UNIVERSITY OF KWAZULU-NATAL (HOWARD COLLEGE)

MASTERS THESIS

Discrete Energy Minimisation Optimisation using Graph Cuts for Fluorescence Microscopy

Author:
Ryan NAIDOO

Supervisor:
Dr. Jules-Raymond TAPAMO

A thesis submitted in fulfillment of the requirements for the degree of Master of Science in Engineering

in the

Department of Electrical, Electronic and Computer Engineering
School of Engineering

September 6, 2016

Declaration of Authorship

I, Ryan NAIDOO, declare that this thesis titled, "Discrete Energy Minimisation Optimisation using Graph Cuts for Fluorescence Microscopy" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

"Thanks to my solid academic training, today I can write hundreds of words on virtually any topic without possessing a shred of information, which is how I got a good job in journalism."

Dave Barry

UNIVERSITY OF KWAZULU-NATAL (HOWARD COLLEGE)

Abstract

Faculty of Engineering
School of Engineering

Master of Science in Engineering

Discrete Energy Minimisation Optimisation using Graph Cuts for Fluorescence Microscopy

by Ryan NAIDOO

The Thesis Abstract is written here (and usually kept to just this page). The page is kept centered vertically so can expand into the blank space above the title too…

Acknowledgements

The acknowledgments and the people to thank go here, don't forget to include your project advisor...

Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviations

ACWE	**A**ctive **C**ontours **W**ithout **E**dges
CED	**C**oherence **E**nhancing **D**iffusion
CCD	**C**harge-**C**oupled **D**evice
CLSM	**C**onfocal **L**aser **S**canning **M**icroscopy
CRF	**C**onditional **R**andom **F**ield
DNA	**D**eoxyribo**n**ucleic **A**cid
EGFP	**E**nhanced **G**reen **F**luorescent **P**rotein
EM	**E**xpectation **M**aximisation
FCS	**F**luorenscence **C**orrelation **S**pectroscopy
FIFO	**F**irst-**I**n **F**irst-**O**ut
FISH	**F**luorenscence **i**n-**s**itu **H**ybridisation
FLIM	**F**luorenscence **L**ifetime **I**maging **M**icroscopy
FRAP	**F**luorenscence **R**ecovery **A**fter **P**hotobleaching
FRET	**F**luorenscence **R**esonance **E**nergy **T**ransfer
GFP	**G**reen **F**luorescent **P**rotein
GMM	**G**uassian **M**ixture **M**odelling
IHC	**I**mmuno**h**isto**c**hemistry
LED	**L**ight **E**mitting **D**iode
MAP	**M**aximum **A** **P**osteriori
MIS	**M**edical **I**mage **S**egmentation
MLP	**M**ulti-**L**ayered **P**erceptron
MRF	**M**arkov **R**andom **F**ield
NA	**N**umerical **A**perture
ORI	**O**ptimised **R**otational **I**nvariance
PSF	**P**oint **S**pread **F**unction
RNA	**R**ibo**n**ucleic **A**cid
TV	**T**otal **V**ariation
UV	**U**ltra **V**iolet

Physical Constants

Speed of Light $c_0 = 2.997\,924\,58 \times 10^{8}\,\mathrm{m\,s^{-1}}$ (exact)

List of Symbols

a	distance	m
P	power	W ($\mathrm{J\,s^{-1}}$)
ω	angular frequency	rad

For/Dedicated to/To my…

Chapter 1

Introduction

1.1 Outline and Contributions

The introduction is here.

1.2 Thesis Overview

The remainder of the thesis outline.

Chapter 3 is where we cover the mathematical foundation to Graph Cut image segmenation.

Chapter ?? is where we cover the mathematical foundation to Graph Cut image segmenation.

Chapter 2 is where we cover the mathematical foundation to Graph Cut image segmenation.

Chapter 4 is where we cover the mathematical foundation to Graph Cut image segmenation.

Chapter 5 is where we cover the mathematical foundation to Graph Cut image segmenation.

Chapter 6 is where we cover the mathematical foundation to Graph Cut image segmenation.

Chapter ?? concludes the thesis with suggestions for further work.

Chapter 2

Introduction to Fluorescence Microscopy

Chapter 3

Mathematical Background

3.1 Graph Theory and Flow Networks

In this section we cover Graph Theory and specifically Flow Networks, which is a branch of Graph Theory, which is fundamental to the understanding of image segmentation via graph cuts. With it roots in Germany where Euler tried to find the solution to the Konigsberg bridge problem, graph theory has since blossomed into a rich field of Mathematics with seemingly endless amounts of application. Graph Theory is a huge topic in mathematics and can be applied to many other sciences. Graph theory is part of another more encompassing field of Mathematics known as Combinatronics. Graph theory and applications are more useful than the average person would recognise. They're used in Google Maps to find shortest routes to destinations, in Molecular Chemistry to model the structure of atoms, and the list goes on for quite a while. It is no surprise that it is also found to be useful in image segmentation.

Graph A graph G is a pair (V, E), where V is the set of nodes/vertices and E is the set of edges consisting of pairs (u, v) where $u, v \in V$. The graph is assumed to be finite i.e. $|V| = n$ and $|E| = m$.

In an **undirected graph**, the edge (u, v) and (v, u) are not distinct. That is, they refer to the same edge. However, in a **directed graph**, the two edge are now distinct. In a directed graph with edge (u, v), u is known as the **tail** and v is know as the **head**. In directed graphs, edges, also known as arcs, are depicted by placing arrowheads at the head of the edge. Given an edge $e = (u, v)$, u and v are said to be **incident** on e. A graph is said to be **simple** if it does not contain any self-loops. A **self-loop** is an edge with of its end points being the same vertex.

Degree The degree of a vertex v is the number of edges incident on it. $deg(v) = |\{(u, v), (v, u) \in E\}|$. A self-loop counts for 2.

