Dissertation
submitted to the
Combined Faculties for the Natural Sciences and Mathematics
of the Ruperto-Carola University of Heidelberg, Germany
for the degree of
Doctor of Natural Science

Presented by
Ir Frederik Roels
Born in Geel, Belgium
Oral examination: 30/9/2010

Analysis of differentiation trees using transcriptome data: application to hematopoiesis

Referees: Prof. Dr. Roland Eils
Prof. Dr. Rainer Haas

I would like to thank Professor Roland Eils and the department of theoretical bioinformatics at the DKFZ for providing the work environment and expertise needed to complete this project. Special thanks goes out to Doctor Benedikt Brors for being the man with an answer to every question.

I would like to thank Professor Rainer Haas and the department of hematology and oncology at the university clinic of Düsseldorf for providing me with the interesting, yet difficult to obtain, data that spawned the idea for this project.

Abstract

Cellular differentiation is a complicated and highly important system in all multicellular organisms. The remarkable aspect about differentiation is that the multitude of different and highly specialised cell types are all descendant from one cell, the zygote. Not surprisingly differentiation is a highly regulated process. A complicated interplay of environmental signals and intracellular regulation defines the ultimate mature state of all cell types.

In this work a method was developed that can analyse differentiation trees computationally. The development of the method was guided by three questions. Do microarrays contain enough information to retrace steps in differentiation? Can this information be used to validate proposed differentiation paths? Can this information be used to compare differentiation in different contexts?

The method starts from microarray data and uses a combination of methods to identify the most likely differentiation tree out of all possibilities. The method has two components, one component identifies the most likely conformation using a scoring system. The other component identifies the most likely root node using a comparison system. The conformation scoring system relies on transcriptional changes in previously defined subnetworks, all possible differentiation conformations are tested in a manner similar to maximum parsimony. Maximum parsimony is used in molecular phylogeny to score possible evolutionary trees, a problem similar to the one tackled in this work. Root node identification is done using a value calculated based on within cell type gene expression correlations, high values indicate the cell is less mature.

The method was tested on microarray data from the myeloid lineage of hematopoiesis. The datasets are comprised of expression data taken from four different cell types: Hematopoietic Stem Cells, Common Myeloid Progenitors, Granulocyte Monocyte Progenitors and Megakaryocyte Erythrocyte Progenitors. Data was gathered from healthy donors and patients suffering Chronic Myeloid Leukemia and Multiple Myeloma respectively.

The method performed well, in most cases the correct differentiation tree could be identified. This indicates that there is indeed enough information

present in microarray data to retrace differentiation. Interesting results where seen for the root node identification component. When analysing the dataset taken from patients with CML, the method predicted known differences in stemness in that particular cancer.

Zusammenfassung

Zelluläre Differenzierung ist ein kompliziertes und äusserst wichtiges System in allen multizellularen Organismen. Der bemerkenswerte Aspekt bei der Differenzierung ist, dass die Vielzahl an unterschiedlichen und enorm spezialisierten Zelltypen alle von einer Zelle abstammen, der Zygote. Es überrascht daher nicht, dass Differenzierung ein stark regulierter Prozess ist. Ein kompliziertes Zusammenspiel von umweltbedingten Signalen und intrazellulärer Regulierung definiert den endgültigen, vollentwickelten Zustand von allen Zelltypen.

In Rahmen dieser Arbeit wird ein Verfahre entwickelt, mit der Differenzierungsbäme programmatisch analysiert werden können. Die Entwicklung dieser Methode wurde von drei Hauptfragen bestimmt: Enthalten Microarrays genügend Informationen, um die Schritte der Differenzierung nachzuverfolgen? Können diese Informationen verwendet werden, um vorgeschlagene Differenzierungs-Wege zu validieren? Können diese Informationen verwendet werden, um Differenzierung in verschiedenen Kontexten miteinander zu vergleichen?

Das im Rahmen dieser Arbeit entwickelte Verfahren verarbeitet Microarray Daten zu einem Differenzierungsbaum, indem es aus allen möglichen den wahrscheinlichsten Differenzierungsbaum ermittelt. Die Transformation der Daten wird im wesentlichen von zwei Komponenten bernommen: Eine Komponente identifiziert die wahrscheinlichste übereinstimmung basierend auf einem Bewertungssystem. Die andere bestimmt den wahrscheinlichsten Wurzelknoten des Differenzierungsbaums durch ein Vergleichssystem. Das Conformation Scoring System bzw. das Bewertungssystem für Übereinstimmungen beruht auf transkriptionellen Änderungen in vorher definierten Subnetzwerken, in denen auf mögliche bereinstimmungen bei der Differenzierung getestet wird, ähnlich wie bei Maximum-Parsimony. Maximum-Parsimony wird im Bereich der molekularen Phylogenie eingesetzt, um die Wahrscheinlichkeit von Stammbäumen zu bewerten, einer Problemstellung, die der in dieser Arbeit besprochenen Problematik sehr ähnlich ist. Die Identifizierung des Wurzelknotens basiert auf einem Wert, der mithilfe der Korrelation von Genexpressionen innerhalb eines Zelltyps berechnet wird. Ein hoher Wert

deutet darauf hin, dass die Zelle noch nicht voll entwickelt ist.

Das Verfahren wurde mit Microarray Daten von hämatopoetischen Zellen der myeloischen Linien getestet. Die Dateien bestehen aus Expressionsdaten, die von vier verschiedenen Zelltypen stammen: hämatopoetischen Stammzellen, Common Myeloid Progenitors, Granulocyte-Monocyte Progenitors and Megakaryocyte-Erythrocyte Progenitors. Die Daten stammen sowohl von gesunden Spendern als auch von Patienten, die an chronischer myeloischer Leukmie (CML) erkrankt sind.

Das Verfahren arbeitete erfolgreich und führte in den meisten Fällen zur Bestimmung des korrekten Differenzierungsbaums. Dies ist ein Indikator dafür, dass Microarray Daten genügend Informationen enthalten, um die Schritte der Differenzierung nachzuverfolgen. Die Komponente zur Identifizierung des Wurzelknotens lieferte besonders interessante Resultate. Bei der Analyse von Datenstzen, die von Patienten mit CML stammen, konnten mithilfe des Verfahrens bekannte Unterschiede in der Stemness dieser Krebsform vorausgesagt werden.

Contents

Chapter 1

Introduction

Cellular differentiation is a complicated and highly important system in all multicellular organisms. The remarkable aspect about differentiation is that the multitude of different and highly specialised cell types are all descendant from one cell, the zygote. Not surprisingly, differentiation is a highly regulated process. A complicated interplay of environmental signals and intracellular regulation defines the ultimate mature state of all cell types. It is clear that errors in this system could lead to disastrous effects. Indeed, such defects may be the underlying cause of some cancers. Although the system is of high importance, a lot of questions remain open.

Research is complicated by several factors. The first issue is that the percentage of stem cells is usually quite small in comparison to that of fully differentiated cells, making isolation and identification troublesome. Up to now, the most studied lineage stem cell is the hematopoietic stem cell. Therefore the system that is generally the most studied in regard to differentiation is the hematopoietic system. Another problem comes from the fact that it is difficult to follow differentiation in vivo while it may not be possible to correctly simulate differentiation in vitro. As will be more explained in the section below, cellular differentiation does not occur in a vacuum. Interaction with surrounding cells provides important guidance throughout the differentiation process. The complexity of these interactions complicates in vitro studies. It may be possible to induce differentiation in vitro but, due to the lack of intercellular contact, the complete in vivo-picture may not be reflected in these experiments. Thus constructing differentiation trees becomes a complicated inference game integrating biological data from several sources. In fact the only organism to date for which the full differentiation tree is known is *Caenorhabditis elegans*: a small transparent nematode counting roughly a thousand somatic cells at maturity (Sulston and Horvitz, 1977). Given these complications it is not surprising that there is a lot of uncertainty surrounding differentiation and proposed differentiation trees are often challenged and need to be readjusted.

In this section the reader will be given an overview on transcriptome analysis techniques and differentiation will be introduced by the example of hematopoietic differentiation. Since this study also includes microarray data from two myeloid malignancies, chronic myeloid leukemia and multiple myeloma, those will also be treated briefly. Finally epigenetic changes will be introduced since they play a pivotal role in the design of the method.

1.1 Background

1.1.1 Transcriptome analysis techniques

Analysis of the transcriptome measures quantitative data about the concentrations of various mRNA molecules. One of the first techniques to analyse gene expression was Northern blotting, the name derives from the analogy to the technique for DNA called Southern blotting (named after the inventor, Southern). In Northern blotting RNA samples are first size-separated using gel electrophoresis. By means of capillary force or vacuum, the RNA molecules are then transferred to a nylon membrane. The RNA molecules have a negative charge and are attracted to the positively charged nylon membrane. The RNA molecules are then cross linked to the membrane by UV radiation. The immobilised RNA molecules can be quantified and identified using labelled complementary probes. Probe detection can occur by a variety of methods including radioisotopes and chemiluminescense. A follow-up method does the reverse, instead of fixating the RNA to the membrane, pieces of genomic DNA are fixated and RNA molecules are used as probe. This method can be seen as a rough first version of the now popular microarrays. Microarrays take reverse Northern blotting to the microscopic level. Essentially the same as reverse Northern blotting, microarrays consist of a support structure, on which short stretches of DNA are fixated. Due to advances in modern technology thousands of these oligomers can be attached to a small surface, similar to how electronic microchips are produced. The oligomers, called probes, correspond to known coding regions. Sets of probes can then be mapped to known genes. The microarray platform offers several possibilities dependant on the probes attached to the support structure, the most common application is expression profiling. RNA is extracted and converted into labelled cDNA by reverse transcriptase, fluorescent dyes are generally used. The labelled cDNA is then hybridized to the oligomers, and specialised machinery takes intensity readings (Lockhart et al., 1996). These readings can then be used to calculate the expression levels of the genes represented on the microarray. There are two possibilities: relative expression levels comparing two cases directly, or absolute measurements on a single case. They are called two-channel and one-channel techniques respectively, “channel” indicates a colour channel (or emission wavelength). In two-channel experiments the cDNA of the different cases to compare are labelled in a different colour. Figure 1.1 illustrates the steps in a two-channel experiment.

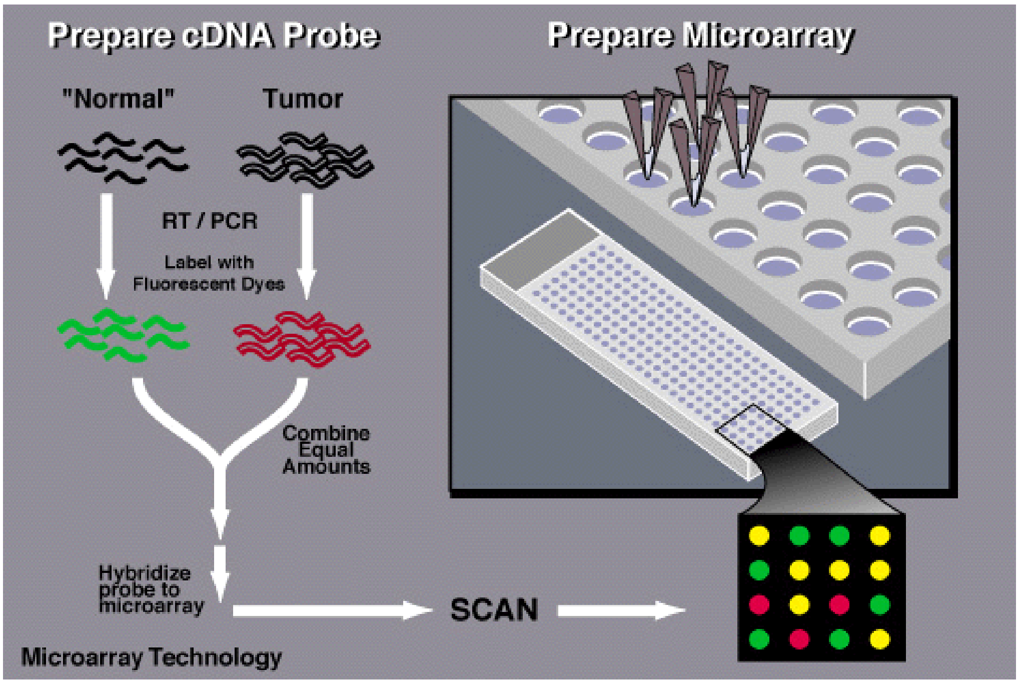


Figure 1.1: Illustration of the steps in a two-channel microarray experiment

The methods described above are all based on hybridisation and base complementarity. Around the time of the first microarray experiments, a competing technique was developed: Serial Analysis of Gene Expression (SAGE) (Velculescu et al., 1995). SAGE tackles the expression profiling issue by means of DNA sequencing. With SAGE, RNA is converted to cDNA and then cut by a restriction enzyme. This cut is then used to connect the cDNA pieces through a known linker sequence. This concatenate can be sequenced and, because the linker sequences are known, the RNA sequences and their abundance in the initial sample can be identified. This technique has a drawback and an advantage when compared to micoarrays. The drawback is that this technique is rather expensive, the advantage is that the sequences don't need to be known beforehand. Recently, with the development of Next Generation Sequencing (NGS), the technique has become viable again because the costs associated with sequencing have decreased drastically.

1.1.2 Similarities between evolution and differentiation

The similarities between the evolution of species and the differentiation of cells has a central place in this work. Evolutionary analysis aims to identify tree structures that explain the relationships between the different organisms that exist today. Central to any method to identify such trees is a component that changes over time. Early efforts in evolution could only use morphological attributes but recently it has become possible to use genomic sequence information in such research. The further apart two organisms are in an evolutionary sense, the more changes there should be in the genome.

Cellular differentiation can be approached in a similar way. In each step of differentiation, cells become more specialised for their particular role and cells with similar functions are organised into specific lineages. This process follows a tree structure with intermediate progenitors stages making up the internal nodes, an analogy can be made with ancestral states in the evolution of species. In the case of differentiation, changes in the transcriptome dictate changes in the identity of the cell. These changes occur gradually during the maturation process of the cell.

There are two important differences between the differentiation of cells and the evolution of species. In the evolution of species the object that changes is the genomic sequence, in the case of cellular differentiation the genome remains constant in all cell types (excluding mutation events and genome reorganisation events in immune cells) and the object that changes is the transcriptome. Another notable difference is that evolution is blind, a combination of random mutation events and how these mutations affect the fitness of the organism in its particular environment are what drive evolution. Clearly such randomness in undesirable in the context of a multicellular organism. Instead cellular differentiation proceeds along a, more or less, defined path towards a defined end point: the mature state. The changes in the transcriptome needed to arrive at this mature state are dictated by the environment through elaborate cell to cell contact. Hence, the environmental aspect of evolution is shared with cellular differentiation.

The analogy between evolution and cellular differentiation is know as the

Evo-Devo concept (David, 2001).

1.1.3 Hematopoietic differentiation

A well-studied differentiation system is that of blood cells, hematopoiesis. It will be outlined here as a general example of differentiation.

Hematopoiesis starts from the Hematopoietic Stem Cell (HSC) and gives rise to all types of blood cell present in the body. The system is highly active and can produce more than a billion of new blood cells each day. The differentiation process is guided by a complex combination of intracellular signals and communication with the environment, in adults this process occurs in the bone marrow.

The HSC, as all stem cells, has self-renewal potential. Cellular division can be performed asymmetrically. The results of this division are two daughter cells of a different type. One daughter cell is again a HSC, and the other cell a more mature progenitor. This is important because this asymmetrical division ensures that the pool of HSC never depletes and new blood cells can be continuously produced. The more differentiated daughter cell marks a new level of maturity and has lost self-renewal potential. These more mature progenitors are destined to terminally differentiate. In this first asymmetric division the lineage is determined. The strict lineage separation is now known as the classic model since new evidence suggests a different model. In this work the classic model will be taken as the working model.

In the classical model there exist two clearly defined and separated lineages: the myeloid lineage and the lymphoid lineage. Both development paths are entered based on the result of the asymmetric division of the HSC. The HSC either generates a Common Lymphoid Progenitor (CLP) or a Common Myeloid Progenitor (CMP). These cells are the progenitors for their respective lines. The lymphoid line consists of T-cells, B-cells and plasma cells. The myeloid line generates all the other blood cell types. There are still distinctions present in the myeloid line since it contains both cells from the innate immune system in addition to red blood cells and platelet-producing cells. This distinction is brought about by two possible progenitors descendant from the CMP: Granulocyte Monocyte Progenitors (GMP) and Megakaryocyte Erythrocyte Progenitors (MEP). This short description already covers a lot of the basics of differentiation in general. The process is stratified with a decrease in differentiation potential in each step. The most important step is the loss of self-renewal potential, which drives the cell towards a determined end point (Giebel and Punzel, 2008). Figure 1.2 illustrates this.

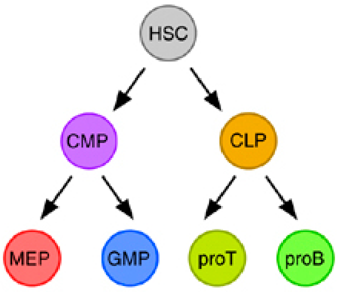


Figure 1.2: Schematic representation of the classical model of hematopoietic differentiation

This reduced potential and increased specialisation is a consequence of gradual yet profound changes in gene expression due to interactions with the environment, the niche. The niche provides stem cells with a favourable yet controlled environment. The niche exerts control over differentiation in a variety of ways, either by keeping stem cells in the stem cell state or inducing differentiation to a certain cell type. Cells in the niche maintain close contact with the stem cells by means of adhesion molecules. In addition, the niche cells influence stem cells by signalling to a variety of receptors. In the case of hematopoiesis the niche exists inside the bone. In the bone marrow niche the main interaction partners of HSCs are the osteoblasts. Osteoblasts produce several hematopoiesis related cytokines such as GM-CSF, which induces differentiation into granulocytes. Interaction with osteoblasts can also regulate self-renewal in HSCs through an elegant mechanism involving Notch signalling. When activated by parathyroid hormone, osteoblasts produce Jagged1, which in turn interacts with the Notch receptors on HSCs and increases self-renewal. Osteoblasts can also influence HSC self-renewal through signalling through the Wnt pathway (Rizo et al., 2006). Osteoblasts also directly influence the retention of the HSC in the niche by producing CXCL12, which interacts with CXCR4. Decreased CXCL12 causes increased mobilization of HSCs. In this light it is interesting that osteoclasts oppose niche retention by osteoblasts. Osteoclasts, once activated by RANKL, cause cleavage of CXCL12 and thus promote mobilisation into the bloodstream. It would appear that osteoblasts and osteoclasts don't only oppose one another in bone formation, but also in their influences on HSCs (Forsberg and Smith-Berdan, 2009; Porter and Calvi, 2008; Rizo et al., 2006).

The previous is but a small example of the complexity and variety of interactions in the bone marrow niche that govern hematopoietic differentiation. The important message is that cellular differentiation does not occur in a vacuum, but that the system is guided and controlled by the environment. Figure 1.3 shows some additional interactions as an illustration.

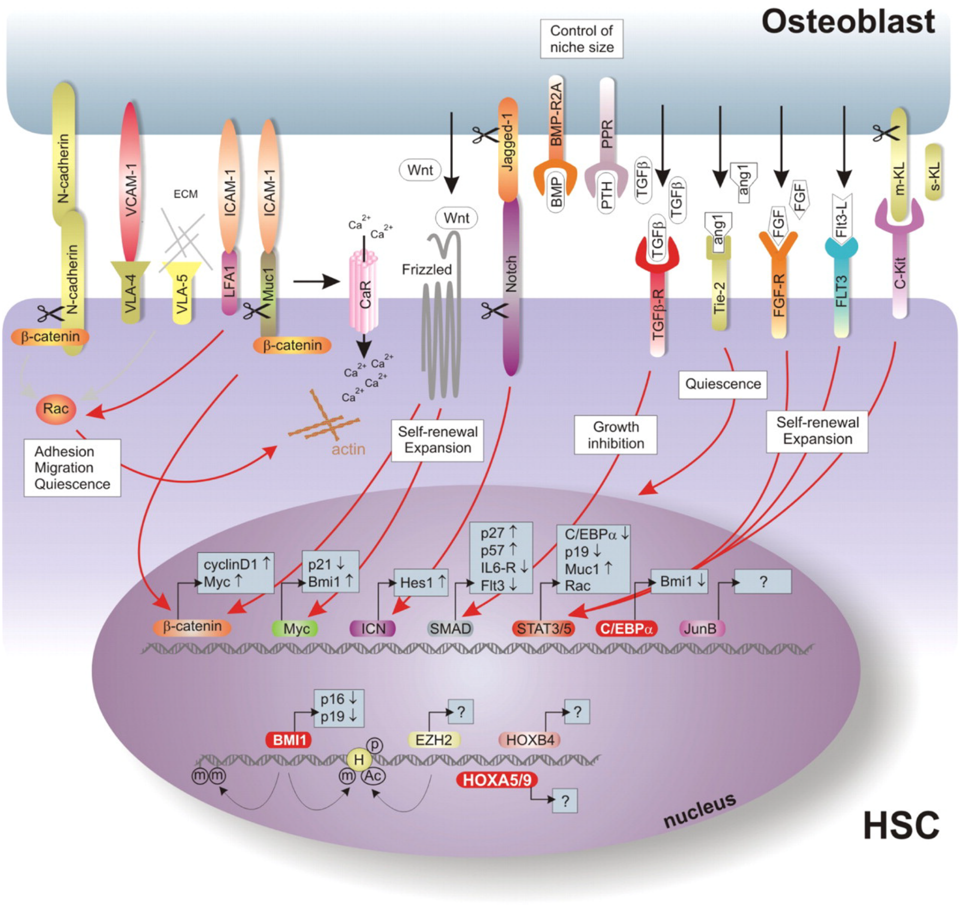


Figure 1.3: Illustration of niche interactions, image was taken from Rizo et al. (2006)

It should be noted that this classical model has recently been challenged by new evidence. The competing model argues that the separation between the the myeloid and the lymphoid line is not as strict as previously assumed. It was shown that progenitors destined for lymphoid differentiation may still differentiate into cells of the myeloid lineage. It is suggested that myeloid cells, in this case granulocytes, are a default or prototype form of blood cell and that all other varieties are a specialised form of this prototype cell. This model is quite elegant and does manage to explain characteristics of the different cell types. The prototypic myeloid cells have functionality that overlaps with the more specialised cells, in this case mobility and phagocytosis. These characteristics are also found in B-cells in addition to the specific functionality they have. This theory may also help explain why there are two branches of the immune system: innate and adaptive. (Kawamoto et al., 2010)

Hematopoietic differentiation is not only governed by regular signalling but also by epigenetic events. DNA methylation is an important factor in hematopoietic differentiation REF showed that disruption of the DNA methyl transferase DNMT1 in HSC caused defects in a variety of systems such as self renewal and niche retention (Towbridge et al., 2009). These adverse effects were not restricted to the HSC alone, but could also be seen in the more mature progenitors. This indicates that correct epigenetic regulation is of high importance for successful differentiation.

1.1.4 Myeloid malignancies

1.1.4.1 Chronic Myeloid Leukemia

Chronic Myeloid Leukemia (CML) is a malignancy of the hematopoietic system which manifests itself in the myeloid line. The system becomes malignant quite early in development at the point of the HSC. Interestingly the transformation to CML is caused by a single specific translocation event in 95 percent of the reported CML cases. The resulting chromosome is called the "Philadelphia" chromosome. The Philadelphia chromosome is the result of a translocation between chromosomes 9 and 22. It generates a fusion gene consisting of parts of the ABL gene and the BCR gene. The BCR-ABL fusion gene transforms HSC by eliminating the the possibility to regulate the tyrosine kinase activity through the SH1 domain of ABL. The fusion results in a constitutively active ABL tyrosine kinase, which overtakes the activities of regulated ABL. Because of the importance of ABL and its regulation in cellular differentiation, the continuous activity of this enzyme has profound implications on normal development. The inability to control signalling from this enzyme results in a wide variety of abberations such as uncontrolled proliferation and growth factor independence, in essence cells retain stem cell-like characteristics while normal differentiation is impaired.

Because of the time point at which the cells become transformed (HSC) it is perhaps not surprising that CML can progress into any of the possible outcomes of hematopoiesis. Interestingly it most cases an expansion of the granulocyte fraction is seen.

CML progresses through three phases: the chronic phase, the accelerated phase, and blast crisis. The chronic phase is the longest phase. In that phase the effects of the disease are not very pronounced. After 3 to 5 years however, the disease progresses into the so-called blast crisis. In this phase the disease is highly similar to Acute Myeloid Leukemia (AML) with the same disastrous effects. It is not entirely clear what causes the progression of the chronic phase into the highly aggressive blast crisis. A possible explanation is the inhibition of DNA repair by the BCR-ABL fusion gene which would allow additional mutations to develop. (Bruns et al., 2009; Burke and Carroll, 2010; Frazer et al., 2007; Jamieson, 2008)

1.1.4.2 Multiple Myeloma

Multiple Myeloma (MM) is a malignancy of the hematopoietic system which manifests itself in plasma cells. Plasma cells are a terminally differentiated form of B-cells. After activation by antigen and T-cells, B-cells can differentiate into plasma cells which behave as factories for a specific type of immunoglobulin. An important step in the differentiation to plasma cells are several rounds of somatic hypermutation. Hypermutation occurs in the hypervariable regions of the immunoglobulin genes. These comprise the complementarity regions of the folded immunoglobulin domains. The desired effect of these mutation events is to allow for small variations in the immunoglobulines that may lead to increased affinity for the specific antigen.

Inducing such mutations also holds certain risks, chromosome breaks may be induced elsewhere in the genome, and oncogenes may be placed near the rather strong immunoglobulin enhancer sequences. Indeed, it is exactly those mutations that are causative for progression into MM. Because of the somewhat random nature of this process there is no single fusion or translocation that is causative for MM, unlike the Philadelphia chromosome in CML. (Dimopoulos et al., 2006)

Although an uncontrolled population of plasma cells is clearly undesirable, it is interesting that the main impact of this malignancy on the system is due to the influence of the MM plasma cells on bone. MM cells cause an imbalance between bone destruction and bone formation. The bone is subject to two opposing forces: bone formation by osteoblasts and bone destruction by osteoclasts. MM cells alter the balance in favour of the osteoclasts by both stimulating osteoclasts and inhibiting osteoblasts. The imbalance results in netto bone destruction and, as a consequence, hypercalciemia. Due to the toxicity of calcium, hypercalciemia has a negative impact on wide variety of organs and systems. In light of what was discussed previously in relation to the involvement of osteoblasts and osteoclasts in the HSC niche the interference with the balance between osteoblasts and osteoclasts becomes even more interesting.

