matrices have not yet been of immediate use in the analysis of the *MCL* process, but knowledge of their existence surely will not harm. The first part of Theorem 11 below is a decomposition of a *dpsd* matrix into mutually orthogonal rank 1 idempotents. This decomposition is in general possible for matrices which are (not necessarily diagonally) similar to a hermitian matrix, but is still of particular interest. It is extensively used in the analysis of rapidly mixing Markov chains, where the relationship between the diagonal matrix transforming a matrix to symmetric form and the stationary distribution is of crucial importance. The second part is a decomposition of a *dpsd* matrix into rank 1 matrices with a particular bordered 0/1 structure. For this decomposition, diagonal similarity is responsible for preserving the bordered structure.

THEOREM 11. *Let A be* dpsd *of dimension n, such that $A = {d_t}^{-1} S d_t$. Then A can be written in the forms*

i) $A = \sum_{i=1}^{n} \lambda_i(A) E_i$, *where the* E_i *are a set of mutually orthogonal rank* 1 *idempotents.*

ii) $A = {d_t}^{-1}(\sum_{i=1}^{n} x_i {x_i}^*) d_t$, *where the last* $i-1$ *entries of* x_i *are zero. If* A *is real, the vectors* x_i *can be chosen real.*

The *reduction* aspect of this statement is that all partial sums $\sum_{i=1}^{k} x_i {x_i}^*$ are positive semi-definite as well (by the property that all hermitian forms are nonnegative), so that all partial sums ${d_t}^{-1}(\sum_{i=1}^{k} x_i {x_i}^*) d_t$ are *dpsd.*

PROOF. Let u_i be a set of orthonormal eigenvectors of S. Then S can be written as the sum of weighted idempotents $U_i = \lambda_i(S) u_i {u_i}^*$. Statement i) now follows from setting $E_i = {d_t}^{-1} U_i d_t$. Statement ii) is adapted from a similar theorem by FitzGerald and Horn [**59**] for hermitian matrices. The proof of ii) follows from their argument for the hermitian case, given in the lemma below. □

LEMMA 16. [**59**] *Let* B *be positive definite of dimension* n. *If* $B_{nn} > 0$ *write* b_n *for the* n^{th} *column of* B, *and let* x *be the vector* b_n *scaled by a factor* ${B_{nn}}^{-1/2}$, *so that* $[xx^*]_{nn} = B_{nn}$. *If* $B_{nn} = 0$ *let* x *be the null vector of dimension* n. *In either case, the matrix* $B - xx^*$ *is positive semi-definite and all entries in the last row and column are zero.*

PROOF. The latter statement is obvious. For the first part, if $B_{nn} = 0$ (and thus $x = 0$) the proof is trivial, as $B - xx^*$ then equals C which is positive semi-definite because it is a principal submatrix of B. Otherwise consider the hermitian form u^*Bu and partition B and u conformally as

$$B = \begin{pmatrix} C & y \\ y^* & B_{nn} \end{pmatrix} \qquad u = \begin{pmatrix} v \\ z \end{pmatrix}$$

where u and v are complex vectors of dimension n and $n-1$ respectively. Expand the hermitian form u^*Bu as $v^*Bv + \overline{z}y^*v + v^*yz + \overline{z}B_{nn}z$. This expression is greater than or equal to zero for any choice of u. For arbitrary v fix z in terms of v as $-(y^*v)/B_{nn}$. In the further expansion of the hermitian form u^*Bu two terms cancel, and the remaining parts are $v^*Cv - (yv^*y^*v)/B_{nn}$ which can be rewritten as $v^*(C - (yy^*)/B_{nn})v$. Because this expression is greater than or equal to zero for arbitrary v, and because $C - (y^*y)/B_{nn}$ equals $B - xx^*$, the lemma follows.

A positive semi-definite matrix B has a decomposition $B = \sum_{i=1}^{n} x_i {x_i}^*$ by repeated application of Lemma 16, yielding the theorem of FitzGerald and Horn. Using diagonal similarity, this decomposition translates to the decomposition given in Theorem 11 for *dpsd* matrices. □

The following theorem provides a reduction of diagonally symmetric and *dpsd* stochastic matrices to smaller dimensional counterparts, by taking two states together. This theorem may be of use for the proof of existence or non-existence of stochastic *dpsd* matrices (e.g. with respect to the associated DAGs).

DEFINITION 15. *Let* M *be a diagonally symmetric stochastic matrix of dimension* n, *and let* π *be its stationary distribution, so that* ${d_\pi}^{-1/2} M {d_\pi}^{1/2}$ *is symmetric. Let* k *and* l *be two*

states corresponding with columns of M. The stochastic contraction M' of M with respect to k and l is the diagonally symmetric matrix in which the states k and l are contracted into a new state $\{kl\}$ as follows.

$$\begin{aligned} M'_{a\{kl\}} &= \frac{M_{ak}\pi_k + M_{al}\pi_l}{\pi_k + \pi_l} \\ M'_{\{kl\}a} &= M_{ka} + M_{la} \\ M'_{\{kl\}\{kl\}} &= \frac{(M_{lk} + M_{kk})\pi_k + (M_{ll} + M_{kl})\pi_l}{\pi_k + \pi_l} \end{aligned}$$

□

THEOREM 12. *Taking the stochastic contraction commutes with taking powers of M. If M is* dpsd *then so is its stochastic contraction M'.*

PROOF. Without loss of generality, assume that 1 and 2 are the two states being contracted. Identify the new state $\{1,2\}$ with the second column and second row. It is easily verified that the equilibrium distribution π' of M' equals $(0, \pi_1 + \pi_2, \pi_3, \ldots, \pi_n)^T$ and that $Md_{\pi'}$ is symmetric. Let S_M and $S_{M'}$ be the matrices to which M and M' are respectively diagonally similar. Then $[S_{M'}]_{2a} = \sqrt{\pi_a/(\pi_1 + \pi_2)}(M_{1a} + M_{2a})$, and all entries of $S_{M'}$ corresponding with column and row indices greater than two are identical to the entries of S_M. This establishes that $S_{M'}$ can be factorized as below (remembering that $[S_M]_{kl}$ equals $\sqrt{(\pi_l/\pi_k)}M_{kl}$).

$$S_{M'} = \begin{pmatrix} 0 & 0 & & & \\ \frac{\pi_1}{\sqrt{\pi_1+\pi_2}} & \frac{\pi_2}{\sqrt{\pi_1+\pi_2}} & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{pmatrix} S_M \begin{pmatrix} 0 & \frac{\pi_1}{\sqrt{\pi_1+\pi_2}} & & & \\ 0 & \frac{\pi_2}{\sqrt{\pi_1+\pi_2}} & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{pmatrix}$$

This factorization establishes both the commuting part of the theorem and the fact that contraction preserves *dpsd*-ness. The latter follows from considering a symmetric form $xS_{M'}x$; using the factorization it is reduced to a particular symmetric form yS_My □.

7.4 Hilbert's projective metric

For convenience, all vectors and matrices in this section are assumed to be positive. This is not strictly necessary, see e.g. [**37**]. Hilbert's projective metric d for two positive vectors x and y both of dimension n is defined as

$$d(x,y) = \ln\left[(\max_i \frac{x_i}{y_i})(\max_j \frac{y_j}{x_j})\right] = \max_{i,j} \ln\left(\frac{x_i y_j}{x_j y_i}\right)$$

It can be defined in the more general setting of a Banach space [**30**]. Hilbert's metric is a genuine metric distance on the unit sphere in $\mathbb{R}^n$, with respect to any vector norm

(see [**30**]). For a positive matrix A define the contraction ratio τ and the cross-ratio number ϕ by

$$\tau A = \sup_{x,y} \frac{d(Ax, Ay)}{d(x, y)} \qquad \phi A = \min_{i,j,k,l} \frac{A_{ik}A_{jl}}{A_{jk}A_{il}}$$

These are related to each other via

$$\tau A = \frac{1 - \sqrt{\phi A}}{1 + \sqrt{\phi A}} \tag{25}$$

For proofs see [**21, 76, 149**]. The quantity τ is used to measure the deviation of large products of nonnegative matrices from the set of rank 1 matrices (see e.g. [**30, 37, 76, 149**]). There is a straightforward connection between Γ_r and ϕ. For M nonnegative stochastic,

$$\phi(\Gamma_r A) = (\phi A)^r \tag{26}$$

It follows immediately from the definition of τ, that for A and B nonnegative,

$$\tau(AB) \leq \tau(A)\tau(B) \qquad \text{c.q.} \qquad \tau(A^k) \leq \tau(A)^k \tag{27}$$

OBSERVATION. Equations (26) and (27) supply the means for a simple[5] proof of Lemma 12. Suppose that M is a rank 1 column stochastic matrix (so that $\phi M = 1$), and that $M' = M + E$ is a perturbation of M such that $\phi M' = 1 - \epsilon$. Equation 25 yields that $\tau M'$ is of order $1 - 1/2\epsilon$. Then $\phi\Gamma_2 M'$ is of order $1 - 2\epsilon$ and $\tau\Gamma_2 M'$ is of order ϵ. So for small perturbations inflation has a linear effect on the contraction ratio, whereas quadratic expansion (i.e. Exp_2) squares the contraction ratio. It follows immediately that the *MCL* process applied to M' will result in a limit that has rank equal to one. □

Experiments with the *MCL* process suggest that if the process converges towards a doubly idempotent limit, then appropriately chosen submatrices of the iterands have the property that their contraction ratio approaches zero. For example, consider an *MCL* process converging towards the limit

$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

partition the iterands M_i as

$$M_i = \left(\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right)$$

with each matrix $A, \ldots, D$ of dimension 2×2. The observation is that all four quantities $\tau(A|B)$, $\tau(B^T|D^T)$, $\tau(C|D)$, and $\tau(A^T|C^T)$ tend to zero as i goes to infinity. This presumption is not of crucial importance for the matter of convergence, since it is already known that the *MCL* process converges quadratically (in one of the usual matrix norms) in the neighbourhood of doubly idempotent matrices. However, the connection between Γ_r and ϕ may just lead to new insights in the *MCL* process. What is needed is results on the square of the matrix M_i above (not assuming anything on the dimension

[5] For simplicity it is still assumed that all vectors under consideration are positive.

of M_i), in term of the inherited partition. Thus, bounds are sought for $\tau(A^2 + BC|AB + BD)$, $\tau((AB + BD)^T|(CB + D^2)^T)$, $\tau(CA + DC|CB + D^2)$, and $\tau((A^2 + BC)^T|(CA + DC)^T)$. For this it may be interesting to investigate a notion like 'mutual contraction ratio', e.g. the quantity τ' defined as

$$\tau'(A, B) = \sup_{x,y} \frac{d(Ax, By)}{d(x, y)}$$

It is difficult to assess the potential of this line of research, but it is interesting to see that inflation and expansion can be described in the same framework. As was the case for majorization though (Chapter 6), the two operators seek different corners of the framework, as exhibited by Equations (25), (26), and (27). For the simple case of rank 1 stochastic matrices this leads to a conceptually more appealing proof of Lemma 12.

7.5 Discussion and conjectures

Theorem 9 and 10 shed light on the structure and the spectral properties of the iterands of the *MCL* process. Theorem 9 also gives the means to associate an overlapping clustering with each *dpsd* iterand of an *MCL* process, simply by defining the end nodes of the associated *DAG* as the unique cores of the clustering, and adding to each core all nodes which reach it.

There is a further contrasting analogy with the usual Markov process. Consider a Markov process with *dpsd* input matrix M. Then the difference $M^k - M^l$, $k < l$, is again *dpsd* (they have the same symmetrizing diagonal matrix, and the spectrum of $M^k - M^l$ is nonnegative). From this it follows that all rows of diagonal entries $M^{(k)}{}_{ii}$, for fixed diagonal position ii, are non-increasing. Given a stochastic *dpsd* matrix M, the Γ_r operator, $r > 1$, (in the setting of *dpsd* matrices) always increases some diagonal entries (at least one). The sum of the increased diagonal entries, of which there are at least k if k is the number of endnodes of the *DAG* associated with both M and $\Gamma_r M$, is a lower bound for the combined mass of the k largest eigenvalues of $\Gamma_r M$ (Lemma 13 a)).

The *MCL* process converges quadratically in the neighbourhood of the doubly idempotent matrices. Proving (near–) global convergence seems to be a difficult task. I do believe however that a strong result will hold.

CONJECTURE 1. *All MCL processes* $(M, e_{(i)}, r_{(i)})$*, with* $e_i = 2, r_i = 2$ *eventually, converge towards a doubly idempotent limit, provided* M *is irreducible,* dpsd*, and cannot be decomposed as a Kronecker product of matrices in which one of the terms is a flip–flop equilibrium state.*

It is a worthy long standing goal to prove or disprove this conjecture. Subordinate objectives are:

i) For a fixed *MCL* process $(\cdot, e_{(i)}, r_{(i)})$, what can be said about the basins of attraction of the *MCL* process. Are they connected?

ii) What can be said about the union of all basins of attraction for all limits which correspond with the same overlapping clustering (i.e. differing only in the distribution of attractors)?
iii) Can the set of limits reachable from a fixed nonnegative matrix M for all *MCL* processes $(M, e_{(i)}, r_{(i)})$ be characterized? Can it be related to a structural property of M?
iv) Given a node set $I = \{1, \ldots, n\}$, and two directed acyclic graphs D_1 and D_2 defined on I, under what conditions on D_1 and D_2 does there exist a *dpsd* matrix A such that the *DAG*s associated with A according to Theorem 9, via respectively rows and columns, equals D_1 and D_2? What if A is also required to be column stochastic?
v) Under what conditions do the clusters in the cluster interpretation of the limit of a convergent *MCL* process $(M, e_{(i)}, r_{(i)})$ correspond with connected subgraphs in the associated graph of M?
vi) For A *dpsd*, in which ways can the *DAG* associated with A^2 be related to the *DAG* associated with M?
vii) Is it possible to specify a subclass S of the stochastic *dpsd* matrices and a subset R' of the reals larger than $\mathbb{N}$, such that $\Gamma_r M$ is in S if $r \in R'$ and $M \in S$?

REMARK. There is no obvious non-trivial hypothesis regarding item vi), unless such a hypothesis takes quantitative properties of M into account. This is due to the fact that the equilibrium state corresponding with a connected component of M corresponds with a *DAG* which has precisely one endclass. The breaking up of connected components which can be witnessed in the *MCL* process is thus always reversible in a sense. With respect to v), I conjecture the following.

CONJECTURE 2. *The clustering associated with a limit of an MCL process with* dpsd *input matrix M, corresponds with subsets of the node set of the associated graph G of M which induce subgraphs in G that are connected.*

There are several lines of research which may inspire answers to the questions posed here. However, for none of them the connection seems so strong that existing theorems can immediately be applied. The main challenge is to further develop the framework in which the interplay of Γ_r and Exp_s can be studied. Hadamard–Schur theory was discussed in Section 7.1. Perron–Frobenius theory, graph partitioning by eigenvectors (e.g. [**138, 139**]), and work regarding the second largest eigenvalue of a graph (e.g. [**6, 41**]), form a natural source of inspiration. These are discussed in the next chapter. The theory of Perron complementation and stochastic complementation as introduced by Meyer may offer conceptual support in its focus on uncoupling Markov chains [**125, 126**]. There are also papers which address the topic of matrix structure when the subdominant eigenvalue is close to the dominant eigenvalue [**77, 135**]. The literature on the subject of diagonal similarity does not seem to be of immediate further use, as it is often focussed on scaling problems (e.g. [**51, 83**]). For the study of flip–flop equilibrium states the many results on circulant matrices are likely to be valuable, for example the monograph [**42**], and the work on group majorization in the setting of circulant matrices in [**66**]. It may also be fruitful to investigate the relationship with *Hilbert's distance* and the *contraction*

ratio for positive matrices, introduced in Section 7.4. Regarding flip–flop states, several interesting questions are open:

i) For the *MCL* process with both parameter rows constant equal to 2, are there orbits of length greater than 2 in the class of *dpsd* matrices?
ii) Must an indecomposable *dpsd* (in terms of the Kronecker product) flip–flop state necessarily be a symmetric circulant? It seems obvious that this must be the case.
iii) For flip–flop states which are symmetric circulants, how close is is Exp_2 to $\Gamma_{1/2}$? Note that both operators have a contracting effect on positive matrices.
iv) For each dimension n, does there exist a flip–flop state which is the circulant of a vector

$$(p_1, p_2, p_3, \ldots, p_{k-1}, p_k, p_{k-1}, \ldots, p_2), \qquad n = 2k-2$$
$$(p_1, p_2, p_3, \ldots, p_{k-1}, p_k, p_k, p_{k-1}, \ldots, p_2), \qquad n = 2k-1,$$

where all p_i are different, $i = 1, \ldots, k$?
v) For which $r > 1$ and $s > 1$ do there exist nonnegative *dpsd* matrices A such that $\Gamma_r(A^s) = A$, where A^s is defined according to Lemma 13e)?

CONJECTURE 3. *For every* dpsd *flip–flop equilibrium state which is indecomposable in terms of the Kronecker product, there is no trajectory leading to this state other than the state itself.*

8

The relationship between $\lambda_2(G)$ and cluster structure

There are many results regarding the relationship between the spectrum of a graph and its connectivity properties. In particular, a connected graph having large subdominant eigenvalue — relative to the dominant eigenvalue — can be separated into two sets of vertices such that the two induced subgraphs have a high degree of connectivity. These statements can be made precise via two (well known) approaches, namely via the spectrum and eigenvectors of a graph $G = (V, E, w)$ and its adjacency matrix A, or via the spectrum and eigenvectors of the associated Laplacian matrix[1]. Section 8.1 gives the basic results regarding the Laplacian; the results for the adjacency matrix itself are given in Section 8.2. The last section contains a short account of the role played by subdominant eigenvalues in respectively the theory of stochastic uncoupling and the theory of rapidly mixing Markov chains.

8.1 Partitioning via the Laplacian

DEFINITION 16. *Let $G = (V, E, w)$ be a weighted undirected graph without loops on n nodes. Let A be the adjacency matrix of G. Let D be the diagonal matrix of dimension n where D_{ii} equals the sum of weights of all arcs incident to i. The Laplacian L of G is defined as*

$$L = D - A \tag{28}$$

□

It is easy to prove that all eigenvalues of the Laplacian L of G are nonnegative, and that the multiplicity of the eigenvalue zero equals the number of connected components of G. For each connected component of G, the characteristic vector of that component is a corresponding eigenvector of G with eigenvalue zero. Henceforth, I will assume that the graph G is connected, that $\lambda_n(G) = 0$ and that $\lambda_{n-1}(G) > 0$, *and that the dimension of G is even.* The latter simplifies the exposition and does not really limit its significance. The first part of this section follows the accounts given in [**53**] and [**137**]. Consider the problem of finding an optimal cut of G, that is, a balanced[2] bipartition (S, S^c) such that the weight $\delta(S, S^c)$ of the cut vertices is minimized. Let x be the characteristic difference

[1] For surveys on the Laplacian matrix of a graph, see [**72, 71, 124**].

[2] For the definition see page 35.

vector associated with the bipartition (S, S^c). The quantity $\delta(S, S^c)$ can be rewritten as follows.

$$\delta(S, S^c) = \frac{1}{4} \sum_{\substack{(i,j)\in E \\ (i,j)\in S\times S^c}} (x_i - x_j)^2 A_{ij} \tag{29}$$

This formulation of the cut problem turns out to be equivalent to the minimization of a quadratic form over all balanced characteristic difference vectors. Let x be the characteristic difference vector associated with a balanced bipartition (S, S^c). Noting that the sum of the entries in D equals twice the sum of all edge weights, one finds that

$$\begin{aligned} \sum_{(i,j)\in E} A_{ij}(x_i - x_j)^2 &= \sum_{(i,j)\in E} {x_i}^2 A_{ij} + {x_j}^2 A_{ij} - 2A_{ij}x_i x_j \\ &= \sum_{i\in V} D_{ii}{x_i}^2 - 2\sum_{(i,j)\in E} A_{ij}x_i x_j \end{aligned}$$

which establishes

$$\begin{aligned} x^T L x &= x^T D x - x^T A x \\ &= \sum_{i\in V} D_{ii}{x_i}^2 - 2\sum_{(i,j)\in E} x_i x_j A_{ij} \\ &= 4\,\delta(S, S^c) \end{aligned} \tag{30}$$

Thus, a minimal solution to the cut problem is equivalent to the minimization of the quadratic form $x^T L x$ under the condition that x is a characteristic difference vector. Of course this problem is no more tractable, but a straightforward relaxation of the new formulation is. That is, if the requirement that x is a characteristic difference vector is replaced by only demanding that $\sum x_i = 0$ (and naturally requiring that $\| x \|_2$ is fixed, say, equal to one), then the problem is tractable and a solution is immediately known. Writing u_i for the eigenvector corresponding with eigenvalue $\lambda_i(L)$, one obtains

$$\begin{aligned} \min_{\substack{\sum x_i = 0 \\ x_i = \pm 1}} x^T L x &= \min_{\substack{\sum x_i = 0 \\ x_i = \pm 1/\sqrt{n}}} n\ x^T L x \\ &\geq \min_{\substack{\sum x_i = 0 \\ \|x\|_2 = 1}} n\ x^T L x \\ &= n\ {u_{n-1}}^T L u_{n-1} \\ &= n\ \lambda_{n-1}(L) \end{aligned} \tag{31}$$

Why is the derivation just given valid? The second equality is the special case $k = 1$ of the famous Raleigh–Ritz variational characterization of the eigenvalues and eigenvectors

of a hermitian matrix H given below, where the vectors u_i are again the eigenvectors corresponding with the eigenvalue $\lambda_i(H)$.

$$\lambda_{n-k}(H) = \min_{\substack{\|x\|_2=1 \\ x \perp u_{n-k+1},\ldots,u_n}} x^* H x$$

The minimum is attained for the eigenvector u_{n-k}. There are many more variational characteristics of eigenvalues such as this; the book [**86**] contains an extensive collection of results. In the specific case of Derivation (31), the eigenvector u_n is the all–one vector corresponding with eigenvalue zero. The vector u_{n-1} of a Laplacian is often called the *Fiedler vector*, honouring the pioneering work of Miroslav Fiedler in this area. The following important result is due to Fiedler.

THEOREM 13. *(Fiedler,* [**56**]*) Let $G = (V, E, w)$ be an undirected weighted graph, let u be the Fiedler eigenvector of the Laplacian of G. For any real number $r > 0$, the subgraphs induced by the sets $\{v_i \in V \mid u_i \geq -r\}$ and $\{v_i \in V \mid u_i \leq r\}$ are connected.*

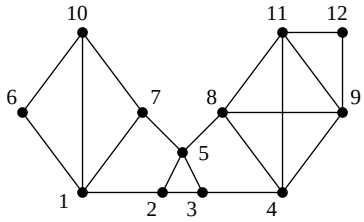
So the Fiedler eigenvector u yields at least one bipartition for which the induced subgraphs are connected, though it need not be balanced. A different method is to compute the median m of the values u_i, to segment V into two parts according to the sign of $u_i - m$, and to distribute the nodes v_i with $u_i = m$ such that the resulting partition is balanced. There is no hard guarantee that this partition induces connected subgraphs, but it was shown by Chan et al [**32**] that the characteristic difference vector x thus obtained is best in any l-norm, that is, the minimum over all balanced characteristic difference vectors y of the quantity $\| y - \sqrt{n} u_{n-1} \|_l$ is achieved for x. These results are the starting point for a proliferation of spectral approaches to partitioning, resulting from imposing additional constraints, (e.g. introducing weights on the vertices), additional objectives (e.g. k-way partitioning), and additional solution techniques (e.g. computing the d eigenvectors corresponding with the d smallest eigenvalues of L — the geometric properties of the eigenvectors can then be used in different ways to obtain partitions of V). See the references on page 28.

EXAMPLE. The Fiedler vector of the simple graph G_3 on page 45, repeated below, is approximately equal to

$$(0.325, 0.121, -0.0354, -0.250, 0.0344, 0.420, 0.276, -0.235, -0.325, 0.381, -0.325, -0.387)^T$$

This vector induces a linear ordering on the nodes of G_3: $12 < \{11, 9\} < 4 < 8 < 3 < 5 < 2 < 7 < 1 < 10 < 6$. In this case, there are precisely six negative and six positive values, yielding the bipartition ($\{1, 2, 5, 6, 7, 10\}$, $\{3, 4, 8, 9, 11, 12\}$) which cuts 3 edges. This partitioning induces two connected subgraphs, where the connection of the node 3 is weakest. By moving the separating value successively to 0.04, 0.0, -0.04, and -0.15, the following partitions are obtained.

$\{1,6,7,10\}$ $\{2,3,4,5,8,9,11,12\}$
$\{1,2,6,7,10\}$ $\{3,4,5,8,9,11,12\}$
$\{1,2,5,6,7,10\}$ $\{3,4,8,9,11,12\}$
$\{1,2,3,5,6,7,10\}$ $\{4,8,9,11,12\}$



These four partitions correspond with four different ways of distributing the elements of one of the three clusters obtained by the *MCL* algorithm over the remaining two clusters.

