A NURSE SCHEDULING USING GRAPH COLOURING

BY

ANANE GIDEON [BED. (HONS) MATHEMATICS]

A THESIS SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL MATHEMATICS

KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY IN

PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

INSTITUTE OF DISTANCE LEARNING

MAY, 2013

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

DECLARATION

I hereby declare that this submission in my own towards the MSc and that, to the best of my knowledge, it contains no material previously published by another person nor material which has been accepted for the award of any other degree of the University, except where due acknowledgement has been made in the text.

ANANE GIDEON PG6316511	……	……
Student Name & ID	**Signature**	**Date**

Certified by

MR CHARLES SEBIL	……	……
Supervisor Name	**signature**	**Date**

Certified by

PROF S. K AMPONSAH	……...	……
Head of Mathematics Department	**Signature**	**Date**

ABSTRACT

The aim of this thesis work is to provide effective method for solving Nurse Scheduling Problem (NSP) by satisfying the nurses, patients and hospital requirements. Nurse schedule problem is a major problem faced by many hospitals all over the world. That is a subclass of scheduling problems that are hard to solve. The work is difficult for the duty planner because the duty planner has to ensure that every scheduling decision made complies with a mixture of hard hospital rules and soft nurse preference rules. The thesis describes the design and implementation of a constraint-based nurse scheduling using graph colouring. A conflict graph was constructed and the vertices of the graph represented the different types of nurses. The vertices were then coloured using Greedy algorithm approach and this removed the various conflicts. The result was then used to create the nurses schedule. Results showed a feasible solution to the problem.

ACKNOWLEDGEMENT

Firstly, I thank Almighty God for giving me the strength, knowledge and vision to come out with this thesis work.

Secondly, I want to express appreciation to my supervisor, Mr. Charles Sebil for his encouragement, support and most of all expertise, May God richly bless you and your family.

Special thanks also go to the head of department of mathematics, Prof S.K. Amponsah and all the lecturers in the department for their encouragement.

I want to express special thanks to my family members who have inspired me, prayed for me and constantly encouraged me in the writing of this thesis work. You are treasures in my life! I appreciate you – Mr. and Mrs. Joyce Anane, Enock Anane, Mr. and Mrs. Esther Kyere, Abigail Anane, Grace Anane, Philip Anane, Pricilla Anane, Emmanuella Anane and Puis Anane not forgetting my Grandfather, Puis Mensah (OFa Kwabena) for his encouragement and inspiration. God bless you GrandPa.

I am also grateful to the principal nurse officer (matron), Cecilia Millicent Anan of the Female/Peadiatric ward of Juaso District Hospital and all the nurses over there for allowing me to use the ward. God bless you all.

My appreciation would not be complete without expressing appreciation to Rev. Osei Agyemang, Antiedu Baffour, Hon. Asiamah Dickson, MrNsiah, Madam Betrice, Madam Linda Octhere, Madam Sabina, Madam Appiagyeiwaa and all teachers of Morso. God use you as a source of inspiration for me. Thank you!

CHAPTER 1

1.0 INTRODUCTION

1.1 BACKGROUND OF THE STUDY

One of the most exciting mathematical development of the twentieth century was the proof, in 1976, of The Four-Color Theorem, whose proof had remained unsolved since 1852.While try to color a map of the counties of England, Francis Guthrie postulated the four colour conjecture, noting that four colours were sufficient to colour the map so that no regions sharing a common border received the same colour. For well over 100 years a vast amount of work and theories were developed to prove it until finally in 1976 Kenneth Appel and Wolfgang Haken found it's prove.

There are several interesting practical and feasible problems that can be modeled by graph colouring. The surge in recent times has resulted in countless real world problem applications, which includes; Time tabling Scheduling problems, Frequency Assignment, Register allocation, Register Allocation, Analysis of Biological and Archaeological Data and pattern Matching

In any Organization that operates continuously, daily work is divided into shifts to minimize the complexity and to carry on the work in an easy manner. In such a context, the scheduling problem consists in assigning a schedule to each worker, which involves building a timetable for a specified period. Scheduling can be thought of as a decision making process which involves the allocation of limited resources to tasks over time. One of the definitions of scheduling is given by Wren1, who stated that "Scheduling is the arrangement of objects into pattern in time or space in such a way that some goals

are achieved, or nearly achieved". Wren has been described about the rostering problem also. Rostering is the placing of resources into slots in a pattern.

Hospital is one example for above mentioned type of organization. In a hospital there are various kinds of employers like doctors, nurses, attendants, etc. and they must be assigned to shifts to do their work. This study mainly considered about the nurse shifts which is given for nurses named as nurse roster.

Hospital care units must provide twenty four hour nursing coverage at levels to match patient demand while adhering to organizational policies designed to protect the health and welfare of patients and staff. The already difficult scheduling problem is further compounded by a shortage of nurses. The schedule has to determine the day-to-day shift assignments of each nurse for a specified period of time in a way that satisfies the given requirements as much as possible, taking into account the wishes of nurses as closely as possible.

1.2 STATEMENT OF THE PROBLEM

Juaso Districts Hospital is a multi-discipline hospital in AsantiAkim South of Ghana. It consists of a number of operation wards and the Female / Paediatric ward is one of them. The female/ Paediatric ward provides 24-hour services to the general public seven days a week. Scheduling nurses to staff shift is usually made by a head nurse (matron) manually. Manually created timetable has created a lot of conflict and problems despite the great effort required to form a duty roaster (timetable).

The allocation of nursing staff is a critical task in hospital management. The allocation of the right number and skill mix of staff to each shift becomes crucial. Nurse schedule policies can have a direct impact on nurse satisfaction and hence on turnover. Schedule requiring nurses to work difficult and tiring combinations of shifts can again impact on the quality and safety of patient care. Hospital management is therefore further concerned with providing rosters that minimise nurse dissatisfaction

1.3 OBJECTIVE OF THE STUDY

The purpose of this thesis work is to apply graph colouring technique to generate efficient and reliable nurses' schedule. This research work seeks to:

1. Produce the right combination of nurses for each shift
2. Remove various conflicts which normally course problems in the nurses schedule
3. Produce the right number of nurses for each shift.

Base on the prepared schedule, nurses will be assigned for working shift with consideration of time, requirement, along with experience and nurse skills.

1.4 METHODOLOGY

As described in above, to solve nurse scheduling problem (NSP), nurses needed to be assigned into shifts. In the proposed solution for the NSP, nurses were divided into shift groups and then nurses in one shift group were assigned to one shift. When creating the shifts, nurses from those different shift groups were not jointly used for a one shift. This is a kind of scheduling problem. Major problem of scheduling problem is allocation of resources in an effective way. Graph coloring was used in creating the shift groups. This was done with matgraph a tool box matlab software.

Because the NSP is a Constraint Satisfaction Problem (CSP), constraints analysis is very important in solving this problem. During the requirements analysis several constraints were identified. After analyzing those constraints, it was divided into two groups according to the effect of those constraints to the final solution. Violating of some constraints will affect the solution directly and violating of some constraints will be affecting the quality of the solution.

Similar to other studies which were carried out to solve NSP, the two groups were named as Hard Constraints and Soft Constraint. Hard constraints are the constraints, which must be satisfied to get feasible solution for use in practice and Soft constraints the constraints, which are used to evaluate the quality of the solution. So soft constraints are not compulsory but are desired to be satisfied as much as possible.

1.5 JUSTIFICATION OF THE STUDY

Making sure that each shift is properly staffed is one of the hardest challenges that head nurse faces. In general, staff members prefer to have more input and flexibility in their scheduling, but the more flexible scheduling is, the harder it is for a head nurse to supervise scheduling and make sure that the ward is correctly staffed. In view of this, manual way of making the nurse schedule is really a trouble for the head nurse since the results of the schedule is highly affecting nurses as well as the patients. Since people need healthcare throughout 24 hours, it is important to design effective automated nurse scheduling software to manage nurse duty activities.