MM cells seem to interfere with bone primarily because of similarities in the optimal niche for both MM cells and osteoclasts. MM cells attempt to create a preferable environment for themselves and, in doing so, activate osteoclasts and inhibit osteoblasts. There is at least some evidence for this, MM cells produce the chemokine $MIP-1\alpha$, which interacts with the CCR5 receptor. Both MM cells and osteoclast precursors express this receptor. In osteoclast precursors, signalling from CCR5 promotes differentiation into mature osteoclasts, for MM cells signalling from CCR5 promotes growth and survival. As mentionned earlier, activity of osteoclasts negatively impacts retention of HSCs in the bone marrow niche. Taken together, the influence of MM on osteoclasts is a nice example of how important the interactions between different cell types are for successful differentiation. (Sezer, 2009)

1.1.5 Epigenetics

Epigenetics is a general term for a series of processes that influence cellular regulation but do not involve changes in DNA sequence while being heritable. Epigenetic regulation provides cells with the ability to regulate over time, in essence it provides cells with a memory. In the case of differentiation, this is a memory of past differentiation stages, but the system can also provide different time-related regulation. In plants for instance, the exact moment to progress from the vegetative stage to the reproductive state is to a large extent governed by epigenetic mechanisms (Sung and Amasino, 2004). Environmental signals are registered over time, and a developmental decision is made based on the accumulated information. Perhaps the most important characteristic of epigenetic regulation is that it can persist after cell division, this quality is of high importance in regard to differentiation.

This time-stable regulation is made possible by an intricate interplay between methylation of DNA and chemical modifications to its support structure, the histones. Histones are proteins that form disc-like structures (nucleosomes), the DNA is wound up around them into an efficiently packed structure. Because of the close contact between histones and DNA modification of these histones can influence transcription. Because of this unique possibility, histones are modified quite heavily with a number of possible molecular attachments. Of those possible modifications, modifications involving the attachment of a methyl group or an acetyl group are the most understood and studied. Histones have protruding stretches of amino acids, called tails, on which the modifying groups are attached. There is some combinatorial variation allowed here, specific groups can be added to specific amino acids in the tail. It appears that attaching a specific modifying molecule to a specific location in the histone provides a specific function. This specificity combined with the almost overwhelming amount of possible combinations has lead some authors to speculated that they may form a specific histone code. (Ruthenburg et al., 2007)

Some of the more known modification will be briefly covered here, because they are important in the interaction between histone modification and DNA methylation. By convention the position of the modified amino acids in the histone tail are indicated like in this example: H3K9. What is indicated is that histone 3 (histones form complexes) is modified at the lysine residue (K) at position 9 in the tail.

Histone modifications, unlike DNA methylation, can promote transcription or suppress it based on which residue is modified. In this regard two methyl modifications are on opposite sides: H3K4 methylation and H3K9 methylation. H3K4 methylation is associated with genes that are actively transcribed. H3K4 methylation can exist in three methylation states: mono-, di- and trimethylated. Although methylation of H3K4 can be found along the entire length of the transcribed region there is a gradient in the methylation state. Trimethylation is found at the transcription start site, monomethylation at the end and dimethylation in the middle (Ruthenburg et al., 2007). The methylation of this residue is closely linked to transcription itself. The enzyme responsible for the methylation of H3K4 in yeast, Set 1, associates with RNA polymerase (Sims et al., 2004).

On the other side of the spectrum is H3K9 mehtylation. Methylation at this residue is associated with transcriptional silencing. The trimethylated form is primarily localised to heterochromatin, but methylated forms of H3K9 can be found in active sites. Since H3K9 methylation is not present in promotor regions and at transcription start sites, it is speculated that it could prevent faulty transcription initiation. H3K9 plays a central role in epigenetic silencing due to its interactions with DNA methylating enzymes (Stewart et al., 2005). It can also be acetylated. Acetylation of histones is associated with open and active chromatin. H3K9 acetylation also excludes recruitment of DNA methyltransferases. Methylated H3K9 can recruit the DNA methyltransferase DNMT1 to the site by means of HP1 that interacts with the methylated H3K9 (Stewart et al., 2005).

Methylated DNA is associated with silenced DNA in mammals. In Drosophila methylation is associated with active transcription. Fortunately the mechanisms depositing the methylation marks are largely similar across different organisms, however the interpretation thereof is not necessarily similar. Methylation marks are generally deposited in areas rich in the CG dinucleotides called CpG islands, where cytosine is the methylated residue. These CG repeats are undermethylated in the promotor regions of active genes.

There are two possible ways in which methylation negatively impacts transcription, either by interfering with the binding sites of proteins that initiate or enhance transcription, or by promoting the binding of repressive proteins.

More important than how methylation silences transcription is how the methylation marks are deposited. Silencing by methylation is a consequence of inactive transcription rather than a cause. Earlier it was mentionned that the transcriptional machinery can indeed deposit epigenetic marks associated with active transcription. As a consequence of transcription-related silencing certain genes will have to be active, at least transcribed, in early development even though they are not required. It was indeed shown that this is the case for some genes. By transcribing these genes, undermethylated CpG islands can be formed in their promotor regions allowing expression later in development when they are required (Bird, 2002).

This last observation is important for the work presented here as it suggests that in early stages of differentiation CpG islands in promotor regions should be mostly undermethyhlated, and methylation in general should be lower. Fortunately there is evidence to support this mechanism. It was shown that in stem cells the amount of methylation in CpG islands is low, while methylation of these sequences increases during differentiation. Since there is almost no detectable demethylation this means epigenetic silencing increases during differentiation (Mohn and Schubeler, 2009). The link between increased methylation, and resulting silencing, and differentiation serves as one of the core principles of the method presented in this work. The increasing silencing provides a possibility to organise cell types in a chronological order. Methylation data was not available for this project, but by looking at transcriptional changes silencing can be indirectly analysed. The way in which this increased silencing is used in the method will be explained in more detail in the Method section.

1.2 Related work

The analysis of transcriptome changes over time and in an evolutionary context was also performed by Giger et al. (2010). In their work, comparisons were made between neuronal cells and endothelial cells. These two cell types were chosen specifically because neuronal cells are highly tissue specific, whereas endothelial cells exist in a variety of tissues. By comparing the endothelial transcriptome to that of neuronal cells two groups of genes were defined: a group of genes preferentially expressed in neuronal cells and a group of genes preferentially expressed in endothelial cells. Those two groups served as a basis to analyse transcriptome changes over time and across related species. They found that during development there is more variation in the group of endothelial genes than in the group of neuronal genes. Additionally genes from the endothelial cells are also expressed in other tissues whereas the neuronal group is very specific for brain (brain being the neuronal tissue analysed). A comparison between humans, chimpanzees and macaques revealed that there were more changes in the neuronal group than in the endothelial group. This is likely a consequence of the rapidly increased evolution of the brain in primates compared to other organs.

Although this work touches on similar topics as the work presented here, the outcome and direction are substantially different. From the work of Giger et al. (2010) it is not clear how this information can be used to recreate or score a differentiation tree. Also, the study is based on fully differentiated mature cells whereas in the work here the emphasis lies specifically on stem and progenitor cells.

Modelling of differentiation, specifically hematopoiesis, has been undertaken by several groups (reviewed by Foster et al. (2009)) but the focus there seems to be more on inferring regulatory networks than specifically reconstructing differentiation trees as is the aim of this work.

The study by Felli et al. (2010) approaches hematopoietic differentiation from a computer-science related angle. They consider the transcriptome of a cell at any given time as a representation of the state of the system at that particular time point. They suggest that cellular differentiation can be seen as the evolution of the system from an unstable state (the progenitor) to an attractor state (mature cell) following a given trajectory. To test this, they induced differentiation in vitro and did a microarray analysis at fixed time points. Using a correlation-based analysis, they could indeed show that the transcriptomic changes follow an identifiable trajectory towards a given attractor state. In addition they showed that this behaviour can be observed when considering the transcriptome as a whole or parts of it.

Clearly this work is of relevance to the work presented here and makes a strong case for the possibility to analyse differentiation trees using transcriptome data. The ability to identify specific differentiation trajectories also indicates that global transcriptome changes are not random but instead follow a clearly defined path. Unfortunately it was not clear from the study how these trajectories run and if it is possible to identify positions in a differentiation tree where a where a lineage separation occurs.

1.3 Aim of the thesis

This thesis will explore the possibilities of analysing differentiation in a computational manner using only transcriptome data, under the form of microarray data, and gene interaction data.

The analysis presented in this work will be guided by three questions. Do microarrays contain sufficient information about the differentiation process to analyse differentiation computationally? Can this information be used to validate or verify proposed differentiation trees? Can this information be used to compare differentiation in different contexts? The first question is general in nature but paves the way for the two following and more important questions. These questions will be addressed by stepwise developing a method that can, based on a combination of microarray data and cellular network topology, score the likeliness of any proposed differentiation tree in comparison to all other possible differentiation trees given a collection of cell types and corresponding microarray data. The method will be developed in such a way that it is easy to implement with tools that are readily available to the bioinformatics community. In addition to this, the method is built up in a modular fashion so the results of the component steps can have merit in their own right. Ideally the method should also be able to raise new questions and not only serve as a validation tool.

Chapter 2

Method

2.1 Principles behind the method

It is not uncommon in nature to see certain themes reappear in different systems. In the case of cellular differentiation there are parallels to be found with the evolution of species. The evolution of species is considered to have started with a single organism, as time progressed the descendants of this organism became more and more distinct and specialised in their role. In the initial phase of evolution the amount of different species was rather small, and the differences between them minor. With time the amount of different species and the differences amongst them gradually increased through stages that would later become ancestral species. This outbranching and rooted structure led to the adoption of the term ”tree of life” or phylogenetic tree. At any point in time, the tree contains ancestral internal nodes and leaf nodes. The leaf nodes represent a set of species that are highly specialised for their environment because of their environment. The environment is what drives the phylogenetic tree.

Cellular differentiation follows similar principles as the evolution of species. Cellular differentiation starts at the zygote. During subsequent cell divisions the amount of cell types increases and they become more specialised for their role until a terminally differentiated or mature cell type is reached. Along the path to mature cell there are several progenitor cell types which can be seen as ancestral to the mature cell types. Based on these ancestral stages the different cell types can be classified according to lineages. An example of this is the early separation into the ectoderm, mesoderm and endoderm germ layers. Similarly to evolution, the characteristics of the leaf nodes are largely due to communication with the environment during a cell's lifetime. The environment drives the differentiation tree.

Although there clearly is a likeness between both systems, there are also substantial differences. The most important difference is that evolution is undetermined, there is no fixed end point. Differentiation on the other hand is a determined process and has a defined end point. Evolution appears to be following a random trial and error system, while differentiation follows a more or less defined program. Because of these substantial differences the similarities between both systems serve a more philosophical purpose in identifying possible methods to analyse differentiation computationally.

In recent years phylogenetic analysis is mostly done by computational methods because of the increasing availability of sequence data. Central to these methods are base changes between genomes or parts of genomes or genes. Two methods are commonly used: maximum parsimony and maximum likelihood. In both methods all possible trees are evaluated and then scored based on some criterion. Maximum parsimony is non-parametric and, starting with an initial alignment of genome sequences, scores the trees based on the total amount of changes needed to arrive at the data analysed. Trees that score low are considered the best. Maximum likelihood follows a similar strategy, but is parametric. All trees are evaluated, but base changes are scored based on probabilities. The tree with the highest likelihood given the data is considered the best. Although both methods looks suitable to reconstruct differentiation trees, there is an important shortcoming due to the nature of phylogeny: the internal nodes are only inferred and the actual data always ends up in the leaves. This is not surprising since in the case of a phylogenetic study, data from the ancestral species is difficult to obtain, so it makes sense in that case to not consider them explicitly. For differentiation, however, the internal nodes are of importance, since they are progenitors, and data is available for them. Nevertheless, the idea of scoring all trees according to a change-based criterion is appealing in the case of differentiation. What remains is to identify what is changing, and how it is expected to change throughout differentiation.

Although every cell contains the full genomic information, fully differentiated cells only call upon a fraction of the possibilities. This is even required: it is undesirable that a liver cell, for instance, expresses gene programs characteristic for neural cells. As a result of differentiation, cells become more specialised and as a consequence loose potential. Cells are progressively programmed for their specific task in the organism. This requires the changes to be constant. Common cellular regulation systems do not carry over between generations, this function is provided by epigenetic regulation.

As mentioned in the introduction (1.1.5 on page 24), the netto methylation of the genome increases as differentiation progresses. As a consequence increasing parts of the genome will be covered by silencing methylation marks. One could make the analogy of an island that becomes progressively more flooded. In this case land represents active, non silenced chromatin while flooded areas represent chromatin that is silenced and therefore no longer accessible to the transcription machinery. Clearly, this decreases the potential of the genome. However, this decreasing amount of available chromatin or genes is exactly what gives cells their specialised nature. Similar to how an ocean oil rig only has very limited available surface but can provide a very specialised function, so can a cell with only a small amount of available genes.

Clearly the decrease of available chromatin does not occur in a random fashion. Instead, the silencing progresses in such a way that the gene programs available to the cell get gradually more restricted until a terminal specialisation stage is reached. In this light, the island analogy can be modified a bit. Instead of looking at differentiation as a gradual flooding of the genome with methylation marks, differentiation can be seen as a blueprint or operations manual out of which sections are deleted or blacked out with progressive differentiation. The zygote starts with a fully accessible and readable operations manual, and each time a cell enters a more mature stage a part of this manual is blocked, ultimately leading to a cell type that is locked into and performs a specific function. There is at least some evidence that the full array of gene programs is available to the cell in the early stages. As pointed out in the introduction (1.1.5), at least one study has stated that in the early stages of development cells can and do express genes that are neither specific nor useful for the early development stages. This is somewhat counter-intuitive, but given the thermodynamically stable nature of methylation it makes sense to not deposit methylation marks prematurely.

Although it is clear what the term "gene program" means it is not clear how these gene programs should be identified. Pathways are an early effort to somehow structure the multitude of genes present. In pathways, connected genes that together provide some kind of functionality, such as signalling from a receptor, are grouped together. The pathway system is inspired by biochemical knowledge derived from text and somtimes appears to be composed in a somewhat arbitrary manner. The most important issue with pathways is that they tend to overlap. This is mostly caused by highly connected "hub" genes. Recently a lot of attention has gone to network topology in order to identify gene programs. The cellular regulation network falls into the category of scale-free networks. This structure is commonly seen in many large naturally occurring networks. The internet and the human social interaction network are good examples. Scale-free networks have an interesting property: they consist of several highly interconnected subnetworks, called communities, which are themselves connected by highly connected hub genes. It is reasonable to assume that the genes in these community structures together provide a function in the cell similar to pathways. Hence, there are two possible ways to identify gene programs: by means of pre existing pathways, or by means of topology analysis. From now on gene programs will be referred to as subnetworks.

After identifying these gene programs, there has to be a way to assess whether or not they have become silenced. This can be done by means of statistical testing. A change from active program to inactive program should be visible in expression changes. A subnetwork that changes along an edge of the differentiation tree is regulated in some different way. This can either mean the subnetwork is epigenetically silenced along the edge, or it is influenced by the regular regulation machinery. For a subnetwork that does not change along an edge there are two possibilties: it was already silenced along that edge or the expression is exactly the same. We will assume here that there is no statistically significant difference for a subnetwork that is already silenced, the issue lies with subnetworks that are not regulated along an edge but are also not silenced. It is difficult to separate both. Nevertheless, it would seem that subnetworks that are neither silenced nor regulated do not confer any specific characteristics to the cell. Housekeeping genes and by extension housekeeping subnetworks are likely candidates here. Subnetworks may provide metabolic functions or they may be specific to the lineage of the cells under study. Hence, they also should not add any significant information to the method. The main principle here is the change from active to epigentically silenced. When scoring conformations there are two extremes to consider: all subnetworks change along every edge in the differentiation tree, and the opposite that none of them changes. In the first case this could mean two things: the subnetworks are all alternatingly silenced and reactivated, or they all remain active but are differentially regulated along every edge. Both scenarios are unlikely since the amount of methylation and therefore the amount of silencing should increase during differentiation. The other extreme is that none of the subnetworks changes during differentiation. This is would mean that either all subnetworks were silenced to begin with or that they are not specific to any cell type in the analysis. Clearly those extremes are not expected to be encountered. A normal differentiation process will probably lie somewhere in between. The main scoring principle will therefore rely on counting the amount of subnetworks that change along multiple edges in the conformation. Subnetworks that change along one edge are not counted because these changes are to be expected given increased silencing. Changes along multiple edges, although possible, are considered unlikely. The scoring method will consider the conformation with the smallest amount of subnetworks that change along multiple edges is the most likely.

The conformation does determine the position of the root node. A possible method to address this issue stems from a gene clustering experiment unrelated to this project. A graphical representation of the correlation coefficients between the genes in the clusters revealed that the complexity of these correlation matrices decreases when differentiation progresses. Figure 2.1 shows this progression for one of the mentioned clusters. The images representing the more mature cell types clearly have lower complexity than the less differentiated hematopoietic stem cell. This observed decrease in complexity shall serve as a basis for identifying the root nodes.

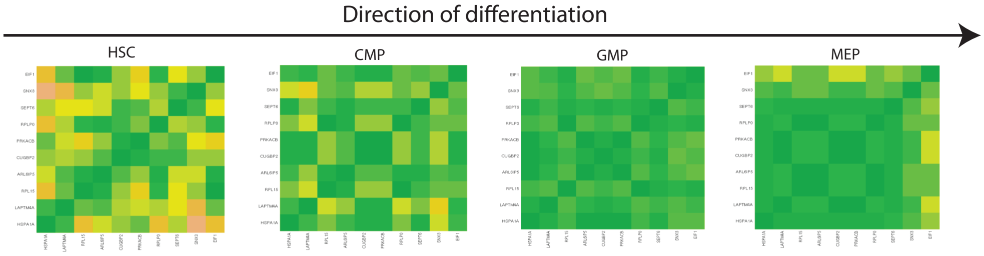


Figure 2.1: Changes in the complexity of the correlation matrices as observed after a clustering analysis

In what follows, the individual parts of the method will be outlined and explained in more detail. There are two main components: conformation scoring similar to maximum parsimony and root node identification based on correlation entropy.

Although this method should be applicable to any kind of high throughput expression data, in this case the method was tailored to microarray data.

2.2 Conformation scoring

Conformation scoring is analogous to maximum parsimony in that all possible conformations are scored based on changes in some vector. In the case of maximum parsimony, every node is assigned a vector, in that case a vector containing genomic sequence information with each base making up one element. In the method described here, edges instead of nodes are assigned vectors. The vectors are binary in nature and of length N where N is the amount of subnetworks, either pathways or topology-derived gene groups. The values of the N elements reflect whether or not a change has occurred in that particular subnetwork along that specific edge. For brevity, these vectors will be referred to as change vectors in what follows. Central here is the identification of the subnetworks.

2.2.1 Subnetworks

2.2.1.1 Predefined Pathways

The pathway system has its origins in an effort to group different cellular metabolic reactions. Cells perform a wide variety of chemical conversions on an equally wide variety of organic molecules. These conversions are usually not simple educt-product reactions and proceed over a range of intermediates, usually each step is performed by a particular enzyme or complex. A specific metabolic pathway contains all the reactions that together perform a certain conversion. For instance the glycolysis pathway contains all the chemical steps involved in the conversion of glucose to pyruvate. This pathway principle has been extended to also include groups of genes that together transmit information in the cell. Transmission of information is similar to metabolic reactions in that information is transmitted by means of chemical modifications, in most cases attachment or detachment of phosphate groups by kinases or phosphatases, respectively. This is referred to as a phosphorylation cascade and starts at a specific receptor.

Pathways are grouped in pathway databases. These databases are projects undertaken by research institutes or private companies. There is a lack of standardisation here, and because of this databases from different sources may not be completely identical and for all but the older metabolic pathways it is likely that the pathways in different databases are not equal or only partially agree. Different databases may also have a different focus, whereas one database may focus on metabolic pathways, another may focus more on signalling. Because of these discrepancies combining different pathways is no trivial task. It is recommended to select one particular database and stay with it. For the method described here, it is important that the database has enough coverage and is not specific to any kind of cellular process. It has to be large enough to cover most of the genes present on the microarray platform used so as not to loose potentially interesting information.

2.2.1.2 Topology-Derived Subnetworks

Cellular interaction networks fall in the category of scale-free or power law networks. This is a type of network commonly seen in naturally occurring networks such as the internet, social networks or predator-prey networks. This type of network has some interesting characteristics. The degree distribution of the nodes in the network follows a power law. This means that the bulk of the nodes in the network are of low degree, while a small percentage of nodes has a comparatively high degree. These nodes are referred to as hub nodes. Another interesting characteristic is that the length of the mean geodesic is surprisingly short and is fairly constant in regard to the overall size of the network. The exact length may vary, but is always fairly close to 5. It was shown by Stanley Milgram that this also holds for social networks (Milgram, 1969). He showed that any two people are connected by an average path length of 6 regardless of who those people are. This somewhat extraordinary fact is rooted in the structure of the network: it consists of a large amount of highly connected subnetworks that are themselves connected through high-degree hub nodes. These subnetworks are referred to as communities. An important consequence of this community structure is that the amount of connections between communities (outgoing connections) is low in comparison to the amount of internal connections.

Although the problem of dividing a network according to those communities is conceptually simple, it has proven quite difficult in practice. So this particular area of graph theory has been subject to a lot of research. Several possible algorithms are proposed. Most try to exploit the expected difference between inter community connections and intra community connections. Exhaustive use and analysis of these different algorithms is beyond the scope of this project. Instead, two methods are selected, one of them being novel modification to the already popular Markov graph clustering algorithm call

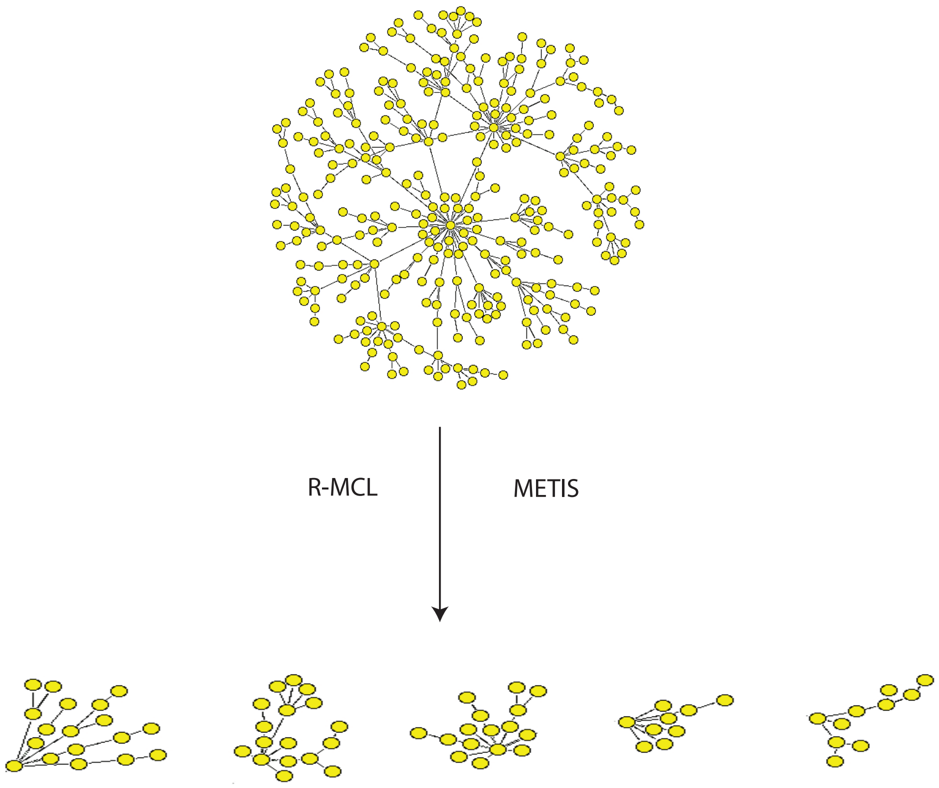


Figure 2.2: Illustration for how scale-free networks can be decomposed into individual subnetworks

regularised markov clustering (Satuluri and Parthasarathy, 2009), the other is an industry standard library from the field of parallel computing called METIS (Abou-Rjeili and Karypis, 2006). Both will be explained in more detail below.

Figure 2.2 illustrates how scale-free networks can be decomposed into individual subnetworks, taking the community structure into account.

2.2.1.2.1 METIS METIS (Abou-Rjeili and Karypis, 2006) is a programming library containing a variety of tools and functions for parallel computing. Specifically, the algorithms were designed to efficiently distribute computation jobs over several processors. Although this problem seems unrelated to community detection in biological networks, both questions are nonetheless similar: efficient distribution of jobs over different machines requires that the connections between the machines is low. This is because moving data between different physical machines or CPUs is a slow step and hence a performance bottleneck. One of the core principles in community detection in scale free networks is exactly that: the amount of connections between communities needs to be low. Communities can be seen as groups of jobs that are assigned to a single physical machine or CPU. This quality makes METIS an interesting candidate for identifying subnetworks to populate the change vectors. In addition to the aforementioned qualities, the algorithm also allows the user to specify the amount of desired groups and attempts to balance group sizes.