8.2 Partitioning via the adjacency matrix

The spectrum of the adjacency matrix of a graph is also related to its structural properties. However, the exposition is not as clean as for the Laplacian. The defining inequality in Theorem 14 below is similar to Theorem 13, except for the appearance of the entries of u.

THEOREM 14. *(Fiedler* [**56**]*). Let $G = (V, E, w)$ be a connected undirected weighted graph with adjacency matrix A of dimension n. Let u and v be the eigenvectors corresponding with eigenvalues $\lambda_1(A)$ and $\lambda_2(A)$ respectively.*

For any $r \geq 0$, the submatrix $A[\alpha_r]$ is irreducible, where α_r is the index list formed from the set

$$\{i \in \{1,\ldots,n\} \mid v_i + r u_i \geq 0\} \tag{32}$$

□

Note that the signs of the eigenvector u can be reversed, so the statement in the theorem has a symmetric counterpart. The statement is easily generalized towards diagonally symmetric matrices, which is again illustrative for the fact that diagonally symmetric (hermitian) matrices are the 'next nice thing' to have after symmetric (hermitian) matrices.

THEOREM 15. *Let A be a diagonally symmetric irreducible matrix, let u and v be the eigenvectors corresponding with the eigenvalues $\lambda_1(A)$ and $\lambda_2(A)$ respectively.*

For any $r \geq 0$, the submatrix $A[\alpha_r]$ is irreducible, where α_r is the index list formed from the set

$$\{i \in \{1,\ldots,n\} \mid v_i + r u_i \geq 0\} \tag{33}$$

PROOF. Let d_t and S be respectively a diagonal matrix and a symmetric matrix such that $S = {d_t}^{-1} A d_t$. Any eigenvector w of A corresponds with an eigenvector ${d_t}^{-1} w$ of S. Applying Theorem 14 to S yields that the set

$$\{i \in \{1,\ldots,n\} \mid [{d_t}^{-1} v]_i + r[{d_t}^{-1} u]_i \geq 0\} \tag{34}$$

is connected in the underlying graph of S. Noting that the underlying (simple) graph of S is identical to the underlying (simple) graph of A, and rewriting the set above as

$$\{i \in \{1,\ldots,n\} \mid [d_t{}^{-1}(v + ru)]_i \geq 0\} \tag{35}$$

proves the theorem, since scaling by a positive factor does not change the sign of a number. □

The following is a generalization of a lemma by Powers [**139**].

LEMMA 17. *Let A be a nonnegative irreducible matrix with Perron eigenvector u. Suppose A has a real positive eigenvalue $\lambda_k(A)$, $k > 1$, corresponding with the eigenvector v. Without loss of generality, assume that v and A are conformally permuted according to the sign structure of v such that*

$$v = \begin{pmatrix} x \\ -y \\ \mathbf{0} \end{pmatrix} \qquad A = \begin{pmatrix} A_{PP} & A_{PN} & A_{P0} \\ A_{NP} & A_{NN} & A_{N0} \\ A_{0P} & A_{0N} & A_{00} \end{pmatrix}$$

Then $\lambda_1(A_{PP}) > \lambda_k(A)$ and $\lambda_1(A_{NN}) > \lambda_k(A)$.

PROOF. The proof of this lemma is very simple. The vector v must have both positive and negative entries, because the Perron vector of a nonnegative irreducible matrix is the only real eigenvector with nonnegative entries only (See Theorem 16), so x and y are both non-void. This theorem also establishes that $\lambda_1(A_{PP})$ and $\lambda_1(A_{NN})$ are both positive real. The identity $A_{PP}x - A_{PN}y = \lambda_k(A)x$ implies that $A_{PP}x > \lambda_k(A)x$, and similarly $A_{NN}y > \lambda_k(A)y$. The result now follows from Theorem 16 given below. Powers proved this lemma for symmetric matrices; the only extra argument needed is that v still must have both positive and negative entries if the requirement of symmetry is dropped. □

EXAMPLE. Consider the associated Markov matrix M of G_3 defined on page 50. Its Perron eigenvector is the equilibrium distribution π which is simply the scaling vector $(5,\ 4,\ 4,\ 5,\ 5,\ 3,\ 4,\ 5,\ 5,\ 4,\ 5,\ 3)^T$. Its second largest eigenvalue is approximately 0.923 with corresponding eigenvector

$$(2.40, .794, -0.0872, -1.66, 0.379, 1.74, 1.64, -1.56, -2.17, 2.18, -2.17, -1.48)^T$$

Setting r=0 in Theorem 15 gives the two sets $\{3,\ 4,\ 8,\ 9,\ 11,\ 12\}$ and $\{1,\ 2,\ 5,\ 6,\ 7,\ 10\}$. Varying r in the range $[-2, 0.5]$ gives exactly the same series of bipartitions as found with the Laplacian (which is accidental). Focusing on the bipartition found for r=0, and using Lemma 17 with k=2, it is established that the submatrices $M[3, 4, 8, 9, 11, 12]$ and $M[1, 2, 5, 6, 7, 10]$ both have the Perron root lower bounded by 0.923. The lower bound implies that for each submatrix, the average mass of their columns is at least 0.923, which indicates that the submatrices are sparsely connected. Numerical verification yields indeed that the respective Perron roots are approximately 0.939 and 0.938. The clustering resulting from the *MCL* process on page 53 on this graph induces submatrices $M[1, 6, 7, 10]$, $M[2, 3, 5]$, and $M[4, 8, 9, 11, 12]$ that have respective Perron roots 0.893, 0.700, and 0.928.

REMARK. Any nonnegative symmetric matrix is diagonally similar to a stochastic matrix. If S is symmetric, and u is the eigenvector corresponding to the largest eigenvalue λ_1, then it is easy to verify that ${\lambda_1}^{-1} d_u S {d_u}^{-1}$ is stochastic and has eigenvector $u \circ u$ for eigenvalue one.

THEOREM 16. [**19**], *page 26 ff. Let A be a square nonnegative matrix.*

i) *The spectral radius $\rho(A)$ is an eigenvalue of A and there is a nonzero nonnegative vector x such that $Ax = \rho(A)x$.*
ii) *If y is a nonnegative vector such that $Ay \geq \alpha y$, $\alpha > 0$, then $\rho(A) \geq \alpha$.*
iii) *If in addition A is irreducible, then $\rho(A)$ is a simple eigenvalue and x is positive, and any nonnegative eigenvector of A is a multiple of x.*
iv) *If A is irreducible and y is nonnegative such that $Ay > \alpha y$, then $\rho(A) > \alpha$.* □

8.3 Stochastic matrices, random walks, and λ_2

There are several results relating the presence or absence of a gap between the dominant eigenvalue and subsequent eigenvalues of a stochastic matrix. Meyer [**125**] developed the elegant framework of stochastic complementation, in which he described the behaviour of nearly completely reducible systems. These systems have the property that the associated matrix M has a block structure such that the mass of the off-diagonal blocks is very small, though the matrix as a whole is irreducible. Assume that the largest eigenvalue of each of the diagonal blocks is well separated from the second eigenvalue. It was shown in [**125**] that in the short run the powers of M behave as though it were completely reducible, with each diagonal block approaching a local equilibrium. After the short–run stabilization, the powers of M converge towards a global equilibrium state. During this long–run convergence the local equilibria are roughly maintained but their relative mass changes to accommodate for the global equilibrium. The theory of stochastic uncoupling is a special case of the theory of Perron uncoupling (uncoupling in nonnegative matrices), which was also researched by Meyer [**126**].

The second largest eigenvalue also plays an important role in the theory of *rapidly mixing Markov chains*. In this setting random walks are used for probabilistic measurement of large sets which are otherwise difficult to quantify. Random walks are simulated on large graphs in which the vertices are the combinatorial structures which are of interest. The graphs are typically exponentially large; the aim is either one of a) approximately counting the number of structures or b) their uniform generation. Without going into this matter too deeply, the idea is to simulate a Markov chain via the neighbour relation attached to the graph. The state space is generally exponentially large. The crucial idea is now that a polynomial number of steps should be sufficient to *get lost* from a given starting position. The necessary condition for this to happen is that the subdominant eigenvalue of the Markov chain is sufficiently separated from the dominant eigenvalue 1, where the notion of 'sufficiency' is governed by the relationship between transition probabilities and the stationary distribution. For the analysis of mixing properties of Markov chains, extensive use is made of the first decomposition given in Theorem 11. Under the mentioned condition the Markov chain approach leads to randomized polynomial time approximation schemes.

Among others, Markov chains can be constructed for the uniform generation of maximal matchings of a simple graph, the counting of the number of linear extension of a partial order, the number of forests in dense graphs, and graphs with good expanding properties. Results in a surprisingly different direction are that both the volume of a convex body and the permanent of a matrix (the computation of both is $\sharp P$–hard) can be approximated (in a satisfactory sense) via Markov chain techniques. A good monograph on the subject of random generation, counting, and rapidly mixing Markov chains is [**154**], and the survey article [**114**] on random walks on graphs gives a short account of ideas and results.

Part III

Markov Cluster Experiments

9

Comparing different graph clusterings

It is in general a non-trivial task to compare or rank different clusterings of the same graph. The reasons for this are much the same as for the fruitlessness of postulating the existence of a 'best' clustering. A comparison procedure for clusterings almost inevitably requires a formula that takes as input a graph and a clustering, and outputs a real number in some range, say between zero and one. Different clusterings of a single graph are then compared by plugging them into the formula and ordering them according to the respective scores. Clusterings with highly differing degrees of granularity are likely to have differing scores, and this may tell little about which levels of granularity are reasonable and which are not.

However, the above primarily implies that automated comparison of clusterings for which little context is known should be handled with care. Useful performance criteria do exist, and if applied in conjunction with information about cluster granularity they can be put to good use. In this chapter two different performance criteria are formulated, one for $0/1$ (simple) graphs, and one for weighted graphs. Both criteria yield a number within the range $[0, 1]$. They are used in Chapter 12 to test the performance of the *MCL* algorithm on randomly generated test graphs.

The criterion for simple graphs is derived by demanding that it yields the perfect score 1 if and only if the graph is a direct sum of complete graphs and the clustering is the corresponding canonical clustering. This criterion is first formulated globally in terms of the incidence matrix associated with the simple graph, and then in terms of an average of scores for all nodes in the graph. The latter formulation is used to derive a formula that measures how efficient a characteristic vector (corresponding with a cluster) captures the mass of a nonnegative vector (corresponding with an element or node contained within the cluster). The average score over all nodes in the graph yields the weighted performance criterion for clusterings of nonnegative graphs. The formula contains a scaling factor which is a Schur convex function on the set of nonnegative nonzero vectors. Several properties of this function are derived.

A formula for a metric distance between two partitions (non-overlapping clusterings) is given in Section 9.3. This distance is computed as the sum of two numbers, where each number represents the distance of one of the two partitions to the intersection of the two partitions. If one of the two numbers is very small compared to the other, this implies that the corresponding partition is nearly a subpartition of the other partition.

9.1 Performance criteria for simple graphs

Consider a simple graph and a clustering of this graph. If the rows and columns of the incidence matrix of the graph are permuted such that nodes in the same cluster correspond with consecutive columns (and rows), then the clustering is pictorially represented by a diagonal of blocks in the permuted matrix. (See Figure 4 on page 11 and Figure 27 on page 131).

It is desirable that the clustering covers as many edges as possible, and does so efficiently. This means that the blocks should cover as many edges or 'ones' as possible, while there should be few zeroes within the blocks and few edges outside. It is easy to cover all of the edges by taking the top clustering (implying a single block consisting of the whole matrix), but then all zeroes are covered as well. This suggests that clusterings should be punished for each edge that is not covered by the clustering, and for each edge that is suggested by the clustering but not present in the graph, i.e. a within–block zero. Definition 17 gives a simple formula expressing this idea.

DEFINITION 17. *Let G be a simple graph on n nodes with or without loops and let $\mathcal{P}$ be a partition of G. The* naive performance criterion *for G and $\mathcal{P}$ is defined to be*

$$\text{Perf}(G,\mathcal{P}) = 1 - \frac{\#^1_{\text{out}}(G,\mathcal{P}) + \#^0_{\text{in}}(G,\mathcal{P})}{n(n-1)} \qquad \textit{(naive)} \tag{36}$$

where $\#^1_{\text{out}}(G,\mathcal{P})$ denotes the number of edges not covered by the clustering (i.e. an edge (i,j) in G for which i and j are in different clusters of $\mathcal{P}$), and where $\#^0_{\text{in}}(G,\mathcal{P})$ denotes the number of edges suggested by $\mathcal{P}$ absent in G (i.e. all pairs $i \neq j$ for which i and j are in the same cluster of $\mathcal{P}$ and (i,j) is not an edge in G). □

Note that by disregarding loops in the definition (i.e. only considering pairs $i \neq j$ for $\#^0_{\text{in}}(G,\mathcal{P})$) it is achieved that clusterings are not punished for putting a node i in the same cluster as itself, if a loop from i to i is absent. This ensures that the best clustering (scoring the maximum performance 1) for the empty graph on n nodes (with no links between different nodes) is the bottom clustering consisting of n singletons. Note that this graph can be viewed as a direct sum of n times the complete graph K_1. In general, direct sums of complete graphs are the only graphs for which a clustering yielding performance 1 exists, and this is exactly what intuition demands. The scaling factor $n(n-1)$ guarantees that the performance criterion lies between 0 and 1 for all simple graphs. The performances of the top and bottom clusterings are related to each other by the fact that they add up to 1, as stated in the following basic property.

PROPERTY. Let G be a simple graph on n nodes with or without loops. Denote the top and bottom clusterings respectively by Top and Bottom. The following identity holds:

$$\text{Perf}(G,\text{Top}) + \text{Perf}(G,\text{Bottom}) = 1 \tag{37}$$

The proof is nearly trivial and is omitted here. The criterion in Definition 36 is useful only for clusterings of graphs which are not too sparse. For large sparse graphs the numerator in (36) usually pales in comparison to the denominator. This implies that for such graphs the naive performance criterion is close to one and almost constant for

different clusterings as long as they are not too absurd. A slight modification of the definition remedies this drawback. To this end, the quality of a clustering is measured per node rather than in one stroke. First, consider what it takes to formulate definition 17 in terms of coverage of the respective nodes.

DEFINITION 18. *Let G be a simple graph on n nodes with or without loops, let M be its incidence matrix, let $\mathcal{P}$ be a partition of G, and let v be a node in G. Denote the cluster in $\mathcal{P}$ containing v by P_v. The* naive coverage measure *of P_v for v is defined to be*

$$\text{Cov}(G, P_v, v) = 1 - \frac{\#^1_{\text{out}}(G, P_v, v) + \#^0_{\text{in}}(G, P_v, v)}{n-1} \qquad \textit{(naive)} \tag{38}$$

where $\#^1_{\text{out}}(G, P_v, v)$ denotes the number of edges going out from v not covered by P_v (i.e. a nonzero entry M_{iv} for which $i \notin P_v$) and where $\#^0_{\text{in}}(G, P_v, v)$ denotes the number of edges suggested by P_v for v absent in G (i.e. all labels $i \neq v$ for which $i \in P_v$ and M_{iv} is zero). □

Averaging the coverage of a node according to Definition 18 over all nodes in a graph, yields the naive performance criterion from Definition 17. The coverage according to Definition 18 has the advantage that it can be easily modified such that its discriminating power applies to clusterings of sparse graphs as well. This is done by making the denominator smaller, whilest maintaining the property that the whole expression lies between 0 and 1.

DEFINITION 19. *Let G be a simple graph on n nodes with or without loops, let M be its incidence matrix, let $\mathcal{P}$ be a partition of G, and let v be a node in G. Denote the set of neighbours of v (excluding v itself) in G by S_v, and denote the cluster in $\mathcal{P}$ containing v by P_v. The* simple coverage measure *of P_v for v is defined to be*

$$\text{Cov}(G, P_v, v) = 1 - \frac{\#^1_{\text{out}}(G, P_v, v) + \#^0_{\text{in}}(G, P_v, v)}{|S_v \cup P_v|} \qquad \textit{(scaled)} \tag{39}$$

where $\#^1_{\text{out}}(G, P_v, v)$ and and $\#^0_{\text{in}}(G, P_v, v)$ are as in the previous definition. □

It is easy to see that the quantity defined by 39 lies between 0 and 1, and is equal to 1 only if the sets S_v and P_v are the same. It should be noted that this definition of coverage no longer yields the property that the respective performances of top and bottom clustering add to 1, which is a small sacrifice for its increased expressive power.

9.2 Performance criteria for weighted graphs

Definition 18 provides the inspiration for a measure of coverage telling how well a certain characteristic vector (representing a cluster) captures the mass of a nonnegative vector. The measure (given in Definition 21) is independent of scaling. An important distinction from the preceding definitions is that the loop (c.q. return probability) is no longer treated as a special case. The measure uses the notion of *mass centre* defined below.

DEFINITION 20. *Let π be a probability vector of dimension n, let r be a real number greater than zero. The* ***mass centre*** *of order r of π is defined as*

$$\operatorname{ctr}_r \pi = \left(\sum_{i=1}^{n} {\pi_i}^r \right)^{\frac{1}{r-1}} \tag{40}$$

If the subscript is omitted, it is understood that the mass centre is of order 2. □

Given a probability vector π, the number $\operatorname{ctr}_r(\pi)$ separates the transition probabilities which grow under Γ_r from those which shrink under Γ_r. If $r < 1$ then those entries smaller (larger) than $\operatorname{ctr}_r(\pi)$ grow (shrink), and vice versa if $r > 1$ then those entries smaller (larger) than $\operatorname{ctr}_r(\pi)$ shrink (grow). A straightforward calculation confirms these statements.

LEMMA 18. *The mass centre of order r is an increasing function in r for all stochastic vectors π. It is monotone increasing for all non-homogeneous vectors π and constant for all homogeneous vectors π.*

PROOF. The derivative of $\operatorname{ctr}_r(\pi)$ equals $\operatorname{ctr}_r(\pi) f_\pi(r) g_\pi(r)$ where

$$f_\pi(r) = -\left(\sum_i {\pi_i}^r\right) \ln\left(\sum_i {\pi_i}^r\right) + (r-1) \sum_i {\pi_i}^r \ln \pi_i$$
$$g_\pi(r) = 1 / \left(\sum_i {\pi_i}^r\right) (r-1)^2$$

The sign of $d/dr \operatorname{ctr}_r(\pi)$ depends on $f_\pi(r)$ only. Jensen's inequality states that

$$\sum_i w_i x_i \ln x_i - \left(\sum_i w_i x_i\right) \ln\left(\sum_i w_i x_i\right) \geq 0$$

where $\sum_i w_i = 1$ and the x_i are all positive. This inequality is strict unless all the x_i are equal. A proof is given in [**18**], page 17. Alternatively, this inequality follows easily from letting ϵ approach zero in the inequality

$$\sum_i w_i {x_i}^{1+\epsilon} - \left(\sum_i w_i x_i\right)^{1+\epsilon} \geq 0$$

The latter inequality follows from the convexity of the mapping $x \to x^y$, $y \geq 1$. Assume without loss of generality that all the π_i are positive (simply by pruning all zero entries of π and rescaling its dimension). Substituting $w_i = \pi_i$ and $x_i = {\pi_i}^{r-1}$ yields

$$\sum_i \pi_i {\pi_i}^{r-1} \ln {\pi_i}^{r-1} - \left(\sum_i \pi_i {\pi_i}^{r-1}\right) \ln\left(\sum_i \pi_i {\pi_i}^{r-1}\right) \geq 0$$

which is equivalent to saying that $f_\pi(r) \geq 0$, with equality iff π is homogeneous. This proves the lemma. □

The behaviour of $\operatorname{ctr}_r(\pi)$ for the limit cases is easily described.

LEMMA 19. *Let π be a probability vector of dimension n, and suppose that it has k positive entries and $n-k$ entries equal to zero. By convention, define 0^0 to be equal to one.*

$$\lim_{r \downarrow 0} \mathrm{ctr}_r(\pi) = 1/k \tag{41}$$

$$\lim_{r \to 1} \mathrm{ctr}_r(\pi) = \prod_i \pi_i{}^{\pi_i} \tag{42}$$

$$\lim_{r \to \infty} \mathrm{ctr}_r(\pi) = \max_i \pi_i \tag{43}$$

PROOF. The first and last identities are elementary. The second follows from the derivation given below.

$$\begin{aligned}
\Big[\sum_i \pi_i{}^{1+\epsilon}\Big]^{1/\epsilon} &= \Big[\sum_i \pi_i e^{\epsilon \ln \pi_i}\Big]^{1/\epsilon} \\
&= \Big[\sum_i \pi_i (1 + \epsilon \ln \pi_i + \mathcal{O}(\epsilon^2))\Big]^{1/\epsilon} \\
&= (1 + \epsilon \sum_i \pi_i \ln \pi_i + \mathcal{O}(\epsilon^2))^{1/\epsilon} \\
&\to e^{\sum \pi_i \ln \pi_i} \qquad (\epsilon \to 0) \\
&= \prod_i \pi_i{}^{\pi_i}
\end{aligned}$$

□

LEMMA 20. *Extend the definition of ctr_r to $\mathbb{R}_{\geq 0}{}^n \setminus \{0^n\}$. Let u and v be nonnegative vectors of the same dimension n having the same weight, that is, $\sum u_i = \sum v_i$. Let r be greater than zero and not equal to one. If u is majorized by v then $\mathrm{ctr}_r(u) \leq \mathrm{ctr}_r(v)$. This inequality is strict if u is not a permutation of v.*

PROOF. When the implication $u \prec v \implies \mathrm{ctr}_r(u) \leq \mathrm{ctr}_r(v)$ holds for all $u, v \in \mathbb{R}_{\geq 0}{}^n$, this is known as the property of *Schur convexity* of the function ctr_r on the set $\mathbb{R}_{\geq 0}{}^n$. A sufficient condition for Schur convexity of a continuously differentiable function ϕ defined on I^n, where I is an open interval in $\mathbb{R}$, is that ϕ is symmetric and that the expression

$$(x_i - x_j)\Big[\frac{\partial \phi(x)}{\partial x_i} - \frac{\partial \phi(x)}{\partial x_j}\Big]$$

is nonnegative for all $x \in I^n$ ([**117**], page 57). Setting ϕ to ctr_r expands the above to

$$\tfrac{r}{r-1}(x_i - x_j)(x_i{}^{r-1} - x_j{}^{r-1})\big(\sum x_i{}^r\big)^{\frac{1}{r-1}-1}$$

Indeed, this expression is nonnegative for all $x \in I^n$, where I is chosen as the interval $(0, \infty)$. A limiting argument now establishes the first statement in the lemma. A sufficient condition for strictness of the majorization is that whenever (for ϕ symmetric)

$$\frac{\partial \phi(X)}{\partial x_i} = \frac{\partial \phi(X)}{\partial x_j} \tag{44}$$

for given $X \in I^n$, one must have

$$\phi_{(i,i)}(X) + \phi_{(j,j)}(X) - \phi_{(i,j)}(X) - \phi_{(j,i)}(X) > 0 \tag{45}$$

where $\phi_{(i,j)}(x)$ denotes the partial derivative $\partial\phi(x)/\partial x_i \partial x_j$. This condition is given in [**117**], pages 56–57. Equation (44) is satisfied only if $x_i = x_j$. Using this equality and the identities

$$\frac{\partial \mathrm{ctr}_r(x)}{\partial x_i \partial x_j} = \tfrac{r^2}{r-1}(\tfrac{1}{r-1}-1){x_i}^{r-1}{x_j}^{r-1}\Big[\sum {x_i}^r\Big]^{\frac{1}{r-1}-2}$$
$$\frac{\partial \mathrm{ctr}_r(x)}{\partial x_i \partial x_i} = \tfrac{r^2}{r-1}(\tfrac{1}{r-1}-1){x_i}^{r-1}{x_i}^{r-1}\Big[\sum {x_i}^r\Big]^{\frac{1}{r-1}-2} + r{x_i}^{r-2}\Big[\sum {x_i}^r\Big]^{\frac{1}{r-1}-1}$$

validates Inequality (45). This proves the second statement for all $u, v \in {\mathbb{R}_{>0}}^n$, and its general form (for $u, v \in {\mathbb{R}_{\geq 0}}^n$) follows again from a limiting argument. □

The Schur convexity of ctr_r implies that ctr_r is a measure for the deviation from homogeneity of a nonnegative vector. The inverse of the mass centre of order r ($r \geq 2$) applied to stochastic vectors has an interpretation as the number of nonzero entries in a weighted sense. This statement is justified by Definition 21 and Theorem 17.