1.6 SCOPE AND LIMITATIONS OF THE STUDY

Specifically, this thesis work considers nurse scheduling system focusing primarily on the shift structure for the Juaso District Hospital. Even within this hospital the work is limited to the female and padeatric ward of the hospital with the hope that the work can be replicated to other wards, district, municipal and general hospital in Ghana. There is more room for using this work as basis or reference for further studies to investigate other schedule problems in other sectors of our economy.

1.7 ORGANISATION OF THE STUDY

This thesis work has been organized into five chapters, chapter one gives a brief account of the history of graph colouring and its application in nurse scheduling and other fields. It discusses the statement of problem, objectives, methodology, justification and the scope and limitation of the study. Chapter two also gives some related literature works on the thesis work. Chapter three presents an overview of graph theory involving graph colouring. Definitions of some basic terms and lemmas to help

in our study are presented and proved. Collection of data and modeling of nurse schedule was discussed in chapter four. Chapter five studies the results obtained and the necessary conclusion made. Recommendations were also included in the chapter five.

CHAPTER 2

2.1 LITERATURE REVIEW

Existing research works has been proposed diverse models and methodologies to improve nurse scheduling problems. Most of the current proposed solutions either make use of random based optimization algorithms which won't be efficient or applicable.

Tao et al., (2011) has done much research work related to Medical informatics and had deep discussions about the role of data warehouse management system to handle hospital and nurse management information. Multidimensional analysis techniques under different angles were used to extract the required data and information.

Silver et al., used data mining technique for data warehouse and published their findings under the title "Case study: How to apply data mining techniques in a Healthcare Data warehouse". This approach has been implemented successfully in many of the American hospitals. Two numerous data mining techniques called; patient rule introduction method (PRMI) and weighted items sets (WLS) were used to analyse large quantities of data.

Villiers et al., (1998) applied data mining techniques for solving clinical data warehouse functionality and proposed Flexible clinical data mining system (CDMS) using SAS statistical software. In addition, research is carried out in two stages. In first stage, controlled environment were provided for CDMS access based systems and transformed it into analytical clinical data. In the later stage, operations were tested

with the row data operations with same data. Peter Villiers proposes genomic based data for further performance enhancements.

Cheng et al., describes the design and implementation of a constraint-based nurse rostering system using a redundant modeling approach. To reduce search time, they proposed redundant modeling, an effective way to increase constraint propagation through cooperation among different models for the same problem. Their problem domain involved around twenty-five to twenty-eight nurses and eleven shift types.

Kundu et al., (2008) described the use of Genetic Algorithm (GA) for solving nursing schedule problem (NSP). They used two different models, Simulated Annealing and Genetic Algorithm to solve this problem. Compare nurse performance at different levels. They have considered soft and hard constraints.

Juhos et al., (2004) described a novel representation and ordering model that, aided by an evolutionary algorithm, was used in solving the graph k-coloring problem. Its strength lies in reducing the number of neighbours that need to be checked for validity. An empirical comparison was made with two other algorithms on a popular selection of problem instances and on a suite of instances in the phase transition. The new representation in combination with a heuristic mutation operator showed promising results

Culberson and Gent (2001) denned the ‘frozen development’ of colouring random graphs and identified two nodes in a graph as frozen if they were the same colour in all legal colourings. This was analogous to studies of the development of a backbone or spine in SAT (the Satis ability problem). The authors described in detail the algorithmic

techniques used to study frozen development and presented strong empirical evidence that freezing in 3-colouring is sudden. A single edge typically caused the size of the graph to collapse in size by 28% and used the frozen development to calculate unbiased estimates of probability of colourability in random graphs, even where this probability was low. The links between frozen development and the solution cost of graph colouring was investigated. In SAT, a discontinuity in the order parameter was correlated with the hardness of SAT instances; data for colouring was suggestive of an asymptotic discontinuity. The uncolourability threshold was known to give rise to hard test instances for graph-colouring. Evidence that the cost of colouring threshold graphs grows exponentially, when using either a specialist colouring program, or encoding into SAT, or even when using the best of both techniques were presented. Theoretical and empirical evidence showed that the size of the smallest uncolourable sub graphs of threshold graphs became large as the number of nodes in graphs increases. The application of their work to the statistical mechanics analysis of colouring was discussed extensively.

The graph-theoritic parameter that has probably received the most attention over the years in the chromatic number. As is well-known, the colouring problem is an NP-Complete problem. Lie et al., (2002) solved by means of molecular biology technique to this effect. The algorithm was highly parallel and has satisfactory fidelity. Their work showed further evidence for ability of DNA computing to solve NP-Complete problems. (Graph colouring problem). The ever number of wireless communications systems deployed around the globe have made the optimal assignment of a limited radio frequency spectrum a problem of primary importance, at issue are planning models for permanent spectrum allocation, licensing, regulation, and network design.

Further as issue are on-line algorithms for dynamically assigning frequencies to users within an established network. Applications include aeronautical mobile, land mobile, maritime mobile, broadcast, land fixed (pointto-point), and satellite systems.

Murphy et al., (1999) surveyed researches conducted by theoreticians, engineers, and computer scientists regarding the frequency assignment problem (FAP) in all of its guises. Their paper began by defining some to the more common types of FAPs. It continued with a discussion on measures of optimality relating to the use of spectrum, models of interference, and mathematical representations of many FAPs, both in graph theoretic terms, and as mathematical programs. Graph theory and, in particular, graph colouring play an important role in the FAP since, in many instances, the FAP in cast in a form which closely resembles a graph colouring. Theoretical results that bound optimal solutions for special FAP structures were presented. Exact algorithms for general FAPs were explained, and since many FAP instances are computationally hard, much space was devoted to approximate algorithms. Their paper concluded with a review of evaluation methods for FAP algorithms, test problem generators, and a discussion of the underling engineering issues that was considered when generating test problem.

Hedetniemi et al., (2003) proposed two new self-stabilizing distributed algorithms for proper *i* (*i* is the maximum degree of node in the graph) colouring of arbitrary system graphs. Both algorithms were capable of working with multiple types of demons (schedulers).

The first algorithm converges in Ç Ň moves while the second coverages in at most Ò moves (Ò is the number of nodes and Ň is the number of edges in the graph). The

second improvement was that neither of the proposed algorithms requires each node to have knowledge of *i*. Further, the colouring produced by their first algorithm provided an interesting special case of colouring e.g., Grundy Colouring.

The problem of properly colouring the vertices (or edges) of a graph using for each vertex (or edge) a colour from a prescribed list of permissible colours, received a considerable amount of attention. Alon (1993) described the techniques applied in his study of this subject, which combined combinatorial, algebraic and probabilistic methods, and discussed several intriguing conjectures and open problems. This was mainly a survey of recent and less results in the area, but it contained several new results as well.

Simulated annealing is also a very successful heuristic for various problems in combinatorial optimization. An application of simulated annealing to the 3-colouring problem was considered by Nolte and Schrader (1999). In contrast ot many good empirical results they showed for a certain class of graphs that the expect first hitting time of a proper colouring, given an arbitrary cooling scheme, was of an exponential size and proved the convergence of simulated annealing to an optimal solution in an exponential time.

D'Hondt (2008) investigated quantum algorithms for graph colouring problems, in particular for 2- and 3- colouring of graphs. The main goal was to establish a set of quantum representations and operations suitable for the problem at hand and proposed a unitary-as well as measurement based quantum computations, also taking inspiration from answer set programming, a form of declarative programming close to traditional logic programming. The approach used was one in which he first generate arbitrary solutions to the problem, then constraining those according to the problem's input.

Though he did not achieve fundamental speed-ups, his algorithms showed quantum concepts could be used for programming and moreover exhibit structural differernces. For example, the computations of all possible colourings at the same time. Comparing, his algorithms with classical ones, highlighting how the same type of difficulties gave a rise to NP-complete behavior, and proposed possible improvements.