METIS, or more specifically kMETIS, the component program used here, works by means of multilevel k-way partitioning. The algorithm first performs a series of coarsening steps, in which the amount of edges is reduced. Edges are collapsed (removed) in such a way that the resulting graph still has the same overall structure as the initial graph. This smaller, coarse graph is then partitioned. After this partition the graph is uncoarsened and the previous partitions are partitioned again, this is referred to as the refinement phase. The effect of this is that instead of working on one large graph, the algorithm operates on a series of smaller graphs instead. This speeds up the algorithm, but also makes it easier to account for the hub structures in the graph. METIS has been heavily optimised and uses a series of modifications that speed up the algorithm and increase the quality of the partitioning. These technicalities are outside of the scope of this project and the interested reader is referred to Abou-Rjeili and Karypis (2006).

2.2.1.2.2 R-MCL Regularized Markov graph clustering (R-MCL) is a modification made to Markov graph clustering by Satuluri and Parthasarathy (2009). Markov graph clustering is based on the manipulation of the transition matrix or flow matrix of the network. This matrix is the column-normalised adjacency matrix. Because all elements in a column sum to one, they can be seen as the transition probabilities away from the node associated with a particular column. By multiplying this flow matrix with itself, a random walk is performed in the network. This is called the expansion step. One multiplication step is the equivalent of a random walk of length two in the network. The expansion step is followed by an inflation step, in which every matrix element is raised to a power between 1 and 2 (the default value being 2) and the matrix is again column-normalised. This has the effect of exaggerating the results of the random walk performed in the expansion step. After iterating both operations for a while the matrix reaches convergence, in this case only one column entry is non-zero. This non-zero entry is the attractor node, a specific node that other nodes cluster around. These nodes define the communities in the graph. In essence, each column is a probability distribution of the flow out of a given node. After every iteration the distribution becomes more centered around a specific node.

Satuluri and Parthasarathy (2009) suggest that this method is not optimal because the initial distribution is lost after the first iteration. This causes divergence of the probability distribution of neighbourring nodes and may lead to community fragmentation. Satuluri and Parthasarathy (2009) prove that minimising the Kullback-Leibler divergence between the distributions (the lower the Kullback-Leibler divergence the more two distributions are alike) of neighbourring nodes can be easily accomplished by right-multiplying the expanded matrix with the initial flow matrix instead of multiplying it by itself in each expansion step. Thus they attempt to overcome the problem of community fragmentation.

As mentioned earlier, scale-free networks may contain nodes with a fairly high degree in comparison to the other nodes in the network. Often degrees as high as 250 are seen. Especially connections between those high degree nodes may have a negative effect on the decomposition because they tend to draw a lot of nodes to them. To down-weight the influence of these nodes Satuluri and Parthasarathy (2009) suggest performing a weight transformation given by the following formula:

$A_{modified}(i,j) = \frac{A(i,j)}{D(i,i)} + \frac{A(i,j)}{D(j,j)}$

Where A is the adjacency matrix of the network and D the degree matrix of the network.

Because the cellular network consists of a large amount of nodes, the size of the flow matrices increases dramatically. Serialised matrix multiplication runs, assuming the standard algorithm is used, in $O(n^3)$ time. This is disastrous for large matrices and it may take as much as three hours to perform a matrix multiplication on a 5000 by 5000 matrix. Clearly this is a serious bottleneck for the algorithm because it expects to perform this operation several times till convergence. Satuluri and Parthasarathy (2009) suggest a multilevel work around, similar to how METIS functions, that uses a coarsening and refining phase.

Alternatively, the algorithm can be speed up dramatically by parallelising all operations. The largest speed boost comes from parallelising the matrix multiplication, but the algorithm also greatly benefits from parallelising the inflation step and the column normalisation procedure. For this project, the regularised Markov graph clustering algorithm was parallelised using the nVidia CUDA architecture. With CUDA, nVidia has managed to bring massive parallelisation down to affordable levels. Although the primary market for 3D accelerators lies in video gaming, it was quickly realised that these relatively cheap dedicated SIMD (Single Instruction, Multiple Data) devices could be used to parallelise mathematical operations in general. Early efforts to harness the potential of these devices had to use openGL to be able to access the raw potential of the card. nVidia developed an architecture especially for such computation called CUDA. CUDA provides an extension to C and obscures most of the technicalities of threading from the user. In addition several mathematical functions have already been ported to the system. The full documented source code for the parallisation of R-MCL is presented in the code appendix, and the description of the code is also presented at the end of this section 2.5.1.

2.2.2 Score calculation

2.2.2.1 Differences in subnetworks

There are two possible ways to test for differential expression: univariate or multivariate. Univariate analysis tests for differences between single

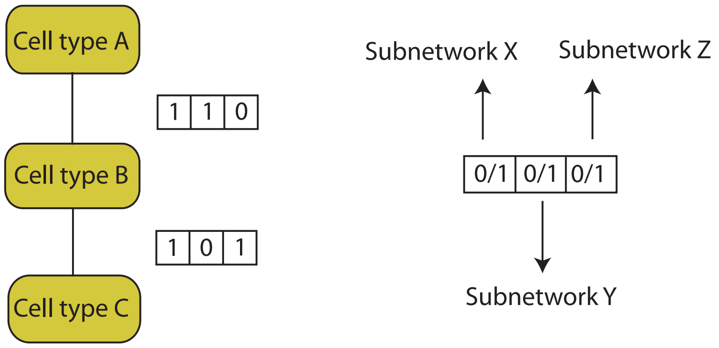


Figure 2.3: Illustration of the meaning of the change vectors and how they relate to differentiation conformations

genes,while multivariate analysis tests for differences in groups of genes as a group. Since the aim here is to test for differences in subnetworks between differentiation states, a multivariate, or group testing, method is required.

There are several possible methods to accomplish this. The test used has no influence on the function of the presented method. For this project the globaltest was chosen (Goeman et al., 2004). The global test uses a linear model approach to test for differences in gene groups. Normally, a linear model is used to classify an unknown instance based on a set of training data. The globaltest takes the reverse path and uses a linear model to assess whether or not there is enough difference between two groups of genes to make a clear distinction between two cell types. The null hypothesis is that none of the genes in the tested group provide any information regarding the two cell types. In other words it test if all the regression coefficients are simultaneously zero.

The process of assigning the change vectors and how they are uses is illustrated in figure 2.3.

2.2.2.2 Conformation Scoring

By applying the globaltest to all subnetworks generated by one of the methods described above, a binary change vector can be constructed for each possible edge between the cell types under study.

Each proposed conformation is scored by taking the sum of all the change vectors in the conformation. The result vector indicates which gene programs have changed and how many times they changed in the tested conformation. As stated earlier it is assumed that a large amount of programs that change along multiple edges is unlikely because it would mean a large percentage of gene programs would stay active which contradicts the assumption that silencing increases with differentiation.

2.3 Rooting the tree

A method to compare the maturity states of different cell types comes from analysing the correlations in the transcriptome, it was mentionned earlier (2.1) that the complexity of the correlation matrices appears to be dependant on the differentiation state of the cell.

Correlation allows for the analysis of interactions between genes. Although correlation does not imply causality, it would seem that interaction information is reflected by the correlation matrix of the genes. Here Spearman rank correlation was used. Spearman rank correlation is calculated with the following formula:

$\rho = 1 - \frac{6\sum d_i^2}{n(n^2-1)}$

where $d_i = x_i - y_i$ the difference between the ranks of both observations, i indicates the location of the variable in the vector and n is the amount of samples.

Spearman correlation is rank-based, this makes the calculation of the correlation entropy more robust. In addition it is also more resistant to noise because it does not require the relation between both variables to be linear. In the case of microarray data this is often the case.

Calculating the correlation coefficient between all genes on a microarray poses technical problems, the calculation is not limited by time but by memory. Common analysis software such as R or Matlab tend to allocate space for the results in memory. This limits the amount of genes between which correlation can be calculated in one run. To address these issues, a custom C implementation was written for the calculation of the Spearman correlation on large matrices. The program handles storage issues by deliberately keeping a small memory footprint and outputting to disk on regular intervals. Caution was taken to balance disk access with memory usage in an effort to reduce overall runtime. The program starts from a pre rank converted matrix and exploits the fact that the bulk of the calculations for Spearman correlation can be handled by vector operations. This makes the code easy to parallelise if needed. The full source code is presented in the code Appendix, and the description of the code is also presented at the end of this section.

Information entropy is the standard tool to analyse complexity or randomness of a dataset. Information entropy was developed by Shannon (Shannon, 1948) as a way to optimise telegraph communications. Shannon developed the following formula:

$H(X) = -\sum_{i=1}^{n} p(x_i) \log_2(p(x_i))$

where i indicates the position in the vector, and $p(x_i)$ the probability of the character that appears at position i in the vector. From the formula it can be seen that low probability leads to high entropy. The total entropy of a source, in the case of Shannons work the sender of the message, is the absolute limit of the best possible lossless compression of that source. The formula has some flexibility and the main principles that it relies on can be applied in a different setting. In this case the message entropy will be calculated, given some distribution of possible characters in a message, the entropy of a message can be calculated. Messages containing a large amount of rare characters will therefore have a high entropy. As an example, the English language rarely contains the letters Q and X. Thus based on the distribution of letters in the English language the message entropy of a text containing only Xs and Qs would be fairly high. In this case the characters are the possible correlation coefficients. The correlation matrix associated with a cell type is the message, and the distribution is derived based on all the messages in the set (the correlation matrices for all the cell types in the dataset).

The correlation values do not have to be discretised for this because the possible outcomes of Spearman correlation are discrete. This is because

Spearman correlation is rank-based and therefore dependant on the possible permutations of numbers between 1 and n, where n is the amount of samples. This makes the amount of outcomes directly dependant on n, hence Spearman correlation is not continuous between -1 and 1 if n is fixed.

2.4 System studied and data used

Although the method should be applicable to any differentiation system, the ease with which data can be gathered from the hematopoietic system in comparison to other differentiation systems makes this particular system an excellent choice to test and apply the presented method. The data used consists of two datasets derived from separate studies. One dataset consists of data gathered from CML patients and a healthy control (Bruns et al., 2009), the other dataset consists of MM data with a healthy control (data not published at time of writing). Samples were taken from bone marrow.

In both datasets HSC and progenitors of the myeloid line were isolated and microarray data was gathered for the following cell types: Hematopoietic Stem Cells (HSC), Common Myeloid Progenitors (CMP), Granulocyte Monocyte Progenitors (GMP) and Megakaryocyte Erythrocyte Progenitors (MEP).

The different cell fractions were obtained by means of immunofluorescence-based cell sorting. The four fractions can be identified based on the presence or absence of particular proteins expressed on the cell surface. The total fraction is defined as: Lin- CD34+. The HSC can be separated from the more mature progenitors because HSC are CD38-, while the more mature progenitors are CD38+. The CD38+ fraction can be further subdivided by using two additional markers: $IL-3R\alpha$ and CD45RA. The marker configurations for the respective fractions are: $IL-3R\alpha^{lo}$ CD45RA- for CMP, $IL-3R\alpha^{lo}$ CD45RA+ for GMP and $IL-3R\alpha-$ CD45RA- for MEP. Additionally, the fractions were tested for the presence of the BCR-ABL fusion gene with in situ hybridisation.

Microarray data was gathered from each of the fractions using the Affymetrix HGU133a2 platform which has 22277 probe sets. The data was normalised using GCRMA (Zhang et al., 2003).

Data from the healthy donors was explicitly not mixed, although the experiments are highly similar it is still possible that there is an experimental bias between both datasets. This bias could complicate the analysis of the results of the method. The datasets from the CML project and the MM project are considered separately. The following list gives an overview of the data in the experiment:

- Healthy set 1: 5 biological replicates for each cell type taken from healthy donors in the CML project

- Healthy set 2: 5 biological replicates for each cell type taken from healthy donors in the MM project

- CML set: 7 biological replicates for each cell type taken from CML donors in the CML project

- MM set: 5 biological replicates for each cell type taken from MM donors in the MM project

2.5 Code details

2.5.1 Code description for rmcl-cuda

This program is a parallelised implementation of Regularised Markov Graph Clustering (R-MCL). R-MCL is a modification of ordinary Markov graph clustering by Satuluri and Parthasarathy (2009). The improved algorithm was presented at SIGKDD 2009 in Paris under the title: Scalable graph clustering with stochastic flows: applications to community discovery

Regarding computation there is no difference between R-MCL and ordinary Markov graph clustering. Both perform an expansion step and an inflation step, as described above. Computationally those are equivalent to matrix multiplication and raising all elements in a vector to a given power. These operations, especially the matrix multiplication, can consume substantial amounts of computation time if the matrices are large. Since the algorithm expects to perform this action at least several times in succession the time cost can become unmanageable rather quickly.

Because of the nature of the operations, they can be easily parallelised. This program uses the nVidia CUDA architecture to perform parallelisation on nVidia GPUs. GPUs are by nature SIMD devices and are therefore suitable for parallelisation.

The code relies on the sgemm function from the CUBLAS library to perform matrix multiplication. CUBLAS is a library containing a variety of linear algebra functions ported to the CUDA framework.

The other operations are parallelised by means of custom functions, in the CUDA world referred to as kernels. Kernels are functions that are ran directly on the GPU, each thread executes the same kernel, but may have different internal variables. Because of this the code has two types of functions: those that are run directly on the device and those that run on the host machine. It is impossible (at least in the current CUDA versions) to access data in the device memory directly from the host. Accessing the data requires a memory copy operation. Because it can be time-expensive to perform this operation multiple times the data is kept on the device until the computation has ended.

It should be noted that the CUBLAS library expects matrices to be in column major format instead of row major format. Because of this, all functions expect the data to be in column major format. The fully documented source code can be found in the code appendix.

2.5.2 Code description for SpearmanPreranked

The program calculates the Spearman rank correlation coefficient for large datasets. The issue with calculating correlation on matrices with a high amount of rows is memory. The program solves this by keeping the usage of memory low and instead calls on disk space to progressively store the result. It is not advised to continuously output to disk since this would mean constant disk access which may slow down the program, and the operating system in general, considerably. In this case, each time a row of the correlation matrix is calculated it is appended to an output file on disk. The program outputs the results as strings instead of binary values so as to ease integration with other programs.

The program starts from a matrix in which the rows have already been converted to ranks. The rows are calculated at once by exploiting the fact that most of the calculation of Spearman correlation can be done by means of vector operations. Note that this also makes the described program easy to parallelise or to distribute.

The program keeps one copy of the original ranked matrix in memory which is used as a one dimensional vector. This is the master.

For the calculation of each row, a slave is loaded, the slave consists of N repeats of a given row, where N is the total amount of rows in the matrix. The resulting vector is hence of the same length as the master. By using vector operations combining both the slave and the master a row of the correlation matrix is calculated. The fully documented source code can be found in the code appendix.

Chapter 3

Results

3.1 Network data

The Transpath database was chosen as a source for pathway data. It contains 1059 pathway entries and covers both signalling and metabolic processes. The mean pathway size is 9.4 and the standard deviation 14.1.

For the topology based methods, data from the STRING database was used (Jensen et al., 2009). STRING consists of a large amount of interactions that come with a confidence score between 0 and 999. Interactions scoring 600 or higher were used to build up the network. This score strikes a balance between the size of the network and the confidence in the interactions. The network should contain a large amount of genes so the genes on the microarray are covered as much as possible.

The resulting network of interactions with a confidence of 600 or more consists of 14,764 nodes and 487,552 interactions.

Because the Transpath database contains the fixed amount of 1059 subnetworks, it was attempted to reach a similar amount of subnetworks with the topology approaches.

This can be easily accomplished with METIS by setting a parameter. The STRING network was decomposed by METIS in 1000 subnetworks, the average size of the subnetworks was 14.1 and the standard deviation 8.3. It should be noted that it may not be possible to divide the network in exactly

1000 subnetworks. In this case and METIS managed to identify 976 distinct subnetworks.

For R-MCL it is not immediately clear how the amount of subnetworks could be influenced. One possible approach is to vary the exponent in the inflation step. However applying the algorithm on very large graphs, such as the one used here, caused major issues with the algorithm. These issues will be discussed in more detail in the discussion section (4.1 on page 71). A method was found to influence the amount of subnetworks directly: by setting the desired amount of expansion-inflation steps and thereby influencing the length of the random walks performed in the network. Figure 3.1 shows the amount of subnetworks in relation to the amount of steps in the random walks. The amount of steps was set to 22 which resulted in a decomposition into 903 distinct subnetworks. the average subnetwork size is 16.3 and the standard deviation 43.9.

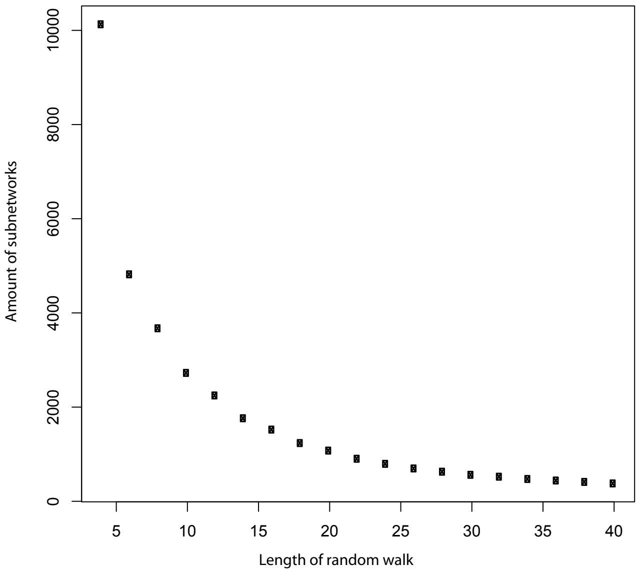


Figure 3.1: Graph showing the amount of iterations and the resulting amount of subnetworks for the R-MCL method

The performance of the graph decomposition algorithms can be scored with measures specific for scale free networks. The idea was to decompose the network into communities, tightly connected subnetworks. It is likely that these communities together perform a specific function. One such measure is the clustering coefficient, also called transitivity, which measures how interconnected the nodes in a given network are. More precise, the clustering coefficient calculates the probability that the nodes adjacent to a node are connected. Assuming the algorithm correctly decomposed the network into the respective communities, the average clustering coefficient of the subnetworks should be close to one. For the R-MCL algorithm the average transitivity of the subnetworks is 0.603 with a standard deviation of 0.274. For the subnetworks generated by METIS the mean transitivity is 0.23 and the standard deviation 0.397. Another possible evaluation criterion is vertex connectivity, also called graph cohesion, which calculates the minimum number of vertices (nodes) that need to be removed in order to make the graph not strongly connected. Intuitively one would expect this number to be quite high in highly interconnected subnetworks. Nevertheless, this number is also dependant on the size of the subnetworks and whether or not the subnetworks also have a large dependence on internal hub nodes. Elimination of one such hub node may severely impact the flow in the network. For R-MCL the average vertex connectivity is 1.294 with a standard deviation of 0.834, for METIS the average is 0.779 with a standard deviation of 0.46. For Transpath is was not possible to calculate the clustering coefficient and vertex connectivity since interaction information was not available for these pathways. Taken together these numbers seem to indicate that the R-MCL algorithm managed to decompose the network according to communities more effectively than METIS. This is perhaps not too surprising since METIS does not implicitly perform community detection but a calculation related to community detection, the minimisation of inter-subnetwork connections.

3.2 Identification of differentiation trees

3.2.1 Change vectors

Based on the subnetworks a change vector was calculated for each possible edge between cell types in each of the four data sets. Differences in subnetwork expression were tested using the globaltest as described in the method section (2.2.2.1 on page 38) . The significance level was set to 0.05. Subnetworks that were found to be differentially expressed were assigned a 1, the others a 0. Table 3.1 gives an overview of the total amount of changing subnetworks in each edge. There are four cell types, which leads to six possible edges per dataset. The table contains the total amount of changes per edge for each of the three subnetwork identification methods.

Edge	METIS	Transpath	R-MCL
Healthy set 1			
Healthy set 1: hsc-cmp	225	287	196
Healthy set 1: hsc-mep	52	76	117
Healthy set 1: gmp-mep	7	13	69
Healthy set 1: cmp-gmp	10	9	69
Healthy set 1: hsc-gmp	28	47	79
Healthy set 1: cmp-mep	15	19	86
CML set			
CML set hsc-cmp	27	15	61
CML set cmp-mep	23	37	65
CML set cmp-gmp	10	16	57
CML set hsc-gmp	17	10	56
CML set gmp-mep	10	20	61
CML set hsc-mep	10	25	62
Healthy set 2			
Healthy set 2 hsc-cmp	64	64	122
Healthy set 2 gmp-mep	222	361	196
Healthy set 2 hsc-gmp	240	292	225
Healthy set 2 cmp-mep	130	168	138
Healthy set 2 cmp-gmp	251	318	236
Healthy set 2 hsc-mep	159	234	164
MM set			
MM set gmp-mep	61	29	98
MM set cmp-gmp	29	42	78
MM set hsc-mep	90	85	104
MM set hsc-cmp	59	35	98
MM set cmp-mep	24	11	74
MM set hsc-gmp	53	68	97

Table 3.1: Total changes per edge in the four dataset using the three described subnetwork identification methods

From the table 3.1 it can be seen that the ranking of the edges, in regard to the total amount of changing networks, is not completely the the same for the three subnetwork identification methods.

Because of the amount of data the change vectors are illustrated graphically for a comprehensive overview. Figures 3.2, 3.3, 3.4 and 3.5 show the different edges and their associated change vectors. Zeroes (no change) are represented by white, ones (changes) are represented by black. Figures 3.6, 3.7, 3.8 and 3.9 show the distance between the change vectors, the distance metric used is the Manhattan distance. The Manhattan distance is defined as:

$\sum_{i=1}^{n} |p_i - q_i|$

where p and q are the vectors between which the distance is calculated, and i indicates the element in those vectors.

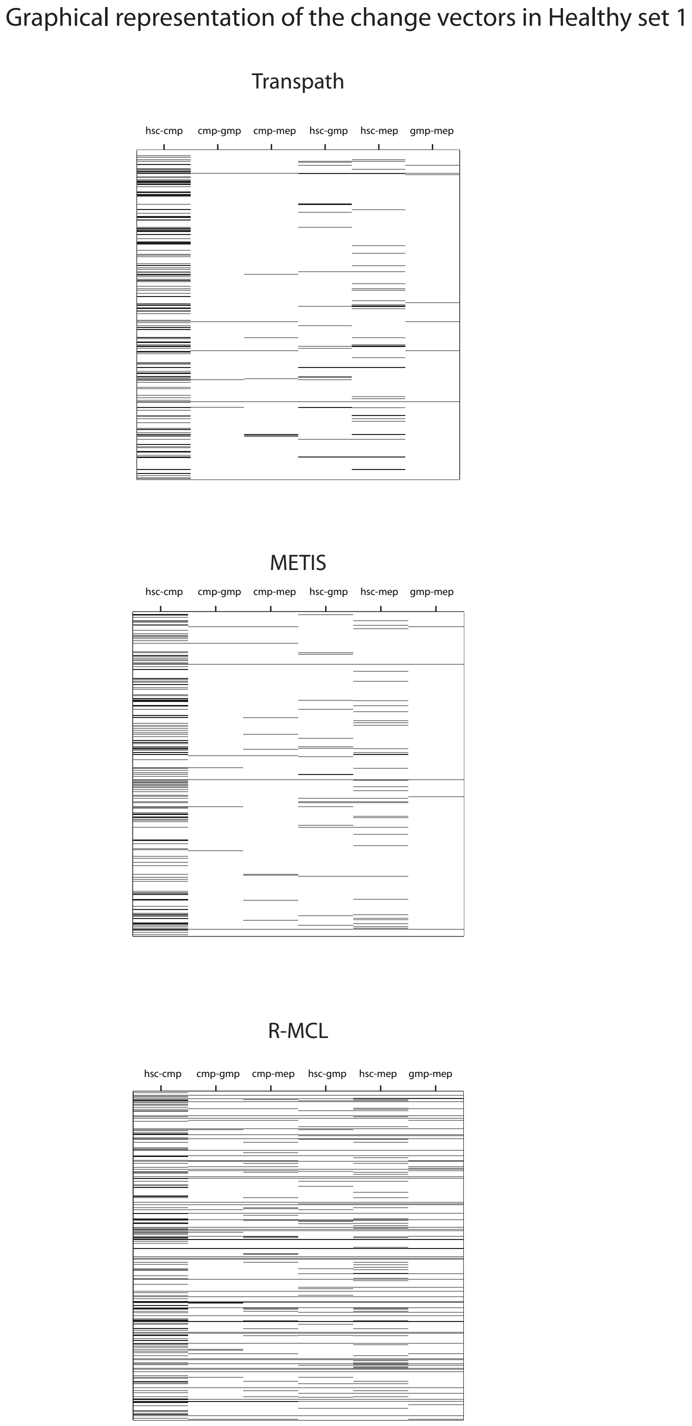
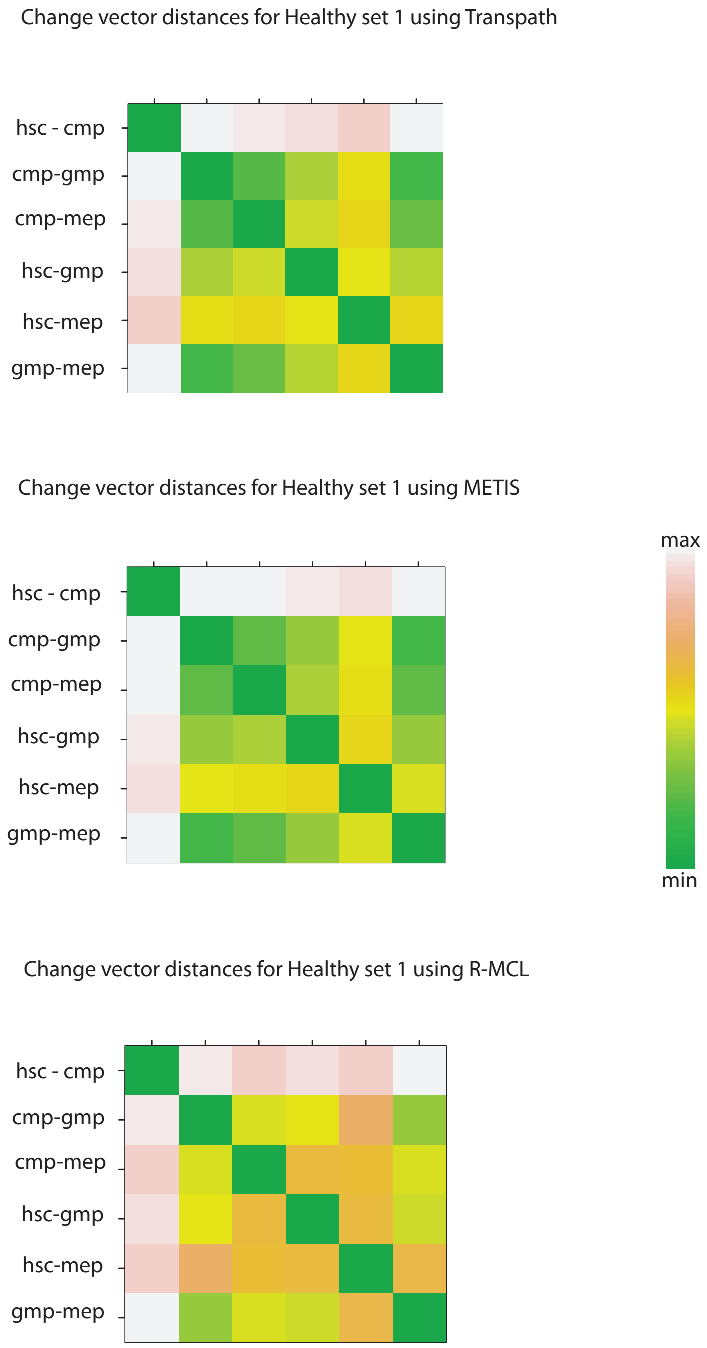