DEFINITION 21. *Let u be a nonnegative vector, and let π be the scalar multiple of u such that π is stochastic. Let P be a subset of $\{1, \ldots, n\}$, let P^c denote its complement in this same set. For $r \geq 2$ the* weighted coverage measure of order r *for P and u is defined to be*

$$\mathrm{Cov}_r(P, u) = 1 - \frac{|P| - \dfrac{1}{\mathrm{ctr}_r(\pi)}\left(\sum_{i\in P}\pi_i - \sum_{i\in P^c}\pi_i\right)}{n} \qquad \textit{(weighted)} \tag{46}$$

If the subscript is omitted, it is understood that the measure is of order 2. □

The interpretation of the measure is as follows. The quantity $1/\mathrm{ctr}_r(\pi)$ can be viewed as the size of the 'ideal' clustering for the vectors u and π. This is certainly true if u is homogeneous. If the actual clustering equals the ideal clustering of u (picking out precisely all positive elements), then the numerator in the fraction cancels, and the measure yields the perfect score one. If the vector u is not homogeneous, then it is impossible for any cluster to yield this perfect score. This makes sense if e.g. the vectors $a = (100, 0, 0)$, $b = (98, 1, 1)$, $c = (58, 21, 21)$, and $d = (50, 25, 25)$ are considered. The cluster (represented by a characteristic vector) $(1, 0, 0)$ should (and does) have a coverage measure equal to one for a, because it perfectly captures all of the mass of a. It does not perfectly capture all of the mass of b, but the clustering $(1, 1, 1)$ for b is inefficient in the sense that the cluster uses two positions accounting for two percent of the mass, and one position accounting for ninety-eight percent of the mass. The performance coefficients (of order 2) of the respective clusterings $(1, 1, 1)$ and $(1, 0, 0)$ for the vectors a, b, c, and d are respectively $(0.333, 1.000)$, $(0.347, 0.999)$, $(0.785, 0.792)$, and $(0.889, 0.667)$. The 'max' coefficients of order $r = \infty$ are respectively $(0.333, 1.000)$, $(0.340, 0.993)$, $(0.575, 0.759)$, and $(0.667, 0.667)$. The coefficient of order 2 tends to be more rewarding if all mass is covered; the max coefficient tends to be more rewarding if relatively small elements are not covered.

The weighted coverage measure $\mathrm{Cov}_r(P,u)$ has the property that it lies between 0 and 1 for all u and P iff $r \geq 2$.

THEOREM 17. *Let π and P be as in Definition 21. Let $\mathrm{Cov}_r(P,\pi)$ denote the weighted coverage measure ($r \geq 2$). Then*

$$0 \leq \mathrm{Cov}_r(P,\pi) \leq 1 \tag{47}$$

with equality in the second inequality if and only if the set of nonzero positions of π coincides with P and the nonzero entries are homogeneously distributed.

For $r < 2$ there exist vectors π and clusterings P such that $\mathrm{Cov}_r(P,\pi) > 1$.

PROOF. For the first part of the theorem only one inequality needs to be proven, since replacement of P with P^c interchanges the inequalities. It is easy to see that $\mathrm{Cov}_r(P,\pi) + \mathrm{Cov}_r(P^c,\pi) = 1$, so that the coverage measure $\mathrm{Cov}_r(P,\pi)$ is equal to zero iff the nonzero entries are homogeneously distributed and P covers all the zero entries of π.

Using the identity $\sum_{i\in P}\pi_i = 1 - \sum_{i\in P^c}\pi_i$ leaves the inequality $2\sum_{i\in P}\pi_i \leq \mathrm{ctr}_r(\pi)|P|+1$ to be proven. This inequality is proven for the special case that $r = 2$; the case $r > 2$ then follows from the monotonicity of $\mathrm{ctr}_r(\pi)$ in r.

The inequality $2x \leq x^2 + 1$ (valid for $x \in [0,1]$) and the inequality $2ab \leq a^2 + b^2$ (valid for all real a and b) validate the following derivation.

$$\begin{aligned}
2\sum_{i\in P}\pi_i &\leq \left(\sum_{i\in P}\pi_i\right)^2 + 1 \\
&= \sum_{i\in P}{\pi_i}^2 + \left(\sum_{i,j\in P, i<j} 2\pi_i\pi_j\right) + 1 \\
&\leq \sum_{i\in P}{\pi_i}^2 + \left(\sum_{i,j\in P, i<j} {\pi_i}^2 + {\pi_j}^2\right) + 1 \\
&= \mathrm{ctr}(\pi) + (|P|-1)\mathrm{ctr}(\pi) + 1
\end{aligned}$$

The second part of the theorem follows from an explicit construction. Let ϵ be an arbitrarily small (fixed) positive number, and set $r = 2-\epsilon$. Let a be a small positive number, let π be the two-dimensional vector $(1-a, a)$, and let P be the cluster $\{1\}$. It is shown that for a small enough the inequality $2\sum_{i\in P}\pi_i \leq \mathrm{ctr}_r(\pi)|P| + 1$ is invalidated. First rewrite this inequality in terms of the chosen parameters as

$$1 - 2a \leq \left[(1-a)^{2-\epsilon} + a^{2-\epsilon}\right]^{\frac{1}{1-\epsilon}}$$

Estimate the right-hand side (which equals $\mathrm{ctr}_r(\pi)$) as follows.

$$\begin{aligned}
\left[(1-a)^{2-\epsilon} + a^{2-\epsilon}\right]^{\frac{1}{1-\epsilon}} &= \left[1 - \left(2a - \epsilon a + \mathcal{O}(a^2)\right) + a^{2-\epsilon}\right]^{1+\epsilon+\mathcal{O}(\epsilon^2)} \\
&= \left[1 - \left(2a - \epsilon a + \mathcal{O}(a^{2-\epsilon})\right)\right]^{1+\epsilon+\mathcal{O}(\epsilon^2)} \\
&= 1 - \left(2a - \epsilon a + \mathcal{O}(a^{2-\epsilon})\right)\left(1 + \epsilon + \mathcal{O}(\epsilon^2)\right) + \mathcal{O}(a^2) \\
&= 1 - 2a + \epsilon a - 2\epsilon a + \mathcal{O}(\epsilon^2 a) + \mathcal{O}(a^{2-\epsilon}) \\
&= 1 - 2a - \epsilon a + \mathcal{O}(\epsilon^2 a) + \mathcal{O}(a^{2-\epsilon})
\end{aligned}$$

It follows that the inequality $2\sum_{i\in P}\pi_i \leq \mathrm{ctr}_r(\pi)|P| + 1$ will fail to be true for the chosen parameters if a is chosen sufficiently small (much smaller than ϵ). Setting $\epsilon = 0.1$ and $a = 0.01$ yields that $\mathrm{ctr}_r(\pi)$ is approximately equal to 0.9792, whereas $1 - 2a$ equals 0.98. □

Combining the modifications from Definitions 19 and 21 yields the following performance criterion for clusterings of weighted graphs.

DEFINITION 22. *Let $r \geq 2$ be real, let u be a nonnegative vector of dimension n, and let π be the scalar multiple of u such that π is stochastic. Denote the set of indices i with u_i nonzero by S. Let P be a subset of $\{1,\ldots,n\}$, let P^c denote its complement in this same set. The* weighted coverage measure of order r *for P and u is defined as*

$$\mathrm{Cov}_r(P,u) = 1 - \frac{|P| - \dfrac{1}{\mathrm{ctr}_r(\pi)}\left(\sum_{i\in P}\pi_i - \sum_{i\in P^c}\pi_i\right)}{|P\cup S|} \qquad \textit{(weighted, scaled)} \tag{48}$$

If the subscript is omitted, it is understood that the measure is of order 2. Next, let G be a graph with associated matrix M, let $\mathcal{P}$ be a partition of $\{1,\ldots,n\}$, and let P_u be the set in $\mathcal{P}$ containing u, $u \in \{1,\ldots,n\}$. The weighted performance measure $\mathrm{Perf}_r(G,\mathcal{P})$ *is obtained by averaging the summed coverage measures* $\mathrm{Cov}_r(P_u,u)$ *for all columns u of M.* □

Both coverage and performance lie between 0 and 1. This is easily seen; if v is the vector u with those zero entries pruned which are not in S (leaving precisely the entries corresponding with $P \cup S$) then the coverage of u according to Definition 22 is the coverage of v according to Definition 21.

9.3 A distance on the space of partitions

The following distance defined on the space of partitions of a given set is used in judging the continuity properties of clusterings generated by the *MCL* algorithm at different levels of granularity, and for measuring the distance between *MCL* clusterings and clusterings of randomly generated test graphs with a priori known density characteristics.

DEFINITION 23. *Let S be the set $\{1,\ldots,n\}$. Let $\mathcal{A}$ and $\mathcal{B}$ be arbitrary partitions of S. The projection number $p_{\mathcal{A}}(\mathcal{B})$ of $\mathcal{A}$ onto $\mathcal{B}$ is defined in terms of the meet of $\mathcal{A}$ and $\mathcal{B}$, that is, the collection of subsets $a \cap b$ with $a \in \mathcal{A}$ and $b \in \mathcal{B}$.*

$$p_{\mathcal{A}}(\mathcal{B}) = \sum_{a\in\mathcal{A}} \max_{c\in\mathcal{A}\cap\mathcal{B}} |a \cap c| \tag{49}$$

The distance d between $\mathcal{A}$ and $\mathcal{B}$ is defined as

$$d(\mathcal{A},\mathcal{B}) = 2n - p_{\mathcal{A}}(\mathcal{B}) - p_{\mathcal{B}}(\mathcal{A}) \tag{50}$$

It will customarily be written as the pair of nonnegative integers $(n - p_{\mathcal{A}}(\mathcal{B}), n - p_{\mathcal{B}}(\mathcal{A}))$, which is equal to the pair $(d(\mathcal{A},\mathcal{A}\cap\mathcal{B}), d(\mathcal{B},\mathcal{A}\cap\mathcal{B}))$ (see below).

□

EXAMPLE. Let $n = 12$, let A and B be the respective partitions $\{\{1,2,3,4\}, \{5,6,7\}, \{8,9,10,11,12\}\}$ and $\{\{2,4,6,8,10\}, \{3,9,12\}, \{1,5,7\}, \{11\}\}$, which have meet $\{\{1\}, \{2,4\}, \{3\}, \{5,7\}, \{6\}, \{8,10\}, \{9,12\}, \{11\}\}$. Then $p_{\mathcal{A}}(\mathcal{B})$ equals $2+2+2$ and $p_{\mathcal{B}}(\mathcal{A})$ equals $2+2+2+1$, the distance $d(\mathcal{A},\mathcal{B})$ equals $24-6-7=11$ and is presented as the pair $(6,5)$.

It is useful to present the distance as a pair, because if either of the pair elements is zero, this implies that the corresponding partition is a subpartition of the other. This is easy to verify, and expressed in the following identities which are valid for all $\mathcal{A}$ and $\mathcal{B}$.

$$\begin{aligned} p_{\mathcal{A}}(\mathcal{A}\cap\mathcal{B}) &= p_{\mathcal{A}}(\mathcal{B}) \\ p_{\mathcal{A}\cap\mathcal{B}}(\mathcal{A}) &= n \\ d(\mathcal{A},\mathcal{B}) &= d(\mathcal{A},\mathcal{A}\cap\mathcal{B}) + d(\mathcal{B},\mathcal{A}\cap\mathcal{B}) \end{aligned}$$

More generally, a small projection number $p_{\mathcal{A}}(\mathcal{B})$ implies that $\mathcal{A}$ is close to being a subpartition of $\mathcal{B}$. This is intuitively clear, and it is formally expressed in the following theorem, by identifying $d(\mathcal{A},\mathcal{B})$ with the weight of a shortest path in a suitable graph.

THEOREM 18. *The distance defined in Definition 23 satisfies the axioms for a metric.*

PROOF. Clearly it is only the triangle inequality that is of concern. The proof follows by showing that the distance corresponds to the shortest distance between $\mathcal{A}$ and $\mathcal{B}$ in a particular undirected weighted graph constructed on the set of all partitions of the set $\{1,\ldots,n\}$. In this graph two partitions are connected via an edge iff one can be constructed from the other by joining two of its sets (equivalently the other is constructed from the first by splitting a set into two). The weight of the edge equals the size of the smallest of the two sets. So the claim is that $d(\mathcal{A},\mathcal{B})$ is the length of a shortest path (in terms of total weight) between $\mathcal{A}$ and $\mathcal{B}$. Denote a split of a set UV into two parts U and V by $(UV)\searrow(U|V)$, denote a join of two parts U and V by $(U|V)\nearrow(UV)$. Denote a path between A and B by a sequence of splits and joins. Now it is easy to verify that $d(\mathcal{A},\mathcal{B})$ is the cost of a path consisting of successive down arrows $()\searrow()$ starting from $\mathcal{A}$ all the way down to the meet $\mathcal{A}\cap\mathcal{B}$, followed by a sequence of up arrows $()\nearrow()$ up to $\mathcal{B}$, and that there is no similar down/up path (with only one reversal in the orientation of the arrows) of lower weight. The crux is now that any other path through the

graph can be converted to a one-reversal down/up path without gaining weight. To this end, it is only necessary to convert an up/down arrow pair to a down/up arrow sequence without gaining weight. Repeated application of this manoeuvre yields the desired result. Two cases must be distinguished. First consider the sequence $(TU|VW) \nearrow (TUVW) \searrow (TV|UW)$, with associated cost

$$\min(|T|+|U|, |V|+|W|) + \min(|T|+|V|, |U|+|W|)$$

It can be converted into the sequence

$$(TU|VW) \searrow (T|U|VW) \searrow (T|U|V|W) \nearrow (TV|U|W) \nearrow (TV|UW)$$

with associated cost

$$\min(|T|, |U|) + \min(|V|, |W|) + \min(|T|, |V|) + \min(|U|, |W|)$$

The following inequalities yield that the latter cost does not exceed the former.

$$\begin{aligned} \min(|T|, |U|) + \min(|V|, |W|) &\leq \min(|T|+|V|, |U|+|W|) \\ \min(|T|, |V|) + \min(|U|, |W|) &\leq \min(|T|+|U|, |V|+|W|) \end{aligned}$$

Second, consider the sequence $(U|V|XY) \nearrow (UV|XY) \searrow (UV|X|Y)$ with associated cost $\min(|U|, |V|) + \min(|X|, |Y|)$. It can be converted to the sequence $(U|V|XY) \searrow (U|V|X|Y) \nearrow (UV|X|Y)$, which has identical cost as the former sequence. The theorem follows. □

10

Clustering characteristics of the MCL algorithm

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

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

Parameter	Meaning	Default setting
`a`	Loop weight	0
`r`	Initial inflation constant	2
`l`	Initial prefix length	0
`R`	Main inflation constant	2

Table 1. *MCL* implementation legend.

10.1 Attractors

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

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

10.2 Overlap

The phenomenon of overlap in the setting of undirected graphs has only been observed so far for input graphs with specific symmetry properties. For these cases, if the *MCL* algorithm produces two clusters C_1, C_2 with nonempty intersection, there exists an automorphism which transforms C_1 into C_2 while leaving the intersection invariant. An example is the line graph on 7 nodes the associated matrix of which is found in Figure 14. The automorphism maps i onto $7 - i$, $i = 1, \ldots, 7$, which leaves the intersection $\{4\}$

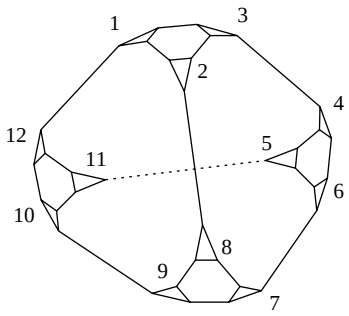


Figure 18. Graph G_4.

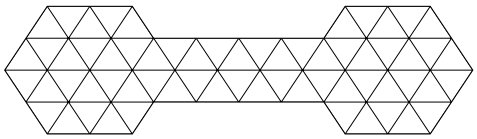


Figure 19. Graph G_5.

invariant. Existence of such an automorphism means that the overlapping part forms a subset of the graph from which the graph looks the same in different directions. If those different directions correspond also with different islands of cohesion, it is rather nice if the overlapping part is not arbitrarily divided among the resulting clusters. Another example of this phenomenon can be found in Figure 20. Overlap occurs at several levels of granularity, and it always corresponds with a symmetry of the graph. For undirected graphs, the amount of possible overlap tends to be proportional to the amount of symmetry present. Naturally, any amount of overlap can be constructed by taking appropriate directed input graphs.

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

10.3 The effect of adding loops

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

10.4 The effect of inflation on cluster granularity

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

The second example in which the inflation parameter is varied while other parameters are kept the same concerns the graph G_4 in Figure 18. It is derived from the graph G_1 in Figure 7 by replacing each of the 12 nodes in G_1 by a triangle. Note that G_4 is a simple graph: The length of the edges in the picture do not correspond with edge weights. Now G_4 clearly allows two extreme clusterings $\mathcal{P}_1 = \{\text{singletons}(V)\}$ and $\mathcal{P}_4 = \{V\}$, a clustering $\mathcal{P}_2$ in which each of the newly formed triangles forms a cluster by itself, and a clustering $\mathcal{P}_3$ with 4 clusters in which each cluster consists of the 9 nodes corresponding

Parametrization `I`	`r`	`R`	Clustering
0	–	1.0 – 1.2	$\mathcal{P}_4$
0	–	1.3 – 1.4	$\mathcal{P}_3$
0	–	1.5 – 3.0	$\mathcal{P}_2$
0	–	3.0 – ∞	$\mathcal{P}_1$
1	1	1.0 – 1.3	$\mathcal{P}_4$
1	1	1.4 – 1.7	$\mathcal{P}_3$
1	1	1.8 – 5.3	$\mathcal{P}_2$
1	1	5.4 – ∞	$\mathcal{P}_1$
2	1	1.0 – 1.4	$\mathcal{P}_4$
2	1	1.5 – 2.4	$\mathcal{P}_3$
2	1	2.5 – 6.8	$\mathcal{P}_2$
2	1	6.9 – ∞	$\mathcal{P}_1$

`a` = 1 set everywhere

Table 2. *MCL* runs for the graph G_4 in Figure 18. The clusterings $\mathcal{P}_1, \ldots, \mathcal{P}_4$ are defined in the text above.

x	`I`	`r`	`R`
5	2	1.2	2.1 – 3.2
	2	1.0	2.1 – 4.0
	2	0.8	2.1 – 5.3
6	2	1.2	2.1 – 2.4
	2	1.0	2.1 – 2.8
	2	0.8	2.1 – 3.3
7	3	1.2	2.1 – 2.7
	3	1.0	2.3 – 3.9
	3	0.8	2.9 – 6.4
8	3	1.2	2.1 – 2.3
	3	1.0	2.3 – 2.9
	3	0.8	2.9 – 4.3
9	3	1.2	2.1
	3	1.0	2.3 – 2.5
	3	0.8	2.9 – 3.3

`a` = 1 set everywhere

Table 3. Parametrizations for which the *MCL* algorithm finds 10 clusters of size x each for the input graph $TORUS(10, x)$, $x = 5, \ldots, 9$.

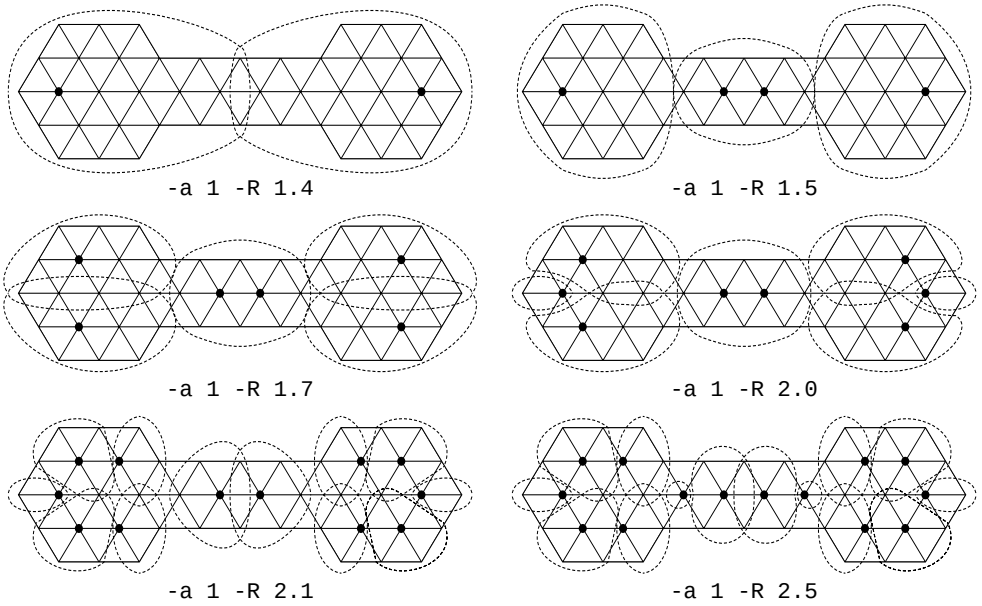


Figure 20. Clusterings resulting from the *MCL* algorithm for the graph in Figure 19. Dotted nodes are attractors.

with 3 newly formed triangles. Clustering with parameters `a` = 1 `E` = 2 `R` = x, where x varies, yields the following. Choosing $x \in [1.0, 1.2]$ results in the top extreme clustering $\mathcal{P}_4$, choosing $x \in [1.3, 1.4]$ in the clustering $\mathcal{P}_3$, choosing $x \in [1.4, 3.0]$ in the clustering $\mathcal{P}_2$, and choosing $x \in [3.1, \infty]$ results in the bottom extreme clustering $\mathcal{P}_1$. The range of x for which the clustering $\mathcal{P}_4$ results is small. This has to do with the fact that the clustering $\mathcal{P}_4$ is rather coarse. The dependencies associated with $\mathcal{P}_4$ correspond with longer distances in the graph G_4 than the dependencies associated with $\mathcal{P}_3$. If the inflation parameter increases, the latter dependencies (in the form of random walks) soon profit much more from the inflation step than the former dependencies. By letting expansion continue a while before starting inflation, this can be remedied. Table 2 shows several parameter settings and the resulting clusterings.

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

10.5 Flow on torus graphs

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

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

DEFINITION 25. *The 1-dimensional* ***torus graph*** *or* ***ring graph*** *of cardinality t is the simple graph defined on the integers modulo t: $0, \ldots, t-1$, where there is an edge between i and j iff $i \equiv j + 1 (\mod t)$ or $j \equiv i + 1 (\mod t)$.*

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

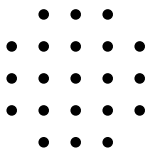
Here I will use only 2- and 3-dimensional simple torus graphs. A 2-dimensional torus graph $TORUS(k, l)$ can be thought of as a rectangular grid of width k and depth l, where nodes lying opposite on parallel borders are connected. In Section 6.3 it appeared that periodic *MCL* limits exist which have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

The last torus example is the 3-dimensional torus graph $TORUS(3, 4, 5)$. A priori it is to be expected that the non-extreme clusterings which the *MCL* algorithm can possibly produce are the clustering $\mathcal{P}_2$ corresponding with 20 subgraphs isomorphic to $TORUS(3)$ and the clustering $\mathcal{P}_3$ corresponding with 5 subgraphs isomorphic to $TORUS(3, 4)$. Denote the top and bottom extreme clusterings by $\mathcal{P}_1 = \{\text{singletons}(V)\}$ and $\mathcal{P}_4 = \{V\}$ respectively. The table below gives four parameter ranges yielding the four clusterings $\mathcal{P}_i$.

Parametrization			
l	r	R	Clustering
2	1.2	1.0 – 2.3	$\mathcal{P}_4$
2	1.2	2.4 – 3.3	$\mathcal{P}_3$
2	1.2	3.4 – 6.4	$\mathcal{P}_2$
2	1.2	6.5 – ∞	$\mathcal{P}_1$

The torus examples illustrate the strong separating power of the *MCL* process. This is mainly interesting for a better understanding of the process, and probably not of much help in using the algorithm. The rewarding aspect of the torus examples is that abstract reasoning about the process applied to extreme cases is confirmed by experiments. The further experiments described below will exhibit a characteristic of the *MCL* algorithm that may be considered a weakness. It concerns the formation of clusters in graphs consisting of weakly connected grids, where certain clusters connect parts of different grids. The phenomenon is rather surprising at first, but it can be be understood in abstract terms. It is indicative for the fact that there are severe problems involved in applying graph cluster methods to neighbourhood graphs derived from vector data.