Graph-colouring register allocation is an elegant and extremely popular optimization for modem machines. But as currently formulated, it does not handle two characteristics commonly found in commercial architectures. First, a single name may appear in multiple register classes, where a class is set of register names that are interchangeable in a particular role. Second multiple register names may be aliases for single hardware register. Holloway et al., (1993) represented a generalization of graph-colouring register allocation that handle these problematic characteristics while preserving the elegance and practicality of traditional graph colouring. Their generalization adapts easily to a new target machines, requiring only the set of names in the register classes and a map of the register aliases. It also drops easily into a well-known graph-colouring allocator, is efficient at compile time, and produces high-quality code.

Duffy el al., (2006) analysed the complexity of decentralized colouring algorithm that had recently been proposed for channel selection in wireless computer networks. Colouring a graph with its chromatic number of colours is known to be NP-hard. Identify an algorithm in which decisions are made locally with no information about the graph's global structure is particularly challenging.

Koyuneu and Secir (2004) used graph colouring algorithm to generate the student weekly time table in a typical university department. Their problem was a Hode-point problem and it could not be solved in the polynomial domain. Various constraints in weekly scheduling such as lecture demands, course hours and laboratory allocations were confronted and weekly time tables were generated for first, second, third and fourth year students in a typical semester.

Burke el al., (1995) developed a general system able to cope with the ever changing requirements of large educational institutions. They presented the methods and techniques behind such a system. Graph Colouring and room allocation algorithms were also presented and it was shown in their basis of a flexible and widely applicable timetabling system.

They intended to overcome the problem of intractability by producing spreadsheet type system that the user could be guided in an informed and useful way. That gives the user control of the search and the possibility of backtracking where no reasonable solution is found, while still letting the heuristic algorithms to the hard work. Their approach cannot guarantee an optimal solution but it can guarantee a solution the user is happy with.

Griesmer (1993) reported a successful application involving 32 nurses. Although the approach is appealing, it is quite difficult to implement in practice because of the impracticality of holding meetings to resolve conflicts, especially for large units. Moreover, the results may not necessarily be perceived as fair. Those who are savvy enough to game the system will always have an advantage over the procrastinators. Controlling the sign-up order and rotatingit over the year is a partial solution.

Burns (1978) studied the case of 10 days on in a 14-day planning horizon with every second weekend off and up to six consecutive days on. Although easy to implement, cyclical schedules have become the bane of the profession because of the rigidity they impose. Many nurses view flexibility as an entitlement that comes with the job.

Warner (1976) was the first to develop a methodology for solving the set covering model for the case in which all nurses work 8-hour shifts. To overcome the unmanageable size of the full IP, only 50 good schedules obtained by a greedy method were included in the model for each nurse. A block pivoting strategy was used to find feasible solutions, which were then improved with a post-processor. The methodology was implemented at two hospitals with staff sizes ranging from 19 to 47 nurses.

Jaumard et al. (1998) extended the basic IP model to include both 8- and 12-hour shifts, and different levels of nursing skills. This extension adopted the concept of demand periods, a unit time bucket that alleviates the problem of overlapping working hours in two different shifts. The modified problem was then solved using Dantzig–Wolfe decomposition with the necessary adjustments for integrality restrictions. Columns were generated at each iteration by solving a resource-constrained shortest path subproblem, one for each nurse.

Miller et al. (1976) used a simplified version of a rotation heuristic for a 4-week problem. The objective function was a weighted combination of personnel costs and preference penalties. A greedy heuristic with feasible neighborhood swaps was adoptedto solve a real instances with up to 12 nurses.

Dowsland (1998) developed a tabu search approach with strategic oscillation. The algorithm starts with an initial roster obtained with a greedy heuristic that ignores the minimum coverage requirement and treats each nurse separately. In the next phase, three swap moves are used to try to satisfy the coverage constraints: shift swaps for a single nurse, two-nurse chain swaps, and rotation swaps. The quality of each schedule produced512 J.F. Bard, H.W. Purnomo / European Journal of Operational Research 164 (2005) 510–534 by the algorithm is measured by a weighted sum of the penalty coefficients associated with the preference violations.

Arthur and Ravindran (1981) were the first to apply goal programming to the preference scheduling problem. They considered the following four goals: contractual requirements, preferences, requests, and staffing requirements. In the first phase of their two-phase approach, a small IP is solved to decide the day on which each nurse is to work. Shift assignments are made in the second phase.

In a similar vein, Berrada et al. (1996) proposed a constraint satisfaction model that viewed demand coverage and the number of working days as the hard constraints and compliance to shift patterns, daily requirementsfor supervisory personnel, and the grouping of days off and weekends off as the soft constraints.

For the same problem,

Ferland et al. (2001) developed a tabu search methodology that included a diversification strategy and an adaptive memory structure. Similarly, Burke et al. (1999) used tabu search to schedule nurses at several Belgium hospitals using a code implemented in the Plane software system.

Subsequent refinements allowed for multiple objectives and more equitable weekend assignments.

CHAPTER 3

3.1 INTRODUCTION

In this chapter some theories and definitions of graph theory were enunciated. Also the algorithm used in colouring the graph was also considered in this chapter.

3.2 OVERVIEW OF GRAPH THEORY

DEFINITION: A graph is a non-empty finite set of vertices V along with a set E of two-element subsets of V.

Let V be a *finite* set, and denote by

$$E(V) = \{\{u, v\} \mid u, v \in V, u = v\} .$$

the2-sets of V, i.e., subsets of two distinct elements.

A pair $G = (V, E)$ with $E \subseteq E(V)$ is called a graph (on$V$). The elements of V are the vertices of G, and those of E the edges of G. The vertex set of a graph G is denoted by $V_G$and its edge set by E_G. Therefore $G = (V_G, E_G)$.

In literature, graphs are also called *simple graphs*; vertices are called *nodes* or *points*; edges are called *lines* or *links*. The list of alternatives is long (but still finite).

A pair $\{u, v\}$ is usually written simply as uv. Notice that then $uv = vu$. In order to simplify notations, we also write $v \in G$ and $e \in G$ instead of $v \in V_G$and$e \in E_G$.

DEFINITION. For a graph G, we denote

$N_G = |V_G|$ and $\varepsilon_G = |E_G|$.

The number $v_G$of the vertices is called the order of G, and $\varepsilon_G$is the size of G.

For an edge $e = uv \in G$, the vertices u and v are its ends. Vertices u and v are

adjacentorneighbours, if $uv \in G$.

Two edges $e1 = uv$ and $e2 = uw$ having a common end, are adjacent with each other.

A graph G can be represented as a plane figure by drawing a line (or a curve) between the points $u$and v (representing vertices) if $e = uv$ is an edge of G.

The figures below are geometric representation of the graph G with

$V_G = \{v1, v2, v3, v4, v5, v6\}$ and

$E_G = \{v1v2, v1v3, v2v3, v2v4, v5v6\}$.

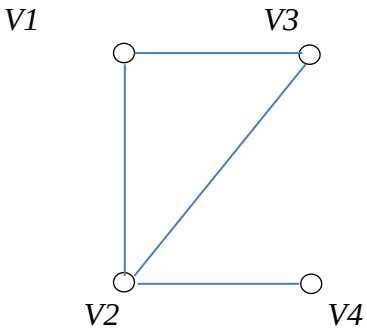

Figure 3.1 A simple graph

Figure 3.2 A simple graph

3.3.1 Multigraph

Graphs can be generalized by allowing loopsvv and parallel (or multiple) edges between vertices to obtain a multigraph$G = (V, E, \psi)$, where $E = \{e1, e2, \ldots, e_m\}$ is a set (of symbols), and

$\psi : E \rightarrow E(V) \cup \{vv \mid v \in V\}$ is a function that attaches an unordered pair of vertices to each $e \in E$: $\psi(e) = uv$.