Figure 3.2: Black and white representations of the binary change vectors for Healthy set 1 using the three described methods for subnetwork identification


Change vector distances for Healthy set 1 using Transpath

Change vector distances for Healthy set 1 using METIS

Change vector distances for Healthy set 1 using R-MCL

Figure 3.6: Colour representations of the distance matrices between the change vectors of Healthy set 1 using the three described methods for subnetwork identification

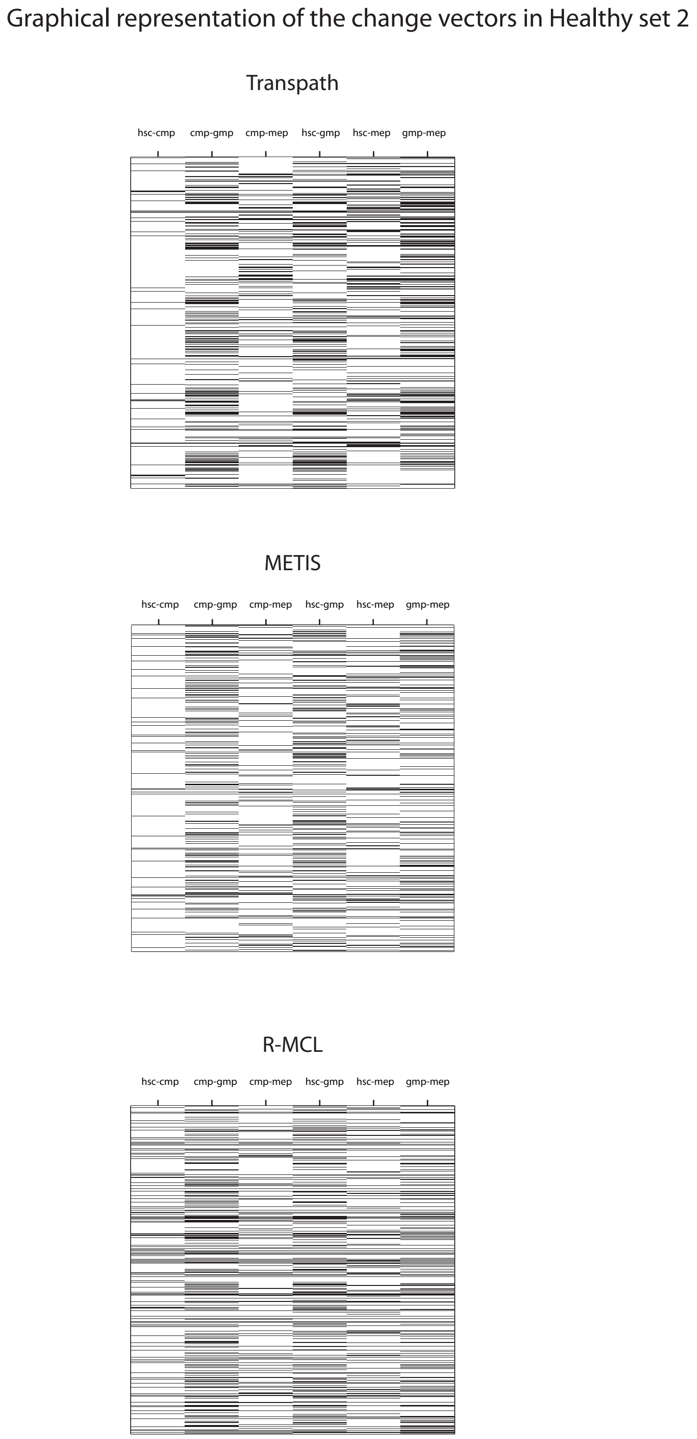


Figure 3.3: Black and white representations of the binary change vectors for Healthy set 2 using the three described methods for subnetwork identification

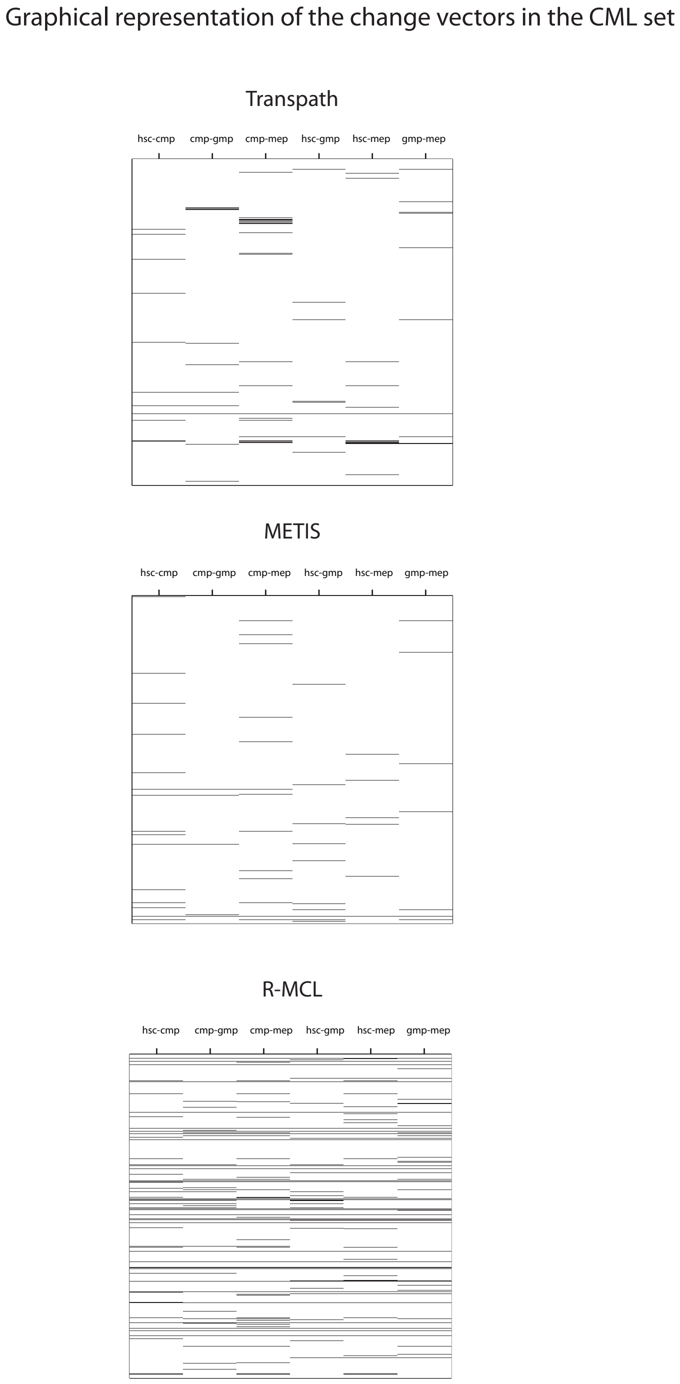


Figure 3.4: Black and white representations of the binary change vectors for CML set using the three described methods for subnetwork identification

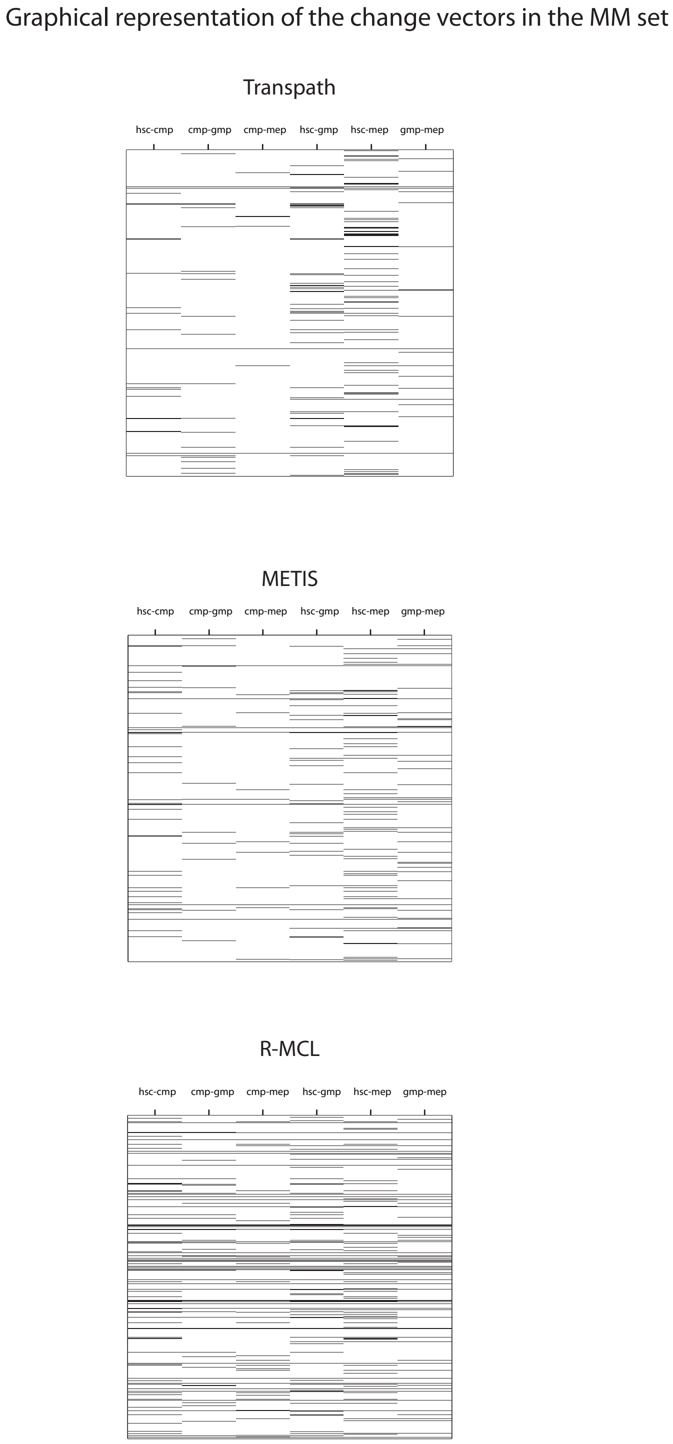
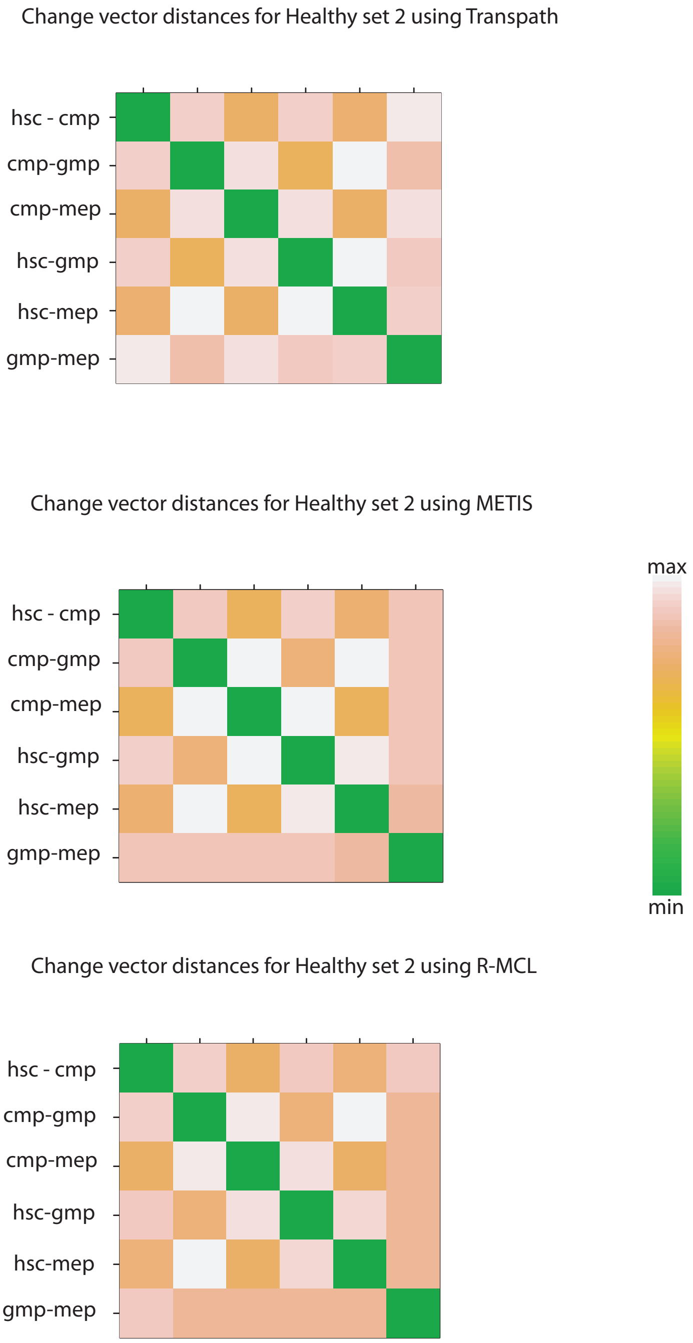


Figure 3.5: Black and white representations of the binary change vectors for MM set using the three described methods for subnetwork identification

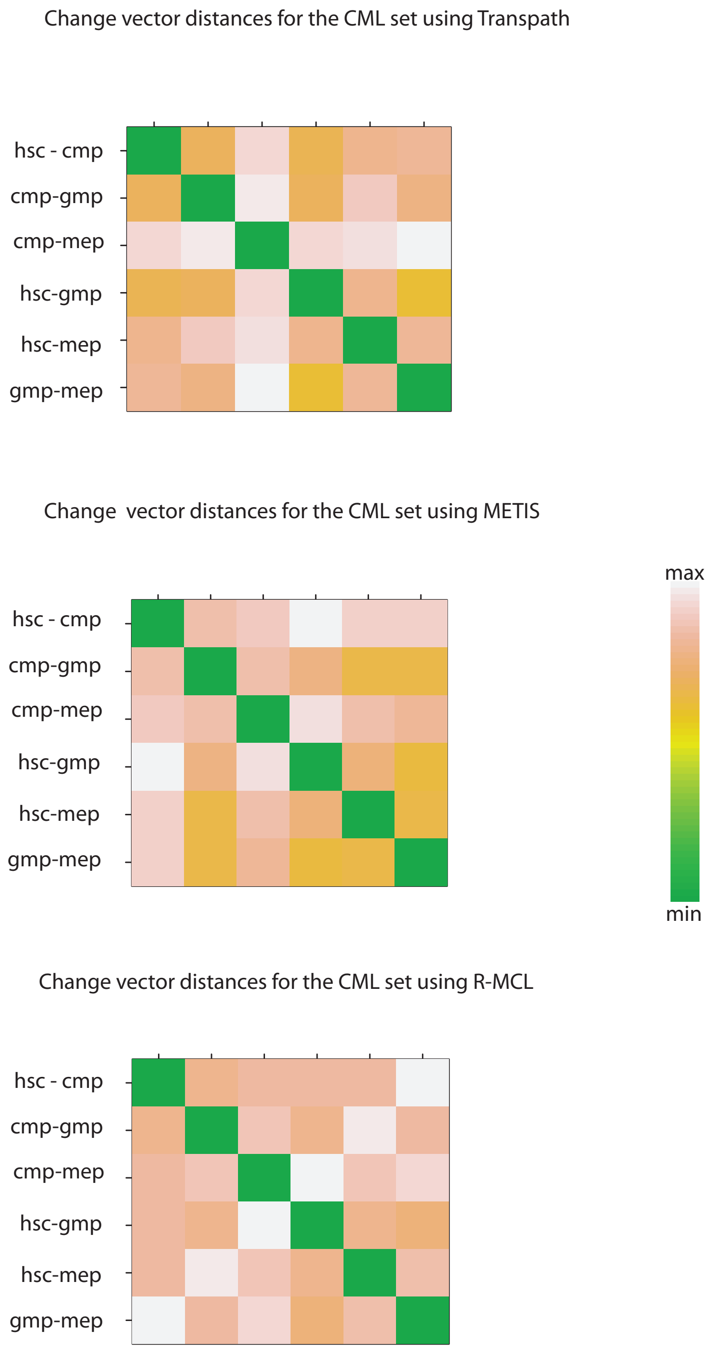


Change vector distances for Healthy set 2 using Transpath

Change vector distances for Healthy set 2 using METIS

Change vector distances for Healthy set 2 using R-MCL

Figure 3.7: Colour representations of the distance matrices between the change vectors of Healthy set 2 using the three described methods for subnetwork identification

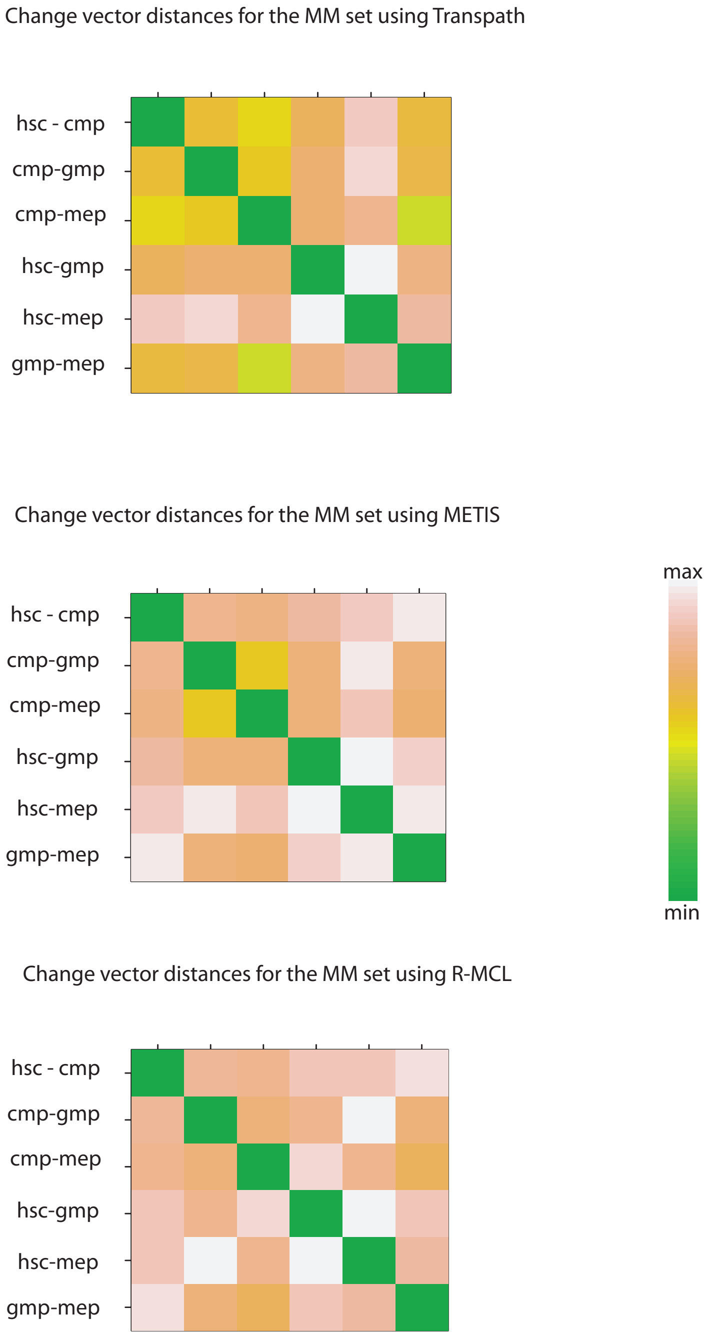


Change vector distances for the CML set using Transpath

Change vector distances for the CML set using METIS

Change vector distances for the CML set using R-MCL

Figure 3.8: Colour representations of the distance matrices between the change vectors of the CML set using the three described methods for subnetwork identification



Change vector distances for the MM set using Transpath

Change vector distances for the MM set using METIS

Change vector distances for the MM set using R-MCL

Figure 3.9: Colour representations of the distance matrices between the change vectors of the MM set using the three described methods for subnetwork identification

The first thing that stands out when looking at the total changes in each change vector (table 3.1) is the large difference between both healthy sets. In Healthy set 1, about 25 percent of the subnetworks change along the edge between HSC and CMP, while there are only few changes when looking at the other edges. In Healthy set 2, on the other hand, it seems to be reversed: the edge between HSC and CMP shows the least change of all edges. It would appear that in Healthy set 2 the CMP cells are rather close or similar to the HSC cells while in Healthy set 1 they are very dissimilar and hence rather far from each other. The difference between Healthy set 1 and Healthy set 2 becomes especially clear when looking at the distance matrices: in Healthy set 2 all change vectors have roughly the same distance to one another, while in Healthy set 1 it is clear that the change vector from HSC to CMP lies rather far from the other vectors while at the same time the other change vectors are fairly close to one another. A similar observation can be made while observing the graphical representations of the change vectors: in Healthy set 1, most changes occur in the step from HSC to CMP while in Healthy set 2 most changes seem to occur between the CMP (or HSC) and the more mature progenitors GMP and MEP.

In addition to showing differences between the datasets, the graphical representations of the change vectors also show noticeable differences between the subnetwork identification methods. Most changes were found when using subnetworks generated with R-MCL.

3.2.2 Scoring possible conformations

In order to score the conformations, all possible trees were built. There are only four cell types in the analysis, so the amount of possible conformations is manageable. A tree contains $v - 1$ edges, where v is the amount of nodes. All possible combinations of three edges out of six were generated. Not all of the twenty possible combinations are biologically plausible, some conformations contain loops which are not allowed. After removing the conformations that contain loops, sixteen conformations remain. Four of those are star conformations, with one cell type in the middle, and the rest are chain conformations. The conformations will be rooted later by means of correlation entropy. Tables 3.2, 3.3, 3.4 and 3.5 give an overview of the three best scores in the respective data sets. The full tables with the scores for each possible conformation can be found in the Appendix. For each conformation in each of the data sets there are three columns, two columns showing the amount of subnetworks that change along two or three edges respectively, and one column giving the total amount of subnetworks that change along more than one edge. This total amount is used to rank the conformations. The amount of subnetworks that changes twice or thrice was explicitly indicated because the amount of subnetworks that changes along every edge (thrice) was found to constant, regardless of the conformation for which these changes were calculated. This observation will be analysed in more detail in the discussion section 4.3.

Edges	Total	Twice	Thrice
METIS			
Healthy set 1: cmp-mep gmp-mep hsc-gmp	5	1	4
Healthy set 1: cmp-mep gmp-mep hsc-mep	6	2	4
Healthy set 1: cmp-gmp gmp-mep hsc-mep	6	2	4
Transpath			
Healthy set 1: cmp-gmp gmp-mep hsc-mep	7	4	3
Healthy set 1: cmp-mep gmp-mep hsc-gmp	9	6	3
Healthy set 1: cmp-gmp gmp-mep hsc-cmp	9	6	3
R-MCL			
Healthy set 1: cmp-gmp cmp-mep hsc-gmp	63	12	51
Healthy set 1: cmp-gmp gmp-mep hsc-cmp	64	13	51
Healthy set 1: cmp-mep gmp-mep hsc-gmp	66	15	51

Table 3.2: Top three conformation scores for Healthy set 1 using the three described subnetwork identification methods

Edges	Total	Twice	Thrice
METIS			
CML set : cmp-gmp gmp-mep hsc-mep	2	0	2
CML set : cmp-gmp hsc-gmp hsc-mep	3	1	2
CML set : cmp-gmp cmp-mep hsc-gmp	4	2	2
Transpath			
CML set : cmp-gmp hsc-gmp hsc-mep	3	1	2
CML set : cmp-gmp cmp-mep hsc-gmp	5	3	2
CML set : gmp-mep hsc-cmp hsc-gmp	6	4	2
R-MCL			
CML set : cmp-gmp cmp-mep hsc-gmp	47	13	34
CML set : cmp-gmp gmp-mep hsc-cmp	48	14	34
CML set : cmp-gmp gmp-mep hsc-mep	48	14	34

Table 3.3: Top three conformation scores for the CML set using the three described subnetwork identification methods

Edges	Total	Twice	Thrice
METIS			
Healthy set 2: cmp-mep hsc-cmp hsc-gmp	51	34	17
Healthy set 2: cmp-gmp cmp-mep hsc-cmp	68	48	20
Healthy set 2: cmp-mep gmp-mep hsc-cmp	78	59	19
Transpath			
Healthy set 2: cmp-mep hsc-cmp hsc-gmp	57	43	14
Healthy set 2: cmp-gmp cmp-mep hsc-cmp	79	65	14
Healthy set 2: hsc-cmp hsc-gmp hsc-mep	99	84	15
R-MCL			
Healthy set 2: cmp-mep gmp-mep hsc-cmp	111	37	74
Healthy set 2: cmp-mep hsc-cmp hsc-gmp	116	43	73
Healthy set 2: cmp-gmp cmp-mep hsc-cmp	122	49	73

Table 3.4: Top three conformation scores for Healthy set 2 using the three described subnetwork identification methods

Edges	Total	Twice	Thrice
METIS			
MM set: cmp-gmp cmp-mep hsc-cmp	13	4	9
MM set: cmp-gmp cmp-mep hsc-gmp	13	4	9
MM set: cmp-mep gmp-mep hsc-cmp	17	8	9
Transpath			
MM set: cmp-mep gmp-mep hsc-cmp	6	1	5
MM set: cmp-gmp cmp-mep hsc-mep	6	1	5
MM set: cmp-mep gmp-mep hsc-gmp	10	5	5
R-MCL			
MM set: cmp-mep gmp-mep hsc-gmp	63	14	49
MM set: cmp-gmp cmp-mep hsc-gmp	64	16	48
MM set: cmp-gmp cmp-mep hsc-mep	65	17	48

Table 3.5: Top three conformation scores for the MM set using the three described subnetwork identification methods

3.2.3 Root node identification: Correlation entropy

In each dataset the message entropy of the correlation matrices of the four cell types was calculated as described in the method section. Table 3.6 summarizes the entropy values, and figure 3.10 shows them in graph form. It is known from biological research that the HSCs are the starting point of hematopoietic differentiation. The entropy values suggest that a high entropy is associated with a less mature cell. The conformations will be rooted according to this observation, the cell with the highest correlation entropy is chosen as the root of the tree. It should be noted here that the shape of the graphs, the downward trend, is considered of more importance than the actual values of the correlation entropy. The graphs seem to indicate that in normal differentiation the correlation entropy of the cell types decreases from less mature to more mature state. Because the graphs for Healthy set 1, Healthy set 2 and the MM set have similar shapes and show a clear decrease in entropy from less mature to more mature cell type, the HSC was chosen as the root node for those particular datasets. The entropy values for the

CML set follow a different pattern, for this set the GMP was chosen as the root node of the conformation.