10.6 Graph clustering and the vector model



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

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

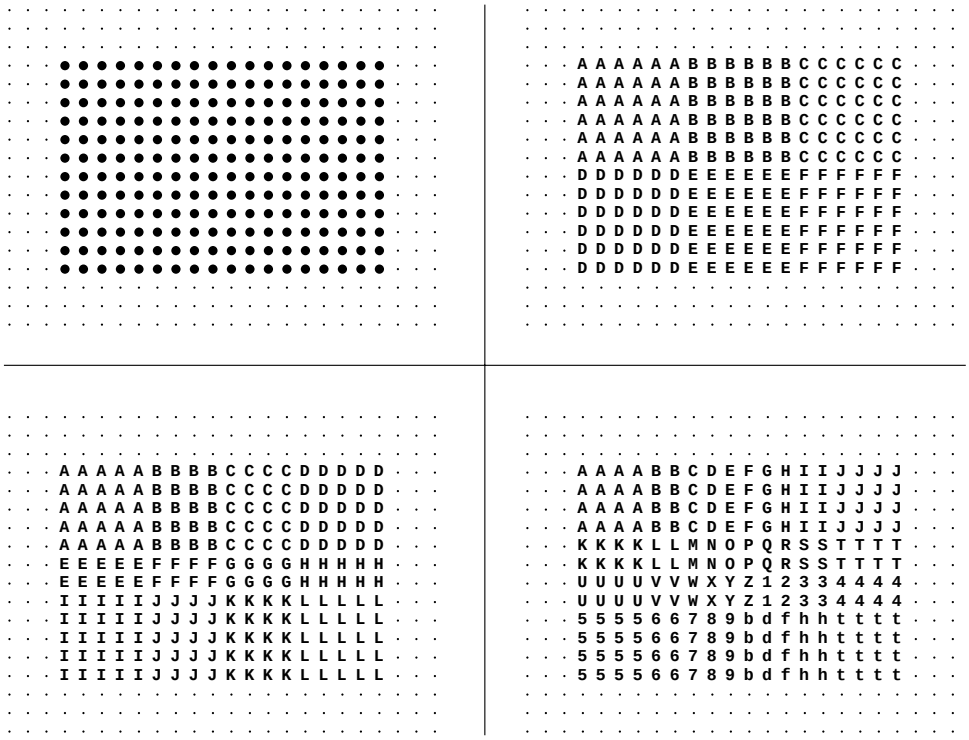


Figure 21. Three clusterings of the 18×12-node graph in the upper left for increasing inflation values. The initial edges are defined by the neighbourhood structure depicted on page 117.

The upper right clustering is remarkable in that the two small satellite subgraphs form clusters together with regions of the large rectangle. This is explained by the effect that the small subgraphs 'inflate' the probability distributions of the nodes of the rectangle lying on the opposing border (i.e. cause them to be less homogeneous). Random walks departing from these border nodes crossing over to the satellite subgraphs have a relatively small probability initially, but due to the low inflation parameter, and the fact that a satellite subgraph has a great absorbing quality, these probabilities win out easily in the end. This is further illustrated by Figure 23, in which all four constellations are again clustered for parametrization $\mathsf{R} = 1.5$. The crossing characteristics of the upper right constellation are now much less dramatic, but still present.

The clustering of the lower left constellation in Figure 22 is remarkable in that the natural bipartition of the large rectangle is rather along the south/north axis than the along the east/west axis. If the rectangle is clustered without satellite systems, then this is the only bipartition which can possibly result, which is explained by the fact that flow is sooner bound along the north/south axis than it is along the east/west axis. This can be

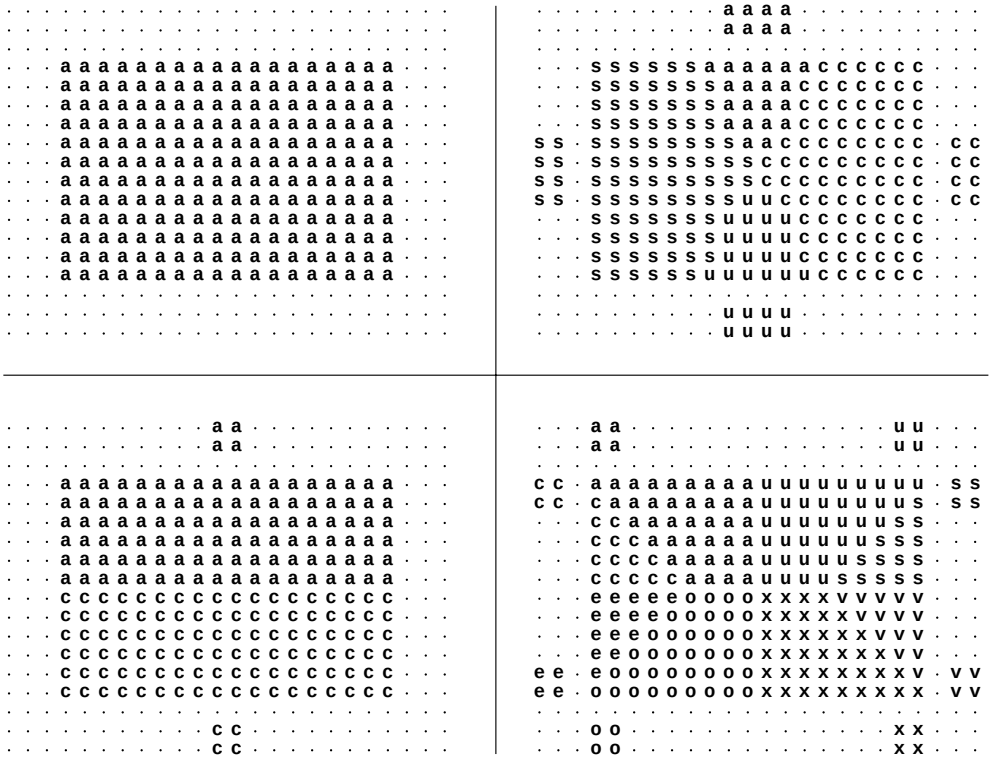


Figure 22. Graph from Figure 21 with different satellite constellations clustered with constant inflation equal to 1.3. The initial edges are defined by the neighbourhood structure depicted on page 117. The upper right clustering exhibits clear border crossing caused by inhomogeneity differentiation. The lower left clustering induces an unnatural bipartition of the large rectangle.

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

In the lower right constellation of Figure 22 it is noteworthy that the corner nodes are all attracted to the clustering corresponding with the neighbouring south or north satellite rather than the east or west satellite. This is probably caused by the fact that the border between the 's' and 'v' clusters is closer by than the border between the 'a' and 'u' clusters, so that the distribution of the corner node transition probabilities is steeper along the south/north axis than it is along the east/west axis. The situation is different for the constellation in Figure 23, because the convergence of the cluster structure corresponding with the satellite systems took place much sooner than the convergence of the four large clusters in the rectangle.

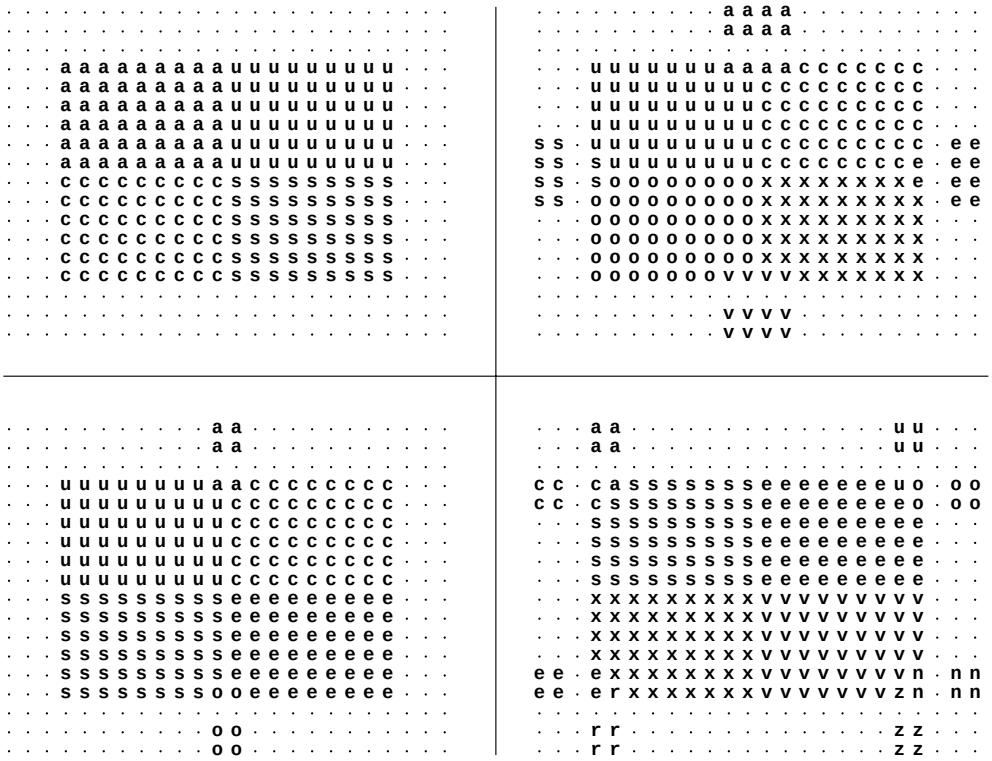


Figure 23. 18×12 Graph with different satellite constellations clustered with constant inflation equal to 1.5. The clusterings exhibit slight border crossing caused by inhomogeneity differentiation.

The behaviour of the *MCL* process for the grid-like graphs in this section has two reasons. The fact that the natural clusters have sizes differing by orders of magnitudes is an important factor. However, examples exhibiting the same border-crossing behaviour can be constructed with sets of grids of the same size, simply by tiling them such that corners are aligned with borders. The most significant factor is the prolonged expansion at low inflation parameter required in order to equalize the probability distributions of opposing corners and opposing borders. The main characteristics of the subgraph corresponding with the large rectangle are that it is rather large (216 nodes) and has relatively large diameter. The process of equalizing distributions via expansion at low inflation values is costly in terms of space due to the large number of elements, and it is costly in terms of time due to the large diameter. The time requirement causes the process to be sensitive to perturbations in the input graph. This was demonstrated by adding small extra grids; similar phenomena occur if for example some nodes are given greater initial return probabilities than other nodes.

The significance of these observations is that one must be careful in applying the *MCL* algorithm to neighbourhood graphs derived from vector data, especially if it is known or unknown whether the diameter of the natural clusters is large. If it is known that this quantity is not too large, then the *MCL* algorithm will work well. This is illustrated by the geometric graph example shown in Chapter 1. The principal cause for the behaviour of the *MCL* algorithm — large diameter and dimension of clusters — will affect any graph clustering algorithm that is grounded on the principles discussed in Chapter 5. The computation of long distance dependencies, be it via random walks, paths, or shortest paths, will in each case be costly and prone to be sensitive to local fluctuations in density of the vectors inducing the neighbourhood graph.

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

10.7 Towards border detection using flow simulation

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

Next consider an image in the form of a bitmap with different grey-levels. An example is given in Figure 25. This image is a bitmap of dimension 284×380, where each pixel may assume 256 grey-levels. A graph was created from this bitmap with a node for each pixel. Horizontally, vertically, and diagonally neighbouring nodes (pixels) where connected via an edge with weight inversely proportional to the difference in grey level between the pixels. Loops were added such that the return probability of a node equalled the mass centre of order 2 of the stochastic vector associated with this node.

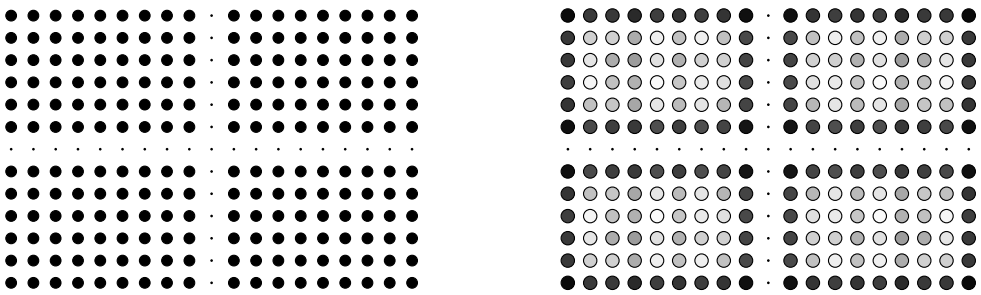


Figure 24. Using flow to find borders. The grey level of the nodes on the right hand side is inversely proportional to the extent to which the nodes attract flow.

A variant of the *MCL* process was applied to this graph which incorporated aggressive pruning. That is, after each matrix multiplication, all nodes were allowed to have at most nine neighbours. The natural choice for this is to pick the nine neighbours with greatest associated probability (see also Chapter 11). After removal of all but the nine largest neighbours of a node, the corresponding pruned column is rescaled to have weight one again. If a pixel is situated in a homogeneous region, then for early stages of the process the neighbours with largest associated transition probability will be just the set of its initial neighbours (including itself), since there is no reason for the symmetry to be broken. Moreover, the return probability will be the largest value in the column, since the symmetry leaves no room for any other direction of attraction. On the other hand, if a pixel is situated near a border or edge in the image, then the distribution of the associated probability vector will be asymmetric with respect to the initial neighbourhood constellation. This will cause the emergence of a direction of attraction, just as in the example in Figure 24. Figure 26 shows the result of interpreting the third iterand of the resulting process using the same principle as in Figure 24 and using a threshold for the indicator value T_{ii}/c_i.

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

Figure 25. San Giorgio Maggiore in Venice.

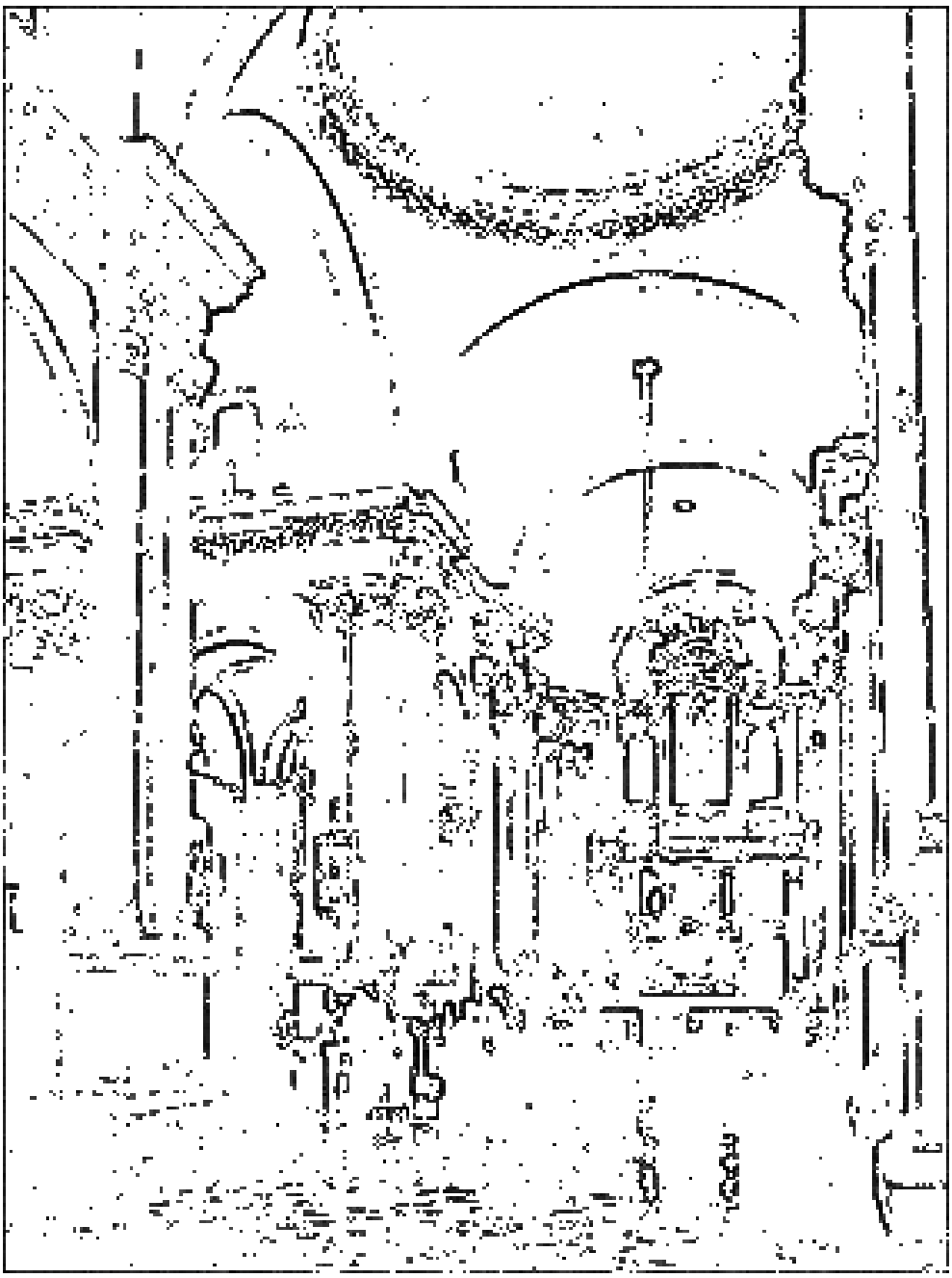


Figure 26. Result of a bordering process based on the *MCL* process.

11

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 in the first of the two sections in this chapter. The last section contains a brief description of the implementation with which the experiments in this thesis were carried out.

11.1 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 (Sections 10.1 and 10.2 in the previous chapter).

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. Section 10.6) 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, and this reasoning will be put to the test by experimenting with randomly generated testgraphs in the next chapter.

11.1.1 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 columns 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 the next chapter this was ensured by using threshold pruning.

11.1.2 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 the previous chapter indicate that the latter will be the case if the diameter of the natural clusters is large. Those examples turned out to vex the the *MCL* algorithm in a much more fundamental way however. In the next chapter I report on experiments using randomly generated testgraphs in which both the graphs themselves and the natural clusters have in general small diameter, and on how the algorithm scales when scaling these dimensions.

11.1.3 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 the following chapter in terms of performance criteria.

11.2 *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 in due course.

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 in the previous section. 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 Chapter 12 indicate that fine-tuning the pruning regime may result in considerably better performance.

12

Randomly generated test graphs

The *MCL* algorithm is tested against randomly generated simple graphs for which it is a priori known that they posses natural cluster structure. The graphs which are used have small diameter and so do (the subgraphs induced by) their natural clusters. The first section describes how test graphs are generated and contains a small-scale example with a graph on 160 nodes. The second section gives an account of two experiments. In the first experiment the *MCL* algorithm was applied to three graphs on 10000 nodes with identical cluster structure but with different density characteristics. Each graph was input to several *MCL* runs. In each run a different pruning constant k was applied while holding other *MCL* parameters fixed. The results indicate that pruning works well, which is explained by the small diameter of the natural clusters. In the second experiment eight graphs were generated (again on 10000 nodes) having the same density characteristics but different granularity characteristics. They were clustered using the exact same *MCL* parametrization for each graph. The resulting clusterings are generally quite good, indicating that the *MCL* algorithm can handle graphs with large natural clusters, and that it depends mainly on the diameter and the density characteristics of the natural clusters which parametrizations give the best result. The experiments indicate that fine-tuning the pruning regime considerably helps the performance of the *MCL* algorithm.

12.1 The generating model

For the purpose of testing the *MCL* algorithm, simple graphs are generated via a simple modification of the generic random graph model, in which each edge is realized with some fixed probability p. In the generic random graph model, the parameters are the dimension of the graph n and the probability p.

DEFINITION 26. *A **random cluster/graph generator** $\mathfrak{G}$ (Fraktur G) is a tuple $(n, p, q, \mathfrak{O})$, where n is a positive integer, p and q are probabilities satisfying $1 \geq p \geq q \geq 0$, and $\mathfrak{O}$ (Fraktur O) is a generator producing partitions of the set $\{1, \ldots, n\}$.*

*The generator $\mathfrak{G}$ generates a **cluster test graph** by obtaining a partition $\mathcal{P}$ from $\mathfrak{O}$, and subsequently realizing an edge (k, l) with respectively probability p if k and l are in the same partition element of $\mathcal{P}$, and probability q if k and l are in different partition elements.* □

The partition generator has not been further specified in this definition, as it is convenient to be able to plug in different types of generators, which differ with respect to the granularity and homogeneity (i.e. variation in the subset sizes) of the partitions generated.

The symmetric nature of the way in which edges are locally realized within partition elements implies that the corresponding subgraphs are unlikely to have long chains. A simple way of seeing this is by envisioning the corresponding incidence submatrix. For any permutation of this matrix, the nonzero entries are expected to be homogeneously distributed, whereas a long chain corresponds with a (part of the) submatrix in which all elements are close to the diagonal. The theory of randomly generated graphs confirms that the expected diameter of the connected components is small [**24, 25**]. The upshot is that the generated cluster structure is spherical rather than chain-like.

The discussion in this chapter will mostly follow heuristic arguments. Whereas the simple setup in Definition 26 should allow mathematical reasoning by probabilistic arguments about the generated graphs, this is entirely beyond the scope of this thesis. The mathematics behind random graph-theory is rather involved, and constitutes a large branch of research in itself (for results on the diameter of random graphs, consult [**24, 25, 136**] and references therein). I do claim however that the random cluster/graph generator is a canonical model for generating cluster test graphs. Consider a graph G generated by a $(n, p, q, \mathfrak{O})$ generator for a partition $\mathcal{P}$ from $\mathfrak{O}$. The graph can be viewed as a random graph generated with parameters (n, q), after which extra edges are added *solely in the subgraphs corresponding with elements of* $\mathcal{P}$. Assuming that p is much larger than q, it follows that $\mathcal{P}$ must approximately be the best clustering of G; it is simply extremely unlikely that any other block diagonalization achieves the same high density p of nonzero entries within the blocks, and the low density q of nonzero entries outside the blocks.

The tests with the *MCL* algorithm were carried out with the randomized parametrized generator defined below.

DEFINITION 27. *A* ***grid partition generator*** $\mathfrak{P}$ *(Fraktur P) is a pair* (n, g)*, where* n *and* g *are positive integers with* $g \leq n$*. Let* r *be the remainder of* n *modulo* g*, and let* k *be the integer part of* n/g*, so that* $n = kg + r$*.*

The generator $\mathfrak{P}$ *generates a partition of* $\{1, \ldots, n\}$ *by generating* k *random permutations of the interval* $\{1, \ldots, g\}$*, and taking as partition sizes the lengths of the cycles. The partition sizes derived from the* i^{th} *permutation,* $i = 1, \ldots, k$*, are mapped onto a consecutive set of elements in the set* $\{(i-1)g + 1, \ldots, ig\}$*. The last* r *entries are partitioned similarly by a random permutation of the set* $\{1, \ldots, r\}$*.* □

NOTE. Throughout this chapter, the word 'partition' will be used to refer to the partition used in generating a test graph. The word 'clustering' is used to refer to a clustering generated by the *MCL* algorithm. A cluster is an element of the clustering, i.e. a set of nodes or node indices. Its counterpart in a partition is called a partition element. The words 'graph' and 'matrix' will be used almost interchangingly. In particular, the diagonal block structure of a (permuted according to cluster structure) matrix is in one–one correspondence with the clustering of the underlying graph of the matrix.

The incidence matrix in Figure 27 was generated with parameters $n = 160$, $g = 60$, $p = 0.25$, and $q = 0.03$. The underlying graph of this particular matrix has diameter four. The generated partition sizes (after rearrangement) equalled 1^5, 2, 2, 3, 16, 20, 35,

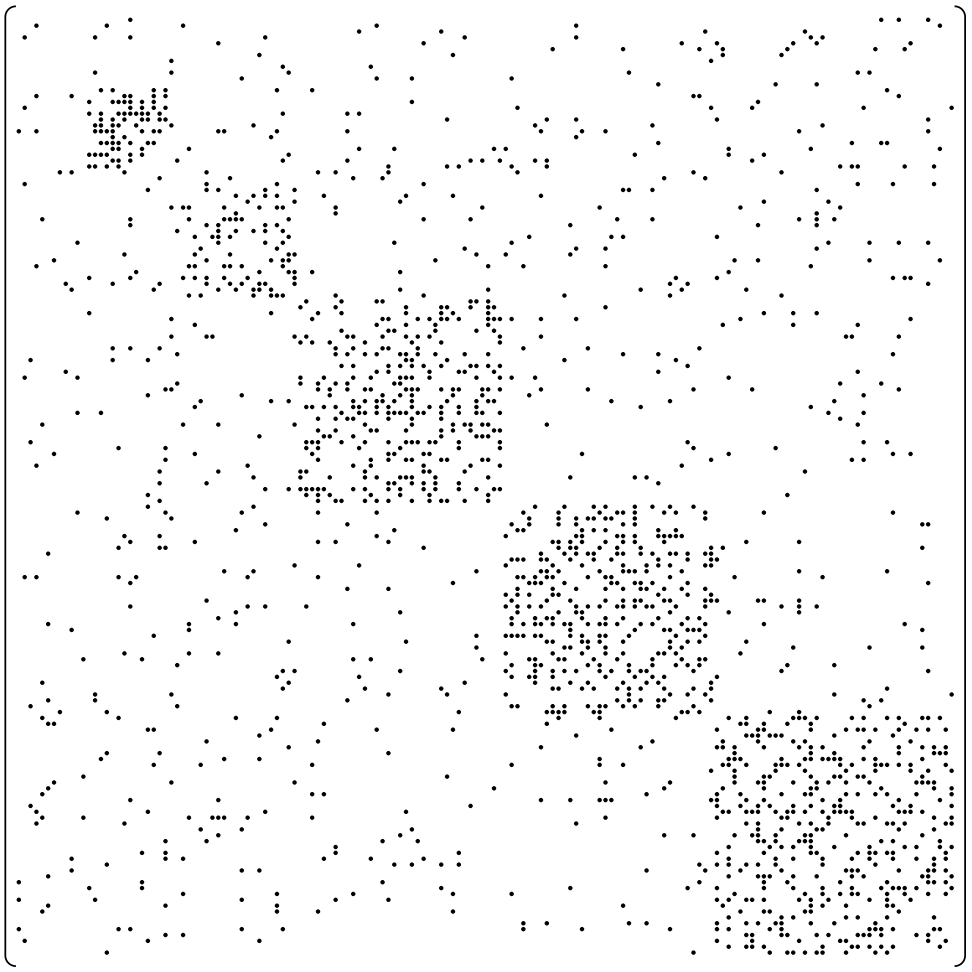


Figure 27. Randomly generated cluster test matrix. Dots indicate nonzero entries, which are all equal to one. The indices were aligned according to the generating partition.