Note that we can have $\psi(e1) = \psi(e2)$. This is drawn in the figure of G by placing two (parallel) edges that connect the common ends. On the right there is (a drawing of) a multigraph G with vertices $V = \{a, b, c\}$

and edges $\psi(e1) = aa$, $\psi(e2) = ab$, $\psi(e3) = bc$, and $\psi(e4) = bc$.

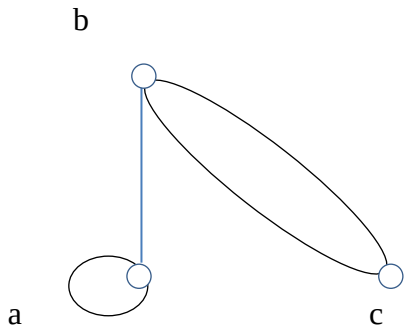

Figure 3.3 Multigraph

3.2.2 Directed Graphs OrDigraphs

DEFINITION.

$D = (V, E)$, where the edges have a direction, that is, the edges are ordered:

$E \subseteq V \times V$. In this case, $uv = vu$.

The directed graphs have representations, where the edges are drawn as arrows.

A digraph can contain edges uv and vu of *opposite directions*.

Graphs and digraphs can also be coloured, labelled, and weighted:

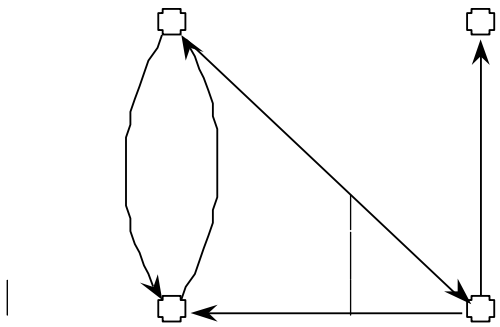

Figure 3.4 A directed graph

DEFINITION. A function α: $V_G \rightarrow K$ is a vertex colouring of G by a set K of colours.

A function $\alpha : E_G \rightarrow K$ is an edge colouring of G. Usually, $K = [1, k]$ for some $k \geq 1$.

If $K \subseteq \mathrm{R}$ (often $K \subseteq \mathrm{N}$), then α is a weight function or a distance function.

3.2.3 Isomorphism of graphs

DEFINITION. Two graphs G and H are isomorphic, denoted by $G \sim= H$, if there exists abijection$\alpha : V_G \rightarrow V_H$ such that

$uv \in E_G \Leftrightarrow \alpha(u)\alpha(v) \in E_H$

for all $u, v \in G$.

Hence G and H are isomorphic if the vertices of H arerenamings of those of G.

Two isomorphic graphs enjoy the same graph theoretical properties, and *they are oftenidentified*. In particular, all isomorphic graphs have the same plane figures (excepting the identities of the vertices). This shows in the figures, where we tend to replace the vertices by small circles, and talk of 'the graph' although there are, in fact, infinitely many such graphs.

The following graphs are isomorphic. Indeed, the required iso-morphism is given by

$V1 \rightarrow 1$, $V2 \rightarrow 3$, $V3 \rightarrow 4$, $V4 \rightarrow 2$, $V5 \rightarrow 5$.

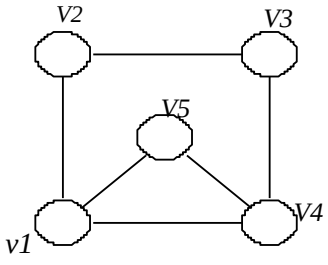

Figure 3.5

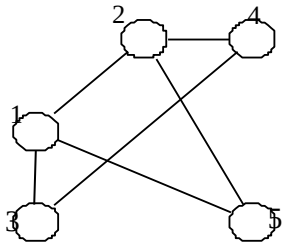

Figure 3.6

Isomorphic graphs

3.3 Other Representations

Plane figures catch graphs for our eyes, but if a problem on graphs is to be *programmed*, then these figures are, to say the least, unsuitable. Integer matrices are ideal for computers, since every respectable programming language has array structures for these, and computers are good in crunching numbers.

3.3.1 Adjacency matrix

Let $V_G = \{v1, \ldots, v_n\}$ be ordered. The adjacency matrix of G is the $n \times n$-matrix M with entries $M_{ij} = 1$ or $M_{ij} = 0$ according to whether $v_iv_j \in G$ or $v_iv_j \notin G$.

For instance, the graph below has an adjacency matrix on the right. Notice that the adjacency matrix is always symmetric (with respect to its diagonal consisting of zeros).

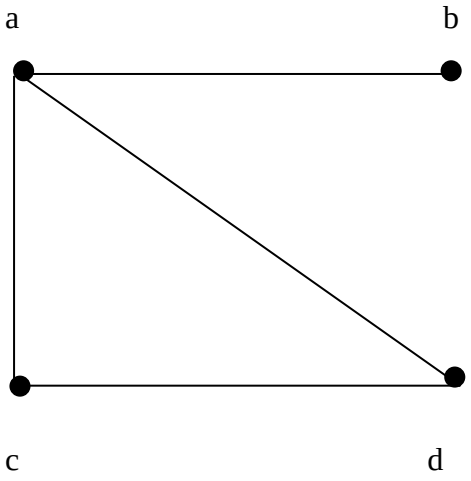

Figure 3.7 G

Table 3.1 Adjacency matrix of graph G

	a	d	c	d
a	0	1	1	1
b	1	0	0	0
c	1	0	0	1
d	1	0	1	0

A graph has usually many different adjacency matrices, one for each ordering of its set V_G of vertices. The following result is obvious from the definitions.

Theorem 3.1.*Two graphs G and H are isomorphic if and only if they have a common adjacency matrix. Moreover, two isomorphic graphs have exactly the same set of adjacency matrices.*

Graphs can also be represented by sets. For this, let X = $\{X1, X2, \ldots, X_n\}$ be a family of subsets of a set X, and define the intersection graph$G_X$as the graph with vertices $X1, \ldots, X_n$, and edges $X_iX_j$for all i and j ($i = j$) with $X_i \cap X_j 6= \emptyset$.

Theorem 3.2.*Every graph is an intersection graph of some family of subsets.*

Proof. Let G be a graph, and define, for all $v \in G$, a set

$$Xv = \{\{v, u\} \mid vu \in G\}.$$

Then $X_u \cap X_v 6= \emptyset$ if and only if $uv \in G$.

Let s (G) be the smallest size of a base set X such that G can be represented as an intersection graph of a family of subsets of X, that is,

$s(G) = \min\{|X| \mid G \sim_= GX \text{ for some } X \subseteq 2^X\}$.

How small can $s(G)$ be compared to the order v_G(or the size ε_G) of the graph? It was shown by Kou, S Tockmeyer And Wong (1976) that it is algorithmically difficult to

determine the number $s(G)$ – the problem is NP-complete.

3.3.2 Degrees of vertices

DEFINITION. Let $v \in G$ be a vertex a graph G. The neighbourhood of v is the set

$N_G(v) = \{u \in G \mid vu \in G\}$.

The degree of v is the number of its neighbours or the number of edges that come out from a vertex:

$d_G(v) = |N_G(v)|$.

If $d_G(v) = 0$, then v is said to be isolated in G, and if $d_G(v) = 1$, then v is a leaf of the graph. The minimum degree and the maximum degree of G are defined as

$\delta(G) = \min\{d_G(v) \mid v \in G\}$ and $\Delta(G) = \max\{d_G(v) \mid v \in G\}$.

The following lemma, due to Euler (1736), tells that if several people shake hands, then the number of hands shaken is even.

Lemma 3.2 (Handshaking lemma).*For each graph G,*

$$\sum_{v \in G} d_G(v) = 2\varepsilon_G.$$

Moreover, the number of vertices of odd degree is even.

Proof.

Every edge $e \in E_G$has two ends. The second claim follows immediately from the first one. Lemma 3.1 holds equally well for multigraphs, when $d_G(v)$ is defined as the number of edges that have v as an end, and when *each loop vv is counted twice.*

Note that the degrees of a graph G do not determine G. Indeed, there are graphs $G = (V, E_G)$ and $H = (V, E_H)$ on the same set of vertices that are *not* isomorphic, but for

which $dG(v) = dH(v)$ for all $v \in V$.