If a graph is directed, also known as a **digraph**, then a node v has an **in-degree** $d_{in}(v)$ and an **out-degree** $d_{out}(v)$. A digraph is said to be **balanced** if $d_{in}(v) = d_{out}(v), \forall v \in V$.

Subgraph A graph $G' = (V', E')$ is said to be a sub-graph of $G = (V, E)$, denoted as $G' \subseteq G$, if $V' \subseteq V$ and $E' \subseteq E$.

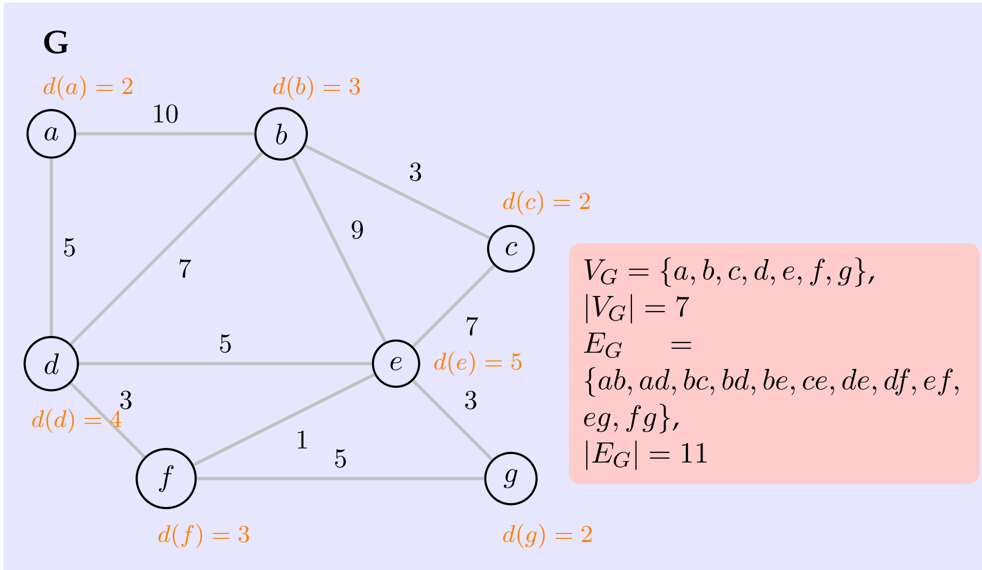

FIGURE 3.1: Undirected weighted graph **G**. The degree of each node is shown next to the corresponding node. The graph is simple. The red box shows the vertex set, V_G, and edge set, E_G, and their corresponding norm.

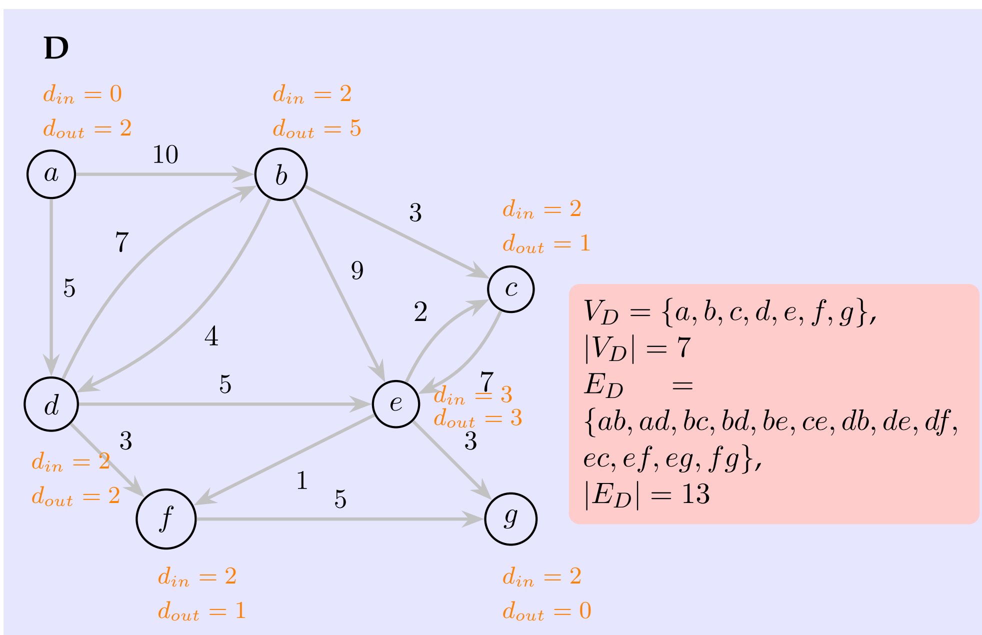

FIGURE 3.2: Directed weighted graph (Digraph) **D**. The in-degree and out-degree is shown next to each node. The graph is simple and not balanced. The red box shows the vertex set, V_D, and edge set, E_D, and their corresponding norm.

Clique A clique is a maximal subgraph.