36, and 41. A random permutation of this matrix is shown in Figure 28, to illustrate the serious challenge of finding block structure in matrices (i.e. clustering the underlying graph).

A singular property of the random cluster/graph generator model is that small partition elements do *not* result in clear clusters in the generated graph, and thus introduce the phenomenon of noise in the testgraphs. This is illustrated by the matrix in Figure 27. The partition elements of size 1, 2, and 3 correspond with nodes of lowest index, inducing the leftmost columns and uppermost rows. Had clusters of size up to approximately

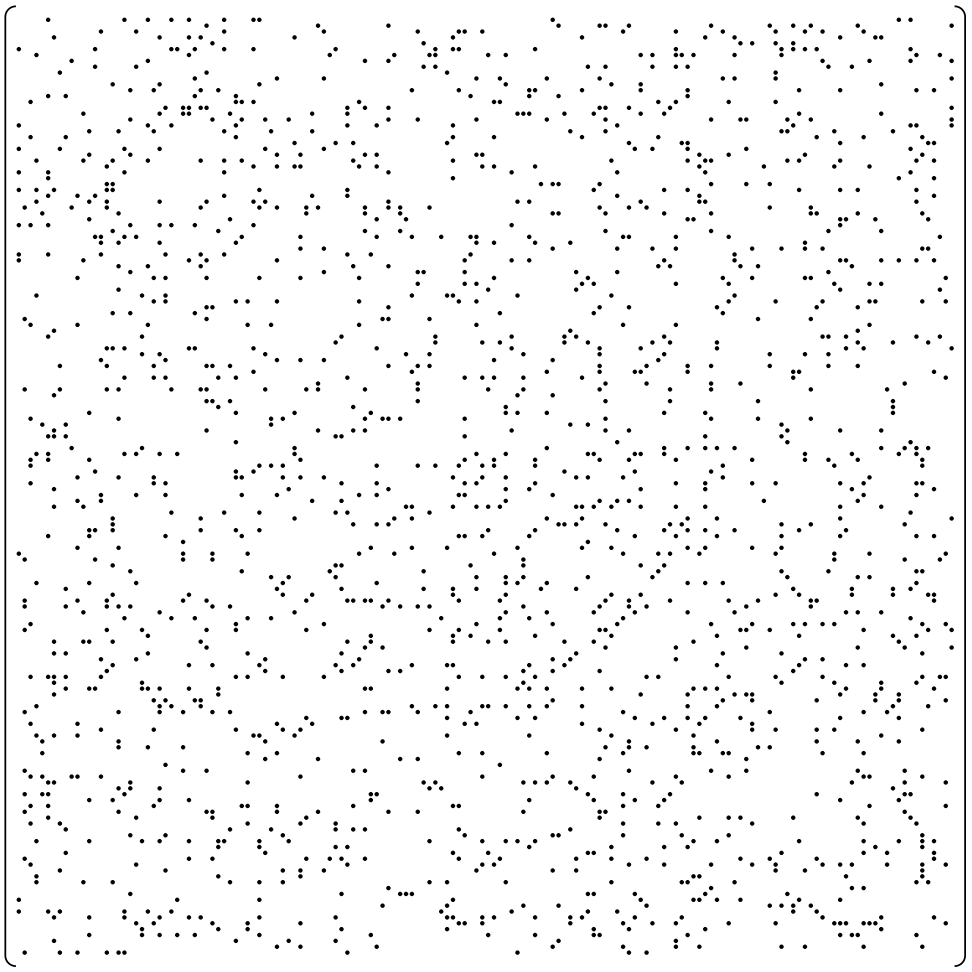


Figure 28. Random permutation of the matrix in Figure 27. A random clustering will have low performance coefficient.

ten been present, the same would apply to them. Apart from modifying the partition generator, this phenomenon can be remedied by extending the random cluster/graph generator model by introducing functions $f(p,q,n,x)$ and $g(p,q,n,x,y)$ such that an edge is realized with probability $f(p,q,n,x)$ if its incident nodes are in the same partition element of size x, and with probability $g(p,q,n,x,y)$ if its incident nodes are in partition elements of respective sizes x and y. This is in fact part of the generator implementation in use at CWI. However, all experiments discussed in this chapter were conducted without using this refinement. Introducing more parameters stand in the

i	a	R	pf(G, C_i)	pf(G^2, C_i)	p_i	q_i	$\lvert C_i\rvert$	$d(C_i, \mathcal{P})$	$d(C_i, C_{i-1})$
1	1	1.3	0.111	0.442	0.100	0.026	2	(84, 4)	–
2	1	1.4	0.149	0.546	0.150	0.029	3	(48, 12)	(0,45)
3	1	1.5	0.149	0.546	0.150	0.029	3	(48, 12)	(0,0)
4	1	1.6	0.193	0.598	0.219	0.032	8	(19, 15)	(9,38)
5	1	1.7	0.201	0.600	0.240	0.034	11	(18, 20)	(6,12)
6	1	1.8	0.186	0.581	0.256	0.042	24	(18, 39)	(3,23)
7	1	1.9	0.165	0.526	0.365	0.055	48	(17, 83)	(8,53)
								$d(C_i, C_{i-7})$	$d(C, C_{i-1})$
8	2	1.3	0.111	0.442	0.100	0.026	2	(0,0)	–
9	2	1.4	0.148	0.547	0.151	0.029	3	(2,2)	(1,47)
10	2	1.5	0.181	0.583	0.189	0.031	6	(7,25)	(6,25)
11	2	1.6	0.183	0.581	0.193	0.032	6	(8,21)	(7,7)
12	2	1.7	0.205	0.601	0.255	0.035	14	(5,9)	(6,28)
13	2	1.8	0.204	0.586	0.281	0.041	24	(13,14)	(5,22)
14	2	1.9	0.181	0.543	0.374	0.052	49	(24,17)	(6,43)

Table 4. Various *MCL* runs for the matrix in Figure 27 — pf is an abbreviation of Perf(ormance).

way of standardizing, repeating, comparing, and benchmarking of experiments. Moreover, the presence of noise (nodes and edges not really fitting in any larger scale cluster structure) is actually interesting in its own right, as it will surely pop up in real-life applications.

As a first elaborate example, several *MCL* runs were carried out for the matrix in Figure 27, with pruning constant set to 50. The results are depicted in Table 4. The partition $\mathcal{P}$ used in constructing this matrix has performance criterion (according to Definition 22) equal to 0.198 for the matrix itself and 0.607 for the square of the matrix. The partition consists of elements with respective sizes 1^5, 2, 2, 3, 16, 20, 35, 36, and 41. The number of clusters found is given in the column under $|C|$. The distance between two different clusterings is measured by the pair-valued function d defined in Section 9.3, Chapter 9. The density of the number of nonzero entries (as a fraction of the cluster area, which is the sum of the squares of the cluster sizes) is given in the column labelled p_i. The corresponding density of nonzero entries *not* covered (as a fraction of the remaining area) is given in the column labelled q_i.

The cluster sizes of the 'best' clustering in row 12 are respectively 1^3, 2^2, 3^2, 4, 6, 14, 15, 33, 34, and 41. The four largest clusters roughly correspond with the four largest partition elements — for the clusters of size 15, 33, 34, and 41 the size of the symmetric difference with a best matching partition element of $\mathcal{P}$ is respectively 3, 4, 2, and 4. The distance pair of this clustering with $\mathcal{P}$ is $(17, 23)$. The data in the figure shows that in general either the clustering or the partition is close to their intersection. The same holds for pairs of clustering at consecutive levels of granularity (corresponding with a small increase of the inflation power coefficient). This shows that the *MCL* algorithm has good properties with respect to continuity. The clustering/partition distances in the

upper part of the table indicate that the maximum performance value is attained for the clustering which is approximately closest to the partition. This can be taken as evidence that the perfomance criterion is a good assistent in measuring the quality of a clustering. The drop in performance from 0.583 to 5.81 at inflation level 1.6 in the lower part of the table is a little peculiar, but the fact that the corresponding clusterings are rather close can easily account for such a small fluctuation. Moreover, it is seen in both parts of the table that all clusterings which are slightly better than the partition with respect to the matrix, are slightly worse than the same partition with respect to the square of the matrix.

Finally, it is interesting to visualize the effect that longer distances have on the resulting clustering. To this end, a champion clustering with performance coefficient equal to 0.214 (resulting from loop weight 3 and inflation 1.7) is used to align the *square* of the matrix in Figure 27. The result is depicted in Figure 29. One sees very clearly off-diagonal bands in the matrix which correspond with (two-step) edges connecting different subgraphs corresponding with different diagonal blocks. These two step connections have 'helped' each of the diagonal blocks to become a cluster in the *MCL* process. Now it is natural to wonder whether this champion clustering cannot be further improved by joining the diagonal blocks connected by the bands. The answer is no, and the reason is that this *does* improve the clustering with respect to the performance criterion applied to the *square of the graph* — which is a mere 0.567 — but it does not improve the clustering with respect to the performance criterion applied to the graph itself.

12.2 Scaled experiments

In this section two experiments are described in which several graphs with 10000 nodes are clustered. In the first experiment three graphs are generated using the same underlying partition $\mathcal{P}$, but with different probabilities for the realization of edges within partition elements. In the second experiment graphs are generated with the same probabilities but with underlying partitions which have different granularities. These graphs are clustered using the same *MCL* parametrization for each. The partition $\mathcal{P}$ which was used for generating the first three test graphs has grid parameter g equal to 500 and has partition element sizes

1^{20}, 2^{10}, 3^{7}, 4^{6}, 5^{5}, 6^{5}, 7, 8^{2}, 9^{3}, 10^{3}, 11, 12^{2}, 13, 14^{3}, 16^{3}, 20, 21^{2} 25, 27, 31, 32, 36, 38, 40^{2}, 41^{2}, 45, 54, 58, 63, 66, 70, 74^{2}, 78, 81, 85, 89, 92^{2}, 97, 108, 111, 116, 121^{2}, 125^{2}, 129, 131, 137, 169^{2}, 183, 226, 255, 284, 298, 314, 343, 350, 352, 374, 392, 401, 422, 428, 444, 484, 499, 500.

The probability p was chosen equal to 0.1 and the probability q was respectively chosen as 0.002, 0.004, and 0.006. Three graphs labelled H_1, H_2, and H_3, were generated this way. A node of H_1 in the partition element of size 500 has on average 50 neighbours on the inside (with respect to this element) and 19 neighbours on the outside. The higher values of q for H_2 and H_3 imply that the cluster structure is more concealed in these graphs. Optimal clustering parameters were sought for each graph using pruning constant $k = 200$. In doing so three parameters were varied, namely the loop weight a, the

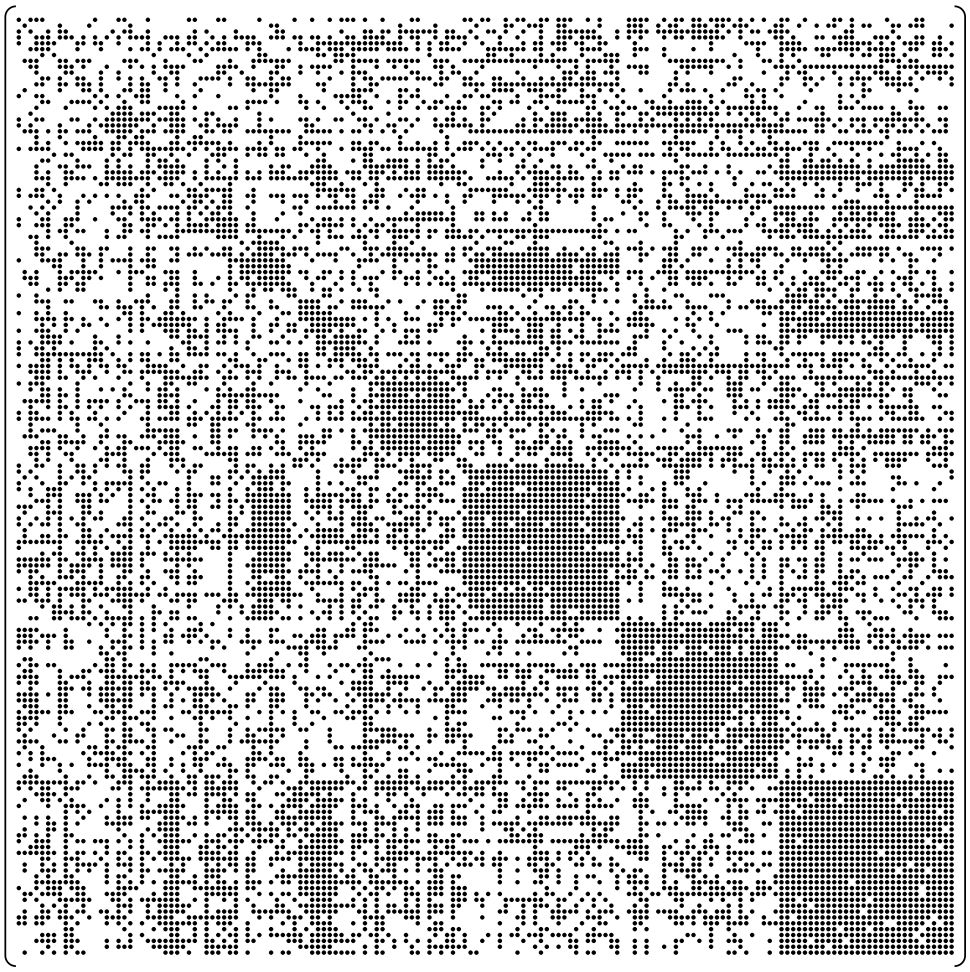


Figure 29. Square of matrix in Figure 27 aligned according to clustering found by the *MCL* algorithm. Dots denote nonzero entries, which are positive integers. The presence of off-diagonal vertical and horizontal bands is explained by the nature of the *MCL* process.

initial inflation constant r, and the main inflation constant R. The length of the initial prefix was in each case chosen equal to 2. Good clusterings were sought for each of the three graphs, judged by performance coefficient only. Parameters were varied according to their observed effect on the performance coefficient; a few different trials sufficed to find good parametrizations. The *MCL* algorithm was again applied holding the corresponding parameters fixed, except for the pruning constant k which was successively decreased with a decrement of 25. The invariant parts of this setup are summarized in Table 5 on page 138.

Table 6 on page 138 shows that the cluster structure found by the *MCL* algorithm matches the generating partition $\mathcal{P}$ rather well. Note that the size of the 22^{nd} largest partition element equals 131. The nodes in this partition element have approximately 13 neighbours within the same element (for all three of H_1, H_2, and H_3), and respectively approximately 20, 40, and 60 neighbours elsewhere (for H_1, H_2, and H_3). For all but two parametrizations this partition element is clearly recognized as a cluster. It is furthermore noteworthy that the clusterings are not very much affected by the value assumed by the pruning constant k, as long as it does not become critically low.

In the second experiment eight graphs were generated on 10000 nodes with probabilities $p = 0.1$ and $q = 0.004$, that is, the same probabilities used for H_2. The underlying partition was generated for each graph separately, where g varied from 300 to 1000. It was ensured (by repeated trials) that each partition generated with grid size g, $g = 300, 400, \ldots, 1000$ had several partition elements of size close to g. Each graph was clustered using the same set of parameters which worked best for H_2, and the pruning constant was chosen equal to 150. The results are depicted in Table 7 on page 139. They clearly show that the same parametrization works for graphs with distinctly different granularity characteristics with respect to their natural cluster structure. The *MCL* algorithm does not perform very well for the graph of lowest cluster granularity (corresponding with grid size g=300), which is explained by the fact that the corresponding generating partition has many clusters of small cardinality — approximately half of the nodes belong to a partition element of size 110 or less.

12.2.1 Pruning. A pruning scheme was used in which threshold pruning is applied first followed by exact pruning. Thresholds of the form $\mathrm{ctr}(c)(1 - t[\max_i(c_i) - \mathrm{ctr}(c)])$ (where c is the stochastic column vector to be pruned) seem to work best in practice, where t is chosen in the range $[1.0, 7.0]$. Threshold pruning was *always* applied for each newly computed column vector, at every stage of the process. No attempt to readjustment was made in case thresholding caused the number of nonzero vector entries to drop below the pruning constant k or if thresholding left a number of nonzero entries much larger than k. However, early stages of the process need thresholding more severely than the middle stages of the process, as the newly computed column vectors during expansion tend to have a much larger number of nonzero entries during early stages. Thresholds that are too harsh during the middle stages have in general an adverse effect on the quality of the resulting clustering. For this reason thresholds of the form $a[\mathrm{ctr}(c)]^b$ appear to be disadvantageous, as they are difficult to tune. Using the threshold $\mathrm{ctr}(c)(1 - t[\max_i(c_i) - \mathrm{ctr}(c)])$ and slowly increasing the parameter t during the process overcomes this problem. This approach is still a bit crude if t is increased regardless of the density characteristics of the iterands (such was the setup in conducting these experiments). It seems quite worthwhile to search for smart pruning schemes which are cheap to compute and effective during the various stages of the *MCL* process.

12.2.2 Clusterings associated with intermediate iterands. The implementation that was used in conducting these experiments computed a (possibly overlapping) clustering for each *MCL* iterand M after every expansion step. This was done by defining a directed graph G on the column indices of M by creating an arc $q \to p$ iff $M_{pq} \geq M_{qq}$. The overlapping clustering was computed as the set of all weakly connected components of G. In an ideal world (where tractability is not an issue and computers use real numbers instead of

approximate values), the matrix M would have been guaranteed to be *dpsd* (since all input matrices are symmetric), and the overlapping clustering could have been computed as the set of all endclasses of the *DAG* associated with M (according to Theorem 9) joined with the nodes reaching them.

For the experiments in this section it took the *MCL* algorithm between 12 and 20 iterations (each iteration consisting of one expansion and one inflation step) to reach a matrix iterand for which the computed clustering was identical to the clustering associated with the eventual limit (which was typically reached 5 iterations later). However, the initial stages exhibited much more dramatic decreases in the number of clusters than later stages — the number of clusters was always decreasing, never increasing. The table below supplies the number of clusters and the amount of overlap associated with each iterand of the run for H_2 using pruning constant 150. The numbers are quite typical for the behaviour of the *MCL* algorithm during the scaled experiments.

Expansion step	#clusters	#nodes in overlap
1	10000	0
2	10000	0
3	7438	139
4	3430	1970
5	2455	166
6	1418	92
7	778	102
8	420	119
9	240	45
10	152	41
11	121	28
12	113	18
13	106	8
14	105	3
15	103	1
16	102	0
17	101	0
18	101	0

One of the many things that has not yet been investigated (theoretically nor empirically) is the relationship (distance) between the clusterings associated with different iterands, and the depths (i.e. the length of a longest path) of the *DAGs* that occur. For the type of graph experimented with here I expect that the depth of the associated *DAG* will in general be small, on average at most 2. It is furthermore interesting to investigate how the *MCL* algorithm performs for graphs which have lower connectivity and natural clusters of somewhat larger diameter, but without the strong homogeneity properties of meshes and grids. Random geometric graphs such as in Figure 2 on page 5 seem suitable candidates, and I expect that for this type of graphs *DAGs* may arise that have larger depth.

Graph label	p	q	Perf$(H_i, \mathcal{P})$	Cluster parameters			
H_1	0.1	0.002	0.0876	a=3.0	r=1.25	l=2	R=1.3
H_2	0.1	0.004	0.0805	a=3.0	r=1.20	l=2	R=1.3
H_3	0.1	0.006	0.0752	a=4.0	r=1.20	l=2	R=1.3

Table 5. Each graph H_i was generated on the same partition $\mathcal{P}$ (on 10000 nodes, grid size 500) with parameters p and q as indicated. Cluster parameters (see Table 1 on page 112) were chosen after a few trials on each graph. Table 6 gives the result of clustering each H_i with these parameters for varying pruning constants k.

k	pf$(C_i(k))$	$d(C_i(k), \mathcal{P})$	Best matches* between $\mathcal{P}$ and $C_i(k)$
			H_1
200	0.0882	(582,738)	0 0 0 0 0 1 0 0 1 0 0 0 0 2 0 1 1 3 4 2 8 7
175	0.0885	(584,780)	0 0 0 0 0 1 0 0 1 0 0 0 1 1 0 1 2 3 4 2 8 5
150	0.0893	(608,881)	1 0 0 0 0 1 0 0 0 0 0 0 1 1 0 1 5 5 4 2 7 5
125	0.0892	(613,938)	1 0 0 1 0 1 0 0 1 0 0 0 1 0 0 1 4 5 2 2 7 6
100	0.0891	(602,927)	1 2 0 0 0 1 0 0 3 0 0 0 0 0 0 1 2 3 6 2 6 9
75	0.0889	(633,1068)	6 8 0 0 0 1 0 0 3 0 0 0 1 0 0 0 1 3 4 3 6 5
			H_2
200	0.0800	(748,1154)	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 3 1 3 9 7
175	0.0801	(730,1043)	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 1 3 8 4
150	0.0796	(890, 722)	0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 2 1 0 9 4
125	0.0794	(764,864)	0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 1 2 1 0 12 9
100	0.0790	(760,964)	1 0 0 0 0 1 9 1 1 0 0 0 0 0 1 0 2 1 3 4 66 8
75	0.0749	(973,993)	5 0 22 8 4 6 67 5 14 7 17 7 3 51 4 2 18 31 21 40 52 26
			H_3
200	0.0731	(1014,1817)	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 4 4 24 8 17 16
175	0.0731	(1004,1791)	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 3 3 24 7 22 21
150	0.0730	(995,1786)	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 14 6 17 12
125	0.0723	(1045,1734)	2 3 1 1 3 1 2 5 0 2 1 0 9 1 1 1 11 2 14 8 41 18
100	0.0662	(1537,2233)	6 13 2 63 37 27 16 32 17 32 24 43 102 30 62 26 57 30 94 102 …
75	0.0444	(3328,6455)	256 320 161 354 236 … (more three digit numbers) …

Table 6. $C_i(k)$ denotes the clustering of graph H_i resulting from an *MCL* process with parameters as in Table 5 and pruning constant equal to k. In the second column, pf$(C_i(k))$ denotes the performance Perf$(H_i, C_i(k))$.

*The last column gives the sizes of the symmetric difference of the j^{th} largest element of $\mathcal{P}$ ($j = 1, \ldots, 22$), with that cluster of $C_i(k)$ which is the best match for the partition element.

g	pf($\mathcal{P}_g$)	pf(C_g)	$d(C_g, \mathcal{P}_g)$	Best matches* between $\mathcal{P}_g$ and C_g
300	0.0681	0.0650	(2321,3206)	3 1 1 0 0 2 11 4 6 8 8 12 29 10 11 22 18 19 15 27
400	0.0755	0.0733	(1198,1574)	2 0 0 4 0 0 0 10 17 1 1 3 0 2 1 3 6 0 0 9
500	0.0810	0.0810	(740,870)	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
600	0.0829	0.0826	(576,632)	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
700	0.0843	0.0836	(575,671)	0 1 1 0 0 0 0 1 0 0 0 0 0 0 0 1 1 1 1 1
800	0.0838	0.0834	(499,635)	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 3 0
900	0.0863	0.0856	(368,407)	1 1 0 0 1 1 1 0 0 0 0 0 0 0 1 0 1 14 4 5
1000	0.0890	0.0888	(192,225)	1 2 2 2 1 1 1 2 1 2 0 0 1 0 1 0 0 0 1 2

Table 7. For each value of g a partition $\mathcal{P}_g$ was generated using g as grid size with n=10000. A graph H_g was generated using $\mathcal{P}_g$ and probabilities $p = 0.1$ and $q = 0.004$. Each graph H_g was clustered using the same *MCL* parameters a=3, r=1.2, l=2, R=1.3, k=150. In the second and third column, pf($\mathcal{P}_g$) and pf(C_g) respectively denote the performance coefficients Perf($H_g, \mathcal{P}_g$) and Perf(H_g, C_g).

*The last column gives the sizes of the symmetric difference of the j^{th} largest element of $\mathcal{P}_g$ ($j = 1, \ldots, 20$), with that cluster of C_g which is the best match for the partition element.

Bibliography

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

[2] ———, ed., *Proceedings of the* 30^{th} *ACM/IEEE Design Automation Conference*, IEEE, June 1993.

[3] ———, ed., *Proceedings of the* 32^{nd} *ACM/IEEE Design Automation Conference*, IEEE, 1995.