3.3.3 Subgraphs

DEFINITION. A graph H is a subgraph of a graph G, denoted by $H \subseteq G$, if $V_H \subseteq V_G$ and $E_H \subseteq E_G$.

A subgraph $H \subseteq G$ spans G (and H is a spanning subgraph of G), if every vertex of G is in H, i.e., $V_H = V_G$.

Also, a subgraph $H \subseteq G$ is an induced subgraph, if $E_H = E_G \cap E(V_H)$. In this case, H is induced by its set V_H of vertices.

In an induced subgraph $H \subseteq G$, the set E_H of edges consists of all $e \in E_G$ such that $e \in E(V_H)$. To each nonempty subset $A \subseteq V_G$, there corresponds a unique induced subgraph

$G[A] = (A, E_G \cap E(A))$.

To each subset $F \subseteq E_G$ of edges there corresponds a unique spanning subgraph of G,

$G[F] = (V_G, F)$.

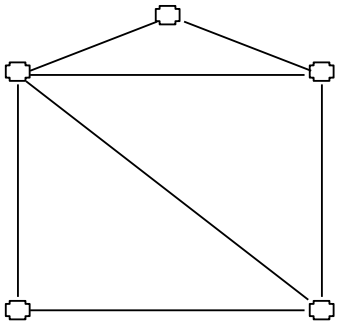

Figure 3.8 G

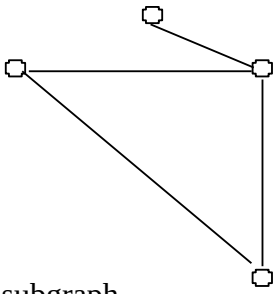

Figure 3.9A subgraph

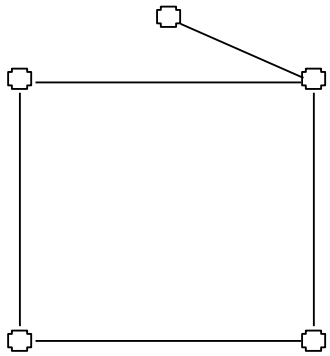

Figure 3.10A spanning

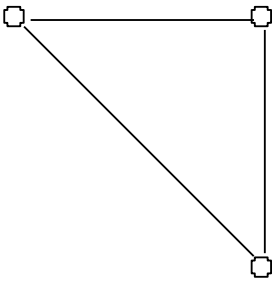

Figure 3.11Induced graph

For a set $F \subseteq E_G$ of edges, let

$$G - F = G[E_G \setminus F]$$

be the subgraph of G obtained by removing (only) the edges $e \in F$ from G. In particular, $G - e$ is obtained from G by removing $e \in G$.

Similarly, we write $G + F$, if each $e \in F$ (for $F \subseteq E(V_G)$) is added to G.

For a subset $A \subseteq V_G$ of vertices, we let $G - A \subseteq G$ be the subgraph induced by $V_G \setminus A$, that is,

$$G - A = G[V_G \setminus A] ,$$

and, *e.g.*, $G - v$ is obtained from G by removing the vertex v together with the edges that have v as their end.

DEFINITION. A graph $G = (V, E)$ is trivial, if it has only one vertex, *i.e.*, $v_G = 1$; otherwise G is nontrivial.

The graph $G = K_V$ is the complete graph on V, if every two vertices are adjacent: $E = E(V)$. All complete graphs of order n are isomorphic with each other, and they will be denoted by K_n.

3.3.4 Discrete and Regular graph

The complement of G is the graph G on V_G, where $E_G = \{e \in E(V) \mid e \in/ E_G\}$. The complements $G = K_V$of the complete graphs are called discrete graphs. In a discrete graph $E_G = \emptyset$. Clearly, all discrete graphs of order n are isomorphic with each other.

A graph G is said to be regular, if every vertex of G has the same degree. If this degree is equal to r, then G is r-regular or regular of degree.

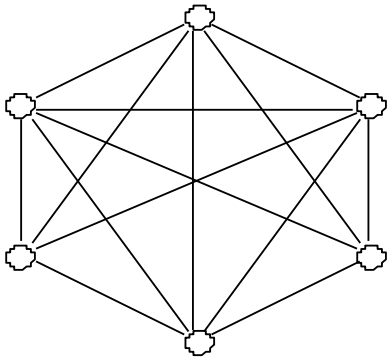

Figure 3.12A regular graph

3.3.5 Planer graph

A planar graph will be a graph that can be drawn in the plane so that no two edges intersect with each other. Such graphs are used, *e.g.*, in the design of electrical (orsimilar) circuits, where one tries to (or has to) avoid crossing the wires or laser beams. Planar graphs come into use also in some parts of mathematics, especially in group theory and topology

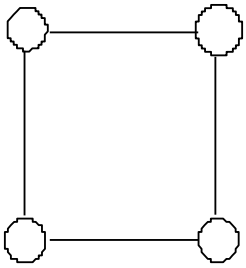

Figure 3.13A planer graph

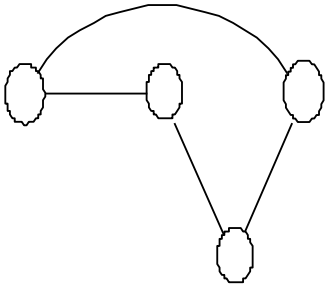

Figure 3.14 A planer graph

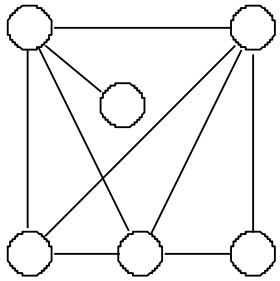

Figure 3.15 Not a planer graph

3.4 Vertex colouring

The vertices of a graph G can also be classified using colourings. These colourings tell that certain vertices have a common property (or that they are similar in some respect), if they share the same colour. In this section, we shall concentrate on proper vertex colourings, where adjacent vertices get different colours.

3.5 The chromatic number

DEFINITION. A k-colouring (or a k-vertex colouring) of a graph G is a mapping α: $V_G \rightarrow [1, k]$. The colouringα is proper, if adjacent vertices obtain a different colour: for all $uv \in G$, we have $\alpha(u)6 = \alpha(v)$. A colour$i \in [1, k]$ is said to be available for a vertex v, if no neighbour of v is coloured by i.

A graph G is k-colourable, if there is a proper k-colouring for G. The (vertex) chromatic number$\chi(G)$ of G is defines as $\chi(G) = \min\{k \mid$ there exists a proper k-colouring of $G\}$.

If $\chi(G) = k$, then G is k-chromatic.

Each proper vertex colouring$\alpha : V_G \rightarrow [1, k]$ provides a partition $\{V1, V2, \ldots, V_k\}$ of the vertex set V_G, where $V_i = \{v \mid \alpha(v) = i\}$.

The graph below, which is often called a wheel (of order 7), is 3-chromatic.

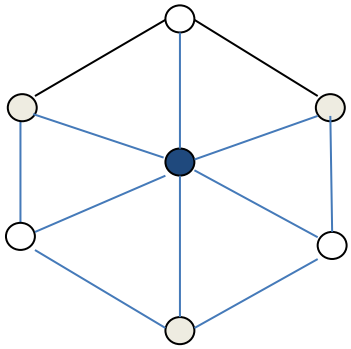

Figure 3.16Wheel

Lemma 3.1:*Letαbe a proper k-colouring of G, and letπbe any permutation of the colours.Then the colouringβ = παis a proper k-colouring of G.*

Proof.Indeed, if $\alpha : V_G \to [1, k]$ is proper, and if $\pi : [1, k] \to [1, k]$ is a bijection, then $uv \in G$ implies that $\alpha(u)6 = \alpha(v)$, and hence also that $\pi\alpha(u)6 = \pi\alpha(v)$.