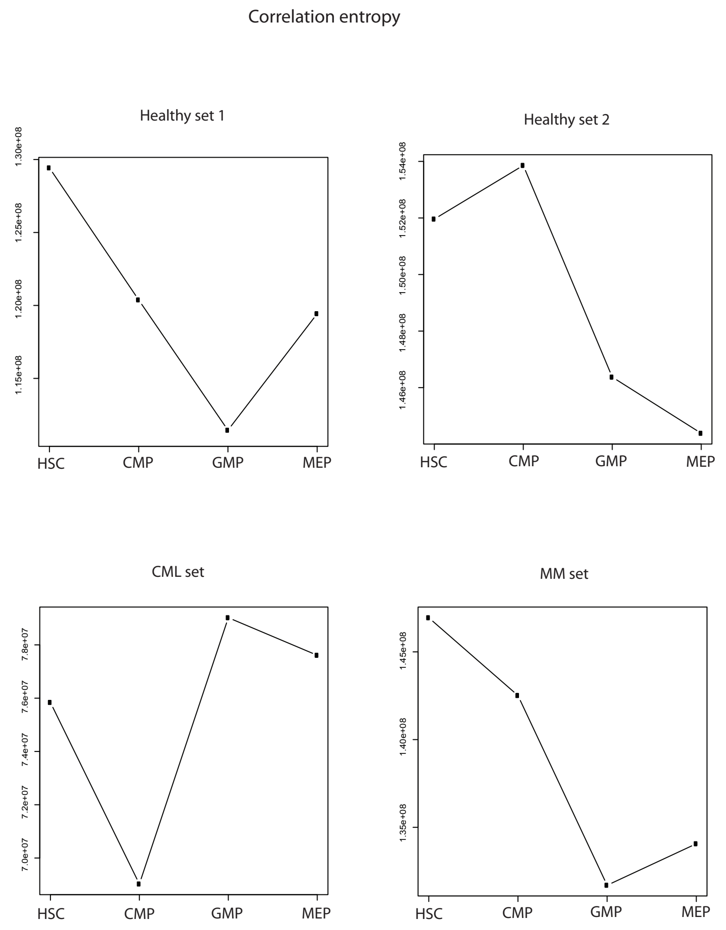


Figure 3.10: Changes in correlation entropy for the four cell types in the four datasets studied

Dataset	hsc entropy	cmp entropy	gmp entropy	mep entropy
Healthy set 1	129427135.486	120378663.211	111449987.890	119433676.774
Healthy set 2	151960435.466	153857752.140	146371679.570	144384082.194
CML set	75839727.320	69025338.092	79020678.721	77610089.118
MM set	146945623.969	142512313.451	131696946.821	134062516.281

Table 3.6: Correlation entropy values for all four datasets in all four cell types

3.2.4 Identifying rooted conformations

By combining the conformations with the most likely root node, the conformations can be rooted. Figures 3.11, 3.12, 3.13 and 3.14 give an overview of the top three best scoring rooted conformations for the four datasets using the three described methods for subnetwork identification.

Aside from identifying the correct differentiation tree according to the classical model, the method should be able to identify the correct differentiation chronology for the data derived from the samples taken from healthy donors. This means the CMP should be positioned between the HSC and the GMP/MEP in the those datasets.

In Healthy set 1, CMP was placed between HSC and GMP/MEP only twice: when using Transpath and R-MCL as subnetwork identification method. In the other cases, the HSC was connected to either the GMP or the MEP, causing the CMP to be connected to the more mature progenitors MEP and GMP.

In Healthy set 2 the differentiation tree according to the classical model could be identified in all of the three subnetwork identification methods. In addition, the edge between the HSC and the CMP was present in all of the top three differentiation trees. Interestingly, the edge between HSC and GMP was added in four of the nine cases. In one particular case (using Transpath), the HSC was linked to all of the more mature progenitors.

In the top three for the CML set, there are noticeable differences between both healthy sets. These difference are primarily due to the fact that the GMP was identified as the root node in this dataset. Interestingly, the HSC was positioned as “least mature” cell in three of the nine differentiation trees. Even when ignoring the rooting of the differentiation trees, the correct differentiation chronology could not be identified in this dataset.

For the MM set, the correct differentiation tree could be identified once when using METIS as subnetwork identification method. The correct differentiation chronology is among the top three in three out of nine cases. The top threes for the MM set are similar to those found for Healthy set 1.

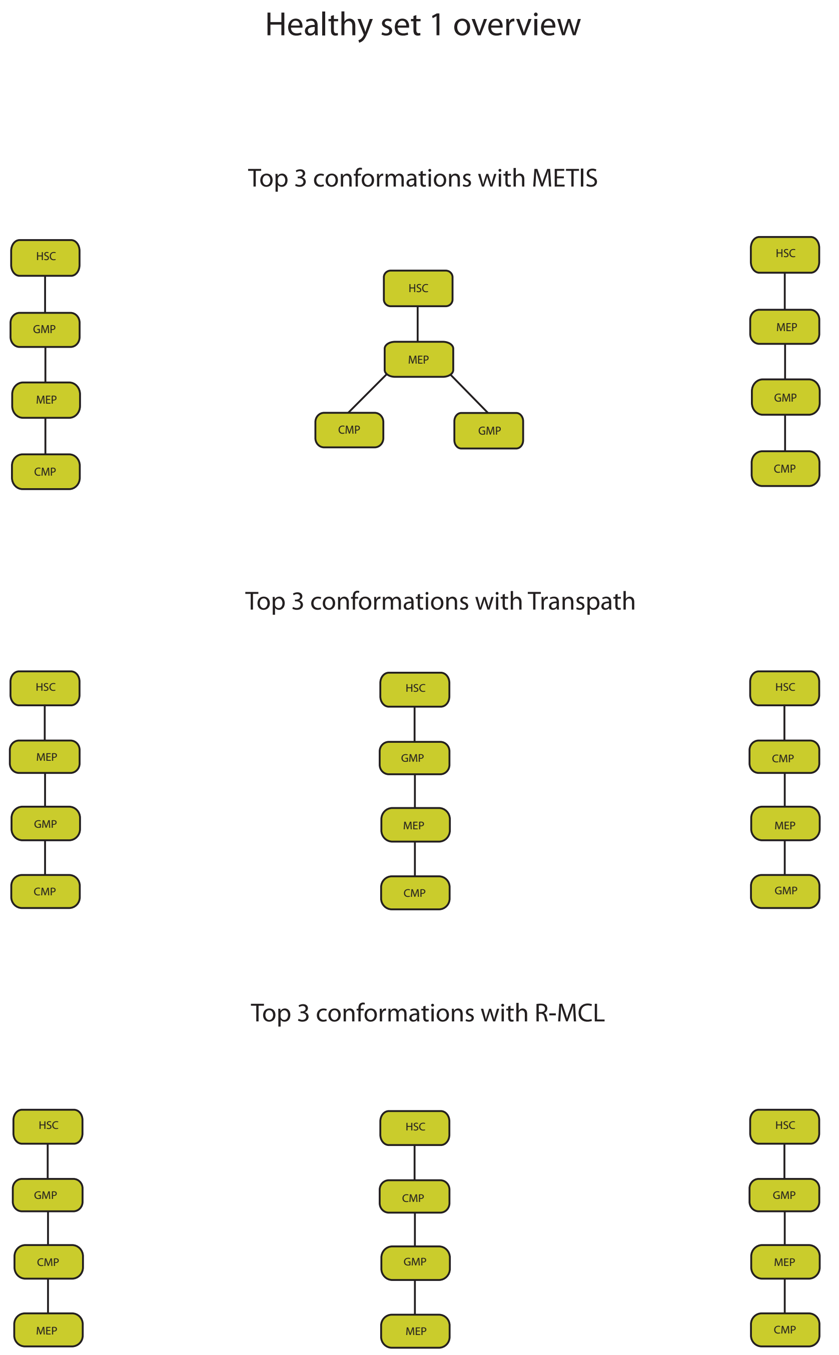


Figure 3.11: Top 3 highest scoring differentiation trees for Healthy set 1 using the three described methods for subnetwork identification

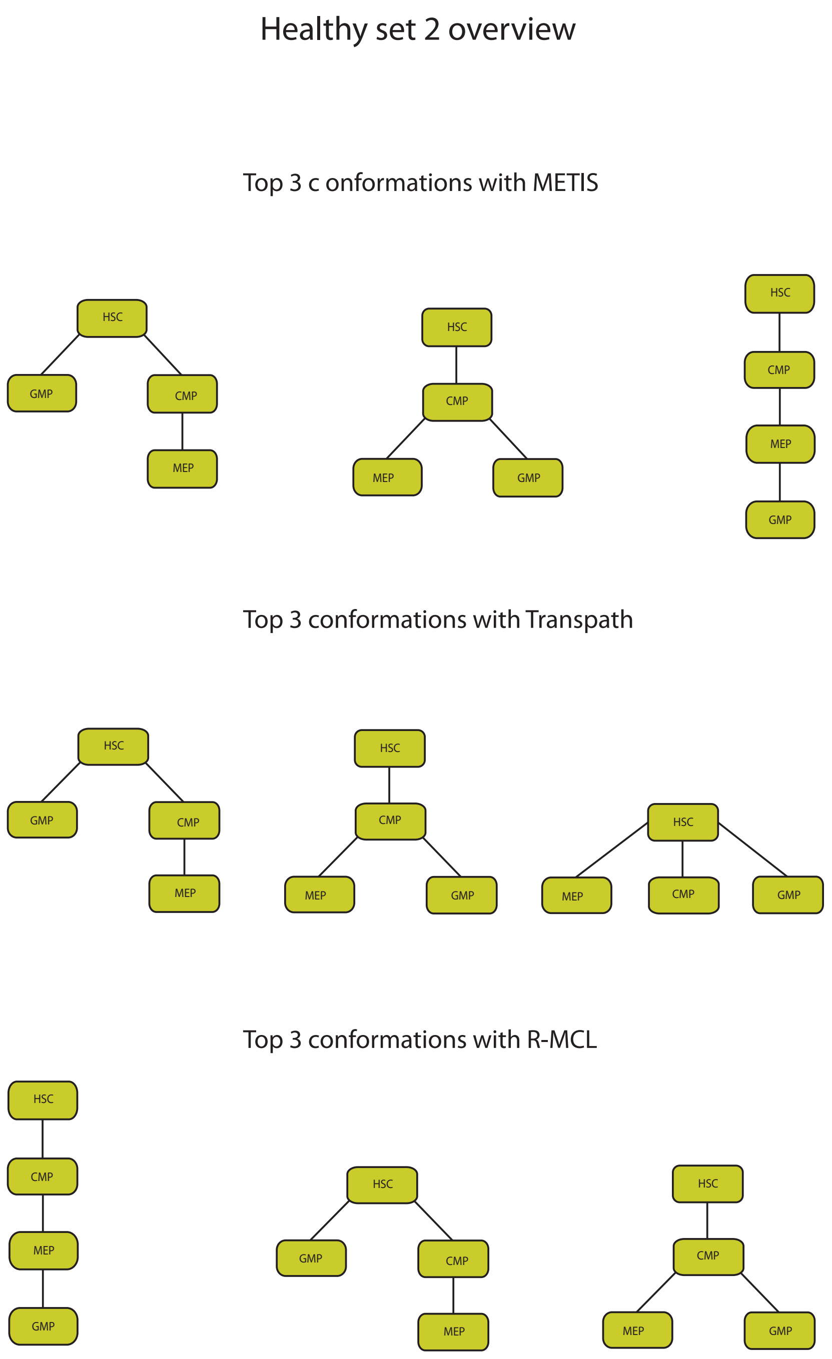


Figure 3.12: Top 3 highest scoring differentiation trees for Healthy set 2 using the three described methods for subnetwork identification

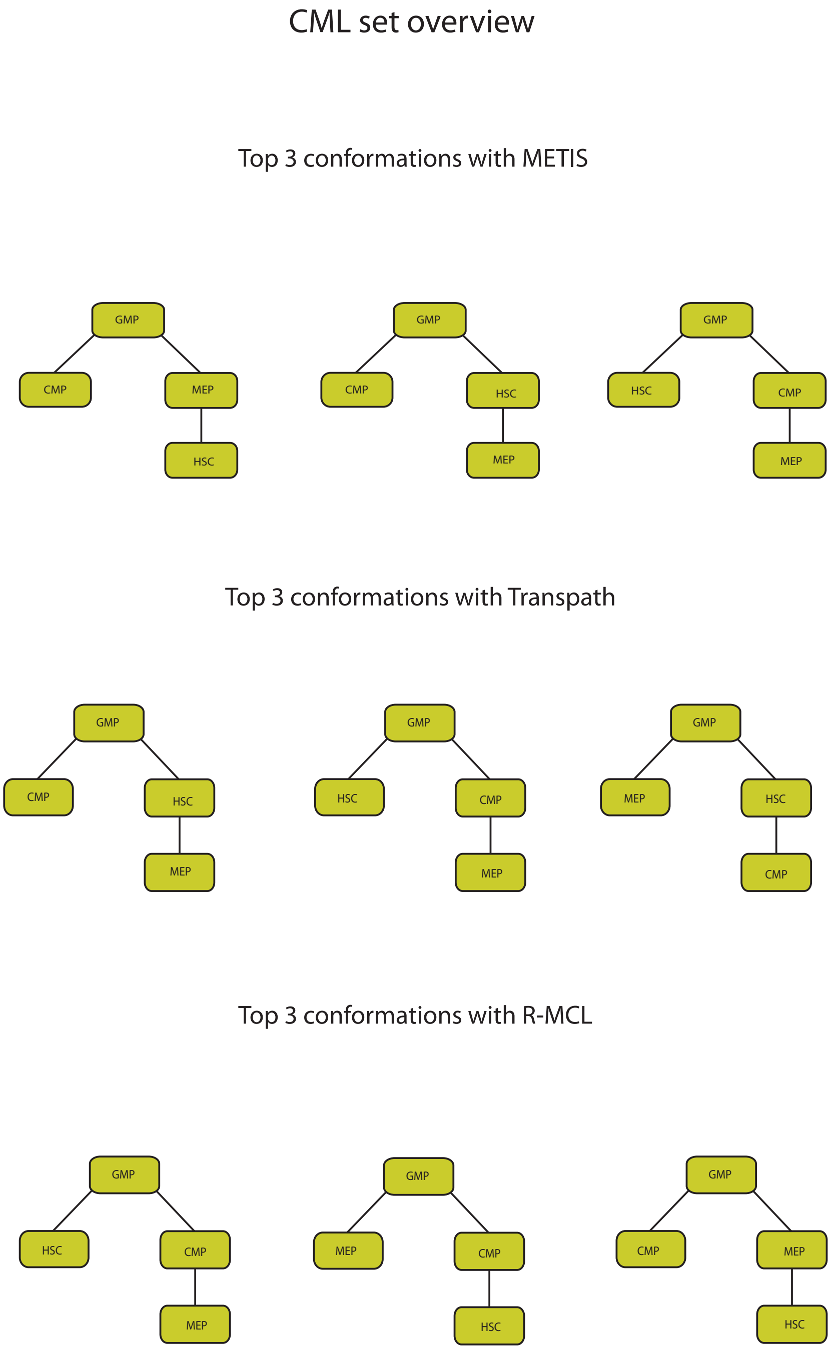


Figure 3.13: Top 3 highest scoring differentiation trees for the CML set using the three described methods for subnetwork identification

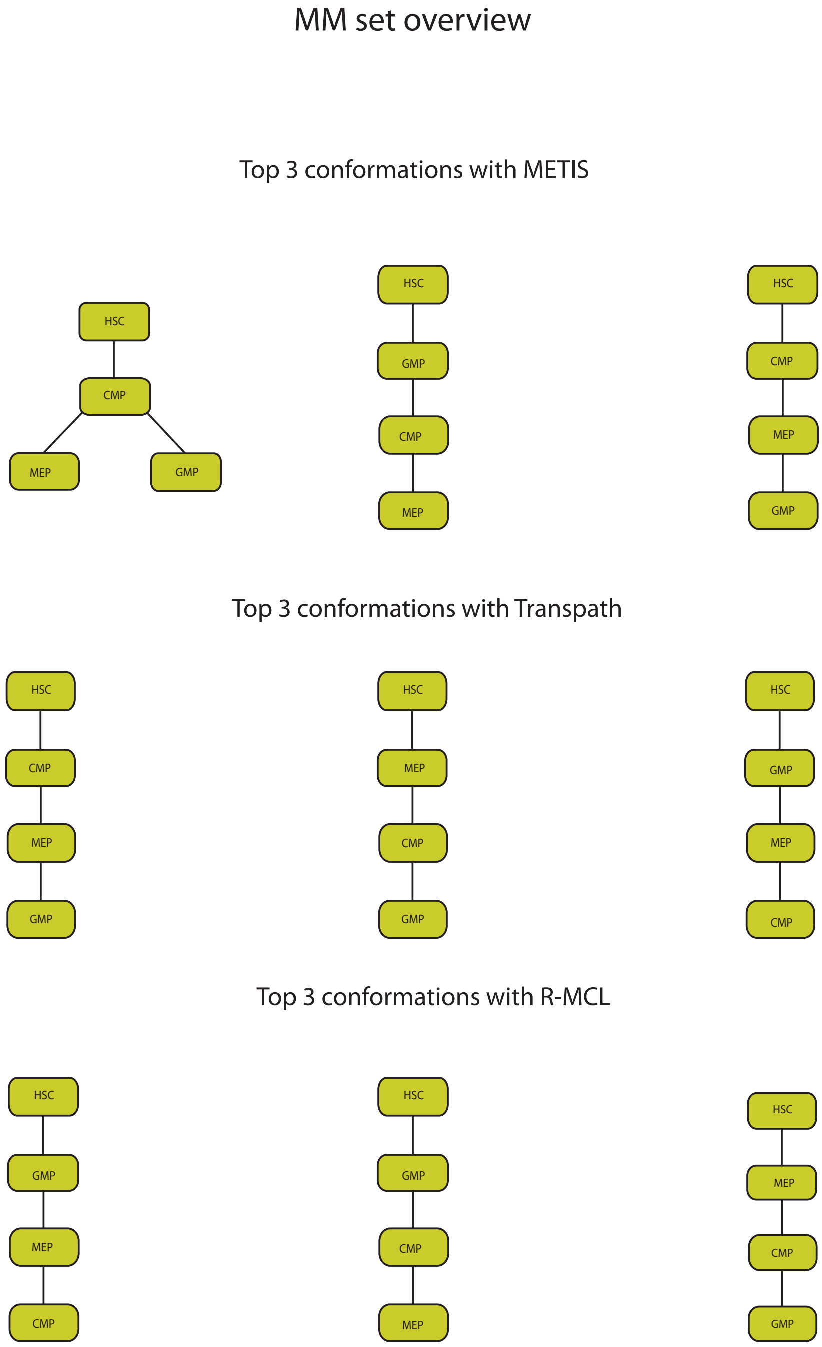


Figure 3.14: Top 3 highest scoring differentiation trees for the MM set using the three described methods for subnetwork identification

Chapter 4

Discussion

The different steps in the calculation will first be discussed in detail and afterwards a general analysis of the method will be presented followed by the outlook section where possible improvements of the method are discussed.

4.1 Subnetwork identification

As briefly mentioned in the result section, the R-MCL algorithm could not be applied as specified by Satuluri and Parthasarathy (2009). The encountered issues and how they were, at least partially, resolved will be covered in more detail here.

The highly connected hub nodes in scale-free networks tend to have a negative influence on graph decomposition, they may cause the entire network to cluster around one highly connected hub node. Clearly, this is undesirable since it defeats the purpose of performing such an operation. Satuluri and Parthasarathy (2009) suggest down-weighting the edges between high-degree hub nodes. These particular edges can be seen as large highways in the network and therefore draw a lot of traffic. Initial tests of the algorithm were run on a small test network composed of all the interactions in the STRING database with a confidence score of 999. The test network consisted of 1789 nodes. During these tests the suggested weight transformation did indeed eliminate the negative effects of the hub nodes. Satuluri and Parthasarathy

(2009) did not report any issues relating to the influence of hub nodes, but it would appear they worked on a network of comparable size. When applying the algorithm to the large network used in this project (the network contains 14,764 nodes) the hub-node related problems showed up, and already quite early in the computation. The presence of highly connected hub can complicate graph decomposition, they have a tendency to draw a lot of nodes to them and in some extreme case a subnetwork around such a node may comprise almost the entire network. A possible yet somewhat aggressive way to reduce the effect the hub nodes is to simply remove them. It should be noted that in the case of the network used in this project some nodes have degrees higher than 800. This means they are directly connected to about 5 percent of the total network. After removing the 5 percent nodes with the highest degree, the results of the computation appeared better in the sense that the decompostion was not clearly influenced by a small number of nodes, but there were still issues. The output consisted of a handful of very large subnetworks and a large amount of subnetworks with only one gene. After further experimentation it was observed that the amount of random walks that the algorithm performs (essentially the amount of iterations times two) is the most sensitive parameter to keep the effect of the hub nodes under control. Coincidently, this also provided a way to influence the amount of subnetworks, something that is required here to be able to compare the three different methods. The Results section refers to a figure (3.1 on page 47) showing the influence of the length of the random walks on the amount of subnetworks. It can be seen that the number of subnetworks decreases dramatically after a certain amount of steps. This is when the influence of the hub nodes starts showing.

The second issue lies with the inability of the algorithm to reach the convergence criterion, both for natural networks and networks simulated by a Barabasi game (Barabasi and Albert, 1999). The convergence criterion states that convergence is reached when all except one element in each of the columns of the flow matrix are zero. More theoretical, this means that the flow distribution out of that particular node (column) has completely collapsed and only peaks at one particular node. When performing standard

Markov graph clustering, the algorithm did reach the mentioned convergence criterion. Perhaps this is not so surprising, because the expansion step (multiplication) is very similar to the power method for computing eigenvectors of a matrix. The eigenvectors of graphs represented as matrices are a central point in spectral graph theory, which attempts to solve a similar problem. It is possible, and likely that the very addition made by Satuluri and Parthasarathy (2009) to Markov clustering is causing this issue. Recall that the addition was made so as to not break up subnetworks, essentially forcing minimal divergence on the probability distributions in the columns. However, this issue has not been reported or encountered by Satuluri and Parthasarathy (2009). The answer may lie in how the convergence criterion was handled. To minimise storage and to speed up the algorithm a pruning step was added that removes small values based on a heuristic. This essentially means that the theoretical convergence criterion was not followed. The pruning step removes small values and turns them into zeroes. In addition, the algorithm was run similar to METIS, using a multi-level approach with a pre-set number of iterations as a means to decrease computation time. If not parallelised, this particular computation may take days to complete even on networks of modest size. Nevertheless, it is tempting to speculate that Satuluri and Parthasarathy (2009) may have unintentionally reduced the amount of random walks and as a consequence avoided the hub node issues.

Because the Transpath database contains a fixed amount of pathways, an attempt was made to influence the topology-based subnetwork identification methods, namely METIS and R-MCL, so as to ensure that each of the three methods has a comparable amount of subnetworks. This is important, because large differences in the number of subnetworks used to score the differentiation conformations could lead to difficulties comparing and analysing the results. This effort turned out to be fruitful for both topological methods. However, looking at the reported mean subnetwork sizes and the standard deviations for the three different methods, it can be seen that mean subnetwork sizes for both METIS and R-MCL are similar while there is a large difference in standard deviation. The standard deviation for the subnetwork sizes generated by R-MCL is fairly large in comparison to METIS. This is not surprising given the discussion regarding R-MCL above: the large standard deviation is caused by the hub nodes. So it seems that removing nodes of high degree and performing a weight transformation step, still can not completely eliminate the effect of hub nodes in large scale-free networks. METIS on the other hand, explicitly tries to keep subnetwork sizes similar.