[4] ACM/SIAM, ed., *Proceedings of the fourth annual ACM–SIAM symposium on discrete algorithms*, ACM, Jan. 1993.

[5] ———, ed., *Proceedings of the* 6^{th} *annual ACM–SIAM symposium on discrete algorithms, San Fransicso, CA, USA, January 22–24, 1995*, ACM, 1995. Edited by K. Clarkson.

[6] N. ALON AND V. MILMAN, λ_1, *Isoperimetric inequalities for graphs, and superconcentrators*, Journal of Combinatorial Theory, Series B, 38 (1985), pp. 73–88.

[7] C. ALPERT AND A. KAHNG, *A general framework for vertex orderings with applications to netlist clustering*, IEEE Transactions on VLSI Systems, 4 (1996), pp. 240–246.

[8] C. J. ALPERT AND A. B. KAHNG, *Recent directions in netlist partitioning: a survey*, Integration: the VLSI Journal, 19 (1995), pp. 1–81.

[9] C. J. ALPERT AND S.-Z. YAO, *Spectral partitioning: The more eigenvectors, the better*, in ACM/IEEE [**3**], pp. 195–200.

[10] M. ANDERBERG, *Cluster analysis for Applications*, Academic Press, London, 1973.

[11] T. ANDO, *Majorizations and inequalities in matrix theory*, Linear Algebra & Applications, 199 (1994), pp. 17–67.

[12] ———, *Majorization relations for Hadamard products*, Linear Algebra & Applications, 223/224 (1995), pp. 57–64.

[13] R. ARNHEIM, *Visual Thinking*, University of California Press, 1969.

[14] F. B. BAKER AND L. J. HUBERT, *A graph–theoretical approach to goodness–of–fit complete–link hierarchical clustering*, Journal of the American Statistical Association, 71 (1976), pp. 870–878.

[15] H. BAKER, ed., *The Collected Mathematical Papers of James Joseph Sylvester*, vol. 1, Chelsea, Corrected reprint of the Cambridge University Press 1904 Edition ed., 1973.

[16] J. BANERJEE, W. KIM, S.-J. KIM, AND J. F. GARZA, *Clustering a DAG for CAD databases*, IEEE Transactions on Software Engineering, 14 (1988), pp. 1684–1699.

[17] E. BARNES, A. VANNELLI, AND J. WALKER, *A new heuristic for partitioning the nodes of a graph*, SIAM Journal on Discrete Mathematics, 1 (1988), pp. 299–305.

[18] E. F. BECKENBACH AND R. BELLMAN, *Inequalities*, Springer–Verlag, 1961.

[19] A. BERMAN AND R. J. PLEMMONS, *Nonnegative Matrices In The Mathematical Sciences*, no. 9 in Classics in Applied Mathematics, SIAM, 1994. Corrected and extended republication of the 1979 book.

[20] M. W. BERRY, S. T. DUMAIS, AND G. W. O'BRIEN, *Using linear algebra for intelligent information retrieval*, SIAM Review, 37 (1995), pp. 573–595.

[21] G. BIRKHOFF, *Lattice Theory*, no. 25 in AMS Colloquium publications, American Mathematical Society, third ed., 1967. Chapter XVI, Positive Linear Operators, pp. 380–396.

[22] R. K. BLASHFIELD, M. S. ALDENDERFER, AND L. C. MOREY, *Cluster analysis software*, in Krishnaiah and Kanal [**110**], pp. 245–266.

[23] H. H. Bock, *Probabilistic models in cluster analysis*, Computational Statistics and Data Analsyis, 23 (1996), pp. 5–28.
[24] B. Bollobás, *The diameter of random graphs*, Transactions of the American Mathematical Society, 267 (1981), pp. 41–52.
[25] ———, *The Evolution of Sparse Graphs*, in Bollobás and Erdös [**26**], 1984, pp. 35–57. Proceedings of the Cambridge combinatorial conference in honour of Paul Erdős.
[26] B. Bollobás and P. Erdös, eds., *Graph theory and combinatorics*, Academic Press, 1984. Proceedings of the Cambridge combinatorial conference in honour of Paul Erdős.
[27] J. V. Bondar, *Comments on and complements to* Inequalities: Theory of Majorization and Its Applications, Linear Algebra & Applications, 199 (1994), pp. 115–129.
[28] R. A. Brualdi, S. V. Parter, and H. Schneider, *The diagonal equivalence of a nonnegative matrix to a stochastic matrix*, Journal of Mathematical Analysis and Applications, 16 (1966), pp. 31–50.
[29] T. Bui, S. Chaudhuri, T. Leighton, and M. Sipser, *Graph bisection algorithms with good average case behavior*, Combinatorica, 7 (1987), pp. 171–191.
[30] P. Bushell, *Hilbert's metric and positive contraction mappings in a Banach space*, Archive for Rational Mechanics and Analysis, 52 (1973), pp. 330–338.
[31] P. K. Chan, martine D.F. Schlag, and J. Y. Zien, *Spectral k–way ratio–cut partitioning and clustering*, in ACM/IEEE [**2**], pp. 749–754.
[32] T. Chan, J. Ciarlet, and W. Szeto, *On the optimality of the median cut spectral bisection graph partitioning method*, SIAM Journal on Scientific Computing, 18 (1997), pp. 943–948.
[33] D. R. Chellappa and A. Jain, eds., *Markov Random Fields*, Academic Press Inc., 1991. Proceedings of the workshop in San Diego, June 1989.
[34] M.-D. Choi, M. B. Ruskai, and E. Seneta, *Equivalence of certain entropy coefficients*, Linear Algebra & Applications, 208/209 (1994), pp. 29–36.
[35] I. T. Christou and R. R. Meyer, *Optimal equi–partition of rectangular domains for parallel computation*, Journal of Global Optimization, 8 (1996), pp. 15–34.
[36] E. Coffman Jr., P.-J. Courtois, E. Gilbert, and P. Piret, *A distributed clustering process*, Journal of Applied Probability, 28 (1991), pp. 737–750.
[37] J. E. Cohen, *Contractive inhomogeneous products of non–negative matrices*, Mathematical Proceedings of the Cambridge Philosophical Society, 86 (1979), pp. 351–364.
[38] J. Cong, W. Labio, and N. Shivakumar, *Multi–way VLSI circuit partitioning based on dual net representation*, in IEEE [**92**], pp. 56–62.
[39] J. Cong and M. Smith, *A parallel bottom–up clustering algorithm with applications to circuit partitioning in VLSI design*, in ACM/IEEE [**2**], pp. 755–760.
[40] R. Cormack, *A review of classification*, Journal of the Royal Statistical Society, 134 (1971), pp. 321–367.
[41] D. Cvetković and S. Simić, *The second largest eigenvalue of a graph (a survey)*, Filomat, 9 (1995), pp. 449–472.
[42] P. J. Davis, *Circulant Matrices*, John Wiley & Sons, 1979.
[43] G. Diehr, *Evaluation of a branch and bound algorithm for clustering*, SIAM Journal on Scientific and Statistical Computing, 6 (1985), pp. 268–284.
[44] S. v. Dongen, *Graph clustering and information structure*, in Kraak and Wassermann [**108**], pp. 13–31.
[45] ———, *A new cluster algorithm for graphs*, Tech. Rep. INS–R9814, National Research Institute for Mathematics and Computer Science in the Netherlands, Amsterdam, Dec. 1998.
[46] ———, *A stochastic uncoupling process with applications to clustering*, submitted, (2000).
[47] R. Dubes and A. Jain, *Clustering techniques: the user's dilemma*, Pattern Recognition, 8 (1976), pp. 247–260.
[48] ———, *Clustering methodologies in exploratory data analysis*, Advances in Computers, 19 (1980), pp. 113–227.

[49] R. DUBES AND A. K. JAIN, *Validity studies in clustering methodologies*, Pattern Recognition, 11 (1979), pp. 235–254.
[50] R. DUDA AND P. HART, *Pattern classification and scene analysis*, Wiley, 1973.
[51] T. ELFVING, *On some methods for entropy maximization and matrix scaling*, Linear Algebra & Applications, 34 (1980), pp. 321–339.
[52] L. ELSNER, C. R. JOHNSON, AND J. DIAS DA SILVA, *The Perron root of a weighted geometric mean of nonnegative matrices*, Linear and Multilinear Algebra, 24 (1988), pp. 1–13.
[53] U. ELSNER, *Graph partitioning, a survey*, Tech. Rep. Preprint SFB393/97–27, Technische Universität Chemnitz, Dec. 1997.
[54] B. S. EVERITT, *Cluster Analysis*, Hodder & Stoughton, third ed., 1993.
[55] J. FALKNER, F. RENDL, AND H. WOLKOWICZ, *A computational study of graph partitioning*, Mathematical Programming, 66 (1994), pp. 211–239.
[56] M. FIEDLER, *A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory*, Czechoslovak Mathematical Journal, 25 (1975), pp. 619–633.
[57] ———, *Special matrices and their applications in numerical mathematics*, Martinus Nijhoff Publishers, 1986.
[58] M. FIEDLER AND T. L. MARKHAM, *Some inequalities for the Hadamard product of matrices*, Linear Algebra & Applications, 246 (1996), pp. 13–16.
[59] C. H. FITZGERALD AND R. A. HORN, *On fractional Hadamard powers of positive definite matrices*, Journal of Mathematical Analysis and Applications, 61 (1977), pp. 633–342.
[60] E. FOWLKES AND C. MALLOWS, *A method for comparing two hierarchical clusterings*, Journal of the American Statistical Association, 78 (1983), pp. 553–584.
[61] J. GARBERS, H. PROMEL, AND A. STEGER, *Finding clusters in VLSI circuits*, in IEEE [**90**], pp. 520–523.
[62] M. GAREY AND D. JOHNSON, *Some simplified* NP*–complete graph problems*, Theoretical Computer Science, 1 (1976), pp. 237–267.
[63] S. B. GELFAND AND S. K. MITTER, *On Sampling Methods and Annealing Algorithms*, in Chellappa and Jain [**33**], 1991, pp. 499–515. Proceedings of the workshop in San Diego, June 1989.
[64] A. GEORGE, J. R. GILBERT, AND J. W. LIN, eds., *Graph Theory and Sparse Matrix Computation*, no. 56 in The IMA Volumes in Mathematics and its Applications, Springer–Verlag, 1993.
[65] A. GIOVAGNOLI AND M. ROMANAZZI, *A group majorization ordering for correlation matrices*, Linear Algebra & Applications, 127 (1990), pp. 139–155.
[66] A. GIOVAGNOLI AND H. P. WYNN, *Cyclic majorization and smoothing operators*, Linear Algebra & Applications, 239 (1996), pp. 215–225.
[67] G. H. GOLUB AND C. F. V. LOAN, *Matrix Computations*, Johns Hopkins University Press, third ed., 1996.
[68] I. GOOD, *The botryology of botryology*, in Van Ryzin [**166**], pp. 73–94.
[69] A. GORDON, *A survey of constrained classification*, Computational Statistics and Data Analsyis, 21 (1996), pp. 17–29.
[70] G. GRIMMETT, *Percolation*, Springer Verlag, 1989.
[71] R. GRONE AND R. MERRIS, *The Laplacian spectrum of a graph* II, SIAM Journal on Discrete Mathematics, 7 (1994), pp. 221–229.
[72] R. GRONE, R. MERRIS, AND V. SUNDER, *The Laplacian spectrum of a graph*, SIAM Journal on Matrix Analysis and Applications, 11 (1990), pp. 218–238.
[73] S. GUATTERY AND G. L. MILLER, *On the performance of spectral graph partitioning methods*, in ACM/SIAM [**5**], pp. 233–242. Edited by K. Clarkson.
[74] A. GUPTA, *Fast and effective algorithms for graph partitioning and sparse–matrix ordering*, IBM Journal of Research & Development, 41 (1997). http://www.research.ibm.com/journal/rd/411/gupta.html.

[75] L. HAGEN AND A. B. KAHNG, *A new approach to effective circuit clustering*, in IEEE [**91**], pp. 422–427.
[76] J. HAJNAL, *On products of non-negative matrices*, Mathematical Proceedings of the Cambridge Philosophical Society, 79 (1976), pp. 521–530.
[77] D. HARTFIEL AND C. D. MEYER, *On the structure of stochastic matrices with a subdominant eigenvalue near* 1, Linear Algebra & Applications, 1272 (1998), pp. 193–203.
[78] D. HARTFIEL AND J. SPEELMAN, *Diagonal similarity of irreducible matrices to row stochastic matrices*, Pacific Journal of Mathematics, 40 (1972), pp. 97–99.
[79] J. HARTIGAN, *Clustering Algorithms*, Wiley, New York, 1975.
[80] M. HAZEWINKEL, *Classification in mathematics, discrete metric spaces, and approximation by trees*, Nieuw Archief voor Wiskunde, 13 (1995), pp. 325–361.
[81] B. HENDRICKSON AND R. LELAND, *An improved spectral graph partitioning algorithm for mapping parallel computations*, SIAM Journal on Scientific Computing, 16 (1995), pp. 452–469.
[82] G. HERDEN, *Some aspects of clustering functions*, SIAM Journal on Algebraic and Discrete Methods, 5 (1984), pp. 101–116.
[83] D. HERSHKOWITZ ET AL., *Minimization of norms and the spectral radius of a sum of nonnegative matrices under diagonal equivalence*, Linear Algebra & Applications, 241–243 (1996), pp. 431–453.
[84] D. HERSHKOWITZ, U. G. ROTHBLUM, AND H. SCHNEIDER, *Classifications of nonnegative matrices using diagonal equivalence*, SIAM Journal on Matrix Analysis and Applications, 9 (1988), pp. 455–460.
[85] D. HEYMAN AND M. SOBEL, eds., *Handbooks in Operations Research and Management Science*, vol. 2, North-Holland, 1990.
[86] R. A. HORN AND C. R. JOHNSON, *Matrix Analysis*, Cambridge University Press, 1990.
[87] ———, *Topics in Matrix Analysis*, Cambridge University Press, 1991.
[88] L. HUBERT, *Spanning trees and aspects of clustering*, British Journal of Mathematical and Statistical Psychology, 27 (1974), pp. 14–28.
[89] L. J. HUBERT, *Some applications of graph theory to clustering*, Psychometrika, 39 (1974), pp. 283–309.
[90] IEEE, ed., *Proceedings IEEE International Conference on Computer-Aided Design*, IEEE, 1990.
[91] ———, ed., *Proceedings of the IEEE international Conference on Computer-Aided Design*, IEEE, Nov. 1992.
[92] ———, ed., *Proceedings IEEE International Conference on Computer-Aided Design*, IEEE, Nov. 1994.
[93] ———, ed., *Proceedings of the IEEE Conf. on Comp. Vision and Pattern Recognition, San Juan, Puerto Rico, June*, IEEE, 1997.
[94] A. K. JAIN AND R. C. DUBES, *Algorithms for Clustering Data*, Prentice Hall, 1988.
[95] N. JARDINE AND R. SIBSON, *Mathematical Taxonomy*, Wiley Series In Probabilistic And Mathematical Statistics, John Wiley & Sons, 1971.
[96] C. R. JOHNSON AND H. M. SHAPIRO, *Mathematical aspects of the relative gain array* $(a \circ a^{-T})$, SIAM Journal on Algebraic and Discrete Methods, 7 (1986), pp. 627–644.
[97] D. S. JOHNSON ET AL., *Optimization by simulated annealing: An experimental evaluation. I: graph partitioning.*, Operations Research, 37 (1989), pp. 865–892.
[98] J.-M. JOLION, P. MEER, AND S. BATAOUCHE, *Robust clustering with applications in computer vision*, IEEE Transactions on Pattern Analysis and Machine Intellligence, 13 (1991), pp. 791–802.
[99] D. R. KARGER, *Global min-cuts in* rnc*, and other ramifications of a simple min-cut algorithm*, in ACM/SIAM [**4**], pp. 21–30.
[100] A. KARR, *Markov process*, in Heyman and Sobel [**85**], ch. 2, pp. 95–123.
[101] G. KARYPIS, *Multilevel algorithms for multi-constraint hypergraph partitioning*, Tech. Rep. 99-034, University of Minnesota, Dept. of Computer Science, 1999.

[102] G. KARYPIS AND V. KUMAR, *Multilevel k-way hypergraph partitioning*, Tech. Rep. 98-036, University of Minnesota, Dept. of Computer Science and Engineering, 1998.
[103] L. KAUFMAN AND P. J. ROUSSEEUW, *Finding Groups in Data*, John Wiley & Sons, 1983.
[104] D. E. KEYES ET AL., eds., *Parallel numerical algorithms. Proceedings of the workshop, Hampton, VA, May 23–25, 1994*, Kluwer Academic Publishers, 1997.
[105] L. KHACHIYAN, *Diagonal matrix scaling is* NP*-hard*, Linear Algebra & Applications, 234 (1996), pp. 173–179.
[106] R. W. KLEIN AND R. C. DUBES, *Experiments in projection and clustering by simulated annealing*, Pattern Recognition, 22 (1989), pp. 213–220.
[107] M. KLINE, *Mathematical Thought From Ancient to Modern Times*, Oxford University Press, 1972.
[108] E. KRAAK AND R. WASSERMANN, eds., *Proceedings Accolade 97*, University of Amsterdam, 1998.
[109] O. KRAFFT AND M. SCHAEFER, *Convergence of the powers of a circulant stochastic matrix*, Linear Algebra & Applications, 127 (1990), pp. 56–69.
[110] P. KRISHNAIAH AND L. KANAL, eds., *Handbook of Statistics*, vol. 2, North-Holland, 1982.
[111] F.-J. LAPOINTE AND P. LEGENDRE, *A classification of pure malt scotch whiskies*, Applied Statistics, 43 (1994), pp. 237–257.
[112] R. LOEWY, *Diagonal similarity of matrices*, Portugaliae Mathematica, 43 (1985–1986), pp. 55–59.
[113] V. LOVASS-NAGY AND D. POWERS, *A note on block diagonalization of some partitioned matrices*, Linear Algebra & Applications, 5 (1972), pp. 339–346.
[114] L. LOVÁSZ, *Random walks on graphs: A survey*, in Miklos et al. [**127**], pp. 353–397.
[115] M. MARCUS, *A unified exposition of some classical matrix theorems*, Linear and Multilinear Algebra, 25 (1989), pp. 137—147.
[116] M. MARCUS AND M. SANDY, *Hadamard square roots*, SIAM Journal on Matrix Analysis and Applications, 12 (1991), pp. 49–69.
[117] A. W. MARSHALL AND I. OLKIN, *Inequalities: Theory of Majorization and Its Applications*, no. 143 in Mathematics In Science And Engineering, Academic Press, 1979.
[118] A. W. MARSHALL, I. OLKIN, AND F. PROSCHAN, *Monotonicity of ratios of means and other applications of majorization*, in Shisha [**152**], pp. 177–197. Wright-Patterson Air Force Base, Ohio, August 19–27.
[119] D. L. MASSART AND L. KAUFMAN, *The Interpretation of Analytical Chemical Data by the Use of Cluster Analysis*, Wiley Series in Probability and Mathematical Statistics, John Wiley & Sons, 1990.
[120] D. W. MATULA, *k-Components, clusters and slicings in graphs*, SIAM Journal on Applied Mathematics, 22 (1972), pp. 459–480.
[121] ———, *Graph theoretic techniques for cluster analysis algorithms*, in Van Ryzin [**166**], pp. 95–129.
[122] D. W. MATULA AND L. L. BECK, *Smallest-last ordering and clustering and graph coloring algorithms*, Tech. Rep. CSE 8104, Southern Methodist University School of Engineering and Applied Science, July 1981.
[123] A. MEHROTRA AND M. A. TRICK, *Cliques and clustering: A combinatorial approach*, Operations Research Letters, (1998), pp. 1–12.
[124] R. MERRIS, *Laplacian matrices of graphs: A survey*, Linear Algebra & Applications, 197, 198 (1994), pp. 143–176.
[125] C. D. MEYER, *Stochastic complementation, uncoupling Markov chains, and the theory of nearly reducible systems*, SIAM Review, 31 (1989), pp. 240–272.
[126] ———, *Uncoupling the Perron eigenvector problem*, Linear Algebra & Applications, 114/115 (1989), pp. 69–74.

[127] D. Miklos et al., eds., *Combinatorics, Paul Erdős is eighty*, vol. II, Janos Bolyai Mathematical Society, 1996.
[128] G. W. Milligan, *An examination of the effect on six types of error perturbation on fifteen clustering algorithms*, Psychometrika, 45 (1980), pp. 325–342.
[129] G. W. Milligan and M. C. Cooper, *An examination of procecures for determining the number of clusters in a data set*, Psychometrika, 50 (1985), pp. 159–179.
[130] H. Minc, *Nonnegative Matrices*, Wiley Interscience Series In Discrete Mathematics And Optimization, John Wiley & Sons, 1988.
[131] H. F. Miranda and R. C. Thompson, *Group majorization, the convex hulls of sets of matrices, and the diagonal element–singular value inequalities*, Linear Algebra & Applications, 199 (1994), pp. 133–141.
[132] B. Mirkin, *Mathematical Classification and Clustering*, Kluwer Academic Publishers, 1996.
[133] M. Niezgoda, *Group majorization and Schur type inequalities*, Linear Algebra & Applications, 268 (1998), pp. 9–30.
[134] C. H. Papadimitriou and K. Steiglitz, *Combinatorial optimization*, Prentice–Hall, 1982.
[135] B. Parlett, *Invariant subspaces for tightly clustered eigenvalues of tridiagonals*, BIT, 36 (1996), pp. 542–562.
[136] T. K. Philips, D. F. Towsley, and J. K. Wolf, *On the diameter of a class of random graphs*, IEEE Transactions on Information Theory, 36 (1990), pp. 285–288.
[137] A. Pothen, *Graph partitioning algorithms with applications to scientific computing*, in Keyes et al. [**104**], pp. 323–368.
[138] A. Pothen, H. D. Simon, and K.-P. Liou, *Partitioning sparse matrices with eigenvectors of graphs*, SIAM Journal on Matrix Analysis and Applications, 11 (1990), pp. 430–452.
[139] D. L. Powers, *Structure of a matrix according to its second eigenvector*, in Uhlig and Grone [**164**], pp. 261–265.
[140] V. V. Raghavan and C. Yu, *A comparison of the stability characteristics of some graph theoretic clustering methods*, IEEE Transactions on Pattern Analysis and Machine Intellligence, 3 (1981), pp. 393–402.
[141] W. M. Rand, *Objective criteria for the evaluation of clustering methods*, Journal of the American Statistical Association, 66 (1971), pp. 846–850.
[142] F. Rendl and H. Wolkowicz, *A projection technique for partitioning the nodes of a graph*, Annals of Operations Research, 58 (1995), pp. 155–179.
[143] F. J. Rohlf, *Single-Link Clustering Algorithms*, vol. 2 of Krishnaiah and Kanal [**110**], 1982, pp. 267–284.
[144] U. G. Rothblum, H. Schneider, and M. H. Schneider, *Scaling matrices to prescribed row and column maxima*, SIAM Journal on Matrix Analysis and Applications, 15 (1994), pp. 1–14.
[145] B. D. Saunders and H. Schneider, *Flows on graphs applied to diagonal similarity and diagonal equivalence for matrices*, Discrete Mathematics, 24 (1978), pp. 205–220.
[146] H. Schneider, *The concepts of irreducibility and full indecomposability of a matrix in the works of Frobenius, König and Markov*, Linear Algebra & Applications, 18 (1977), pp. 139–162.
[147] H. Schneider and M. H. Schneider, *Max-balancing weighted directed graphs and matrix scaling*, Mathematics of Operations Research, 16 (1991), pp. 208–222.
[148] S. Schwartzman, *The Words of Mathematics. An Etymological Dictionary of Mathematical Terms Used in English*, Spectrum Series, The Mathematical Association of America, 1994.
[149] E. Seneta, *Non-negative matrices and Markov chains*, Springer, second ed., 1981.
[150] F. Shahrokhi and D. Matula, *The maximum concurrent flow problem*, Journal of the Association of Computing Machinery, 37 (1990), pp. 318–334.
[151] J. Shi and J. Malik, *Normalized cuts and image segmentation*, in IEEE [**93**], pp. 731–737.
[152] O. Shisha, ed., *Inequalities*, Academic Press, 1965. Wright–Patterson Air Force Base, Ohio, August 19–27.