It follows that $\pi\alpha$ is a proper colouring.

3.6 Special graphs

DEFINITION. A graph $G = (V, E)$ is trivial, if it has only one vertex, *i.e.*,$vG = 1$; otherwise G is nontrivial.

The graph $G = KV$ is the complete graph on V, if every two vertices are adjacent:

$E = E(V)$. All complete graphs of order n are isomorphic with each other, and they will be denoted by Kn.

The complement of G is the graph G on VG, where $EG = \{e \in E(V) \mid e \in EG\}$. The complements $G = KV$ of the complete graphs are called discrete graphs. In a discrete graph $EG = \emptyset$. Clearly, all discrete graphs of order n are isomorphic with each other.

A graph G is said to be regular, if every vertex of G has the same degree. If this degree is equal to r, then G is r-regular or regular of degree r.

Theorem 3.2*Each connected graph has a* spanning tree, *that is, a spanning graph that is a tree.*

Proof. Let $T \subseteq G$ be a maximum order subtree of G (i.e., subgraph that is a tree). If $VT = VG$, there exists an edge$uv \in EG$ such that $u \in T$ and $v \in T$. But then T is not maximal; a contradiction. ⊔

Corollary 3.1.*For each connected graph* G, $\varepsilon\, G \geq vG - 1$. *Moreover, a connectedgraph* G *is a tree if and only if* $\varepsilon\, G = vG - 1$.

Proof. Let$T$ be a spanning tree of G. Then $\varepsilon\, G \geq \varepsilon\, T = vT - 1 = vG - 1$. The second claim is also clear.

Example. In Shannon's switching game a positive player P and a negative player N play on a graph G with two special vertices: a source s and a sink r. P and N alternate turns so that P designates an edge by +, and N by −. Each edge can be designated at most once. It is P's purpose to designate a path $s \stackrel{\star}{\longrightarrow} r$ (that is, to designate all edges in one such path), and N tries to block all paths $s \stackrel{\star}{\longrightarrow} r$ (that is, to designate at least one edge in each such path). We say that game (G, s, r) is

3.7 Computational Complexity of Algorithms

The *complexity* of a problem is related to the resources required to compute a solution as a

function of the size of the problem. The size of a problem is measured by the size of the input N ,and the resources required are usually measured by time (number of steps) and space (maximum amount of memory measured appropriately). *Decision problems* or *yes-or-no questions* are very common. Read HOPCROFT & ULLMAN for classical

complexity theory.

To make computational complexities comparable, we need to agree on some specific mathematical models for algorithms. For example, consider computing with Turing Machines and refer to courses in Theoretical Computer Science and Mathematical Logic. We have *deterministic* and *nondeterministic* version of algorithm models. In the deterministic version, there are no choices to be made. In the nondeterministic version, there is a choice to be made somewhere on the way. For a nondeterministic algorithm, we have to make the following assumptions so that we can actually solve problems:

1. The algorithm terminates at some point no matter how we choose the steps.
2. The algorithm can terminate without yielding a solution.
3. When the algorithm terminates and yields a solution, the solution is correct (it is possible to have more than one solution).
4. For decision problems, if the algorithm fails to give a positive answer (yes), then the answer is interpreted to be negative (no).
5. If the problem is to compute a value, then the nondeterministic algorithm has to give a solution for every input (value of the function).

Nondeterministic algorithms are best treated as *verification procedures* for problems rather than procedures for producing answers.

Computational complexity is considered *asymptotically*, that is for large problems, time or space complexities that differ by constant coefficients are not distinguished because linear acceleration and compression of space are easy to perform in any kind of algorithm model.

Although the choice of an algorithm model has a clear impact on the complexity, it is not an essential characteristic, i.e. it does not change the complexity class. Often, we

use the *big*-O *notation* for complexities. O(f (N)) refers to the class of functions g(N) such that if $N \geq N0$ holds, then $|g(N)| \leq Cf (N)$ holds, where C is a constant.

Without exploring algorithm models any further, we define a couple of important complexity classes. The time complexity class P (*deterministic polynomial time problems*) consists of problems of (input) size N where it takes at most p(N) steps to solve the problem using deterministic algorithms. p(N) is some problem dependent polynomial of N . The time complexity class N P (*nondeterministic polynomial time problems*) consists of problems of size N where it takes at most p(N) steps to solve the problem using nondeterministic algorithms. Once again,
p(N) is some problem dependent polynomial of N .

Time complexity class co−N P (*complements of nondeterministic polynomial time problems*) consists of decision problems whose complements are in N P . (The *complement* of aproblem is obtained by swapping the positive and the negative answer.)

Obviously, $P \subseteq N P$ and (for decision problems) $P \subseteq$ co−N P . Whether or not the inclusion is proper is an open problem, actually quite a famous problem. It is widely believed that both of the inclusions are proper. It is not known if the following holds for decision problems:

$N P =$ co−N P

or

$P = N P \cap$ co−N P

Most researchers believe that they do not hold.

The space complexity class PSPACE (*deterministic polynomial space problems*)

consists

of problems of (input) size N where it takes at most p(N) memory units to solve the problem using deterministic algorithms. p(N) is some problem dependent polynomial of N . The space complexity class N PSPACE (*nondeterministic polynomial space problems*) consists of problems of size N where it takes at most p(N) memory units to solve the problem using non-deterministic algorithms. Once again, p(N) is some problem dependent polynomial of N . It is known that

N P ⊆ PSPACE = N PSPACE ,but it is not known whether the inclusion is proper or not.

An algorithm may include some ideally generated random numbers. The algorithm is then called *probabilistic* or *stochastic*. The corresponding polynomial time complexity class is BPP

(*random polynomial time problems* or *bounded-error probabilistic polynomial time problems*).

Some stochastic algorithms may fail occasionally, that is, they produce no results and terminate prematurely. These algorithms are called *Las Vegas algorithms*. Some stochastic algorithms may also produce wrong answers (ideally with a small probability). These kind of algorithms are called *Monte Carlo algorithms*. Some stochastic algorithms seldom yield exact solutions.

Nevertheless, they give accurate approximate solutions with high probability. These kind of algorithms are called *approximation algorithms*.

The task of an algorithm may be to convert a problem to another. This is known as *reduction*.

If problem A can be reduced to another problem B by using a (deterministic) polynomial time algorithm, then we can get a polynomial time algorithm for problem A

from a polynomial time algorithm for B. A problem is N P -*hard* if every problem in N P can be reduced to it by a polynomial time algorithm. N P -hard problems are N P -*complete* if they are actually in N P .

N P -complete problems are the ”worst kind”. If any problem in N P could be shown to be deterministic polynomial time, then every problem in N P would be in P and P = N P .Over one thousand N P -complete problems are known currently.

The old division of problems into *tractable* and *intractable* means that P problems are tractable and others are not. Because we believe that P = N P in general, N P –complete problems are intractable. In the following, graph algorithms are either in P or they are approximations of some more demanding problems. The size of an input can be for example thenumber of nonzero elements in an incidence matrix, the number of vertices n or the number of edges m or some combination of n and m.

3.7.1 Reachability: Warshall’s Algorithm

We only deal with directed graphs in this section. The results also hold for ”undirected” graphs

if we interpret an edge as a pair of arcs in opposite directions.

Problem. *We are given an adjacency matrix of the digraph* G = (V, E). *We are to construct*

the reachability matrix R = (rij) *of* G, *where*

1 *if* G *has a directed* vi–vj *path* 0 *otherwise.*

(Note that V = {v1, . . . ,vn}.*) In particular, we should note that if*rii = 1, *then*vi *is in a directed circuit.*

Warshall's Algorithm constructs a series of n × n matrices E1, . . . , En where

1. elements of Ei are either zero or one.

2. Ei ≤ Ei+1 (i =

3. E0 is obtained from the adjacency matrix D by replacing the positive elements with ones.

4. En = R.

In this case, the maximizing operation is sometimes called the *Boolean sum*:Let us show that Warshall's Algorithm gives us the desired results. Let Ei denote the value of E after i steps.