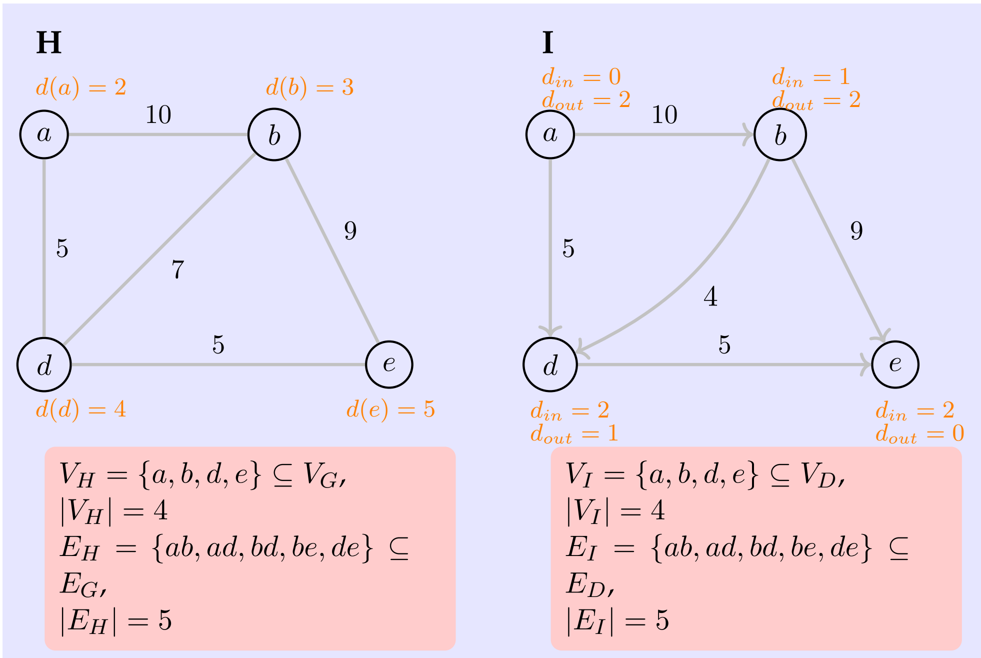

FIGURE 3.3: Undirected weighted graph **H** is a subgraph of **G** in Figure XX, $\mathbf{H} \subseteq \mathbf{G}$. Directed weighted graph **I** is a subgraph of **D** in Figure XX, $\mathbf{I} \subseteq \mathbf{D}$. The degree of each node is shown next to the corresponding node. The red box shows the vertex set, the edge set and their corresponding norms.

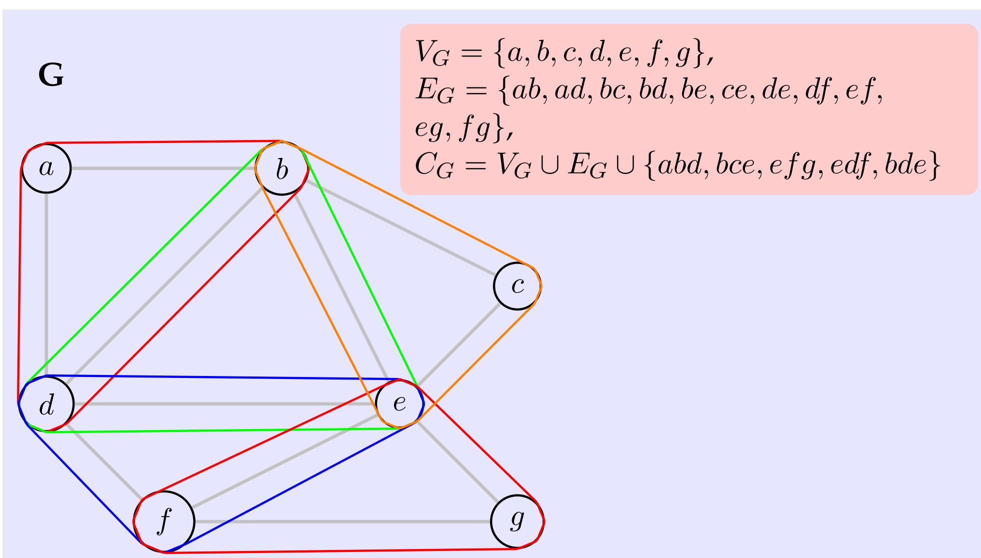

FIGURE 3.4: Cliques of the undirected weighted graph **G**. The maximal cliques are shown by the hyperedges that encompass the nodes of that clique.

Network A network $N = (V, E)$ is a directed graph with a source node s, a sink node t and a strictly positive capacity on every edge. That is, for each edge $e \in E$, the capacity, $c(.)$, obeys $c(e) \in \Re^+$.

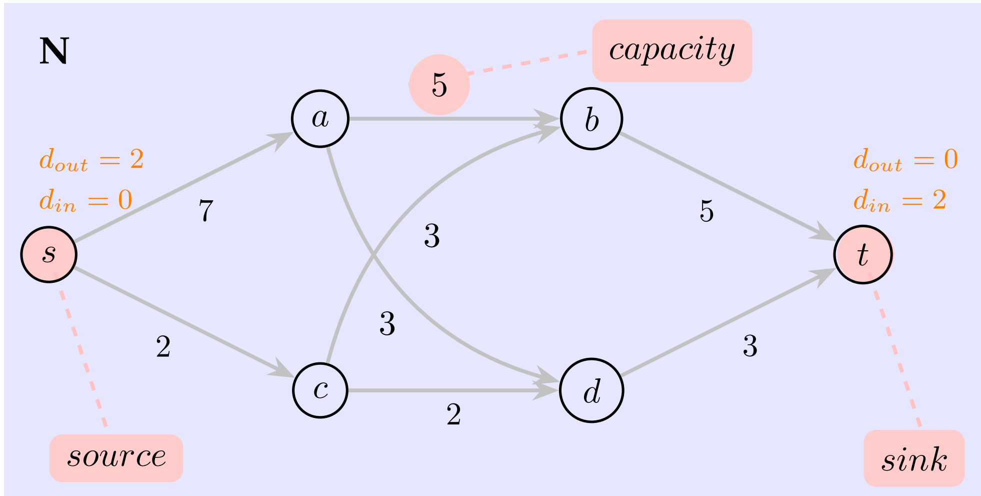

FIGURE 3.5: Network **N** with no flow. The in-degree and out-degree for the source, **s**, and the sink, **t**, are shown next to the corresponding node.

The **source node** only has out-going edges, $d_{in}(s) = 0$ and $d_{out}(s) \geq 0$. The **sink node** only has incoming edges, $d_{in} \geq 0$ and $d_out = 0$.

Flow A flow $f : V^2 \longrightarrow \Re^+$ is associated with each edge $e = (u, v)$ such that:

1. for each edge $e \in E$ we have $0 \leq f(e) \leq c(e)$. That is, the flow is positive and cannot excees the capacity of the edge.