4.2 Change vectors

It is unclear what could have caused the differences between the datasets observed in 3.2.1. Both datasets consist of samples taken from healthy donors albeit in different experiments. Because only those two healthy sets are available for analysis, it is difficult to assess whether HSC should be close to CMP or not. It is possible that this difference has a biological background. It could be that the CMP state is not very clearly defined and is instead comprised of a mixture of progenitors along the differentiation path from the HSC to the GMP or MEP. A similar noticeable difference comes from looking at the possible edge between GMP and MEP cells. In Healthy set 1, there appear to be very little differences between both cell types while the opposite is true in Healthy set 2. An alternative explanation for these observed differences could be that there is some experimental bias between both datasets. The procedure to generate a dataset such as the one used here, is made up of several complicated steps (cell sorting and RNA extraction, for example) and even small variations in any of these steps may lead to different results. Because the possible conformations are scored based on operations on the change vectors, the differences between both healthy sets will manifest themselves throughout the analysis. An interesting question to ask is whether these differences are also clear in a hierarchical clustering of the samples in the respective datasets. Figures 4.1 and 4.2. For Healthy set 1 the samples cluster predominantly according to the donor, and not according to cell type. For Healthy set 2 the GMPs cluster together in a small cluster. Although the majority of the samples also seem to cluster together according to the donor, CMPs and HSCs do group close together which is in line with what was observed from the change vectors.

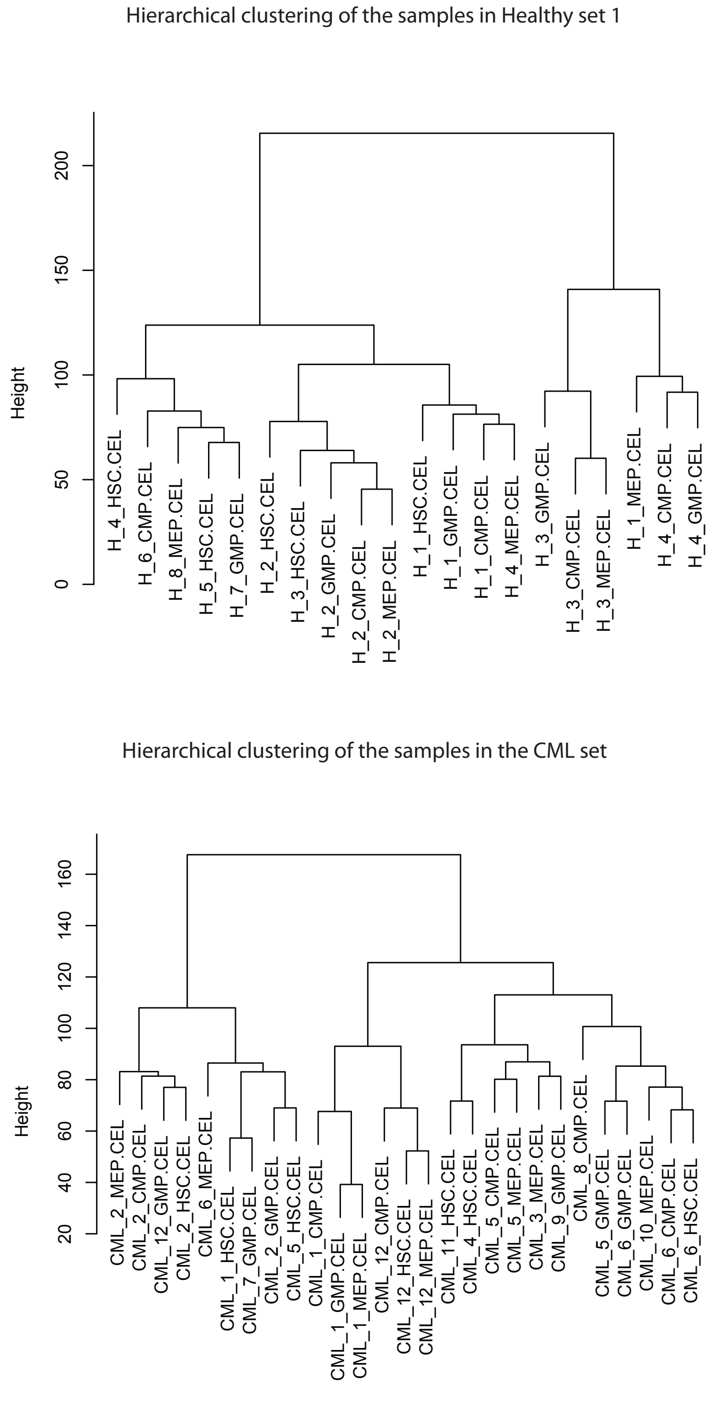
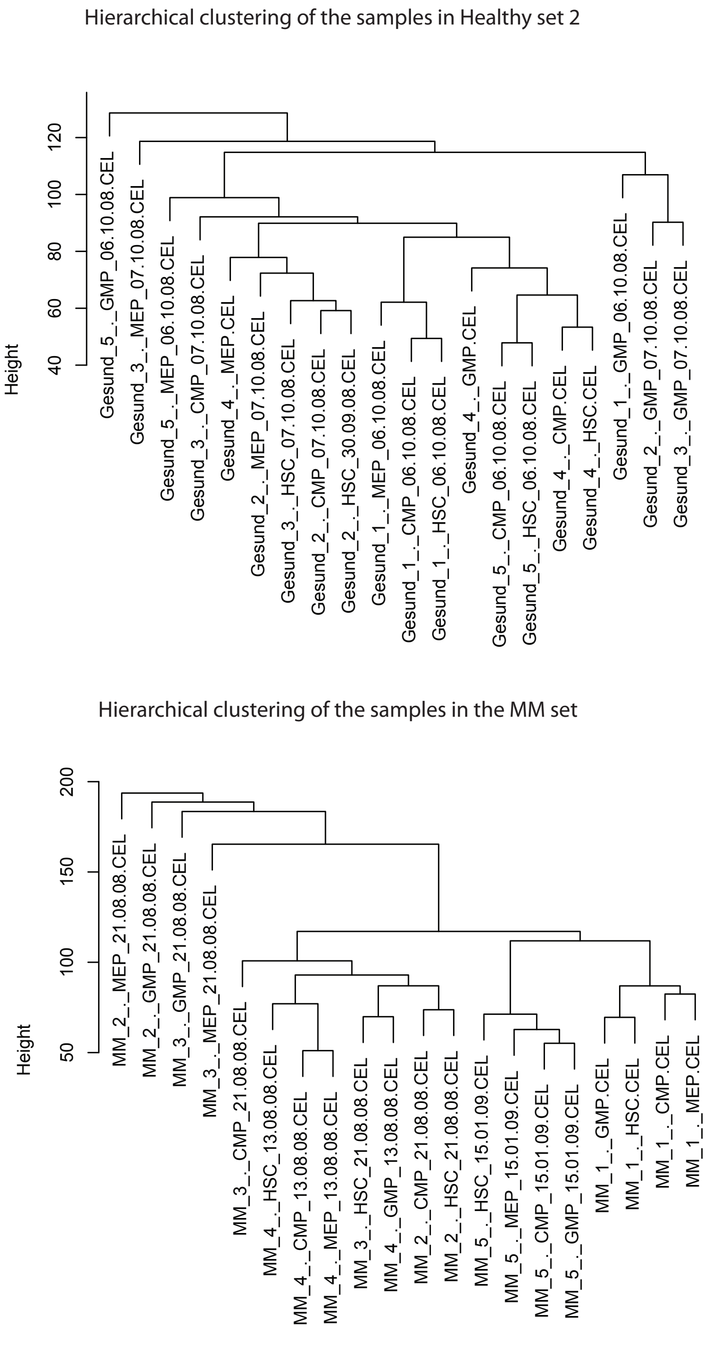


Figure 4.1: Hierarchical clusterings for the samples in Healthy set 1 and the CML set respectively



Hierarchical clustering of the samples in Healthy set 2

Hierarchical clustering of the samples in the MM set

Figure 4.2: Hierarchical clusterings for the samples in Healthy set 2 and the MM set respectively

Looking at all the datasets in a more general sense, it appears that the R-MCL method can on average identify the most changes in the transcriptome. The graphical representations of the change vectors show this clearly (figures 3.2, 3.3, 3.4 and 3.5). The R-MCL algorithm performed the best at decomposing the global regulation network according to communities (the highly connected subnetworks in the global scale-free network). This can be seen from the values for the vertex connectivity and clustering coefficients. Combined with the observation that the most changes can be found in subnetworks generated by this method provides evidence for the fact that cellular function indeed does overlap with the community structure as obtained by R-MCL.

There appear to be large differences in the mean amount of changes between all datasets. When comparing the datasets within the individual projects, Healthy set 1 versus CML set and Healthy set 2 versus MM set, the cancer datasets always show lower mean changes. In light of the theories underlying this method this would seem to indicate that there are more subnetworks switched off in the cancer cells studied here because a smaller percentage of them can change, or they are constantly active.

An important general conclusion to draw from the analysis of the change vectors is that there are noticeable systematic differences between the data in the two different projects. In light of this, the decision to keep the healthy data out of both projects separated is well justified. Most notably the differences between HSC and CMP cells could have caused issues in interpreting the results.

4.3 Conformation scoring

It is assumed in this discussion that at least the differentiation of the myeloid line occurs according to the classical model of hematopoiesis. Although this was covered in the Introduction, the steps in myeloid differentiation according to the classical model will be quickly outlined here again.

According to this model, myeloid differentiation starts from the HSC cell which in turn differentiates into a CMP cell. The CMP cell, as the full name suggests, is a common ancestral cell type to all the cell types comprising the myeloid line. As such, it has a similar position as the CLP cell (Common Lymphoid Progenitor) which itself also descends from a HSC cell and is ancestral to all the cell types in the lymphoid line. The CMP cell can further differentiate into either a GMP cell or an MEP cell. This means that the conformation that should score the best, given that the method is able to recover the truth, is the combination of the following edges: hsc-cmp, cmp-gmp and cmp-mep. In the discussion that follows the focus will be on the top three best scoring conformations in each of the datasets and for each of the subnetwork identification methods.

In Healthy set 1, the expected conformation does not appear among the top three of conformations for the three subnetwork identification methods. The best ranking for the assumed conformation is 9 when using the Transpath database. Although this seems to be a poor result, the top threes of both the Transpath-based and R-MCL based conformations contain a conformation that puts CMP in between HSC and the more mature progenitors GMP and MEP.

In Healthy set 2, the correct conformation did appear in all the top three results, making this a very favourable result. In addition, all conformations contain the edge between HSC and CMP. In this case the differences between Healthy set 1 and Healthy set 2 show up again. While the edge between HSC and CMP was present in all conformations in the top three for Healthy set 2, it appears only twice in the top three of Healthy set 1.

Interestingly, in the MM set the expected conformation ranks number one when using subnetworks based on METIS. For the other subnetworks the results are not very good, especially in the top three based on R-MCL where the edge between HSC and CMP does not appear in any of the conformations among the top three. In the case of the Transpath based conformation the situation is slightly better, the number one ranking conformation does place CMP in between HSC and GMP and MEP. In addition, the expected conformation is ranked sixth with only a minor difference to the conformation ranked third when it comes to multiple changes.

An unexpected observation is that the amount of subnetworks that change along all edges (in this case three times) is rather constant in all conformations in each dataset. Those subnetworks are the same for all conformations in the respective datasets, including the top scoring conformations. This means they add no information to the analysis. This is interesting in a biological sense since it could mean that they are pathways that are actively involved in myeloid differentiation because they are differentially regulated in each cell type. The Transpath-based conformations allow for a quick verification of these subnetworks since they are established pathways. The biology behind the pathways will be briefly covered here. The following lists show the Transpath pathways that change along every edge in the respective datasets:

Healthy set 1

* CD14 pathway
* Bub1 pathway
* Smurf-1 pathway

CML set

* Bub1 pathway
* NICD pathway

Healthy set 2

* E2F pathway
* SHP-1 pathway
* Aurora-A pathway
* Beta Catenin pathway
* APP pathway
* Usp7 pathway
* Caspase 8 pathway

* Cyclin D pathway

* PI3K pathway

* Bub1 pathway

MM set

* ZAP70 pathway

* Usp7 pathway

* Beta catenin pathway

* Caspase 8 pathway

Here there is a small overlap between Healthy set 1 and Healthy set 2: the Bub1 pathway. This pathway is involved in cell cycle regulation, more specifically the spindle assembly check point (Williams et al., 2007). Additionally, the list for Healthy set 1 contains the pathway involving Smurf-1 which in involved in cell adhesion and mobility (Wang, 2010). It is possible that this pathway plays a role in release and retention from the stem cell niche.

Interestingly, apart from the Bub1 pathway, the pathway around NICD also seems heavily regulated in CML. NICD is formed as a result of the cleavage of the Notch receptor. This pathway is of high importance for stem cell maintenance in hematopoiesis, and signalling form this receptor is a consequence of niche interactions (Porter and Calvi, 2008). The fact that this pathway is differentially expressed throughout differentiation in CML also means it remains active throughout differentiation which may, in part, explain the stem cell like behaviour of CML cells. Although Bub1 seems to be heavily regulated in both healthy sets in addition to the CML set, this does not necessarily mean that this regulation has the same cause and effect. It was shown that in CML cells with the BCR-ABL fusion gene expression of the Bub1 pathway is negatively influenced (Wolanin et al., 2010). This influence may in part explain the increased mutation rate in CML cells.

In the pathway listing for Healthy set 2 there are some interesting entries. SHP-1 is a tyrosine phospathase involved in hematopoietic differentiation (Wu et al., 2003). The beta catenin pathway is part of the important niche interactions and is required for maintenance of hematopoietetic progenitors (Nemeth et al., 2009). In this light one would expect this pathway to be regulated throughout hematopoiesis. The Aurora-A pathway is related to the function of the Bub1 pathway. While Bub1 performs quality control on spindle formation, Aurora-A performs quality control on successful segregation of chromatin (Libertini et al., 2010). Cyclin D is an important player in cell cycle regulation. This specific variety of cyclin controls transition into the S-phase (Assoian and Klein, 2008). The PI3K pathway and the Caspase 8 pathway both play a role in inducing apoptosis (Duronio, 2008; Ghavami et al., 2009), regulation of apoptosis is expected in differentiation. For the APP pathway, no specific connection to hematopoiesis could be found, however this particular protein has been linked to glial cell differentiation (Kwak et al., 2010) which could mean it also plays a role in hematopoietic differentiation. Usp7, also called HAUSP, is an ubiquitine ligase that interacts with a variety of targets and in doing so influences the activity of the important tumor suppressor p53 (Shan et al., 2008). The E2F pathway has similar tumour suppressor activity and is an important regulator of cellular proliferation (van den Heuvel and Dyson, 2008). It is possible that there is cross talk between the E2F pathway and the p53 pathway (Polager and Ginsberg, 2009). Taken together the findings in Healthy set 2 are in line with current knowledge which expects these pathways to be actively regulated throughout differentiation.

Comparing the pathways listed for Healthy set 2 to those listed for the MM set there is a large overlap, although clearly some important players are missing. Interestingly ZAP70 is in the list for MM. This particular protein is associated with the lymphoid line and seems to be expressed by all types of B-cells independant of maturity stage and whether or not they are malignant (Scielzo et al., 2006). This is highly interesting in this case because MM is primarily a malignancy of the B-cell population. It is especially peculiar to see this pathway here, since the cells analysed are from the myeloid line. It would appear that this observation is in line with the recent observations that the distinction between the myeloid and lymphoid line is not as strict as assumed by the classic model of hematopoiesis.

When comparing the pathways listed for Healthy set 1 to those listed for the CML set the most striking difference is the absence of the CD14 pathway in the CML set list. The CD14 pathway is specific for the myeloid lineage and influences the differentiation to macrophages upon contact with LPS. Looking deeper into the changes reveals that the CD14 pathway does not change along any edge (Jerala, 2007; Seta and Kuwana, 2010). Looking back at the principles outlined in the method section this could mean two things: either the pathway is constantly active but never regulated, or the pathway was already silenced at the HSC stage and remained so during differentiation.

From this brief look into pathways that are regulated differently in all the cell types of the differentiation, it can be concluded that these pathways are indeed either lineage-specific or are housekeeping pathways that perform tasks required in the differentiation process, in this case cell cycle regulation pathways.

Although the aim of this work is not to identify pathways or subnetworks that are specific to differentiation, the change vectors do offer the possibility to look into differences in regulation between healthy differentiation and differentiation in malignancies. An important step in hematopoietic differentiation is the step from HSC to CMP; differences in this step will be briefly covered here. To ease interpretation only the Transpath pathways were considered. Healthy set 1 was compared to the CML set, and Healthy set 2 was compared to the MM set. Comparison of the edge between HSC and CMP for Healthy set 1 and CML set revealed that 5 Transpath pathways were specifically regulated in CML, whereas no regulation was seen in the healthy set. Those pathways were the pathways: procaspase 9, GATA3, ATF-2, trkA and AF6. Caspases are important for apoptosis, a system that is often deregulated in cancers (Ghavami et al., 2009). The transcription factor GATA3 has recently also been linked to cancer progression (Chou et al., 2010). The transcription factor ATF-2 activates a variety of targets, although its role in cancer is somewhat ambiguous. There are indications that ATF-2 may be involved in cancer maintenance (Vlahopoulos et al., 2008). TrkA, although strictly speaking a receptor for nerve growth factor, was shown to be expressed on CML cells transformed by the BCR-ABL fusion gene making this a very interesting result (Koch et al., 2008). The activity of AF6 is related to cell adhesion. Overexpression of this particular gene decreases cell adhesion (Zhang et al., 2005), this is very interesting in relation to the importance of the niche in hematopoietic differentiation. In MM, 21 pathways are regulated in MM that are not regulated in healthy differentiation. Among those are some interesting pathways: the SOCS family, PDGF and p300. The SOCS family has a negative effect on signalling in several cytokine pathways, most importantly the JAK/STAT pathway. At least for SOCS1 there is evidence that this gene is hypermethylated, and therefore silenced, in MM (Galm et al., 2003). PDGF is a growth factor. Greco et al. (2006) show that the interaction of the BB form of PDGF with c-MYC can confer resistance to the chemotherapeutic agent melphalan in MM cells, indicating the possible importance of PDGF in MM. p300 is both a transcriptional co-factor and a histone acetyl transferase. Although there doesn't appear to be any specific function of this protein in MM, it has been implicated in several malignancies and is involved in known tumor-related pathways such as the pathways around p53 and TGF-beta (Lyer et al., 2004).

4.4 Correlation entropy

The results from the correlation message entropy are encouraging. The graphs are in accordance with what was stated in the Method section. It would seem that there is in fact an entropy decrease between HSC cells and the more mature progenitors GMP and MEP. This would mean that the behaviour observed in the gene clusters mentioned in the method section can indeed be extended to whole microarrays and is not a phenomenon specific to certain groups of genes. All but the CML set show a clear downward trend starting at the HSC. More important perhaps, is that there is a clear distinction between HSC and CMP on the one side and GMP and MEP on the other. This is interesting, because both groups form distinct developmental stages. It should be noted that in the entropy analysis there don't appear to be very noticeable differences between both healthy sets. This is in sharp contrast with the change vector-related analysis, where there were substantial differences between both datasets. The CML set shows clearly divergent behaviour, here the situation seems to have reversed: the more mature progenitors have a higher entropy than the HSC and CMP cells. Assuming that the observed trend in both healthy sets and the MM set are caused by changes due to differentiation, it could be concluded that differentiation is indeed impaired in patients with CML. In addition it could indicate that all the cell types have retained stem cell like behaviour and characteristics. The results for both the MM set and the CML set seem to agree with what is known about both diseases. Bruns et al. (2009) showed that in CML progenitors do retain stem cell-like qualities Interestingly the GMP fraction exhibits the most stem cell-like behaviour, this is in accordance with the results of the entropy method. For MM it could be found that this is very close to the normal situation. This also seems to hold because MM is a malignancy that manifests itself in B-cells. It is therefore likely that normal differentiation and the correct loss of stem cell characteristics in not severely impacted by this disease. Nevertheless, MM plasma cells do influence the niche and so may interfere with differentiation.

As was mentioned in the Method section, the idea to use this correlation entropy came from observing this peculiar behaviour in an unrelated clustering experiment. While observing colour-coded images of the correlation matrices, more accurately the correlation distance matrices, it can be seen that the image complexity of those images decreases with progressing differentiation. Here it was shown that this behaviour was not limited to the gene clusters analysed in that experiment, but in fact seems to be a characteristic of the entire microarray and, by extension, for the part of the transcriptome represented by it. However, it is not at all clear why this is the case. The decrease in entropy indicates that in the later stages more and more matrix elements start taking on the same value. Or more general, that the probability distributions in the individual matrices starts narrowing around certain values.

An attractive possibility is that due to the increased silencing the overall interaction complexity in the network decreases. Since the increasing silencing should cause the amount of active genes to decrease, one would expect that the overall interaction possibilities should also decrease. In this sense correlation coefficients allow for an increased flexibility because a strong correlation does not imply a direct interaction. Given the optimised communication within the cellular network, the correlation profile also captures non-direct and long-range effects. Unfortunately it did not prove possible to analyse this in more detail at this given time. It would be desirable to analyse time-resolved data for this dynamic process, rather than the static data available for this study. The interpretation is also further complicated because of the question what the correlation between two silenced genes should be. Using the correlation entropy to identify the degree of maturity in differentiation should therefore be considered as a grey box approach.

4.5 General conclusions

For the last part of the discussion the focus will be on the results of the complete method. The top three scoring conformations rooted by means of the entropy method in each dataset are illustrated in figures 3.11, 3.12, 3.13 and 3.14 in the Results section. From now on these conformations will be referred to as differentiation trees to indicate the combination of the results of change vector-based conformation scoring and correlation entropy.

Validating the performance of the method is complicated in this case by the differences in both healthy datasets. For Healthy set 1, the expected differentiation tree could not be identified at all, while in Healthy set 2 it could be identified regardless of the subnetwork identification method used. Probably more important than this is that there are almost no similarities in the top three sets of both datasets. This divergence, and the lack of a third dataset with healthy samples, makes it difficult to decide which dataset is telling the truth. This makes evaluating the method based on the data alone impossible.

An alternative method would be to look at how well the method manages to identify the expected differentiation tree. The tree is referred to as expected and not correct because, as was mentioned in the Introduction, it is possible that the classical model of differentiation may not be correct. Nevertheless, we will assume here that the expected differentiation tree according to the classical model is indeed the correct one. Based on this fact it can be concluded that the method performed well. It is of interest whether the appearances of the correct differentiation tree among the top-three set is due to some random chance event. This question can be easily answered by considering the presence of the correct differentiation tree in the top three as a Bernouili trial. The chance for the correct differentiation tree to appear in the top three is 3 over 16. If all the results are considered, all data sets using the three different subnetwork identification methods, there are 12 trials and 4 successes. This puts the p-value at 0.2561. This can be made more favourable by excluding Healthy set 1 based on the differences with Healthy set 2 and the fact that in Healthy set 2 it was possible to identify the correct differentiation tree regardless of the subnetwork identification method used. Excluding Healthy set 1 brings the p-value to a substantially lower 0.07003. In this case it is not possible to get really low P-values due to the limited number of possible conformations.

Some biological evidence can be gathered from the brief look at the pathways that are regulated differently in each cell type. As mentioned earlier, those pathways are either lineage-specific or perform some kind of general utility function, as one would expect in differentiation such as cell cycle regulation.

The method should be evaluated against the questions posed at the beginning of this work. The main question was if microarray data contains enough information to analyse and reconstruct the differentiation tree given only transcriptome data of the individual cells. Based on the data presented it would appear that microarrays do indeed contain useful information specific to the differentiation process. Even though the correct differentiation tree could not be identified in some cases, the trees that the method suggested were usually close to the correct one, meaning that they positioned the CMP cells in between the HSC cells and the GMP and MEP cells.

The other two questions are similar in nature: can this information be used to verify proposed differentiation paths and if so can it also be used to spot changes in differentiation. Validation of proposed differentiation trees may not be possible with the method given the divergent results between both healthy datasets. It was not possible to fully validate the method on the data used here. On the other hand, it was possible to identify differences between normal differentiation and differentiation in cancer. The clearest differences come from the correlation entropy analysis, the CML cells really stand out here with a practically reversed curve.

The method presented here should be approached as a proof of concept rather than a fully functional analysis tool. In the outlook section improvements and suggestions for further development will be discussed. Nevertheless, it was shown that differentiation trees can indeed be reconstructed based on transcriptome data alone. In addition, the correlation entropy-based method for root node identification could in principle be deployed as a stand alone method, it could potentially be adapted as a diagnostic method for certain cancers such as was demonstrated here for CML. Further, aside from its main goal of identifying correct differentiation trees the method also managed to identify pathways and subnetworks that are actively regulated and involved in the differentiation process. The work presented here may further the understanding of the differentiation process and could lead to the development of new diagnostic tools.

4.6 Outlook

An interesting path of development to follow is to convert the method from a non-parametric method to a parametric method. This is similar to how maximum parsimony and maximum likelihood compare in the case of phylogenetics and combines nicely with the evolution theme the method is based on. Central to maximum likelihood is the existence of some probability of change. The previous discussions already hinted at the fact that the chance of changing or becoming silenced along an edge is likely to be somewhat subnetwork- and context-specific. It was observed that some subnetworks do not change at all while some subnetworks change in each cell type. It is also possible that changes in one subnetwork influence the change probabilities of other subnetworks. A simple example would be that a pathway for neural cell-specific signalling should not be expressed in cells that have entered the myeloid line and have active hematopoiesis-specific pathways. Although this system of conditional change probabilities is clearly more elaborate than the change probabilities in phylogentics, there is an analogy to be made with the so called mutation hot spots in the genome. Clearly this requires a substantial amount of data to learn these complicated probability interactions, and preferably in vitro-data. Generating this data is not a trivial task, however.

Another aspect from the method that could benefit from parameterisation is the correlation entropy decay. Here it would be helpful to have a function that defines the expected entropy decrease. If the expected trend is known, observed entropy values can be better evaluated based on confidence intervals. The critical reader may have noticed that for the generation of the differentiation trees for Healthy set 2, HSC was chosen as root node, while the entropy of CMP was slightly higher. This decision was made based on the general trend in the other graphs and prior knowledge about the system. With a known expected entropy decrease it would have been easier to make this decision more accurately.