Statement. (i) *If there is a directed path from*vs *to*vt *such that apart from*vs *and*vt, *the path*

only includes vertices in the set {v1, . . . , vi}, *then* (Ei)st = 1.

(ii) *If vertex*vs *belongs to a directed circuit whose other vertices are in the set* {v1, . . . , vi},

then (Ei)ss = 1

Proof. We will use induction on i.

Induction Basis: i = 1. (E1)st = 1 if (E0)st = 1, or (E0)s1 = 1 and (E0)1t = 1. We have one of the following cases:

Induction Hypothesis: The statement is true for i < ℓ. (ℓ ≥ 2)

Induction Statement: The statement is true for i = ℓ.

Induction Statement Proof: Let us handle both statements together. The proof for (ii) is given in square brackets. We have two cases: · vℓ belongs to the directed path [resp. directed circuit] but ℓ = s, t [resp. ℓ = s]. Then, we

use the Induction Hypothesis:(Eℓ−1)sℓ = 1

and (Eℓ−1)ℓt = 1 [resp.

(Eℓ−1)sℓ = 1 and

(Eℓ−1)ℓs = 1],

so (Eℓ)st = 1 [resp. (Eℓ)ss = 1].

· vℓ is either vs or vt [resp. vℓ is vs] or it does not belong to the directed path [resp. directed

circuit] at all. Then, by the Induction Hypothesis

(Eℓ−1)st = 1 [resp.

(Eℓ−1)ss = 1],

so (Eℓ)st = 1 [resp. (Eℓ)ss = 1].

In Warshall's Algorithm, the maximizing operation is performed at most n3 times.

3.8 Greedy algorithm for vertex colouring

Greedy algorithm is the most popular algorithm for vertex colouring in graph theory algorithm

Start with a graph G and list colours say 1,2,3,4……..

Step 1

Label the vertices say v1, v2, v3…… in any manner.

Step 2

Identify the uncoloured vertex labeled with the earliest letter in the vertices w1, w2, w3…. Colour it with the first colour in the list not used for any adjacent coloured vertex. Repeat step 2 until all the vertices are coloured, and then stop.

Step 3

A vertex colouring of graph G has been obtained. The number of colours used depends

on the labeling chosen for the vertices in step 1. By using the greedy algorithm we have drawn the graph and coloured the vertices based on the definition vertex colouring. After drawing the graph, adjacent vertices were coloured with different colours.

3.9 Edge colourings

Colourings of edges and vertices of a graph *G* are useful, when one is interested in classifying relations between objects.

There are two sides of colourings. In the general case, a graph *G* with a colouring *a*is given, and we study the properties of this pair *Ga*= (*G*, *a*). This is the situation, *e.g.*, in transportation networkswith bus and train links, where the colour (*buss, train*) of an edge tells the nature of a link.

In the chromatic theory, *G* is first given and then we search for a colouring that the satisfies required properties.One of the important properties of colourings is 'properness'.

In a proper colouring adjacent edges or vertices are coloured differently.

3.9.1 Edge chromatic number

DEFINITION. A *k*-edge colouring*a* :*EG* → [1, *k*] of a graph *G* is an assignment of *k* colours to its edges.We write *Ga*to indicate that *G* has the edge colouring*a*.

A vertex *v* ∈*G* and a colour*i* ∈[1, *k*] are incident with each other, if *a*(*vu*) = *i* for some *vu* ∈*G*. If *v* ∈*G* is not incident with a colour*i*, then *i* is available for *v*.

The colouring*a* is proper, if no two adjacent edges obtain the same colour: *a*(*e*1) 6= *a*(*e*2) for adjacent *e*1 and *e*2.

The edge chromatic number $c'(G)$ of *G* is defined as$c'(G) = \min\{k \mid$ there exists a proper *k*-edge colouring of *G*$\}$.

A *k*-edge colouring*a* can be thought of as a partition {*E*1, *E*2, . . . ,*Ek*} of *EG*, where *Ei*= {*e* | *a*(*e*) = *i*}. Note that it is possible that *Ei*= Æ for some *i*. We adopt asimplified notation

Ga[*i*1, *i*2, . . . , *it*] = *G*[*Ei*1 ∪*Ei*2 ∪ · · · ∪*Eit*] for the subgraph of *G* consisting of those edges that have a colour*i*1, *i*2, . . . , or *it*. That is, the edges having other colours are removed.

Lemma. 3.3*Each colour set Ei in a proper k-edge colouring is a matching.Moreover, for each*

graph G, D(*G*) ≤ *c*′(*G*) ≤ *#G*.

Proof.This is clear.

Example.The three numbers in Lemma 4.1 can be equal. This happens, for instance, when*G* = *K*1,*n* is a star. But often the inequalities are strict.

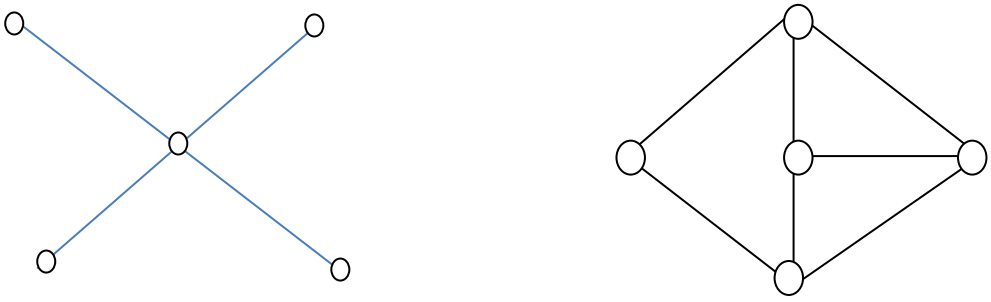

Figure 3.17 A star, and a graph with $\chi'(G) = 4$.

3.9.2 Optimal colourings

We show that for bipartite graphs the lower bound is always optimal: *c*′(*G*) = D(*G*).

Lemma 3.3*Let G be a connected graph that is not an odd cycle. Then there exists a 2-edgecolouring (that need not be proper), in which both colours are incident with each vertex v withdG*(*v*) ≥ 2.

Proof. Assume that *G* is nontrivial; otherwise, the claim is trivial.

(1) Suppose first that *G* is eulerian. If *G* is an even cycle, then a 2-edge colouringexists as required. Otherwise, since now *dG*(*v*) is even for all *v*, *G* has a vertex *v*1 with *dG*(*v*1) ≥ 4. Let *e*1*e*2 . . . *et*be an Euler tour of *G*, where *ei*= *vivi*+1 (and *vt*+1 = *v*1).

Define

$$a(ei) = \begin{cases} 1, if\ i\ is\ odd \\ 2, if\ i\ is\ even \end{cases}$$

Hence the ends of the edges *ei*for $i \in [2, t-1]$ are incident with both colours. Allvertices are among these ends. The condition *dG*(*v*1) ≥ 4 guarantees this for *v*1.Hencethe claim holds in the eulerian case.

Suppose then that *G* is not eulerian. We define a new graph *G*0 by adding avertex *v*0 to *G* and connecting *v*0 to each *v* ∈*G* of odd degree.In *G*0 every vertex has even degree including *v*0 (bythe handshaking lemma), and hence *G*0 is eulerian.

Let*e*0*e*1 . . . *et*be an eulerian tour of *G*0, where *ei*= *vivi*+1.

By the previous case, there is a required colouring*a* of*G*0 as above. Now, *a* restricted to *EG* is a colouring of *G*as required by the claim, since each vertex *vi* with odddegree *dG*(*vi*) ≥ 3 is entered and departed at least oncein the tour by an edge of the original graph *G*: *ei*−1*ei*.