2. for each intermediate node $v \in V \setminus \{s, t\}$ the in- and out-flow of that node $\sum_{u \in V^-(v)} f(u, v) = \sum_{u \in V^+(v)} f(v, u)$.

The **total flow** F of a network is then what leave the source s or reaches the sink t:

$$F(N) := \sum_{u \in V} f(s, u) - \sum_{u \in V} f(u, s) = \sum_{u \in V} f(u, t) - \sum_{u \in V} f(t, u) \qquad (3.1)$$

Cut A cut of a network $N = (V, E)$ is a partitioning of the vertex set $V = P \bigcup \bar{P}$ into two disjoint sets P containining the source node s and $\bar{P}$ containing the sink node t. $P \bigcap \bar{P} = \emptyset$.

The **capacity** of a cut is the sum of the edges $(u, v) \in V$ where $u \in P$ and $v \in \bar{P}$:

$$\kappa(P, \bar{P}) = \sum_{u \in P; v \in \bar{P}} c(u, v) \qquad (3.2)$$

Maximal Flow The largest amount of flow that can be sent through the source that is able to reach the sink is known as the maximal flow.

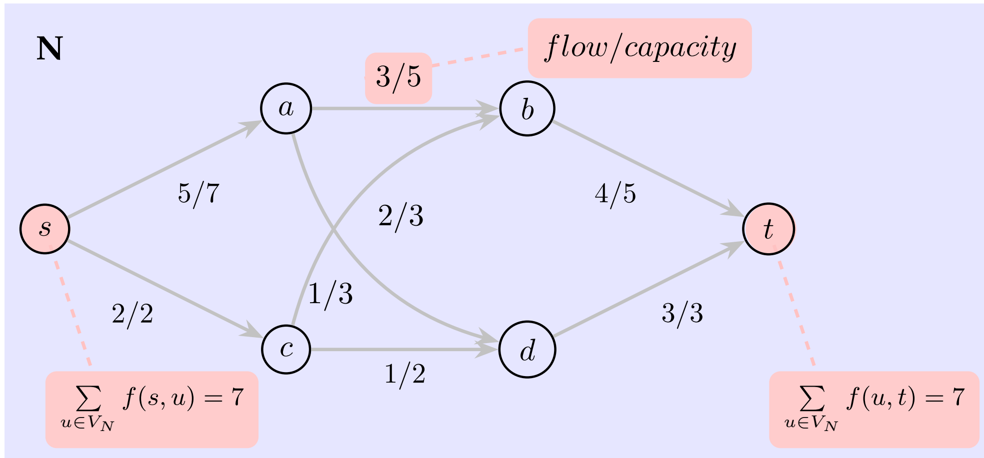

FIGURE 3.6: Network **N** with flow. The flow out of the source node, **s**, is equal to the flow into the sink node, **t**. For all other nodes, the flow-in is equal to the flow-out. This is the conservation of flow principle. This is only part of the network. The remaining part is the residual graph which shows the amount of reverse flow is available on an edge.

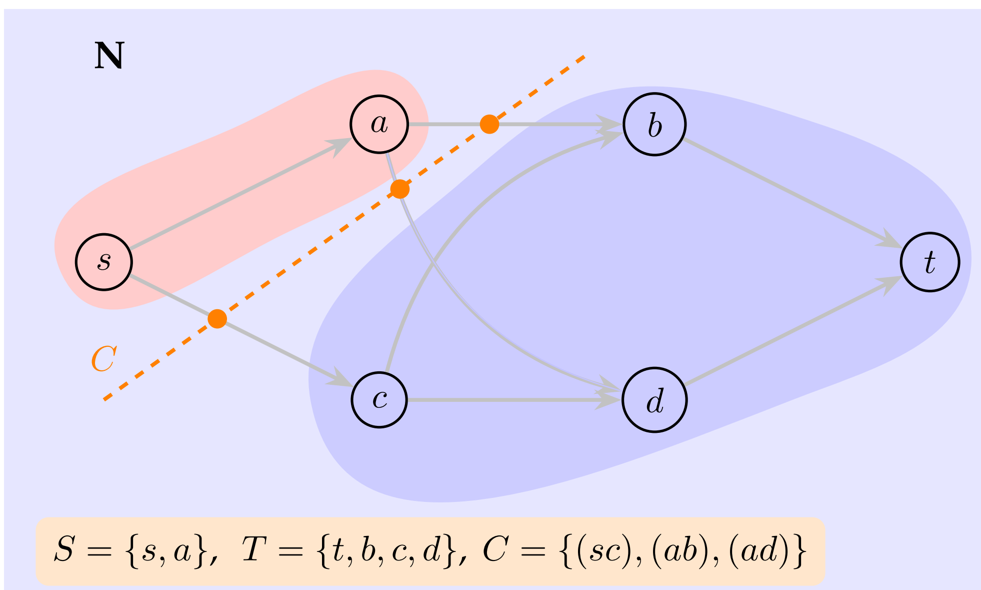

FIGURE 3.7: Network **N** with with a valid cut **C**. The nodes within the red region are reachable from the source and the nodes within the blue region are able to reach the sink. The cut set, **C**, is show in the orange filled block.

Minimal Cut A cut C on a network $N = (V, E)$ is a minimal cut if there exists no other cut C' where $\kappa(C') < \kappa(C)$.

In the next section we show that the Maximal Flow problem and the Minimal Cut problem are duals of each other, commonly known as the Max-Flow/Min-Cut problem.