What would benefit the method and the understanding of the underlying principles the most is of course methylation data. The method essentially tests for silencing methylation indirectly by looking at the amount of changes. This causes practical problems when interpreting what it means if a subnetwork does not change along an edge: it could be silenced or it could be active and not regulated. When looking at methylation data directly, this duality does not exist. Additionally, using methylation data makes the method more similar to maximum parsimony and maximum likelihood since there would be a vector per node instead of a vector per edge. The transcriptome can still be viewed as an array of subnetworks, but in this case the vectors will indicate whether or not a subnetwork is silenced by methylation. The core principle that methylation should increase with differentiation can still be used to score the conformations in that case. The main difference would be that the method analyses the methylome and not the transcriptome.

Bibliography

A. Abou-Rjeili and G. Karypis. Multilevel algorithms for partitioning power-law graphs. *IEEE International Parallel & Distributed Processing Symposium*, 2006.

R. K. Assoian and E. A. Klein. Growth control by intracellular tension and extracellular stiffness. *Trends Cell Biol*, 2008.

A. L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 1999.

A. Bird. Dna methylation patterns and epigenetic memory. *Genes and Development*, 2002.

I. Bruns, A. Czibere, JC. Fischer, F. Roels, RP. Cadeddu, S. Buest, D. Bruennert, AN. Huenerlituerkoglu, NH. Stoecklein, R. Singh, LF. Zerbini, M. Jger, G. Kobbe, N. Gattermann, R. Kronenwett, B. Brors, and R. Haas. The hematopoietic stem cell in chronic phase cml is characterized by a transcriptional profile resembling normal myeloid progenitor cells and reflecting loss of quiescence. *Leukemia*, 2009.

B. A. Burke and M. Carroll. Bcr-abl: a multi-faceted promoter of dna mutation in chronic myelogeneous leukemia. *Leukemia*, 2010.

J. Chou, S. Provot, and Z. Werb. Gata3 in development and cancer differentiation: cells gata have it. *J Cell Physiol*, 2010.

J. R. David. Evolution and development: some insights from evolutionary theory. *An Acad Bras Cienc*, 2001.

M. A. Dimopoulos, E. Kastritis, and A. Anagnostopoulos. Hematological malignancies: myeloma. *Anals of oncology*, 2006.

V. Duronio. The life of a cell: apoptosis regulation by the pi3k/pkb pathway. *Biochem J*, 2008.

N. Felli, L. Cianetti, E. Pelosi, A. Care, C. G. Liu, G. A. Calin, S. Rossi, C. Peschle, G. Marziali, and A. Giuliani. Hematopoietic differentiation: a coordinated dynamical process towards attractor stable states. *BMC Systems Biology*, 2010.

E. C. Forsberg and S. Smith-Berdan. Parsing the niche code: the molecular mechanisms governing hematopoietic stem cell adhesion and differentiation. *Haematologica*, 2009.

S. D. Foster, S. H. Oram, N. K. Wilson, and B. Gotgens. From genes to cells to tissues–modelling the haematopoietic system. *Mol Biosyst*, 2009.

R. Frazer, A. E. Irvine, and M. F. McMullin. Chronic myeloid leukaemia in the 21st century. *Ulster Med J*, 2007.

O. Galm, H. Yoshikawa, M. Esteller, R. Osieka, and J. G. Herman. Socs-1, a negative regulator of cytokine signaling, is frequently silenced by methylation in multiple myeloma. *Blood*, 2003.

S. Ghavami, M. Hashemi, S. R. Ande, B. Yaganeh, W. Xiao, M. Eshragi, C. J. Bus, K. Kadkhoda, E. Wiechec, A. J. Halayko, and M. Los. Apoptosis and cancer: mutations within caspase genes. *J Med Genet*, 2009.

B. Giebel and M. Punzel. Lineage development of hematopoietic stem and progenitor cells. *Biol. Chem.*, 2008.

T. Giger, P. Khaitovich, M. Somel, A. Lorenc, E. Lizanot, L. W. Harris, M. M. Ryan, M. Lan, M. T. Wayland, S. Bahn, and S. Paabo. Evolution of neuronal and endothelial transcriptomes in primates. *Genome Biol. Evol.*, 2010.

J. J. Goeman, S. A. van de Geer, F. de Kort, and J. C. van Houwelingen. A global test for groups of genes: testing association with a clinical outcome. *Bioinformatics*, 2004.

C. Greco, I. Dagano, G. Vitelli, R. Vona, M. Marino, M. Mottolese, C. Zuppi, E. Capaluongo, and F. Ameglio. c-myc deregulation is involved in melphalan resistance of multiple myeloma: role of pdgf-bb. *Int J Immunopathol Pharmacol*, 2006.

C. H. Jamieson. Chronic myeloid leukemia stem cells. *Hematology Am Soc Hematol Educ Program*, 2008.

L. J. Jensen, M. Kuhn, M. Stark, S. Chaffron, C. Creevey, J. Muller, T. Doerks, P. Julien, A. Roth, M. Simonovic, P. Bork, and C. von Mering. String 8–a global view on proteins and their functional interactions in 630 organisms. *Nucleic acid res*, 2009.

R. Jerala. Structural biology of the lps recognition. *Int J Med Microbiol*, 2007.

H. Kawamoto, H. Wada, and Y. Katsura. A revised scheme for developmental pathways of hematopoietic cells: the myeloid-based model. *International Immunology*, 2010.

A. Koch, M. Scherr, B. Breyer, A. Mancini, C. Kardinal, K. Battmer, M. Eder, and T. Tamura. Inhibition of abl tyrosine kinase enhances nerve growth factor-mediated signaling in bcr-abl transformed cells via the alteration of signaling complex and the receptor turnover. *Oncogene*, 2008.

Y. D. Kwak, E. Dantuma, S. Merchant, S. Bushnev, and K. Sugaya. Amyloid-beta precursor protein induces glial differentiation of neural progenitor cells by activation of the il-6/gp130 signaling pathway. *Neurotox Res*, 2010.

S. Libertini, A. Abagnale, C. Passaro, G. Botta, and G. Portella. Aurora a and b kinases - targets of novel anticancer drugs. *Recent pat anticancer drug discov*, 2010.

D. J. Lockhart, H. Dong, M. C. Byrne, M. T. Follettie, M. V. Gallo, M. S. Chee, M. Mittman, C. Wang, M. Kobayashi, H. Horton, and E. L. Brown. Expression monitoring by hybridization to high-density oligonucleotide arrays. *Nat Biotechnol*, 1996.

N. G. Lyer, H. Ozdag, and C. Caldas. p300/cbp and cancer. *Oncogene*, 2004.

S. Milgram. An experimental study of the small world problem. *Sociometry*, 1969.

F. Mohn and D. Schubeler. Genetics and epigenetics: Stability and plasticity during cellular differentiation. *Trends in Genetics*, 2009.

M. J. Nemeth, K. K. Mak, Y. Yang, and D. M. Bodine. beta-catenin expression in the bone marrow microenvironment is required for long-term maintenance of primitive hematopoietic cells. *Stem Cells*, 2009.

S. Polager and D. Ginsberg. p53 and e2f: partners in life and death. *Nat Rev Cancer*, 2009.

R. L. Porter and L. M. Calvi. Communications between bone cells and hematopoietic stem cells. *Arch Biochem Biophys*, 2008.

A. Rizo, E. Vellenga, G. de Haan, and J. J. Schuringa. Signaling pathways in self-renewing hematopoietic and leukemic stem cells: do all stem cells need a niche? *Hum Mol Genet*, 2006.

A. J. Ruthenburg, C. D. Allis, and J. Wysocka. Methylation of lysine 4 on histone h3: intricacy of writing and reading a single epigenetic mark. *Molecular cell*, 25, 2007.

V. Satuluri and S. Parthasarathy. Scalable graph clustering with stochastic flows: applications to community discovery. *SIGKDD*, 2009.

C. Scielzo, A. Camporeale, M. Guena, M. Alessio, A. Poggi, M. R. Zocchi, M. Chilosi, F. Caligaris-Capio, and P. Ghia. Zap-70 is expressed by normal and malignant human b-cell subsets of different maturational stage. *Leukemia*, 2006.

N. Seta and M. Kuwana. Derivation of multipotent progenitors from human circulating cd14+ monocytes. *Exp Hematol*, 2010.

O. Sezer. Myeloma bone disease: recent advances in biology, diagnostics and treatment. *The Oncologist*, 2009.

J. Shan, C. Brooks, N. Kon, M. Li, and W. Gu. Dissecting roles of ubiquitination in the p53 pathway. *Ernst Shering Found Symp Proc*, 2008.

C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 1948.

R. J. Sims, R. Belotserkovskaya, and D. Reinberg. Elongation by rna polymerase ii: the short and long of it. *Genes & Development*, 18:2437–2468, 2004.

M. D. Stewart, J. Li, and J. Wong. Relationship between histone h3 lysine 9 methylation, transcription repression, and heterochromatin protein 1 recruitment. *Molecular and cellular biology*, 25, 2005.

J. E. Sulston and H. R. Horvitz. Post-embryonic cell lineages of the nematode, caenorhabditis elegans. *Dev Biol*, 1977.

S. Sung and R. M. Amasino. Vernalization and epigenetics: how plants remember winter. *Curr Opin Plant Biol*, 2004.

J. J. Towbridge, J. W. Snow, J. Kim, and S. H. Orkin. Dna methyltransferase 1 is essential for and uniquely regulates hematopoietic stem and progenitor cells. *Cell Stem Cell*, 2009.

S. van den Heuvel and N. J. Dyson. Conserved functions of the prb and e2f families. *Nat Rev Moll Cell Biol*, 2008.

V. E. Velculescu, L. Zhang, B. Vogelstein, and K. W. Kinzler. Serial analysis of gene expression. *Science*, 1995.

S. A. Vlahopoulos, S. Logotheti, D. Mikas, A. Giarika, V. Gorgoulis, and V. Zoumpourlis. The role of atf-2 in oncogenesis. *Bioessays*, 2008.

C. Wang. Roles of e3 ubiquitin ligases in cell adhesion and migration. *Cell Adh Migr*, 2010.

G. L. Williams, T. M. Roberts, and O. V. Gjoerup. Bub1: escapades in a cellular world. *Cell Cycle*, 2007.

K. Wolanin, A. Magalska, M. Kusio-Kobialka, P. Podszywalow-Bartnicka, S. Vejda, S. L. McKenna, G. Mosieniak, E. Sikora, and K. Piwocka. Expression of oncogenic kinase bcr-abl impairs mitotic checkpoint and promotes aberrant divisions and resistance to microtubule-targeting agents. *Mol Cancer Ther*, 2010.

C. Wu, M. Sun, L. Liu, and G. W. Zhou. The function of the protein tyrosine phosphatase shp-1 in cancer. *Gene*, 2003.

L. Zhang, M. F. Miles, and K. D. Aldape. A model of molecular interactions on short oligonucleotide microarrays. *Nat Biotechnol*, 2003.

Z. Zhang, H. Rehmann, L. S. Price, J. Riedl, and J. L. Bos. Af6 negatively regulates rap1-induced cell adhesion. *J Biol Chem*, 2005.

Appendix A

Tables

Edges	Total	Twice	Thrice
Healthy set 1: cmp-mep gmp-mep hsc-gmp	5	1	4
Healthy set 1: cmp-mep gmp-mep hsc-mep	6	2	4
Healthy set 1: cmp-gmp gmp-mep hsc-mep	6	2	4
Healthy set 1: cmp-gmp gmp-mep hsc-gmp	6	2	4
Healthy set 1: cmp-gmp cmp-mep hsc-mep	7	3	4
Healthy set 1: cmp-gmp gmp-mep hsc-cmp	8	4	4
Healthy set 1: cmp-gmp cmp-mep hsc-gmp	8	4	4
Healthy set 1: cmp-mep hsc-gmp hsc-mep	9	5	4
Healthy set 1: cmp-gmp hsc-gmp hsc-mep	10	6	4
Healthy set 1: cmp-mep gmp-mep hsc-cmp	12	8	4
Healthy set 1: cmp-gmp cmp-mep hsc-cmp	15	10	5
Healthy set 1: gmp-mep hsc-cmp hsc-gmp	21	17	4
Healthy set 1: cmp-mep hsc-cmp hsc-gmp	28	24	4
Healthy set 1: gmp-mep hsc-cmp hsc-mep	38	34	4
Healthy set 1: cmp-gmp hsc-cmp hsc-mep	40	36	4
Healthy set 1: hsc-cmp hsc-gmp hsc-mep	51	43	8

Table A.1: Ranked conformation scores for Healthy set 1 using METIS

Edges	Total	Twice	Thrice
Healthy set 1: cmp-gmp gmp-mep hsc-mep	7	4	3
Healthy set 1: cmp-mep gmp-mep hsc-gmp	9	6	3
Healthy set 1: cmp-gmp gmp-mep hsc-cmp	9	6	3
Healthy set 1: cmp-gmp cmp-mep hsc-gmp	9	6	3
Healthy set 1: cmp-mep gmp-mep hsc-mep	11	8	3
Healthy set 1: cmp-gmp cmp-mep hsc-mep	11	8	3
Healthy set 1: cmp-gmp gmp-mep hsc-gmp	11	8	3
Healthy set 1: cmp-mep gmp-mep hsc-cmp	18	15	3
Healthy set 1: cmp-gmp cmp-mep hsc-cmp	20	17	3
Healthy set 1: cmp-gmp hsc-gmp hsc-mep	29	26	3
Healthy set 1: cmp-mep hsc-gmp hsc-mep	31	28	3
Healthy set 1: gmp-mep hsc-cmp hsc-gmp	35	32	3
Healthy set 1: cmp-mep hsc-cmp hsc-gmp	44	41	3
Healthy set 1: gmp-mep hsc-cmp hsc-mep	60	57	3
Healthy set 1: cmp-gmp hsc-cmp hsc-mep	62	59	3
Healthy set 1: hsc-cmp hsc-gmp hsc-mep	70	45	25

Table A.2: Ranked conformation scores for Healthy set 1 using Transpath

Edges	Total	Twice	Thrice
Healthy set 1: cmp-gmp cmp-mep hsc-gmp	63	12	51
Healthy set 1: cmp-gmp gmp-mep hsc-cmp	64	13	51
Healthy set 1: cmp-mep gmp-mep hsc-gmp	66	15	51
Healthy set 1: cmp-gmp gmp-mep hsc-mep	67	16	51
Healthy set 1: cmp-gmp hsc-gmp hsc-mep	68	17	51
Healthy set 1: cmp-gmp gmp-mep hsc-gmp	69	18	51
Healthy set 1: gmp-mep hsc-cmp hsc-gmp	70	19	51
Healthy set 1: cmp-mep gmp-mep hsc-cmp	81	30	51
Healthy set 1: cmp-gmp cmp-mep hsc-mep	81	30	51
Healthy set 1: cmp-gmp cmp-mep hsc-cmp	85	33	52
Healthy set 1: cmp-mep hsc-cmp hsc-gmp	85	34	51
Healthy set 1: cmp-mep hsc-gmp hsc-mep	86	35	51
Healthy set 1: cmp-mep gmp-mep hsc-mep	88	37	51
Healthy set 1: cmp-gmp hsc-cmp hsc-mep	95	44	51
Healthy set 1: gmp-mep hsc-cmp hsc-mep	98	47	51
Healthy set 1: hsc-cmp hsc-gmp hsc-mep	106	50	56

Table A.3: Ranked conformation scores for Healthy set 1 using R-MCL

Edges	Total	Twice	Thrice
CML set : cmp-gmp gmp-mep hsc-mep	2	0	2
CML set : cmp-gmp hsc-gmp hsc-mep	3	1	2
CML set : cmp-gmp cmp-mep hsc-gmp	4	2	2
CML set : gmp-mep hsc-cmp hsc-mep	4	2	2
CML set : cmp-gmp cmp-mep hsc-mep	4	2	2
CML set : cmp-mep hsc-gmp hsc-mep	5	3	2
CML set : cmp-mep gmp-mep hsc-mep	5	3	2
CML set : hsc-cmp hsc-gmp hsc-mep	5	3	2
CML set : gmp-mep hsc-cmp hsc-gmp	6	3	3
CML set : cmp-gmp gmp-mep hsc-cmp	6	4	2
CML set : cmp-gmp hsc-cmp hsc-mep	6	4	2
CML set : cmp-gmp gmp-mep hsc-gmp	6	4	2
CML set : cmp-mep gmp-mep hsc-gmp	7	4	3
CML set : cmp-mep hsc-cmp hsc-gmp	10	7	3
CML set : cmp-mep gmp-mep hsc-cmp	11	8	3
CML set : cmp-gmp cmp-mep hsc-cmp	12	9	3

Table A.4: Ranked conformation scores for the CML set using METIS

Edges	Total	Twice	Thrice
CML set : cmp-gmp hsc-gmp hsc-mep	3	1	2
CML set : cmp-gmp cmp-mep hsc-gmp	5	3	2
CML set : gmp-mep hsc-cmp hsc-gmp	6	4	2
CML set : cmp-gmp gmp-mep hsc-cmp	6	4	2
CML set : hsc-cmp hsc-gmp hsc-mep	6	4	2
CML set : cmp-mep hsc-cmp hsc-gmp	7	5	2
CML set : cmp-gmp hsc-cmp hsc-mep	7	5	2
CML set : cmp-mep gmp-mep hsc-gmp	8	5	3
CML set : cmp-gmp gmp-mep hsc-gmp	8	6	2
CML set : cmp-mep gmp-mep hsc-cmp	9	7	2
CML set : cmp-gmp gmp-mep hsc-mep	9	7	2
CML set : cmp-gmp cmp-mep hsc-cmp	10	8	2
CML set : gmp-mep hsc-cmp hsc-mep	10	8	2
CML set : cmp-mep hsc-gmp hsc-mep	12	10	2
CML set : cmp-gmp cmp-mep hsc-mep	12	10	2
CML set : cmp-mep gmp-mep hsc-mep	18	16	2

Table A.5: Ranked conformation scores for the CML set using Transpath

Edges	Total	Twice	Thrice
CML set : cmp-gmp cmp-mep hsc-gmp	47	13	34
CML set : cmp-gmp gmp-mep hsc-cmp	48	14	34
CML set : cmp-gmp gmp-mep hsc-mep	48	14	34
CML set : cmp-gmp hsc-gmp hsc-mep	48	14	34
CML set : gmp-mep hsc-cmp hsc-gmp	49	15	34
CML set : cmp-mep gmp-mep hsc-gmp	49	15	34
CML set : cmp-mep gmp-mep hsc-cmp	50	16	34
CML set : cmp-mep hsc-cmp hsc-gmp	50	16	34
CML set : cmp-gmp cmp-mep hsc-mep	50	16	34
CML set : cmp-mep hsc-gmp hsc-mep	51	17	34
CML set : gmp-mep hsc-cmp hsc-mep	51	17	34
CML set : cmp-gmp hsc-cmp hsc-mep	51	17	34
CML set : cmp-gmp gmp-mep hsc-gmp	55	21	34
CML set : cmp-gmp cmp-mep hsc-cmp	57	22	35
CML set : cmp-mep gmp-mep hsc-mep	57	23	34
CML set : hsc-cmp hsc-gmp hsc-mep	57	23	34

Table A.6: Ranked conformation scores for the CML set using R-MCL

Edges	Total	Twice	Thrice
Healthy set 2: cmp-mep hsc-cmp hsc-gmp	51	34	17
Healthy set 2: cmp-gmp cmp-mep hsc-cmp	68	48	20
Healthy set 2: cmp-mep gmp-mep hsc-cmp	78	59	19
Healthy set 2: cmp-gmp hsc-cmp hsc-mep	79	57	22
Healthy set 2: hsc-cmp hsc-gmp hsc-mep	81	65	16
Healthy set 2: gmp-mep hsc-cmp hsc-mep	97	73	24
Healthy set 2: cmp-mep hsc-gmp hsc-mep	113	90	23
Healthy set 2: cmp-gmp cmp-mep hsc-mep	122	99	23
Healthy set 2: cmp-mep gmp-mep hsc-mep	124	78	46
Healthy set 2: gmp-mep hsc-cmp hsc-gmp	133	114	19
Healthy set 2: cmp-gmp gmp-mep hsc-cmp	142	118	24
Healthy set 2: cmp-mep gmp-mep hsc-gmp	162	139	23
Healthy set 2: cmp-gmp gmp-mep hsc-mep	180	141	39
Healthy set 2: cmp-gmp cmp-mep hsc-gmp	182	155	27
Healthy set 2: cmp-gmp hsc-gmp hsc-mep	195	159	36
Healthy set 2: cmp-gmp gmp-mep hsc-gmp	220	133	87

Table A.7: Ranked conformation scores for Healthy set 2 using METIS

Edges	Total	Twice	Thrice
Healthy set 2: cmp-mep hsc-cmp hsc-gmp	57	43	14
Healthy set 2: cmp-gmp cmp-mep hsc-cmp	79	65	14
Healthy set 2: hsc-cmp hsc-gmp hsc-mep	99	84	15
Healthy set 2: cmp-mep gmp-mep hsc-cmp	103	89	14
Healthy set 2: cmp-gmp hsc-cmp hsc-mep	108	85	23
Healthy set 2: cmp-mep hsc-gmp hsc-mep	156	124	32
Healthy set 2: gmp-mep hsc-cmp hsc-mep	161	137	24
Healthy set 2: cmp-gmp cmp-mep hsc-mep	177	143	34
Healthy set 2: gmp-mep hsc-cmp hsc-gmp	184	170	14
Healthy set 2: cmp-gmp gmp-mep hsc-cmp	206	182	24
Healthy set 2: cmp-mep gmp-mep hsc-mep	208	148	60
Healthy set 2: cmp-gmp cmp-mep hsc-gmp	232	187	45
Healthy set 2: cmp-mep gmp-mep hsc-gmp	236	200	36
Healthy set 2: cmp-gmp hsc-gmp hsc-mep	251	199	52
Healthy set 2: cmp-gmp gmp-mep hsc-mep	287	222	65
Healthy set 2: cmp-gmp gmp-mep hsc-gmp	294	152	142

Table A.8: Ranked conformation scores for Healthy set 2 using Transpath

Edges	Total	Twice	Thrice
Healthy set 2: cmp-mep gmp-mep hsc-cmp	111	37	74
Healthy set 2: cmp-mep hsc-cmp hsc-gmp	116	43	73
Healthy set 2: cmp-gmp cmp-mep hsc-cmp	122	49	73
Healthy set 2: gmp-mep hsc-cmp hsc-mep	126	50	76
Healthy set 2: cmp-gmp hsc-cmp hsc-mep	128	52	76
Healthy set 2: cmp-mep hsc-gmp hsc-mep	134	52	82
Healthy set 2: hsc-cmp hsc-gmp hsc-mep	135	59	76
Healthy set 2: cmp-gmp cmp-mep hsc-mep	137	59	78
Healthy set 2: cmp-mep gmp-mep hsc-mep	144	59	85
Healthy set 2: gmp-mep hsc-cmp hsc-gmp	163	85	78
Healthy set 2: cmp-gmp gmp-mep hsc-cmp	169	91	78
Healthy set 2: cmp-mep gmp-mep hsc-gmp	172	91	81
Healthy set 2: cmp-gmp gmp-mep hsc-mep	173	79	94
Healthy set 2: cmp-gmp cmp-mep hsc-gmp	193	112	81
Healthy set 2: cmp-gmp hsc-gmp hsc-mep	200	110	90
Healthy set 2: cmp-gmp gmp-mep hsc-gmp	211	83	128

Table A.9: Ranked conformation scores for Healthy set 2 using R-MCL

Edges	Total	Twice	Thrice
MM set: cmp-gmp cmp-mep hsc-cmp	13	4	9
MM set: cmp-gmp cmp-mep hsc-gmp	13	4	9
MM set: cmp-mep gmp-mep hsc-cmp	17	8	9
MM set: cmp-gmp gmp-mep hsc-cmp	17	7	10
MM set: cmp-gmp cmp-mep hsc-mep	18	8	10
MM set: cmp-mep gmp-mep hsc-gmp	19	10	9
MM set: cmp-mep hsc-cmp hsc-gmp	20	11	9
MM set: cmp-gmp gmp-mep hsc-gmp	21	11	10
MM set: cmp-gmp hsc-gmp hsc-mep	24	15	9
MM set: gmp-mep hsc-cmp hsc-gmp	26	17	9
MM set: cmp-mep hsc-gmp hsc-mep	27	18	9
MM set: cmp-gmp gmp-mep hsc-mep	31	20	11
MM set: cmp-mep gmp-mep hsc-mep	36	25	11
MM set: cmp-gmp hsc-cmp hsc-mep	37	28	9
MM set : hsc-cmp hsc-gmp hsc-mep	45	32	13
MM set: gmp-mep hsc-cmp hsc-mep	51	41	10

Table A.10: Ranked conformation scores for the MM set using METIS

Edges	Total	Twice	Thrice
MM set: cmp-mep gmp-mep hsc-cmp	6	1	5
MM set: cmp-gmp cmp-mep hsc-mep	6	1	5
MM set: cmp-mep gmp-mep hsc-gmp	10	5	5
MM set: cmp-mep hsc-gmp hsc-mep	11	6	5
MM set: cmp-mep gmp-mep hsc-mep	11	5	6
MM set: cmp-gmp cmp-mep hsc-cmp	13	8	5
MM set: cmp-gmp gmp-mep hsc-mep	13	8	5
MM set: gmp-mep hsc-cmp hsc-mep	13	8	5
MM set: cmp-gmp gmp-mep hsc-cmp	15	10	5
MM set: cmp-gmp hsc-cmp hsc-mep	15	10	5
MM set: cmp-mep hsc-cmp hsc-gmp	19	14	5
MM set: cmp-gmp cmp-mep hsc-gmp	19	14	5
MM set: hsc-cmp hsc-gmp hsc-mep	22	15	7
MM set: gmp-mep hsc-cmp hsc-gmp	23	18	5
MM set: cmp-gmp gmp-mep hsc-gmp	23	17	6
MM set: cmp-gmp hsc-gmp hsc-mep	24	19	5