[153] J. SIMON, E. BACKER, AND J. SALLENTIN, *A Unifying Viewpoint on Pattern Recognition*, vol. 2 of Krishnaiah and Kanal [**110**], 1982, pp. 451–477.
[154] A. SINCLAIR, *Algorithms for Random Generation and Counting, A Markov Chain Approach*, no. 7 in Progress in Theoretical Computer Science, Birkhäuser, 1993.
[155] P. SNEATH AND R. SOKAL, *Numerical Taxonomy*, W.H. Freeman, San Fransisco, 1973.
[156] R. R. SOKAL, *Clustering and classification: Background and current directions*, in Van Ryzin [**166**], pp. 1–15.
[157] M. SONKA, V. HLAVAC, AND R. BOYLE, *Image Processing, Analysis, and Machine Vision*, PWS Publishing, second ed., 1998.
[158] R. SRIKANTH, *A graph theory–based approach for partitioning knowledge bases*, ORSA Journal on Computing, 7 (1995), pp. 286–297.
[159] C. J. T. BUI, C.HEIGHAM AND T. LEIGHTON, *Improving the performance of the Kernighan–Lin and simulated annealing graph bisection algorithms*, in ACM/IEEE [**1**], pp. 775–778.
[160] S. TAMURA, *Clustering based on multiple paths*, Pattern Recognition, 15 (1982), pp. 477–483.
[161] D. TOMI, *k–Connectivity and the overlapping stratified clustering algorithm*, in Tosic et al. [**162**], pp. 63–75.
[162] R. TOSIC ET AL., eds., *Graph Theoy, Proceedings* 8^{th} *Yugoslavian Seminar, Novi Sad, Yugoslavia, 1987*, University of Novi Sad, 1989.
[163] V. TZERPOS AND R. HOLT, *Software botryology, automatic clustering of software systems*, in Wagner [**168**], pp. 811–818.
[164] F. UHLIG AND R. GRONE, eds., *Current trends in matrix theory. Proceedings of the Third Auburn Matrix Theory Conference, Auburn University, Auburn, Alabama, U.S.A., March 19–22, 1986*, Elsevier Science Publishing Co., 1987.
[165] R. URQUHART, *Graph theoretical clustering based on limited neighbourhood sets*, Pattern Recognition, 15 (1982), pp. 173–187.
[166] J. VAN RYZIN, ed., *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.
[167] R. VARADARAJAN, *Partitioning multi–edge graphs*, BIT, 30 (1990), pp. 450–463.
[168] R. WAGNER, ed., *Proceedings of the* 9^{th} *International Workshop on Database and Expert Systems Applications*, IEEE, Aug. 1998.
[169] S. WATANABE, *Pattern recognition as a quest for minimum entropy*, Pattern Recognition, 13 (1981), pp. 381–387.
[170] R. A. WILSON AND F. C. KEIL, eds., *The MIT encyclopedia of the cognitive sciences*, MIT Press, 1999.
[171] Z. M. WÓJCIK, *A natural approach in image processing and pattern recognition: Rotating neighbourhood technique, self–adapting threshold, segmentation and shape recognition*, Pattern Recognition, 18 (1985), pp. 299–326.
[172] H. WOLKOWICZ AND G. P. STYAN, *Bounds for eigenvalues using traces*, Linear Algebra & Applications, 29 (1880), pp. 471–506.
[173] H. WOLKOWICZ AND G. P. H. STYAN, *More bounds for eigenvalues using traces*, Linear Algebra & Applications, 31 (1980), pp. 1–17.
[174] C.-W. YEH, C.-K. CHENG, AND T.-T. Y. LIN, *Circuit clustering using a stochastic flow injection method*, IEEE Transactions on Computer–Aided Design of Integrated Circuits and Systems, 14 (1995), pp. 154–162.
[175] C. T. ZAHN, *Graph–theoretical methods for detecting and describing Gestalt clusters*, IEEE Transactions on Computers, C–20 (1971), pp. 68–86.

A cluster miscellany

This appendix is aimed at a general audience. Topics include the cognitive aspects of cluster structure, the role of the computer in cluster analysis, general aims and ideas, a short history of clustering and classification, and some remarks on the usage and etymology of words in mathematics, in particular those words often occurring in this thesis. The appendix concludes with a short account of the main contribution of this thesis, the Markov Cluster algorithm.

1 Of clusters and cognition

A cluster is a) *a close group or bunch of similar things growing together* or b) *a close group or swarm of people, animals, faint stars, gems, et cetera*, according to the Concise Oxford Dictionary[1]. Examples of usage are 'a cluster of berries', 'galaxy cluster', 'cluster of molecules', and 'cluster of computers'. The usage of the word *cluster* in the mathematical discipline *Cluster Analysis* is really the same, except that only the generic parts of the definition remain. Thus, plants, animals, people, computers, molecules, and galaxies are submerged in the sea of things, and as consolation 'things' is changed to the classier 'entities'. The meaning of cluster in a mathematical setting then becomes *a close group of entities*. Clearly, a 'close group' indicates a remarkable degree of fellowship that is in contrast with the surrounding parts. Thus, if a garden has seven yew trees (Taxus Baccata) which are dispersed homogeneously across the garden, they cannot be regarded as forming a cluster together. A point of interest is that in mathematics it is perfectly acceptable for a cluster to consist of a single element, as this makes reasoning about clusters a lot easier. So, in this case, the yews are better viewed as seven separate clusters, just considering the garden they are in. However, if all neighbouring gardens have exactly one yew, then on a larger scale the seven yews may be seen as forming a single cluster. Furthermore, if a great yew baron has a plantation farming thousands of yew trees, then a group of seven yews picked out in the middle is hardly a cluster. On a larger scale again, all of the thousands of yews do form a giant cluster in the surrounding landscape of meadows and pastures.

More complex arrangements can be pictured, as in a garden with twelve groups (Figure 30), each consisting of three yews, where at each different point of the compass three groups of yews are planted together. This results in different clusters on different scales, one where twelve groups of three yews are distinguished, and, on a larger scale, one where four groups of nine yews are distinguished. The picture becomes blurred if a few extra yews are scattered around the (neglected) garden, some of them standing close to the neatly arranged groups (Figure 31). At a certain degree of closeness the mind and

[1] Ninth Edition.

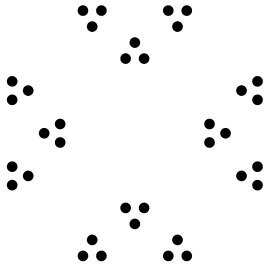


Figure 30. Garden with yews.

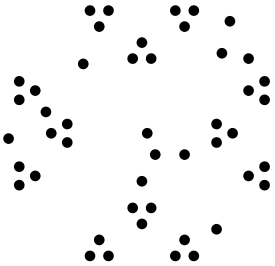


Figure 31. Neglected garden.

eye will tend to see a group of yews swallowing up yews standing closely near, but there is no rule predicting exactly when this will happen. The process is influenced by the relative size and location of other groups. In general, the way in which local cluster structure is perceived is affected by the overall perceived structure; a number of yews which the eye tends to see as a cluster in one configuration may be seen as less related in another configuration. On the other hand, the perception of cluster structure is also influenced by the relative sizes of the different yews, i.e. by highly local parameters. Furthermore, the eye has a tendency to group things together in such a way as to produce balanced groups, and the degrees of regularity and symmetry of an arrangement also plays a role. In the picture on the left of Figure 32 the force of regularity tends to prevail, i.e. enforcing the perception of two clusters of three elements and four clusters of a single element. On the right the scales tend to favour a balanced grouping, with two groups of three elements and one of four. Even in these simple examples it is seen that perceiving cluster structure is a cognitive event influenced by many parameters. The concept of cluster is inherently vague, and much to the chagrin of mathematicians, the situation is more or the less the same in mathematics.

Figure 32. Regularity and balancedness competing.

2 Aims and ends I

Cluster analysis can be described as the study of characterization and recognition of close groups in a class of entities. The study of recognition means the study of methods that label entities as belonging to the same group or to different groups. A *clustering* is such a labelling or division of the entities into groups, and a *method* is a recipe which, if

followed, produces a clustering if one is given a class of entities and a notion of similarity (closeness) or distance between those entities. Usually, recipes are called *algorithms*. With this terminology settled, let us try and see if it is useful, and why the problems in cluster analysis are interesting and difficult.

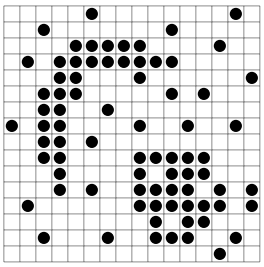


Figure 33. Image A.

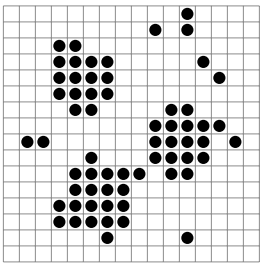


Figure 34. Image B.

What is the use of finding clusterings? One use is as a means of creating a *classification* of the entities by partitioning them into groups. The classification then corresponds with 'the big picture'; it means that structure is attached to some source of information or observations on entities. This is perhaps best explained by looking at what it takes to equip a machine with a simple form of vision. Images A and B in Figure 33 represent camera generated pictures which are sent to a computer, that has to take some action (follow some recipe) depending on how many objects are present in a picture. For interpreting a picture, it needs another recipe, because in the picture itself there is no information about objects, there is only a grid of boxes (called pixels) which may be black or white. Thus, the entities in this case are black pixels, and a close group of black pixels represents some object. Clustering amounts to recognizing higher level entities (shapes or objects) from lower level entities (black pixels).

Animals, c.q. hominides, are very good at extracting patterns from this kind of image. In fact, *they see nothing but structure*, so it is hard to recognize the difficulty of the task. First, it should be stressed that the task is not to find a way of analysing a particular picture, but to find a method that can be used for analysing an enormous range of possible pictures. Second, the images look deceptively simple to us because of our cognitive skills. What if one is asked to recognize 'objects' in the following array of pairs of numbers?

(0 7)	(2 14)	(3 11)	(5 11)	(8 4)	(9 12)	(10 12)	(12 5)	(14 0)
(1 3)	(3 3)	(4 2)	(6 2)	(8 7)	(9 13)	(10 14)	(12 9)	(14 7)
(1 12)	(3 4)	(4 3)	(6 3)	(8 9)	(9 14)	(11 7)	(12 10)	(14 14)
(2 1)	(3 5)	(4 4)	(6 6)	(8 10)	(10 1)	(11 9)	(12 12)	(15 4)
(2 5)	(3 6)	(4 5)	(6 14)	(8 11)	(10 3)	(11 10)	(12 13)	(15 11)
(2 6)	(3 7)	(5 0)	(7 2)	(8 12)	(10 5)	(11 11)	(13 2)	(15 12)
(2 7)	(3 8)	(5 2)	(7 3)	(9 3)	(10 9)	(11 12)	(13 11)	
(2 8)	(3 9)	(5 3)	(8 2)	(9 9)	(10 10)	(11 13)	(13 12)	
(2 9)	(3 10)	(5 8)	(8 3)	(9 11)	(10 11)	(11 14)	(13 15)	

This mess of numbers contains exactly the same information[2] as image A, and is in fact essentially the only way that an image can be stored in a computer. In designing a recipe one is forced to create a list of instructions that can be applied to arbitrary lists of numbers.

Suppose it is known in advance that the image may contain any number of objects between 0 and 5, and that an object is never hiding part of another object, but that objects might be located close to each other. That is at least something; this knowledge can be used and incorporated into the recipe that the computer is going to follow in deciding in how many objects there are. Now, it is not hard at all to design a recipe which works for the picture in Figure 33. But that is not what is required: a recipe is needed which works well for all different kinds of pictures that can be sent to the computer. There may be *noise* present in the picture, like the many loose black pixels in image A, the objects may have different sizes, shapes can be long and stretched or compact, bent or straight, and one shape can posses all of these characteristics simultaneously. Two shapes close together can be hard to distinguish from one long bent shape and noise may further cloud the issue.

The previous examples illustrate some of the typical challenges in cluster analysis. Devising a good clustering method for this kind of problem is at least as difficult as answering how the eye and mind perceive cluster structure. In cases where the complexity of a problem can be visualized, people, scientists included, expect the results of cluster methods to match their own interpretation. Now on the one hand the perception of cluster structure is influenced by changes in context, but on the other hand the perception of cluster structure may equally wel lead to a perceived change in context. Perception of cluster structure has to do with interaction of low–level and high–level cognitive functions, and such interaction is very difficult to catch in a recipe.

The general rule of method design applies that the more restricted the problem area is (i.e. the less uncertainty there is about possible contexts), the easier it is to design methods excelling in this area. Different methods that are respectively optimized for recognizing different kinds of images such as in Figure 31 and Figure 33 will perform not as well on other kinds of images, and methods that are widely applicable are unlikely to excel everywhere. The best thing possible is to have a method that can be easily tuned to different contexts.

3 Aims and ends II

A second use of finding clusters lies in cutting apart a structure such that it stays intact as much as possible. That sounds funny, so consider Figure 35. Twelve cities are pictured as little circles, labelled $1, \ldots 12$, and a number of roads connects the cities. Suppose that the cities should be grouped into different regions, where each region gets its own road maintenance department. One region would be inefficient (for reasons unknown to us), and if there are more regions then the number of roads between regions should

[2] e.g. the $(7, 2)$ pair says that starting from the upper left pixel, a black pixel is found when going 7 pixels to the right and 2 to below.

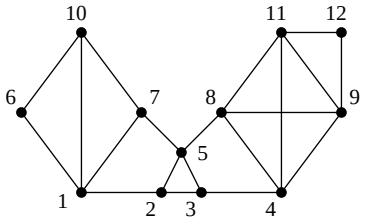


Figure 35. Cities and roads.

be minimal (in order to minimize the chance of colliding maintenance teams). A good candidate for such a grouping or clustering is $\{1, 6, 7, 10\}$, $\{2, 3, 5\}$, $\{4, 8, 9, 11, 12\}$. The main contribution of this thesis is a powerful new method, *the Markov Cluster Algorithm*, that is specifically suited to attacking this kind of problem — where the number of dots may be orders of magnitudes larger.

There are many real-life domains which can be modelled as a structure like the one in Figure 35, that is, as a combination of dots and lines. These structures are called *graphs* and they play a role whenever there is a notion of objects (or entities) being connected or not. Entities are then called *nodes*, and lines are called *edges* or *links*. The most appealing and largest in scale examples are the Internet and the World Wide Web. The Internet consists of millions of computers which are linked via intricate networks. The World Wide Web consists of an ever growing amount of web pages[3], which refer to each other in myriads of ways via the so called hyperlinks. Diving into the computer itself, circuits in chips consist of transistors (nodes), wired together (links). The list of examples is virtually without limit. Take people as entities, and say there is a connection between two people if they ever shook hands. This is a graph where the dots are people and a line corresponds with (at least) one handshake. It is a theory or legend of some kind that there are on average only six handshakes between two arbitrary different people on earth[4]. Note that in this example the link does not correspond with a connection that also resembles a physical distance, contrary to the cities and roads example from Figure 35. However, this figure is an abstract picture; it could equally well represent twelve people, the lines telling which pairs of people ever shook hands.

Many different kinds of connections between people can be considered; e.g. being relatives of the first (or second, third, ...) degree; both having a parent born in the same town; having worked for the same company. In the scientific community, one may define being connected as having co-authored an article. Many mathematicians know their Erdős number, which is the distance from the famous and prolific mathematician Paul Erdős[5]. If mathematician P has co-authored an article with Q, where Q has co-authored an article with Erdős, then Q has Erdős number 1, and P has Erdős number 2.

[3] Currently probably within the billions.

[4] This may seem improbable, but giving it some thought should at least refute the spontaneous disbelief. The origin of this conjecture is unknown to me.

[5] Paul Erdős wrote more than 1500 articles in his career, many of which were jointly written with other mathematicians.

Another graph constructed from science takes as nodes, say, all mathematical articles ever written; two articles are connected in this graph if the subject lists describing their content share one or more common elements. Such a graph is in a sense a map of the whole of mathematics, and it is interesting to investigate properties of this map. One idea is that articles may be related to each other without sharing any element of their subject lists, perhaps because the same subject is known under different wordings. It is then natural to assume that the articles must still be close to each other on the map previously introduced.

Finally it should be noted that the examples in this chapter were chosen because they have some visual appeal and a low degree of complexity. In the vector model, for slightly more complex input data, the problems can no longer be visualized such that an observer or practitioner can test methods against her own intuition. This applies even more to the graph model; graphs with dozens of nodes and hundreds of links already yield a picture of an inextricably entangled web.

4 Perspectives

Cluster analysis, aiming at dissecting structures such that they stay intact as much as possible, yields a relatively new perspective, inspired by the increasing number of phenomena that can be modelled as graphs (structures with nodes and links), such as computer networks and computer chips, databases, document collections, c.q. web pages, et cetera. In this thesis I argue that there is a subtle but significant difference with the classic applications of cluster analysis, which are better described by the notion of *vector models* rather than graphs. This notion will be introduced in the course of a short account of cluster analysis history.

The early origins of cluster analysis are found in the classification of populations in species[6] in biology, a discipline which is commonly known as *taxonomy*. Aristotle was already herein engaged (writing the book *Scala Naturae*, i.e. scale of nature), and Carl Linnaeus is its most famous contributor. In taxonomy, the entities are different populations of animals, and observations on how different populations differ in their characteristics establish a notion of similarity (or conversely, distance) between them. The characteristics chosen by current taxonomists vary among others from morphological attributes (e.g. skeleton or bone related measures like type, curvature, weight, measures on fur, feather, teeth, digestive system and so on) to ecological and geographical data describing the habitat of populations. Essentially, populations are described by their number scores on the chosen characteristics, and two populations are 'close' if their respective scores are close. This picture is far from complete, as taxonomists take other factors into account, such as the ability between populations to produce fertile offspring, the extent of DNA hybridization, and the number and degree of pairing between chromosomes (quoted from [**95**], page 133). Sticking to the simple model, a list of numbers (here characterizing a population) is in mathematics called a *vector*, hence the phrase vector model.

[6] And, following species, genera, families, classes, orders, phyla.

In the 20^{th} century, taxonomists began to seek objective, unbiased methods using such numerical characteristics of individuals and populations (see [**155**], page 13). This research is collectively labelled *numerical taxonomy*. The division between method and application, i.e. formulation of methods in such abstract terms that they can be applied to behavioural, biological, or any other kind of data, eventually lead to the recognition of *cluster analysis* as a research area of its own, where generic clustering and classification methods are studied not tied to any particular context. Taxonomists did not get rid of the issue of objectivity however, because it turns out that different methods and different ways of preparing the data yield different results, and that it is impossible to establish that one method is better than another, except in very specific cases.

Following taxonomy, subsequent applications of cluster analysis are still best described by the vector model. These include the grouping of medical records in order to find patterns in symptoms and diseases related to characteristics of patients, and the analysis of behavioural and sociological data for similar purposes. In the first case the data for each entity (a medical record or patient) is a set of scores on symptoms, body characteristics, or a combination of both, in the second case the entities are either people or populations (e.g. cities), and the scores can pertain to economic status, education, crime, health, or any other sociological phenomenon of interest. Everitt lists several of such applications ([**54**], page 8).

The difference between vector and graph settings is one of genuinely different types of topology. In both settings there is a notion of distance or similarity between entities, but they are conceived in different ways. In the vector model, the distance between two entities is derived by comparing a set of scores, and calculating a number which represents the distance between the two entities. The vector model can be applied to the entities in images A and B, which are the black pixels. Their 'scores' are just their coordinates, and the distance between the black pixels $(7, 3)$ and $(2, 1)$ in image A is then for example calculated as the horizontal distance plus the vertical distance, amounting to $5 + 2 = 7$.

In the graph model, there are the two notions of a) being connected or not and b) longer distance paths going from one entity to another, notions lacking in the vector model. The Markov Cluster algorithm was inspired by the implications of the *path* notion for properties of clusters in the settings of graphs. It hardly could have been conceived in the classic setting of vector models, and experiments indicate that it is very powerful particularly in the setting of graphs. Still, the vector model and the graph model have so many resemblances that it is easy to try and apply the Markov Cluster algorithm to vector models. This honours a good engineering and scientific principle that theories, methods, designs, and even machines should always be tried to the limit of what is possible. By doing so, the practitioner learns about the strengths and weaknesses of that what is tested, and it may well lead to new insights. This is also the case for the Markov Cluster algorithm. It is applied to (highly abstract variants of) images such as in Figures 33 and 34. For some cases it works well, and for others it does not, and it can be explained and predicted for which classes this is the case. This leads to a shift of perspective as it is shown that the *MCL* process can be put to a different use, namely that of border detection in images. These issues are discussed in Chapter 10.

5 Words of history and history of words

Since this thesis is all about a recipe called the *Markov Cluster Algorithm*, some explanatory words may be of interest.

1. *Algorithm* has the meaning of 'recipe', a set of instructions for the purpose of achieving some goal. The man who fathered this word did not live to know that he did. Ja'far Mohammed Ben Musa lived at the court of the caliphs of Bagdad, and was also known as al–Khowarazmi (also often transliterated as al–Chwarizmi), meaning 'the man from Khwarazm'. Around the year 825 he wrote an arithmetic book explaining how to use (i.e. giving methods or recipes) the Hindu–Arabic numerals[7]. This book was later translated with the Latin title *Liber Algorismi*, meaning 'Book of al–Khowarazmi'. Schwartzman writes in [**148**]: "The current form *algorithm* exhibits what the *Oxford English Dictionary* calls a 'pseudo-etymological perversion': it got confused with the word *arithmetic* (which was one of its meanings, and which has several letters in common with it); the result was the current *algorithm*."

An algorithm is something which can be programmed on a computer, that is, the computer can carry out the instructions on the recipe. The most important part of cluster analysis is cooking up good recipes, and understanding why certain ingredients and procedures work well together, and why others fail to do so. It should be noted that all the hard work is done outside of the computer; humans have to supply the recipes, the ingredients, and the cooking equipment. The computer is just a wired together piece of junk[8] that does exactly what the recipe tells it to do, using the ingredients and equipment that come with the recipe.

2. By pulling himself up by his bootstraps *Baron von Münchhausen* fathered the word *bootstrapping*. In science its abstract meaning is to derive high–level structural descriptions from low–level data without using (a lot of) a priori knowledge. The origin suggests that bootstrapping problems require some miraculous feat. However, the dull truth is that no method solves bootstrapping problems entirely satisfactory, which is exactly what makes them so interesting. The phrase can be used in sentences like *The basic problem in cluster analysis is that of bootstrapping*, or *Building a sophisticated software environment requires a lot of bootstrapping*, and perhaps *Life is the mother of all bootstraps.*

3. *Botryology* is an obscure term which was meant as a dignified label for the discipline of cluster analysis, meaning 'the theory of clusters'. It is formed from the Greek $\beta o \tau \rho \upsilon \sigma$, meaning 'a cluster of grapes'. In the article 'The Botryology of Botryology' the author I.J. Good puts in an heroic effort to lift the term into mainstream usage, though his argumentation seems somewhat humorous ([**68**], page 73):

[7] Source: [**148**], page 21.

[8] Of course, computer manufacturers put a lot of hard work in wiring the junk such that it can carry out very large recipes very swiftly.

> It seems to me that the subject of clustering is now wide enough and respectable enough to deserve a name like those of other disciplines, and the existence of such a name enables one to form adjectives and so on. For example, one can use expressions such as "a botryological analysis" or "a well-known botryologist said so and so". There is another word that serves much the same purpose, namely "taxonomy", but this usually refers to biological applications whereas "botryology" is intended to refer to the entire field, provided that mathematical methods are used. The subject is so large that it might not be long before there are professors and departments of botryology. Another possible name would be aciniformics, but it sounds inelegant. On the other hand, "agminatics" is a good contender, forming "agminaticist", etc.

It is an example of scientific word usage that never quite made it. Occasionally, the term still surfaces though, as in the title of [**163**]. In my mind I.J. Good will always be well-known as a botryologist, and as one who profoundly appreciates the aesthetic aspects of word usage. He gives several references to earlier uses of the word. The earliest reference is an article written by himself, which may indicate that he fathered the word. It is left as an exercise for the reader to find the etymology of the constructions *aciniformics* and *agminatics*.

Perhaps it is a witness to the fragmented origin of cluster analysis, i.e. its simultaneous growth in different disciplines, that a wealth of labels has been associated with it. Hartigan gives the following list in [**79**], page 1: *numerical taxonomy, taximetrics, taxonorics, morphometrics, botryology, nosology, nosography, and systematics*.

4. *Cluster* is related to the Low German *kluster*, and the Old English *clott*, meaning lump or mass, courtesy of Webster's dictionary. Webster's concludes its etymological summary of cluster with 'more at CLOT'. Looking up 'clot' yields the related Middle High German *kloz*, meaning lumpy mass or ball, *and* the indirection 'more at CLOUT'. Then 'clout' yields among others Russian *gluda* or lump, Latin *galla* (gall-nut), and the reference 'more at GALL'. The word *gall* is 'perhaps akin to Greek *ganglion* cystic tumor, mass of nerve tissue, [and] Sanskrit *glau* round lump; basic meaning: ball, rounded object'. The intricate ways of language and dictionaries! This gives yet another example of graph structure in daily life; the nodes or entities are dictionary entries and the links are the cross-references between them. Linguists study properties of this type of structure.