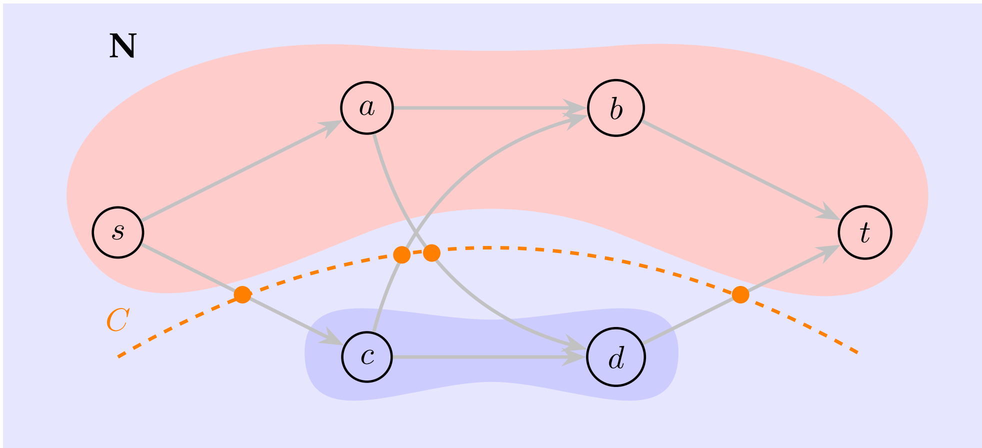

FIGURE 3.8: Network **N** with with a invalid cut **C**. The cut does not partition source node **s** and sink node **t** into distinct sets.

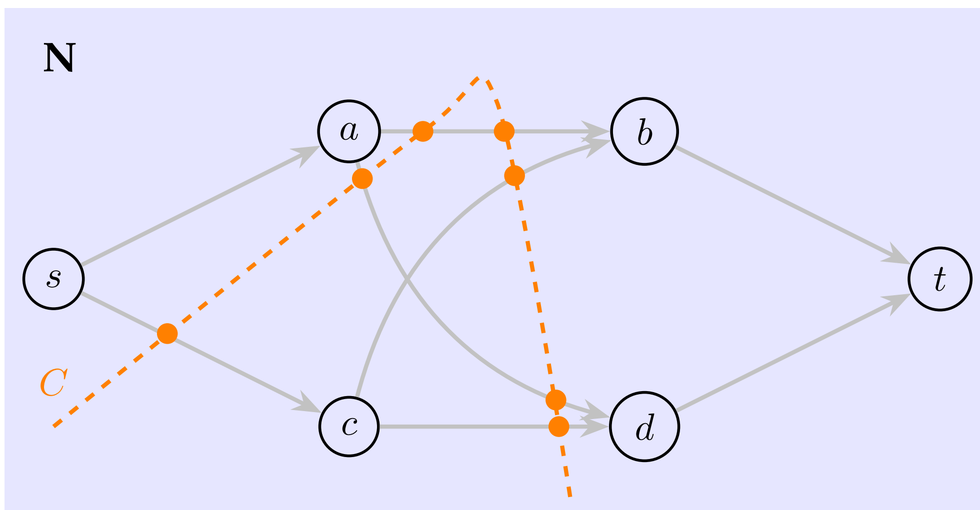

FIGURE 3.9: Network **N** with with a invalid cut **C**. The cut partition partitions the graph into more than two sets and the cut intersects the edges ab and ad twice.

3.2 Markov Random Fields

In this section we review MRF's as a pure mathematical/statistical tool used used specifically for vision. That is, we only cover the necessary concepts related to understanding the problem of modelling images for analysis and inference purposes. In sub-section 1 we cover the basics, in sub-section 2 we cover how to model an image using MRF, and in sub-section 3 we cover how to find the MAP, or a close enough approximation, to the MRF.

3.2.1 Markov Random Fields Theory and Concepts

MRF theory and concepts. Purely mathematical/statistical. Markov Properties Markov Blankets

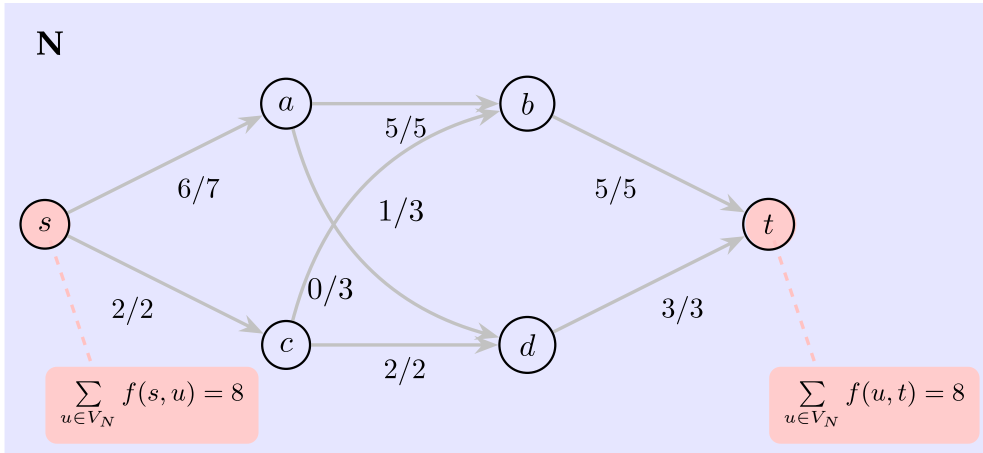

FIGURE 3.10: Network **N** with maximum flow. There is no way to push more flow out of the source into the sink without breaking the rules for conservation of flow.