Table A.11: Ranked conformation scores for the MM set using Transpath

Edges	Total	Twice	Thrice
MM set: cmp-mep gmp-mep hsc-gmp	63	14	49
MM set: cmp-gmp cmp-mep hsc-gmp	64	16	48
MM set: cmp-gmp cmp-mep hsc-mep	65	17	48
MM set: cmp-gmp gmp-mep hsc-cmp	66	18	48
MM set: cmp-gmp gmp-mep hsc-mep	67	19	48
MM set: cmp-mep gmp-mep hsc-cmp	69	21	48
MM set: cmp-gmp hsc-gmp hsc-mep	71	23	48
MM set: cmp-gmp gmp-mep hsc-gmp	71	22	49
MM set: gmp-mep hsc-cmp hsc-gmp	72	22	50
MM set: cmp-mep hsc-gmp hsc-mep	72	23	49
MM set: cmp-gmp cmp-mep hsc-cmp	73	25	48
MM set: cmp-mep hsc-cmp hsc-gmp	78	30	48
MM set: cmp-mep gmp-mep hsc-mep	78	27	51
MM set: cmp-gmp hsc-cmp hsc-mep	80	32	48
MM set: gmp-mep hsc-cmp hsc-mep	83	34	49
MM set: hsc-cmp hsc-gmp hsc-mep	90	38	52

Table A.12: Ranked conformation scores for the MM set using R-MCL

Appendix B

Source code

1 R-MCL source code

```
/*
rmcl-cuda by Ir Frederik Roels

Description:

This program is a parallelised implementation Regularised Markov graph clustering (R-MCL).
R-MCL is an addition made to normal Markov graph clustering by Satuluri. The improved algorithm
was presented by Venu Satuluri at SIGKDD 2009 in Paris under the title: Scalable graph
clustering using stochastic flows: applications for community discovery.

Regarding computation there are no differences between R-MCL and normal Markov graph
clustering. Both perform an expansion step and an inflation step. Computationally those
are equivalent to matrix multiplication and raising all elements in a vector to a given power.
These operations, especially the matrix multiplication, can consume incredible amounts of
computation time if the matrices are large. Since the algorithm expects to perform this
action at least several times in succession the time cost can become unmanageable
rather quickly.

Because of the nature of the operations, they can be easily parallelised. This program uses
the nVidia CUDA architecture to perform parallelisation on nVidia GPUs. GPUs are by nature
SIMD devices and are therefor suitable for parallelisation.

The code relies on the sgemm function from the CUBLAS library to perform matrix
multiplication. CUBLAS is a library containing a variety of linear algebra functions
ported to the CUDA architecture.

The other operations are parallelised by means of custom functions, in the CUDA world
referred to as kernels. Kernels are functions that are ran directly on the GPU, each thread
executes the same kernel, but may have different internal variables. Because of this the code
has two types of functions: those that are ran directly on the device and those that run on the
host machine. It is impossible (at least in the current CUDA versions) to access data in the
device memory directly from the host, accessing the data requires a memory copy operation.
Because it can be time expensive to perform this operation multiple times the data is kept on
the device until the computation has ended.

It should be noted that the CUBLAS library expects matrices to be in column major format
instead of row major format. Because of this, all functions expect the data to be in
column major format.



*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>

#include "/opt/cuda/include/cublas.h"


/*

An upper limit is hard coded, check your hardware for the optimal value, this one requires a
maximum of 4.8 gigabyte of video RAM and 1.6 gigabyte of host RAM

*/
```

```
#define MAX_MAT 20000 * 20000


// Datastructure used to hold the matrices

typedef struct Matrix {
        int length;
        int rows;
        int cols;
        float *data_ptr;
} Matrix;


// Function reading in a matrix from file

Matrix *read_matrix( char *filename, int rows, int cols ) {

        int i,j;
        int counter;
        float *data_ptr;
        float *data_ptr_resize;
        float *seek_ptr;
        float *data_ptr_colmajor;
        float *tmp_ptr;
        int total_size = rows * cols;
        Matrix *return_struct;

        FILE *f_ptr = fopen(filename, "r");
        if( !f_ptr ) {
                fprintf(stderr, "error opening file");
        }

// We’ll allocate as much as possible and later resize, because user input can’t be trusted

        data_ptr = (float *)malloc(MAX_MAT * sizeof(*data_ptr));

        if( !data_ptr ) {
                fprintf(stderr, "error allocating memory for %s\n", filename);
                exit(1);
        }

        seek_ptr = data_ptr;
        counter = 0;

        while( fscanf(f_ptr, "%f", seek_ptr) != EOF) {
                seek_ptr++;
                counter++;
         if( counter > MAX_MAT ) { fprintf(stderr, "Data too large\n");
          free(data_ptr); exit(1); }
        }

        if( counter != rows * cols ) {
         fprintf(stderr, "Dimensions dont fit size for matrix %s\n", filename); exit(1);
        }

        // Shrink the memory to size

        data_ptr_resize = (float *)realloc(data_ptr, counter * sizeof(*data_ptr));
        data_ptr_colmajor = (float *)malloc(counter * sizeof(*data_ptr));

        if( data_ptr_resize ) {
                data_ptr = data_ptr_resize;
        }
        else{
  fprintf(stderr, "Realloc failed\n"); free(data_ptr); free(data_ptr_resize); free(data_ptr_colmajor);
        }


/*
Here the matrices are converted from Row Major to Column Major, the CUBLAS library expects
Column Major
*/

        for( i = 0; i < rows; i++ ) {
                for( j = 0; j < cols; j++ ) {
                        *(data_ptr_colmajor + j * rows + i) = *(data_ptr + i * cols + j);
                }
        }

        tmp_ptr = data_ptr; // Hold on to that memory !!!
        data_ptr = data_ptr_colmajor;

        free(tmp_ptr);

        /* Return everything as a Matrix struct */

        return_struct = (Matrix *)malloc(sizeof(*return_struct));

        return_struct->length = counter;
        return_struct->rows = rows;
        return_struct->cols = cols;
        return_struct->data_ptr = data_ptr;


        return(return_struct);

}

/* Prints out the Column Major stored matrices in Row Major */

void print_mat(const Matrix *mat) {

        int i,j;
        float *matrix_ptr = mat->data_ptr;

        for( i = 0; i < mat->rows; i++ ) {
                for( j = 0; j < mat->cols; j++ ) {
                        printf("%.40f ", *(matrix_ptr + j * mat->rows + i));
                }
                printf("\n");
        }

}


/*

Calculates the degree of every node based on the adjacency matrix. One thread is
executed per column (node)

*/
```

```
__global__ void degreevec_gpu(int size, int *vector, float *matrix) {

        int idx = blockIdx.x * blockDim.x + threadIdx.x;
        int i;

        if( idx < size ) {
                vector[idx] = 0;
                for( i = idx * size; i < (idx + 1) * size; i++ ) {
                        vector[idx] = vector[idx] + *(matrix + i);
                }
        }
}


/*

Performs weight transformation on edge weight based on the degree of both
nodes in the edge. Influence of connections between high degree nodes is
reduced. One thread per column (node)

*/

__global__ void weighttransform_gpu(int size, int *degreevec, float *matrix) {

        int idx = blockIdx.x * blockDim.x + threadIdx.x;
        int i;

        if( idx < size ) {
                for( i = idx * size; i < (idx + 1) * size; i++ ) {
  *(matrix + i) = ( *(matrix + i) / *(degreevec + idx) )
  + ( *(matrix + i) / *(degreevec + i - idx * size) );
                }
        }
}


/* Adds self loops to the network. One thread per column (node) */

__global__ void selfloops_gpu (int size, float *matrix) {

        int idx = blockIdx.x * blockDim.x + threadIdx.x;

        if( idx < size ) {
                *(matrix + idx * size + idx) = *(matrix + idx * size + idx) + 1;
        }
}


/*

Converts colum entries to transition probabilities by making all the entries
sum up to one

*/

__global__ void col2prob_gpu (int size, float *matrix) {

        int idx = blockIdx.x * blockDim.x + threadIdx.x;
        int i;
        float denom = 0.0f;
        if( idx < size ) {

                for( i = idx * size; i < (idx + 1) * size; i++ ) {
                 denom = denom + *(matrix + i);;
                }

                for( i = idx * size; i < (idx + 1) * size; i++ ) {

                        *(matrix + i) = *(matrix + i) / denom;
                }
        }
}

/*

Matrix multiplication on the gpu. Function calls a function from the CUBLAS
linear algebra library

*/

void mult_gpu( float *mat1Dev_ptr, float *mat2Dev_ptr, float *mat3Dev_ptr, int dim ) {

cublasSgemm('n', 'n', dim, dim, dim, 1.0f, mat1Dev_ptr, dim,
mat2Dev_ptr, dim, 0.0f, mat3Dev_ptr, dim);

}

/* Raises each element of a given matrix to a power. One thread per column */

__global__ void power_gpu( int size, float *matrix, float power) {

        int idx = blockIdx.x * blockDim.x + threadIdx.x;
        int i;

        if( idx < size ) {
                for( i = idx * size; i < (idx + 1) * size; i++ ) {
                        *(matrix + i) = pow(*(matrix + i), power);
                }
        }
}

/* The following two functions work together */

/*

Function checks whether there are more than on non zero element in a column
(convergence criterion) and returns the information as a binary vector

*/

__global__ void checkconverge_gpu(int size, int *vector, float *matrix) {

        int idx = blockIdx.x * blockDim.x + threadIdx.x;
        int i;
        int counter = 0;

        if( idx < size ) {
                vector[idx] = 0;
                for( i = idx * size; i < (idx + 1) * size; i++ ) {
                        if( *(matrix + i)  > 0 ) {
                                counter++;
                        }
                }
                if( counter == 1 ) {
                        vector[idx] = 1;
                }
        }
}

/*

Host function calling checkconverge_gpu(). Verifies if all columns have converged based
on the sum of the vector elements, if the sum is equal to the dimension all columns
have converged

*/

int checkconverge( int dim, int *dev_ptr, int *host_ptr, float *matrix) {
        dim3 dimBlock(1);
        dim3 dimGrid(dim);
        int i;
        cublasStatus status;
        int sum = 0;

        checkconverge_gpu<<<dimGrid, dimBlock>>>(dim, dev_ptr, matrix);

        status = cublasGetVector(dim, sizeof(int), dev_ptr, 1, host_ptr, 1);
        if (status != CUBLAS_STATUS_SUCCESS) {
                fprintf (stderr, "!!!! device access error (read C)\n");
                exit(1);
        }

        for( i=0; i < dim; i++ ) {
                sum = sum + *(host_ptr + i);
        }

        if( sum == dim ) {
                return(1);
        }
        else {
                return(0);
        }
}



/*

Function identifies the row index of the largest value in a column (greedy search).
Greedy search is not a problem here since there should only be one non zero.
This is the center of the subnetwork and the attractor node of the column

*/

__global__ void colmax_gpu (int *vector, int size, float *matrix) {

        int idx = blockIdx.x * blockDim.x + threadIdx.x;

        int max_index = 0;
        float last_max = 0.0f;
        int i;
        if( idx < size ) {

                for( i = idx * size; i < (idx + 1) * size; i++ ) {
                         if( *(matrix + i) > last_max ) {
                                last_max = *(matrix + i);
                                max_index = i;
                        }
                }
                vector[idx] = max_index - idx * size;
        }
}


/* Main loop */

int main(int argc, char** argv) {

        Matrix *adjacency_ptr;
        Matrix *intermed_ptr;
        Matrix *result_ptr;

        float *devA_ptr = 0;
        float *devB_ptr = 0;
        float *devC_ptr = 0;


        int dim = atoi(argv[2]);
        float exp = atof(argv[3]);
int max_runs = atoi(argv[4]);
        int i;
/*

dimBlock and dimGrid are used to set the thread configuration of CUDA. The grid
can be subdivided in blocks, all threads in a block can access a certain amount of
shared memory. In the case of this algorithm, communication between threads
is not needed and the block size is set to contain only one thread. The grid holds
all the blocks, an hence all the threads. The maximum grid size is 256 * 256
which is also the maximum amount of available threads.

*/
        dim3 dimBlock(1);
        dim3 dimGrid(dim);

        cublasStatus status;

        if( argc < 3 ) { fprintf(stderr, "Not enough parameters\n");  return(1);}



/* Allocation for the matrices on host memory */

adjacency_ptr = read_matrix(argv[1], dim, dim);

intermed_ptr = (Matrix *)malloc(sizeof(intermed_ptr));
        intermed_ptr->length = dim * dim;
        intermed_ptr->rows = dim;
        intermed_ptr->cols = dim;
        intermed_ptr->data_ptr = (float *)malloc(intermed_ptr->length * sizeof(float));
        result_ptr = (Matrix *)malloc(sizeof(result_ptr));
        result_ptr->length = dim * dim;
        result_ptr->rows = dim;
        result_ptr->cols = dim;
        result_ptr->data_ptr = (float *)malloc(result_ptr->length * sizeof(float));


        status = cublasInit();

        if (status != CUBLAS_STATUS_SUCCESS) {
                fprintf (stderr, "Couldn't start CUBLAS\n");
                exit(1);
        }

/* Allocation memory for the matrices in the video RAM */

        status = cublasAlloc(dim * dim, sizeof(float), (void**)&devA_ptr);
        if (status != CUBLAS_STATUS_SUCCESS) {
                fprintf (stderr, "Couldn't allocate memory for matrix A in video RAM\n");
                exit(1);
        }


        status = cublasAlloc(dim * dim, sizeof(float), (void**)&devB_ptr);
                if (status != CUBLAS_STATUS_SUCCESS) {
                fprintf (stderr, "Couldn't allocate memory for matrix B in video RAM\n");
                exit(1);
        }


        status = cublasAlloc(dim * dim , sizeof(float), (void**)&devC_ptr);
        if (status != CUBLAS_STATUS_SUCCESS) {
                fprintf (stderr, "Couldn't allocate memory for result matrix in video RAM\n");
                exit(1);
        }

/* Copy the matrices from the main RAM to the allocated space in the video RAM*/

        status = cublasSetVector(dim * dim, sizeof(float), adjacency_ptr->data_ptr, 1, devA_ptr, 1);
        if (status != CUBLAS_STATUS_SUCCESS) {
                fprintf (stderr, "Couldn't write matrix A to the card");
                exit(1);
        }

        status = cublasSetVector(dim * dim, sizeof(float), adjacency_ptr->data_ptr, 1, devB_ptr, 1);
        if (status != CUBLAS_STATUS_SUCCESS) {
                fprintf (stderr, "Couldn't write matrix B to the card\n");
                exit(1);
        }

        status = cublasSetVector(dim * dim, sizeof(float), result_ptr->data_ptr, 1, devC_ptr, 1);
        if (status != CUBLAS_STATUS_SUCCESS) {
                fprintf (stderr, "Couldn't initialize the result matrix on the card\n");
                exit(1);
        }

int *hostvector_ptr;
        int *devicevector_ptr;

        hostvector_ptr = (int *)malloc(adjacency_ptr->cols * sizeof(*hostvector_ptr));
        status = cublasAlloc(adjacency_ptr->cols , sizeof(int), (void**)&devicevector_ptr);
        if (status != CUBLAS_STATUS_SUCCESS) {
                fprintf (stderr, "Couldn’t initialize device vector\n");
                exit(1);
        }


/* Calculation part starts here */

/*

In the following steps the adjacency matrix is converted to the cannonical flow matrix on
which the calculations will be performed

*/

// Calculate the degree for each node

degreevec_gpu<<<dimGrid, dimBlock>>>(dim, devicevector_ptr, devA_ptr);

// Calculate the degree for each node

weighttransform_gpu<<<dimGrid, dimBlock>>>(dim, devicevector_ptr, devA_ptr);

// Add self loops

selfloops_gpu<<<dimGrid, dimBlock>>>(dim, devA_ptr);

// Normalise the columns converting them to probabilities

col2prob_gpu<<<dimGrid, dimBlock>>>(dim, devA_ptr);


int run = 1;
int runs = 0;
float *buffer_ptr;

/*

The expansion step (mult_gpu) and the inflation step (power_gpu) are performed in succession
until either the convergence criterion is reached, or a maximum amount of runs is exceeded.

*/
while( run ) {
runs++;
                mult_gpu(devA_ptr, devB_ptr, devC_ptr, dim);
                power_gpu<<<dimGrid, dimBlock>>>(dim, devC_ptr, exp);

// Columns are renormalised after the expansion and inflation steps

                col2prob_gpu<<<dimGrid, dimBlock>>>(dim, devC_ptr);

                buffer_ptr = devA_ptr;
                devA_ptr = devC_ptr;
                devC_ptr = buffer_ptr;


if( checkconverge( dim, devicevector_ptr, hostvector_ptr, devA_ptr) ) {
                        run = 0;
                }
if (runs >= max_runs ) { run = 0; }

}


/*

After either the convergence criterion or a maximum amount of runs is reached, for each
column the row index of the largest element is calculated and outputted to console

*/

colmax_gpu<<<dimGrid, dimBlock>>>(devicevector_ptr, dim, devA_ptr);


        status = cublasGetVector(dim, sizeof(int), devicevector_ptr, 1, hostvector_ptr, 1);
        if (status != CUBLAS_STATUS_SUCCESS) {
                fprintf (stderr, "Could not retrieve resulst from device\n");
                exit(1);
        }

        for( i = 0; i < dim; i++ ) {
                printf("%d\n", *(hostvector_ptr + i));
        }
        printf("\n");


}
```

2 SpearmanPreranked source code

```
/*

SpearmanPreranked by Ir Frederik Roels.

Description:

The program calculates the Spearman rank correlation coefficient for large datasets.
The issue with calculating correlation on matrices with  a high amount of rows is memory.
The program solves this by keeping the usage of memory low and instead calls on disk
space to progressively store the result. It is not advised to continuously output to disk
since this would mean constant disk access which may slow the program, and the operating
system in general, down considerably. In this case, each time a row of the correlation matrix is
calculated it is appended to an output file on disk. The program outputs the results as strings
instead of binary so as to ease integration with other programs.

The program starts from a matrix in which the rows have already been converted to ranks.
The rows are calculated at once by exploiting the fact that most of the calculation of Spearman
correlation can be done by means of vector operations. Note that this also makes the described
program easy to parallelise or to distribute if one would be so inclined.

The program keeps one copy of the original ranked matrix in memory which is used as a one
dimensional vector, this is the master.

For the calculation of each row, a slave is loaded, the slave consists of N repeats of a given row,
where N is the total amount of rows in the matrix. The resulting vector is hence the same length
as the master. By using vector operations combining both the slave and the master a row of
the correlation matrix is calculated.
```

```
Usage:  SpearmanPreranked <path to preranked matrix> <rows> <columns> <path to output file>

*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>


#define MAX_MAT 6000 * 6000  // Hard coded upper limit to matrix size

/*

General purpose data structure to hold matrices. Even if the matrices are considered
as vectors, the same struct will be used

*/

typedef struct Matrix {
        int length;
        int rows;
        int cols;
        float *data_ptr;
} Matrix;


/*

Utility function that reads in a matrix from file and returns it as a Matrix data structure.
The function first allocates MAX_MAT memory and later resizes the memory when to
fit the size of the matrix

*/

Matrix *read_matrix( char *filename, int rows, int cols ) {

        int counter;
        float *data_ptr;
        float *data_ptr_resize;
        float *seek_ptr;
        int total_size = rows * cols;
        Matrix *return_struct;

        FILE *f_ptr = fopen(filename, "r");
        if( !f_ptr ) {
                fprintf(stderr, "error opening input file");
        }

        data_ptr = (float *)malloc(MAX_MAT * sizeof(*data_ptr));

        if( !data_ptr ) {
                fprintf(stderr, "error allocating space for input matrix\n");
                exit(1);
        }

        seek_ptr = data_ptr;
        counter = 0;
        while( fscanf(f_ptr, "%f", seek_ptr) != EOF) {
                seek_ptr++;
                counter++;
                if( counter > MAX_MAT ) { fprintf(stderr, "Data too large\n"); exit(1); }
        }

        if( counter != (rows * cols ) ) {
         fprintf(stderr, "Dimension does not match array size, rows:
          %d cols: %d array size: %d (for %s)\n", rows, cols, counter, filename);
          exit(1);
        }

        data_ptr_resize = (float *)realloc(data_ptr, counter * sizeof(*data_ptr));

        if( data_ptr_resize ) {
                data_ptr = data_ptr_resize;
        }
        else{ fprintf(stderr, "Realloc failed\n"); exit(1); }


        return_struct = (Matrix *)malloc(sizeof(*return_struct));

        return_struct->length = counter;
        return_struct->rows = rows;
        return_struct->cols = cols;
        return_struct->data_ptr = data_ptr;

return(return_struct);

}


// Utility function that prints out the matrix

void print_mat(const Matrix *mat) {

        int i,j;
        float *matrix_ptr = mat->data_ptr;

        for( i = 0; i < mat->rows; i++ ) {
                for( j = 0; j < mat->cols; j++ ) {
printf("%f ", *matrix_ptr);
                        matrix_ptr++;
                }
printf("\n");
        }

}


/*
Function that loads the slave vector from the master vector based on a
    row index (the offset in the data vector)

*/

void load_slave( Matrix *mat, Matrix *slave_mat, int offset ) {

int i, j;
float *master_ptr = mat->data_ptr;
float *slave_ptr = slave_mat->data_ptr;
int cols = mat->cols;
int rows = mat->rows;
```

```
for( i = 0; i < rows; i++ ) {
for( j = 0; j < cols; j++ ) {
*(slave_ptr + ( (i * cols) + j) ) = *(master_ptr + ( (offset * cols) + j) );
}
}
}


/* Function that calculates the difference between two vectors */

void vector_diff( Matrix *matA, Matrix *matB, Matrix *ret_mat) {

int i,j;
int length = matA->length;
float *matA_ptr = matA->data_ptr;
float *matB_ptr = matB->data_ptr;
float *ret_ptr = ret_mat->data_ptr;

for( i = 0; i < length; i++ ) {
*(ret_ptr + i) = *(matA_ptr + i) - *(matB_ptr + i);
}
}

/* Function that squares every element of a given vector */

void vector_square( Matrix *mat ) {

int i,j;
int length = mat->length;
float *mat_ptr = mat->data_ptr;

for( i = 0; i < length; i++ ) {
*(mat_ptr + i) = *(mat_ptr + i) * *(mat_ptr + i);
}
}

/*

Function that takes the sum of parts of a given vector and returns the results of
these partial sums as a vector of reduced length This has the effect of converting
the master vector and slave vector operations to the value used in the correlation
formula. Generating one value to be used in calculating the correlation value for
one column in the row that is being calculated of the correlation matrix.

*/

void sum( Matrix *mat, float *row_vec, int offset ) {

int i;
int cols = mat->cols;
float *mat_ptr = mat->data_ptr;
float sum;
sum = 0.0;

for( i = offset * cols; i < (offset + 1) * cols; i++ ) {
sum = sum + *(mat_ptr + i);
}
*(row_vec + offset) = sum;
}

/*
```

```
Final step in calculating the correlation for all columns in the correlation
matrix row that is being calculated

*/

void calc_cor( float *row_vec, float *ret_vec, float factor, int length ) {

int i;

for( i = 0; i < length; i++ ) {
*(ret_vec + i) = 1.0 - ( *(row_vec + i) * factor );
}
}

/*

Wrapper function which computes the steps required to calculate one
row of the correlation matrix

*/

void calc_row( Matrix *rank_mat, Matrix *inter1_ptr, Matrix *inter2_ptr,
float *row_ptr, float *ret_ptr, float factor, int row ) {

int i;

load_slave(rank_mat, inter1_ptr, row);
vector_diff(inter1_ptr, rank_mat, inter2_ptr);
vector_square(inter2_ptr);

for( i = 0; i < rank_mat->rows; i++ ) {
                sum(inter2_ptr, row_ptr, i);
        }

calc_cor(row_ptr, ret_ptr, factor, rank_mat->rows);
}


/* Utility function that writes a row to a specified file */

void write_row( float *row_ptr, int size, FILE *file_ptr ) {

int i;

for( i = 0; i < size; i++ ) {
                fprintf(file_ptr, "%f ", *(row_ptr + i));
        }
fprintf(file_ptr, "\n");
}


/* Main loop */

int main(int argc, char** argv) {


int i, j;
Matrix *matrix_ptr;
Matrix *intermediate_ptr;
Matrix *intermediate2_ptr;
Matrix *intermediate3_ptr;
float *row_vec1;
float *row_vec2;
float factor;
int n;

matrix_ptr = read_matrix(argv[1], atoi(argv[2]), atoi(argv[3]));

FILE *output_ptr = fopen(argv[4], "w");

        if( !output_ptr ) {
                fprintf(stderr, "error creating output file");
        }


/* Some space is allocated to hold intermediate resulst of the computations */

intermediate_ptr = (Matrix *)malloc(sizeof(*intermediate_ptr));
intermediate_ptr->length = matrix_ptr->length;
intermediate_ptr->rows = matrix_ptr->rows;
intermediate_ptr->cols = matrix_ptr->cols;
intermediate_ptr->data_ptr = (float *)malloc(intermediate_ptr->length * sizeof(float));

if( !intermediate_ptr->data_ptr ) {
         fprintf(stderr, "error allocating space for intermediate storage\n");
         exit(1);
        }


intermediate2_ptr = (Matrix *)malloc(sizeof(*intermediate2_ptr));
        intermediate2_ptr->length = matrix_ptr->length;
        intermediate2_ptr->rows = matrix_ptr->rows;
        intermediate2_ptr->cols = matrix_ptr->cols;
        intermediate2_ptr->data_ptr = (float *)malloc(intermediate2_ptr->length * sizeof(float));

if( !intermediate2_ptr->data_ptr ) {
         fprintf(stderr, "error allocating space for intermediate storage\n");
                exit(1);
        }

row_vec1 = (float *)malloc(matrix_ptr->rows * sizeof(float));
row_vec2 = (float *)malloc(matrix_ptr->rows * sizeof(float));


/* Calculation of the coeficient used in the calculation of the correlation */

n  = matrix_ptr->cols;

int denom = ((n * n) - 1) * n;

factor = 6.0 / denom;
float test = 6.0 / 2;

/* Loop that calculates all the rows of the correlation matrix and outputs them to file */

        for( i = 0; i < matrix_ptr->rows; i++ ) {
calc_row( matrix_ptr, intermediate_ptr, intermediate2_ptr, row_vec1, row_vec2, factor, i );
write_row(row_vec2, matrix_ptr->rows, output_ptr);
}
}
```

Appendix C

List of figures

List of Figures

Appendix D

List of tables

List of Tables