5. *Andrei Andreyevich Markov* was a famous Russian mathematician, born 14 June 1856 in Ryazan, died 20 July 1922 in St Petersburg. He is best known for his contributions to probability theory. His name is connected to many mathematical notions, among them *Markov chain, Markov matrix, Markov moment, and Markov process.* It should be noted that such attributions are made by other mathematicians, usually after the principal contributor's work has gained widespread acceptance. A Markov matrix is a special kind of matrix (see below) which has the property that it is nonnegative, i.e. all its elements are greater than or equal to zero, and that all its columns (or rows, depending on the chosen orientation) sum to 1. This thesis rests mainly on two pillars; Markov theory, and the theory of nonnegative matrices, for which Markov theory formed a thriving source of inspiration.

In the mathematical landscape, this thesis is situated somewhere near the disciplines of *the study of nonnegative matrices* and *matrix analysis*. Many concepts are named after people who made profound contributions, concepts such as Geršgorin discs, Hermitian matrices, the Hadamard–Schur product, the Jordan canonical form, the Kronecker product, Perron–Frobenius theory, and the Schur product theorem.

6. A *matrix* is the mathematical object which lies at the heart of the Markov Cluster algorithm, in particular the matrix subspecies *Markov matrix*. A matrix is a rectangular array of entities that are usually just numbers or indeterminates. Examples of matrices can be found on pages 50, 53, and 66. The word matrix has a rather impressive heritage and several meanings. Most of them refer to something in which the evolution of new life or substance is embedded; the word is etymologically related to *mater* or mother. Examples of this are the meanings *uterus* or *womb*, *cradle*, and *mould* (note: the Dutch word for mould is *matrijs*, which is etymologically very close to matrix). Webster's dictionary[9] gives the example 'an atmosphere of understanding and friendliness that is the matrix of peace'. This generic meaning must be certainly what inspired the makers of the 1999 Warner Bros science-fiction action movie *The Matrix* in titling their creation, in which robots running amok have subjected mankind and cast nearly all humans into artificial wombs. That is not even the worst part, as the humans are wired into a computer, which causes them to believe that they are leading a normal live. Thus, bodies are grown, supported, and constrained by a matrix in the form of artificial wombs, and the mind is similarly treated by a computer–created virtual matrix. In mathematics however, matrices are very likable beasts, which are used in many different disciplines to great avail.

Schwartzman ([**148**], page 132) names two possible entry points for the word *matrix* in mathematics. However, Jeff Miller gives a much clearer explanation[10] and even cites the person who actually introduced the word, James Joseph Sylvester (1814–1897). Schwartzman's first remark is that mathematically speaking a matrix *generates* geometric or algebraic transformations. The second is that matrices are arrays of numbers *surrounded* by large brackets or parentheses. In both cases the meaning of the verb refers to womb or matrix-like qualities, and Schwartzman suggests that these congruences lead to the use of the word matrix. Sylvester himself has the following to say ([**15**], page 150):

> This homaloidal law has not been stated in the above commentary in its form of greatest generality. For this purpose we must commence, not with a square, but with an oblong arrangement of terms consisting, suppose, of m lines and n columns. This will not in itself represent a determinant, but is, as it were, a Matrix out of which we may form various systems of determinants by fixing upon a number p, and selecting at will p lines and p columns, the squares corresponding to which may be termed determinants of order p.

Miller gives the following citation of Kline, found in [**107**], page 804:

---

[9] Webster's Third New International Dictionary, 1971.

[10] Source: http://members.aol.com/~jeff570/mathword.html.

> The word matrix was first used by Sylvester when in fact he wished to refer to a rectangular array of numbers and could not use the word determinant, though he was at that time concerned only with the determinants that could be formed from the elements of the rectangular array.

It is clearly the determinant–generating ability that inspired Sylvester.

7. *Walk, random.* A random walk is a walk that is governed by the flipping of a coin, the rolling of a dice, or more generally by any event for which the outcome is a priori uncertain. Suppose you are walking in a city, and each time you arrive at a crossing you flip two coins, one after another. There are four possible outcomes, writing H for heads and T for tails. and you decide to: turn left if the outcome is TH (the first coin is T, the second is H), go straight on if the outcome is TT, turn right if it is HT, and turn around if the outcome is HH. This is a good example of a random walk. One important aspect is that if you start two or more times from the same departure point, then the resulting walks will in general be different; a random walk cannot be predicted. Still, a lot of things can be said about the *probability* that certain things happen. For example, one may ask what the chances are that you return to the point from which you departed after visiting a hundred crossings, or what the chances are that you have returned at least one time. What are the odds that you visit only different crossings, or that you visit no more than fifty different crossings? If ten thousand people start a random walk from the same point, how far will they be away from the departure point after a hundred steps, on average? One random walk is not predictable, but if you combine very many of them, then it is often possible to make surprisingly strong statements about them. The concept of a random walk is a very rich source for scientific research, because many questions about them (such as posed here) can be answered using mathematical tools such as Markov matrices, and because many real-world phenomena are well described by the concept of a random walk.

The Dutch physicist and publicist Ad Lagendijk dedicated an entire column to the random walk in *De Volksrant* d.d. Saturday 11 December 1999. The column is titled[11] *Walk of the century*. Lagendijk lists several phenomena which can be described using random walks: the transport of fluids through porous media (e.g. oil through rock), the diffusion of molecules in gas mixtures or chemical solutions, the way in which people navigate supermarkets and large stores, and the transport of molecules through the cells of organisms. He argues that the concept of a random walk deserves to be called the biggest scientific discovery of the 20^{th} century, because of the power and the fundamental nature of the concept, its wide applicability, its common use in many different scientific disciplines, and because it has become part of scientific mainstream to the extent that scientists use it even unconsciously.

The *MCL* process utilizes random walks for the retrieval of cluster structure in graphs. It is described in some more detail in the next section. Perhaps confusingly, the word *flow* is also used throughout this thesis to describe the working of the *MCL* process. Obviously one associates flow with a good swim rather than a random walk, at least for ordinary mortals. The right perspective is found by pairing the concept of flow with a

[11] The original title is *Wandeling van de eeuw*.

vast collection of random walks. Picture ten thousand people each starting their own random walk from the same departure point. An observer floating high above them will see the crowd slowly swirling and dispersing, much as if a drop of ink is spilled into a water–filled tray.

8. The Markov Cluster Method is *Yet Another* Cluster Method, in the sense 1 as listed below. The Jargon File 4.0 has this to say[12] about the qualifier *yet another*, which is an example of popular language from the world of computers and computer science.

> Yet Another: /adj./ [From Unix's 'yacc(1)'. 'Yet Another Compiler-Compiler', a LALR parser generator] 1. Of your own work: A humorous allusion often used in titles to acknowledge that the topic is not original, though the content is. As in 'Yet Another AI Group' or 'Yet Another Simulated Annealing Algorithm'. 2. Of others' work: Describes something of which there are already far too many.

6 The Markov Cluster algorithm

The main contributions in this thesis are centred around a new method in cluster analysis, which I named the *Markov Cluster algorithm*, abbreviated as *MCL* algorithm. As stated before, the algorithm is specifically suited to graph structures such as in Figure 35. The *MCL* algorithm is inspired by a simple yet powerful idea. The aim of a cluster method is to dissect a graph into regions with many edges inside, and with only few edges between regions. A different way of putting this is that if such regions exist, then if inside one of them, it is relatively difficult to get out, because there are many links within the region itself, and only a few going out. The idea is now to simulate random walks or flow within the whole structure, and to further strengthen flow where the current is already strong, and to weaken flow where the current is weak. In different wordings, random walks are promoted, e.g. by broadening the pavement, where the number of pedestrians (i.e. current) is already high, and random walks are demoted where this number is low, e.g. by narrowing the pavement. The hypothesis supporting this idea is that cluster structure corresponds with regions of strong current (many random walks pass through), separated by borders where the current is weak (relatively few random walks pass through). If this idea is put to the test with the right tools, it turns out to work. Flow can be manipulated in this way such that it eventually stabilizes, where most of the flow weakens to such a large extent that it actually dries up. Different regions of constant flow remain which are separated by dry borders; these regions can be sensibly interpreted as clusterings. A symbolic picture representing such a situation for the graph in Figure 35 is seen in Figure 36, and a sequence of pictures representing different stages of flow is found on page 7. In these pictures the grey level of a node indicates how many random walks pass through at a given stage: the darker the node, the more walks.

[12] An interesting web resource edited by famous Open Source advocate Eric S. Raymond, found at http://www.tuxedo.org/~esr/jargon/. Its sibling the *Free On–line Dictionary Of Computing*, http://foldoc.doc.ic.ac.uk/, edited by Denis Howe, is also noteworthy.

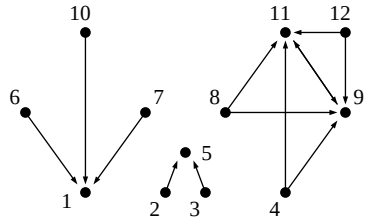


Figure 36. Regions of flow.

Markov theory supplies almost everything needed to manipulate flow as described above. Flow simulation can be done by taking a suitable Markov matrix, and computing powers of this matrix. This is the well-known concept of a discrete Markov chain. The only thing lacking is the strengthening and weakening of flow. That part is supplied by inserting a new operator in the Markov chain, which can be described in terms of the so called *Hadamard–Schur* product. In effect the *MCL* algorithm draws upon two well-developed disciplines of mathematics, and this crossbreeding yields fertile offspring.

The beauty of the *MCL* algorithm is that the method is not actively engaged in finding clusters. Instead, it simulates a process which is inherently affected by any cluster structure present. The cluster structure leaves it marks on the process, and carrying the process through eventually shows the full cluster structure. The process parameters can be varied, i.e. the flow can be strengthened and weakened to a greater or lesser extent. This parametrization affects the scale on which the cluster structure leaves its marks. It is shown in Chapter 7 that the process (in particular, the matrices created in it) has mathematical properties which have a straightforward interpretation in terms of cluster structure. These results are of particular interest, as it is for the first time that cluster structure is found via and within a simple algebraic process.

The *MCL* algorithm generates many new research questions. In this thesis a few of them are answered, and these answers help in gaining insight in the algebraic process employed by the algorithm. Mathematics is, contrary to common belief, an ever changing field of research where results lead to new questions and questions lead to new results.

Samenvatting

Dit proefschrift heeft als onderwerp het clusteren van grafen door middel van simulatie van stroming, een probleem dat in zijn algemeenheid behoort tot het gebied der clusteranalyse. In deze tak van wetenschap ontwerpt en onderzoekt men methoden die gegeven bepaalde data een onderverdeling in groepen genereren, waarbij het oogmerk is een onderverdeling in groepen te vinden die *natuurlijk* is. Dat wil zeggen dat verschillende data-elementen in dezelfde groep idealiter veel op elkaar lijken, en dat data-elementen uit verschillende groepen idealiter veel van elkaar verschillen. Soms ontbreken zulke groepjes helemaal; dan is er weinig patroon te herkennen in de data. Het idee is dat de aanwezigheid van natuurlijke groepjes het mogelijk maakt de data te categoriseren. Een voorbeeld is het clusteren van gegevens (over symptomen of lichaamskarakteristieken) van patienten die aan dezelfde ziekte lijden. Als er duidelijke groepjes bestaan in die gegevens, kan dit tot extra inzicht leiden in de ziekte. Clusteranalyse kan aldus gebruikt worden voor *exploratief onderzoek*. Verdere voorbeelden komen uit de scheikunde, taxonomie, psychiatrie, archeologie, marktonderzoek en nog vele andere disicplines. Taxonomie, de studie van de classificatie van organismen, heeft een rijke geschiedenis beginnend bij Aristoteles en culminerend in de werken van Linnaeus. In feite kan de clusteranalyse gezien worden als het resultaat van een steeds meer systematische en abstracte studie van de diverse methoden ontworpen in verschillende toepassingsgebieden, waarbij methode zowel wordt gescheiden van data en toepassingsgebied als van berekeningswijze.

In de cluster analyse kunnen grofweg twee richtingen onderscheiden worden, naargelang het type data dat geclassificeerd moet worden. De data-elementen in het voorbeeld hierboven worden beschreven door vectoren (lijstjes van scores of metingen), en het verschil tussen twee elementen wordt bepaald door het verschil van de vectoren. Deze dissertatie betreft cluster analyse toegepast op data van het type 'graaf'. Voorbeelden komen uit de patroonherkenning, het computer–ondersteund ontwerpen, databases voorzien van hyperlinks en het World Wide Web. In al deze gevallen is er sprake van 'punten' die verbonden zijn of niet. Een stelsel van punten samen met hun verbindingen heet een graaf. Een goede clustering van een graaf deelt de punten op in groepjes zodanig dat er weinig verbindingen lopen tussen (punten uit) verschillende groepjes en er veel verbindingen zijn in elk groepje afzonderlijk. Het eerste deel van de dissertatie, bestaande uit de hoofdstukken 2 en 3, behandelt de positie van clusteranalyse in het algemeen en de positie van graafclusteren binnen de clusteranalyse in het bijzonder, alsmede de relatie van graafclusteren tot het aanverwante probleem van het *partitioneren* van grafen. In het cluster probleem zoekt men een 'natuurlijke' onderverdeling in groepjes en is het aantal en formaat van de groepjes niet voorgeschreven. In het partitie probleem zijn aantal en afmetingen wel voorgeschreven en zoekt men gegeven deze restricties een toewijzing van de elementen aan de groepjes zodanig dat er een minimale hoeveelheid verbindingen tussen de groepjes is.

De dissertatie beschrijft voorts theorie, implementatie en abstracte toetsing van een krachtig nieuw cluster algoritme voor grafen genaamd Markov Cluster algoritme of *MCL* algoritme. Het algoritme maakt gebruik van (en is in feite niet meer dan een schil om) een algebraisch proces (genaamd *MCL* proces) gedefinieerd voor Markov grafen, i.e. grafen waarvoor de geassocieerde matrix stochastisch is. In dit proces wordt de aanvangsgraaf successievelijk getransformeerd door alternatie van de twee operatoren *expansie* en *inflatie*. Expansie is het nemen van de macht van een matrix volgens het klassieke matrix product. Stochastisch gezien betekent dit het uitrekenen van de overgangskansen behorend bij een meerstapsrelatie. Inflatie valt samen met het nemen van de macht van een matrix volgens het elementsgewijze Hadamard–Schur product, gevolgd door een kolomsgewijze herschaling zodat het uiteindelijke resultaat weer een (kolom) stochastische matrix is. Dit is een ongebruikelijke operator in de wereld van de stochastiek; zijn introductie is geheel en al gemotiveerd door de beoogde werking op grafen waar clusterstructuur aanwezig is. Het is namelijk te verwachten dat bij meerstapsrelaties die corresponderen met puntparen liggend binnen een natuurlijke cluster grotere overgangskansen zullen horen dan bij puntparen waarvan de punten in verschillende clusters liggen. De inflatie operator bevoordeelt meerstapsrelaties met grote bijbehorende kans en benadeelt meerstapsrelaties met kleine bijbehorende kans. De verwachting is dus dat het *MCL* proces meerstapsrelaties zal creeeren en bestendigen die horen bij relaties liggend in één cluster, en dat het alle meerstapsrelaties zal decimeren die behoren bij relaties tussen verschillende clusters. Dit blijkt inderdaad het geval te zijn. Het *MCL* proces convergeert over het algemeen naar een idempotente matrix die zeer ijl is en bestaat uit meerdere componenten. De componenten worden geïnterpreteerd als een clustering van de aanvangsgraaf. Doordat de inflatie operator geparametrizeerd is kunnen clusteringen op verschillend niveau van granulariteit ontdekt worden.

Het *MCL* algoritme bestaat ten eerste uit een transformatiestap van een gegeven graaf naar een stochastische aanvangsgraaf, gebruik makend van het standaard concept van een willekeurige wandeling op een graaf. Ten tweede vergt het de specificatie van twee rijen van waarden die de opeenvolgende expansie en inflatie parametrizeringen definieeren. Tenslotte berekent het algoritme het bijbehorende proces en interpreteert het de resulterende limiet. Het idee om willekeurige wandelingen te gebruiken om clusterstructuur te ontdekken is niet nieuw, maar de wijze van uitvoering wel. Het idee wordt als 'graafcluster paradigma' geïntroduceerd in hoofdstuk 5, gevolgd door enige combinatorische voorstellen tot het clusteren van grafen. Getoond wordt dat er een verband is tussen de combinatorische en probabilistische clustermethoden, en dat een belangrijk onderscheid de localisatiestap is die probabilistische methoden over het algemeen introduceren. Het hoofdstuk besluit met een voorbeeld van een *MCL* proces en de formele definitie van zowel proces als algoritme. Notaties en definities zijn dan reeds geïntroduceerd in hoofdstuk 4. In hoofdstuk 6 wordt de interpretatiefunctie van idempotente matrices naar clusteringen geformaliseerd, worden simpele eigenschappen van de inflatie operator beschreven, en wordt de stabiliteit van *MCL* limieten en de geassocieerde clusteringen geanalyseerd. Het fenomeen van overlappende clusters is in principe mogelijk[13] en maakt intrinsiek deel uit van de interpretatiefunctie, maar blijkt

[13] De tot nu toe waargenomen overlap van clusters correspondeerde altijd met een graafautomorfisme dat het overlappende deel van clusters op zichzelf afbeeldde.

instabiel te zijn. Hoofdstuk 7 introduceert de klassen van *diagonaal symmetrische* en *diagonaal positief semi-definiete* matrices (matrices die diagonaal gelijkvormig zijn met een symmetrische respectievelijk positief semi-definiete matrix). Beide klassen worden in zichzelf overgevoerd door zowel expansie als inflatie[14]. Getoond wordt dat diagonaal positief semi-definiete matrices structuur bevatten die de interpretatiefunctie van idempotente matrices naar clusteringen generaliseert. Hieruit volgt een preciezere duiding van het inflatoire effect van de inflatie–operator op het spectrum van de argumentmatrix. Ontkoppelingsaspecten van grafen en matrices zijn altijd nauw verbonden met karakteristieken van de geassocieerde spectra. Hoofdstuk 8 beschrijft een aantal bekende resultaten die ten grondslag liggen aan de meest gebruikte technieken ten behoeve van het partitioneren van grafen. De hoofdstukken 4 tot en met 8 vormen het tweede deel van de dissertatie.

Het derde deel doet verslag van experimenten met het *MCL* algoritme. Hoofdstuk 9 is theoretisch van aard en introduceert functies die gebruikt kunnen worden als maat voor de kwaliteit van een graafclustering. Ondermeer wordt een generieke maat afgeleid die uitdrukt hoe goed een karakteristieke vector de massa van een andere (niet negatieve) vector representeert. Elements– of kolomsgewijze toepassing van de maat geeft een uitdrukking voor de mate waarin een clustering de massa van een gewogen graaf of matrix representeert. Tevens wordt een metriek op de ruimte van clusteringen of partities afgeleid, die gebruikt wordt om de continuïteitseigenschappen en het onderscheidend vermogen van het *MCL* algoritme te toetsen in hoofdstuk 12. Hoofdstuk 10 doet verslag van experimenten op kleine symmetrische grafen met welbepaalde dichtheidskarakteristieken zoals rastervormige grafen. Het *MCL* algoritme blijkt — experimenteel — een sterk scheidend vermogen te hebben. Experimenten met buurgrafen[15] wijzen uit dat het algoritme niet geschikt is indien de diameter van de natuurlijke clusters groot is. Dit verschijnsel kan begrepen worden in termen van de (stochastische) stromingseigenschappen van het algoritme. Hoofdstuk 11 gaat in op de schaalbaarheid van het algoritme. Cruciaal is dat de limiet van het *MCL* proces over het algemeen zeer ijl is en dat de iteranden van het proces ijl zijn in een gewogen interpretatie van het begrip ijl. Dat wil zeggen, de inflatie operator zorgt ervoor dat de meeste nieuwe niet-nul elementen (corresponderend met meerstapsrelaties) zeer klein blijven en uiteindelijk weer verdwijnen. Dit is des te meer waar naarmate de diameter van de natuurlijke clusters klein is, en naarmate de connectiviteit van de totale graaf laag is. Dit suggereert dat tijdens elke expansie stap — die ervoor zorgt dat de matrix vol loopt — de kolommen van de nieuw berekende matrix uitgedund kunnen worden door simpelweg de k grootste elementen van een nieuw berekende (stochastische) kolom te nemen, en deze elementen te herschalen op 1, waar k afhangt van de aanwezige rekencapaciteit. Omdat het berekenen van de k grootste waarden van een vector in principe niet in lineaire tijd kan, blijkt het in praktijk noodzakelijk een verfijnder schema te hanteren waarin de vector eerst uitgedund wordt door middel van drempelwaardes die afhangen van homogeniteitseigenschappen van de vector. Dit leidt in principe tot een complexiteit in de orde van grootte $\mathcal{O}(Nk^2)$, waar N de dimensie van de matrix is. Hoofdstuk 12 doet verslag van

[14] Voor diagonaal positief semi-definiete matrices geldt dit voor slechts een deel van de parametrizeringsruimte van de inflatie operator.

[15] Rasterachtige grafen gedefinieerd op punten in de Euclidische ruimte.

experimenten op testgrafen met tienduizend punten waarvan de verbindingen op zo'n manier (willekeurig) zijn gegenereerd dat een a priori beste clustering bekend is. Deze grafen hebben natuurlijke clusters met kleine diameter maar hebben als geheel hoge tot zeer hoge connectiviteit. Het geschaalde *MCL* algoritme blijkt zeer goede clusteringen te genereren die dicht bij de a priori bekende clustering liggen. De parameter k kan laag gekozen worden, maar de prestaties van het algoritme nemen sterker af naarmate k lager is en de totale connectiviteit van de input graaf hoger. De appendix *A cluster miscellany* beginnend op pagina 149 is geschreven voor een algemeen publiek en bevat korte uiteenzettingen over diverse aspecten van clusteranalyse, zoals de geschiedenis van het vakgebied en de rol van de computer.

Acknowledgements

Through the enthusiasm and efforts of Jan van Eijck and Michiel Hazewinkel I obtained a PhD position at the CWI. This was on a project called *Concept building from key-phrases in scientific documents*, which was complementary to the twin project *Bottom up classification methods in mathematics*. When I found an opportunity to do interesting research in the latter framework, they encouraged me to pursue it. This act of trust was the decisive moment leading to this thesis. It seems evidently true that the exploratory nature of science, the charting of uncharted waters, requires the willingness to change course if the circumstances require so.

Nada Mitrovic was a cheerful neighbour during the first years in CWI's outbuilding, with its unique atmosphere of permanent temporariness in the beautiful green surroundings. Paul ten Hagen kept a wakeful eye throughout my assignment, and Zsofi Ruttkay and Marc Pauly were very pleasant roommates. The people at the CWI library were very helpful, especially Karin van Gemert and Hans Stoffel. A well equipped and smoothly functioning library is of inestimable value.

Jan van der Steen and Annius Groenink contributed significantly to the matrix section of the software used in conducting the experiments described in Part III of this thesis. Additionally, Jan taught me a lot about LaTeX, PostScript, and *C*. Generating figures by attaching PostScript output routines to the cluster software was a joyous experience and surely saved me from many annoyances.

Moniek was very supportive and enduring during the preparation of the manuscript, which was no small feat. My parents were supportive as ever. Several people proofread early versions of parts of the manuscript, notably Henk, Moniek, and Zsofi. For different reasons — my gratitude goes to all.

April 2000,
Amsterdam

Stijn van Dongen

Curriculum Vitae

The author was born in Drachten in the Province of Friesland, on 4 September 1969, as the son of Henk and Lenie van Dongen and the brother of elder sisters Diana and Iris. Drachten is part of the rural district called Smallingerland and this is the official place of birth[16]. After various moves across the country and a one-year intermezzo in Sant'Ambrogio, Italy, the family settled in Eindhoven in 1973. This is where the author finished secondary education, Gymnasium β, at the Van Maerlantlyceum in 1987. He then went to the Technical University in Eindhoven, where he completed the mathematics propaedeuse in 1989 and his cum laude master's degree in mathematics in 1994. His specialization was Discrete Mathematics and computer algebra in the setting of algebraic number theory. As a conscientious objector he was fortunate to perform his alternative community service at the *Centrum voor Wiskunde en Informatica* in Amsterdam, carrying out a variety of small programming jobs.

After this one-year assignment the author became a PhD student at the same institute, pursuing the subject *Concept Building from Key–Phrases in Scientific Documents*. In 1996 he devised a cluster algorithm for graphs, motivated by research in the twin project *Bottom Up Classification Methods*. This led quickly to the Markov Cluster Process and a shift of focus towards cluster analysis in the setting of graphs. During the course of this research he studied topics from the rich collection of disciplines known as *matrix analysis*. In 1998 the author was one of the founders of Eidetica, a company making text-mining software. The turmoil of this start-up in its first tottering steps, combined with the task of completing a PhD thesis, proved to be too taxing. Mid–1999 saw him leaving Eidetica to work full-time on the thesis which you are now reading. The author's favourite pastime is the game of go. He participated four times in the Dutch Go Championship, his best performance so far ranking 7^{th} in the 2000 edition.

[16] Because the name of Drachten is much better known than that of Smallingerland, voices are now heard to change the name of the district to Drachten. The author is opposed against this, as he likes the rythm and ring of 'Smallingerland' much better than that of 'Drachten'.