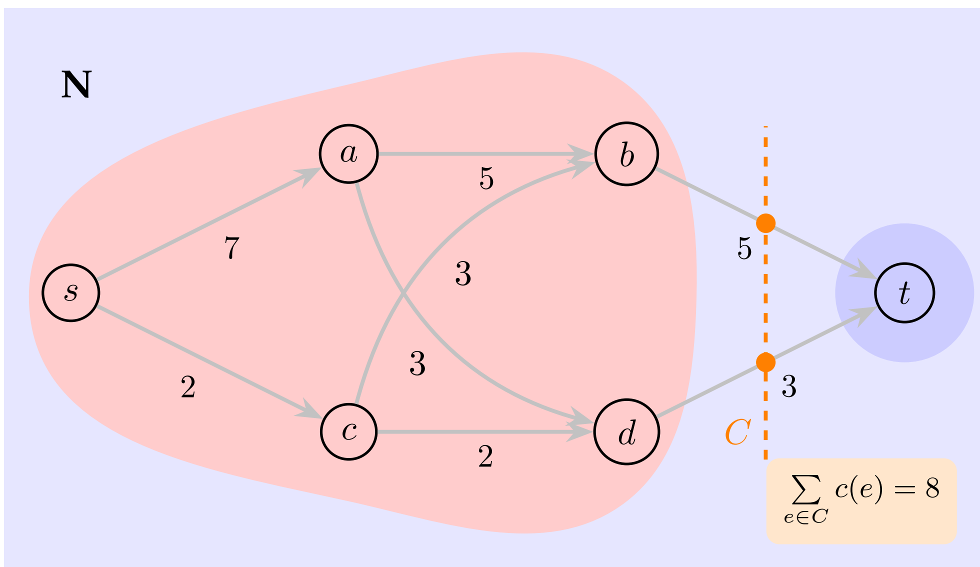

FIGURE 3.11: Network **N** with minimal cut **C**. The sum of the capacity of all the edges in the cut set is the minimum of all possible valid cuts on the network **N**.

3.2.2 Markov Random Fields in Image Modelling for Segmentation

Modelling the joint probability of of image using MRFs. Nearby pixels exhibit high correlation in natural images. This is where we can take advantage of Markov Modelling.

3.2.3 MAP-MRF Approxiamtion via Graph Cuts

How to make MAP estimates on the MRF. How does the graph-cut approach ensure an MAP solution.

3.3 Max-Flow/Min-Cut Problem

Sed ullamcorper quam eu nisl interdum at interdum enim egestas. Aliquam placerat justo sed lectus lobortis ut porta nisl porttitor. Vestibulum mi dolor, lacinia molestie gravida at, tempus vitae ligula. Donec eget quam sapien, in viverra eros. Donec pellentesque justo a massa fringilla non vestibulum metus vestibulum. Vestibulum in orci quis felis tempor lacinia. Vivamus ornare ultrices facilisis. Ut hendrerit volutpat vulputate. Morbi condimentum venenatis augue, id porta ipsum vulputate in. Curabitur luctus tempus justo. Vestibulum risus lectus, adipiscing nec condimentum quis, condimentum nec nisl. Aliquam dictum sagittis velit sed iaculis. Morbi tristique augue sit amet nulla pulvinar id facilisis ligula mollis. Nam elit libero, tincidunt ut aliquam at, molestie in quam. Aenean rhoncus vehicula hendrerit.

3.4 Modelling Images as MRFs

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam ultricies lacinia euismod. Nam tempus risus in dolor rhoncus in interdum enim tincidunt. Donec vel nunc neque. In condimentum ullamcorper quam non consequat. Fusce sagittis tempor feugiat. Fusce magna erat, molestie eu convallis ut, tempus sed arcu. Quisque molestie, ante a tincidunt ullamcorper, sapien enim dignissim lacus, in semper nibh erat lobortis purus. Integer dapibus ligula ac risus convallis pellentesque.

3.4.1 Sub-modularity Conditions for Discrete Systems

Nunc posuere quam at lectus tristique eu ultrices augue venenatis. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Aliquam erat volutpat. Vivamus sodales tortor eget quam adipiscing in vulputate ante ullamcorper. Sed eros ante, lacinia et sollicitudin et, aliquam sit amet augue. In hac habitasse platea dictumst.

3.4.2 Connectivity

Morbi rutrum odio eget arcu adipiscing sodales. Aenean et purus a est pulvinar pellentesque. Cras in elit neque, quis varius elit. Phasellus fringilla, nibh eu tempus venenatis, dolor elit posuere quam, quis adipiscing urna leo nec orci. Sed nec nulla auctor odio aliquet consequat. Ut nec nulla in ante ullamcorper aliquam at sed dolor. Phasellus fermentum magna in augue gravida cursus. Cras sed pretium lorem. Pellentesque eget ornare odio. Proin accumsan, massa viverra cursus pharetra, ipsum nisi lobortis velit, a malesuada dolor lorem eu neque.

3.4.3 Distance Metrics

Morbi rutrum odio eget arcu adipiscing sodales. Aenean et purus a est pulvinar pellentesque. Cras in elit neque, quis varius elit. Phasellus fringilla, nibh eu tempus venenatis, dolor elit posuere quam, quis adipiscing urna leo nec orci. Sed nec nulla auctor odio aliquet consequat. Ut nec nulla in ante ullamcorper aliquam at sed dolor. Phasellus fermentum magna in augue gravida cursus. Cras sed pretium lorem. Pellentesque eget ornare odio. Proin accumsan, massa viverra cursus pharetra, ipsum nisi lobortis velit, a malesuada dolor lorem eu neque.

Euclidean Distance

Morbi rutrum odio eget arcu adipiscing sodales. Aenean et purus a est pulvinar pellentesque. Cras in elit neque, quis varius elit. Phasellus fringilla, nibh eu tempus venenatis, dolor elit posuere quam, quis adipiscing urna leo nec orci. Sed nec nulla auctor odio aliquet consequat. Ut nec nulla in ante ullamcorper aliquam at sed dolor. Phasellus fermentum magna in augue gravida cursus. Cras sed pretium lorem. Pellentesque eget ornare odio. Proin accumsan, massa viverra cursus pharetra, ipsum nisi lobortis velit, a malesuada dolor lorem eu neque.

Riemannian Distance

Morbi rutrum odio eget arcu adipiscing sodales. Aenean et purus a est pulvinar pellentesque. Cras in elit neque, quis varius elit. Phasellus fringilla, nibh eu tempus venenatis, dolor elit posuere quam, quis adipiscing urna leo nec orci. Sed nec nulla auctor odio aliquet consequat. Ut nec nulla in ante ullamcorper aliquam at sed dolor. Phasellus fermentum magna in augue gravida cursus. Cras sed pretium lorem. Pellentesque eget ornare odio. Proin accumsan, massa viverra cursus pharetra, ipsum nisi lobortis velit, a malesuada dolor lorem eu neque.

Learned Distance from Seeds

Morbi rutrum odio eget arcu adipiscing sodales. Aenean et purus a est pulvinar pellentesque. Cras in elit neque, quis varius elit. Phasellus fringilla, nibh eu tempus venenatis, dolor elit posuere quam, quis adipiscing urna leo nec orci. Sed nec nulla auctor odio aliquet consequat. Ut nec nulla in ante ullamcorper aliquam at sed dolor. Phasellus fermentum magna in augue gravida cursus. Cras sed pretium lorem. Pellentesque eget ornare odio. Proin accumsan, massa viverra cursus pharetra, ipsum nisi lobortis velit, a malesuada dolor lorem eu neque.

3.5 Max-Flow/Min-Cut Algorithms

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam ultricies lacinia euismod. Nam tempus risus in dolor rhoncus in interdum enim tincidunt. Donec vel nunc neque. In condimentum ullamcorper quam non consequat. Fusce sagittis tempor feugiat. Fusce magna erat, molestie eu convallis ut, tempus sed arcu. Quisque molestie, ante a tincidunt ullamcorper, sapien enim dignissim lacus, in semper nibh erat lobortis purus. Integer dapibus ligula ac risus convallis pellentesque.

3.5.1 Ford-Fulkerson

Nunc posuere quam at lectus tristique eu ultrices augue venenatis. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Aliquam erat volutpat. Vivamus sodales tortor eget quam adipiscing in vulputate ante ullamcorper. Sed eros ante, lacinia et sollicitudin et, aliquam sit amet augue. In hac habitasse platea dictumst.

Algorithm 1 Euclid's algorithm

```
procedure EUCLID(a, b)                    ▷ The g.c.d. of a and b
    r ← a mod b
    while r ≠ 0 do                        ▷ We have the answer if r is 0
        a ← b
        b ← r
        r ← a mod b
    end while
    return b                              ▷ The gcd is b
end procedure
```

3.5.2 Dinic/Edmond-Karp

Morbi rutrum odio eget arcu adipiscing sodales. Aenean et purus a est pulvinar pellentesque. Cras in elit neque, quis varius elit. Phasellus fringilla, nibh eu tempus venenatis, dolor elit posuere quam, quis adipiscing urna leo nec orci. Sed nec nulla auctor odio aliquet consequat. Ut nec nulla in ante ullamcorper aliquam at sed dolor. Phasellus fermentum magna in augue gravida cursus. Cras sed pretium lorem. Pellentesque eget ornare odio. Proin accumsan, massa viverra cursus pharetra, ipsum nisi lobortis velit, a malesuada dolor lorem eu neque.

3.5.3 Push-Relabel

Originally developed by Andrew V. Goldberg and Robert E. Tarjan. Previous algorithms, such as Ford-Fulkerson, used the concept of residual networks and augmenting paths to determine max-flow. Push-Relabel used the concept of preflow to determine max-flow instead of augmenting paths. Sometimes referred as the Preflow-Push Algorithm. Preflow is a concept originally developed by A.V. Karzanov.

The algorithm works at converting a preflow, f, into a normal flow and then terminates. This flow also turns out to be the maximum flow. Goldberg and Tarjan defined a generic Push-Relabel algorithm which solves the maximum flow problem.

Preflow A preflow is a real-valued function, f, on vertice pairs. The total flow into a vertex can exceed the flow out of a vertex but not vice versa.

A preflow where all $v \in V - \{s,t\}$ has a flow excess of zero, $e_f(v) = 0$, is a normal flow. The preflow function is also referred to as the **s-t preflow**.

Preflow must satisfy:

1. Capacity Constraint
   $\forall u, v \in V, f(u,v) \leq c(u,v)$
2. Antisymmetry/Skew Symmetry
   $\forall u, v, \in V, f(u,v) = -f(v,u)$
3. Nonnegative Constrain
   The flow into $v \in V - \{s\}$ must be greater than or equal to the flow out of v. $\forall u \in V, v \in V - \{s\}, \sum f(u,v) > 0$

Flow Excess Flow excess, $e_f(v)$, is the net flow into v where $v \in V$ for some preflow f.

$$e_f(v) = \begin{cases} \infty & \text{if } v = s \\ \sum_{u \in V} f(u,v) & \text{if } v \in V - \{s\} \end{cases}$$

Active Vertex An active vertex/node is a vertex v which satisfies all of the properties:

1. Not a source or sink, $v \in V - \{s,t\}$
2. Positive flow excess, $e_f(v) > 0$
3. Has a valid label, $d(v) < \infty$

Push-Relabel also uses the concept of a residual graph, $G_f = (V, E_f)$.

Residual Capacity The residual capacity of a preflow is defined as $r_f(v,w) = c(v,w) - f(v,w)$.

Residual Edges The residual edges for a preflow f is defined as the set of edges with positive residual capacity. $E_f = \{(v,w)\}|r - f(v,w) > 0$.

Labelling Push-Relabel also use a valid labelling function, d, to determine which vertex pairs should be selected for the push operation.

A valid labelling , d, is a nonnegative integer function applied to all vertices to denote a label. The labelling is often referred as the height or distance from the sink node, t. This function is sometimes compared to the physical intuition that liquids naturally flow downhill.

A valid labelling for a preflow consists of:

1. For $v \in V, 0 \leq d(v) \leq \infty$

2. $d(s) = |V|$ (source condition)
3. $d(t) = 0$ (sink condition)
4. $d(v) = d(w) + 1$ for every residual edge $(v, u) \in E_f$

A labelling d and a preflow f are said to be compatible id d adheres to the properties above.

The algorithm pushes flow excess starting at the source, s, along all vertices towards the sink, t. The algorithm maintains a compatible vertex labelling function, d, to the preflow, f. The labelling is usedto determine where to puch the flow excess. The algorithm repeatedly performs either a push or a relabel operation so long as there is an active vertex in G_f.

Push Operation The push operation is used to move flow from one vertex to another. The transfer of excess can be performed across the vertex pair $(v, w) \in E_f$ if:

1. v is an active vertex
2. the edge has positive residual capacity, $r_f(v, w) > 0$
3. the label distance $d(v) = d(w) + 1$

This allows the algorithm to move δ excess flow: $\delta = min(e_f(v), r_f(v, w))$ from v to w. A push is considered **saturating** if no more flow can be sent over the edge, $\delta = r_f(v, w)$. A push is considered to be **non-saturating** if all the excess from v the push over the dge and the edge still has some cpacity, $\delta = e_f(v)$.

Algorithm 2 Push Operation

Input: Preflow f, labels d, and (v, w) where $v, w \in V$
Output: Preflow f
Applicable: if $v \in V - \{s, t\}$, $d(v) < \infty$, $e_f(v) > 0$, $r_f(v, w) > 0$ and $d(v) = d(w) + 1$

```
1: procedure PUSH(v, w)
2:     δ := min(e_f(v), r_f(v, w)
3:     f(v, w) := f(v, w) + δ
4:     f(w, v) := f(w, v) − δ
5:     e_f(v) := e_f(v) − δ
6:     e_f(w) := e_f(w) + δ
7:     return f
8: end procedure
```

Relabel Operation The relabel operation is used to increase the label value of a single active vertex so that excess flow can be pushed out of the active vertex. The relabel operation is performed when all the residual edges of the active vertex have positive residual capacity, $r_f(v, w) > 0$. This implies that v's label is less than or equal to all vertices, $d(v) \leq d(w)$, meaning that no push operation across the edges is possible given the push condition $d(v) = d(w) + 1$.

The relabel operation for some vertex v selects the smallest label for the vertices with positive residual edges, $r_f(v, w) > 0$. The active vertex is then assigned the smallest label value $+1$ such that $d(v) := mind(v) + 1|(v, w) \in E_f$. This will alow the vertex v to potentially push its excess flow to atleast one of the othe vertices during the algorithm's next iteration.

Algorithm 3 Relabel Operation

Input: Preflow f, labels d, and $v \in V - \{s, t\}$
Output: Labels d
Applicable: if $v \in V - \{s, t\}$, $d(v) < \infty$, $e_f(v) > 0$, and $\forall w \in V, r_f(v, w) > 0$ which implies $d(v) \leq d(w)$

```
procedure RELABEL(v)
    if {(v,w) ∈ E_f} ≠ 0 then
        d(v) := min(d(w) + 1|(v,w) ∈ E_f)
    else
        d(v) := ∞
    end if
    return d
end procedure
```

The algorithm initialises the following values in the residual graph before the push andrelabel operations in the main loop.

1. Initialise the preflow of all edges in the residual graph
2. Initialise the labellings such that:
   (a) $d(s) = |V|$
   (b) $d(v) = 0$for $v \in V - \{s\}$
3. Performs saturation, pushes along all residual edges out of the source $(s, v) \in E_f$ and $v \in V$.

Once complete the algorithm repeatedly performs either a push or a relabel operation against all vertices. The algorithm continues until no operation can be performed. The algorithm terminates when there are no more active vertices.

The analaysis and the proof of correctness of the Push-Relabel algorithm can be found in Appendix B.

Push-Relabel Speed Optimisation Heuristics

Discharge Push-Relabel also use a valid labelling function ,d, to determine which vertex pairs should be selected for the push operation.

FIFO Push-Relabel also use a valid labelling function ,d, to determine which vertex pairs should be selected for the push operation.

Highest Label First Push-Relabel also use a valid labelling function ,d, to determine which vertex pairs should be selected for the push operation.

Algorithm 4 Push-Relabel Main-loop

Input: Network flow graph $G = (V, E)$, s, t and c
Output: Maximum flow f

procedure $\textsc{Main}(v)$
    for all $(v, w) \in (V - \{s\})(V - \{s\})$ **do**
        $f(v, w) \coloneqq 0$
        $f(w, v) \coloneqq 0$
    end for

    for all $v \in V$ **do**
        $f(s, v) \coloneqq r_f(s, v)$
        $f(v, s) \coloneqq -r_f(s, v)$
    end for

    $d(s) = |V|$

    for all $v \in V - \{s\}$ **do**
        $d(v) \coloneqq 0$
        $e_f(v) \coloneqq f(s, v)$
    end for

    While there exists an active vertex
    while $\exists v \in V - \{s, t\}$ **do** ▷ with either applicable PUSH() or RELABEL() operation
        Perform either a PUSH or a RELABEL operation on v
    end while

    return f
end procedure

Global Relabel Push-Relabel also use a valid labelling function ,d, to determine which vertex pairs should be selected for the push operation.

Gap Relabel Push-Relabel also use a valid labelling function ,d, to determine which vertex pairs should be selected for the push operation.

Chapter 4

Semi-Analytical Determination of λ and σ for Gaussian Smoothing Energies and Non-Uniform Graph Construction for Automatic Graph Cuts

4.1 Determining FG and BG seeds/probability distributions

4.2 Determining Optimal Paramters Settings

4.3 Single-Channel Data

4.4 Multi-Channel Data

Chapter 5

Graph Cut Solution to ACWE Chan-Vese Segmentation

Chapter 6

Pre-Processing and Post-Processing Scheme for Fluorescence Microscopy Images

6.1 Removal of Artifacts using Connected Components

6.2 Anisotropic Diffusion

6.2.1 Coherence Enhancing Diffusion

6.2.2 Coherence Enhancing Diffusion with Optimised Rotational Invariance

6.3 Poisson Denoising

6.3.1 Total-Variation Denoising

6.4 Contrast Enhancement

Appendix A

Proof of Maximum Flow - Minimum Cut Equivalence

Write your Appendix content here.

Appendix B

Push-Relabel Algorithm Analysis and Proof of Correctness

Write your Appendix content here.