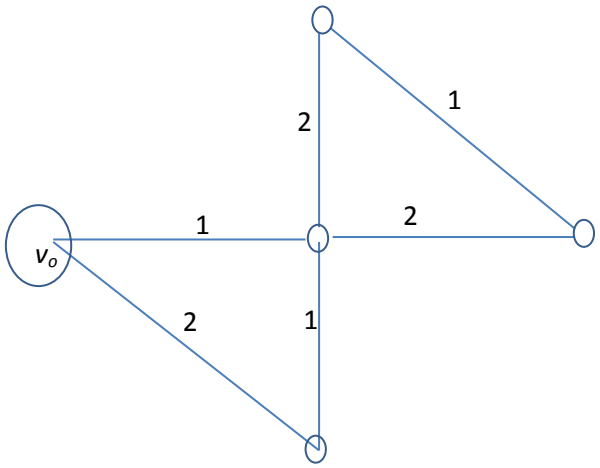

Figure 3.18 Edge

DEFINITION. For a *k*-edge colouring*a* of *G*, let *ca*(*v*) = |{*i* | *v* is incident with *i* ∈[1, *k*]}| .

A *k*-edge colouring*b* is an improvement of *a*, if

$$\sum_{v\in G} C\beta(v) > \sum_{v\in G} c\alpha(v)$$

Also, *a* isoptimal, if it cannot be improved.

Notice that we always have *ca*(*v*) ≤ *dG*(*v*), and if *a* is proper, then *ca*(*v*) = *dG*(*v*),

and in this case *a* is optimal. Thus an improvement of a colouring is a change towards

a proper colouring. Note also that a graph *G* always has an optimal *k*-edge colouring,

but it need not have any proper *k*-edge colourings.

The next lemma is obvious.

Lemma 3.4.*An edge colouring a of G is proper if and only if ca*(*v*) = *dG*(*v*) *for all vertices* *v* ∈*G*.

Lemma. 3.5*Let a be an optimal k-edge colouring of G, and let v ∈G. Suppose that the colour i is available for v, and the colour j is incident with v at least twice. Then the connected component H of Ga*[*i, j*] *that contains v, is an odd cycle.*

Proof.Suppose the connected component *H* is not an odd cycle. By Lemma 4.2, *H* has a 2-edge colouring*g*: *EH* → {*i, j*}, in which both *i* and *j* are incident with each vertex*x* with *dH*(*x*) ≥ 2. (We have renamed the colours 1 and 2 to *i* and *j*.)We obtain arecolouring*b* of *G* as follows:

$$b(e) = \begin{cases} \gamma(e), & if\ e \in H, \\ \alpha(e), & if\ e \notin H. \end{cases}$$

Since *dH*(*v*) ≥ 2 (by the assumption on the colour*j*) and in *b* both colours *i* and *j* are now incident with *v*, *cb*(*v*) = *ca*(*v*) + 1. Furthermore, by the construction of *b*,we have *cb*(*u*) ≥ *ca*(*u*) for all *u* 6= *v*. Therefore å*u*∈*Gcb*(*u*) >å*u*∈*Gca*(*u*), which contradicts the optimality of *a*. Hence *H* is an odd cycle. ⊓⊔

Theorem 3.3; *If G is bipartite, then c′*(*G*) = D(*G*).

Proof. Let *a* be an optimal D-edge colouring of a bipartite *G*, where D = D(*G*). If there were a *v* ∈*G* with *ca*(*v*) <*dG*(*v*), then by Lemma 4.4, *G* would contain an odd cycle. But a bipartite graph does not contain such cycles. Therefore, for all vertices *v*, *ca*(*v*) = *dG*(*v*). By Lemma 4.3, *a* is a proper colouring, and D = *c′*(*G*) as required.

Vizing's theorem

In generalwe can have *c′*(*G*) > D(*G*) as one of our examples did show. The following important theorem, due to VIZING, shows that the edge chromatic number of a graph *G* misses D(*G*) by at most one colour.

Theorem. 3.4 *For any graph G,* D(*G*) ≤*x′*(*G*) ≤ D(*G*) + 1.

Proof. Let D = D(G). We need only to show that $c'(G) \leq D + 1$. Suppose on the contrary that $c'(G) > D + 1$, and let a be an optimal (D + 1)-edge colouring of G.

We have (trivially) $dG(u) < D + 1 < c'(G)$ for all $u \in G$, and so

Claim 1.*For each $u \in G$, there exists an available colour $b(u)$ for u.*

Moreover, by the counter hypothesis, a is not a proper colouring, and hence there exists a $v \in G$ with $ca(v) < dG(v)$, and hence a colour $i1$ that is incident with v at least twice, say $a(vu1) = i1 = a(vx)$. (4.1)

Claim 2.*There is a sequence of vertices $u1, u2, \ldots$ such that*

$a(vuj) = ij$ and $ij+1 = b(uj)$.

Indeed, let $u1$ be as in (4.1). Assume we have already found the vertices $u1, \ldots, uj$, With $j \geq 1$, such that the claim holds for these. Suppose, contrary to the claim, that v is not incident with $b(uj) = ij+1$.

We can recolour the edges $vu\ell$ by $i\ell+1$ for $\ell \in [1, j]$, and obtain in this way an improvement of a. Here v gains a new colour $ij+1$. Also, each $u\ell$ gains a new colour $i\ell+1$ (and may loose the colour $i\ell$). Therefore, for each $u\ell$ either its number of colours remains the same or it increases by one. This contradicts the optimality of a, and proof Claim 2.

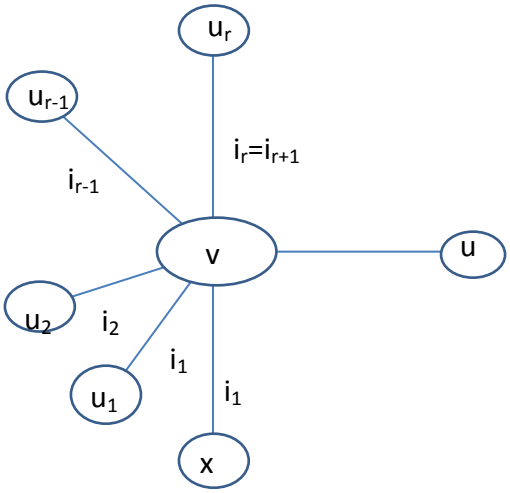

Figure 3.19 Graph of claim 1

Let *t* be the smallest index such that for some *r* <*t*,

it+1 = *ir*. Such an index *t* exists, because *dG*(*v*) is finite.

Let *t* be the smallest index such that for some *r* <*t*, *it*+1 = *ir*. Such an index *t* exists, because *dG*(*v*) is finite.

Let then the colouring*g* be obtained from *b* by recolouring

the edges *vu j*by *ij*+1 for $r \leq j \leq t$. Now,

*vut*is recoloured by *ir*= *it*+1.

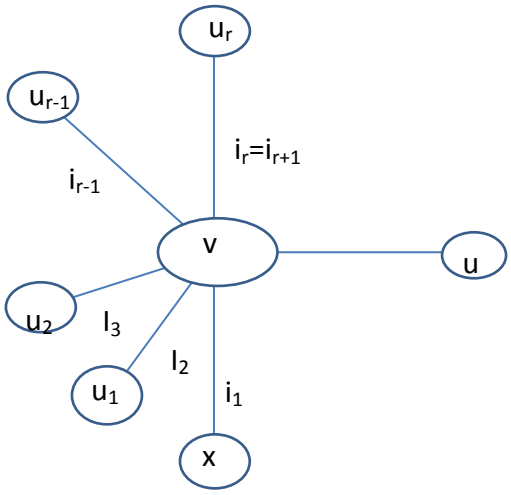

Figure 3.20 Graph of Claim 3

Claim 4.*g is an optimal* (D +1)-*edge colouring of G.*

Indeed, the fact *ir*= *it*+1 ensures that *ir*is a new colour incident with *ut*, and thus that $cg(ut) \geq cb(ut)$.

For all other vertices, $cg(u) \geq cb(u)$ follows as for *b*.

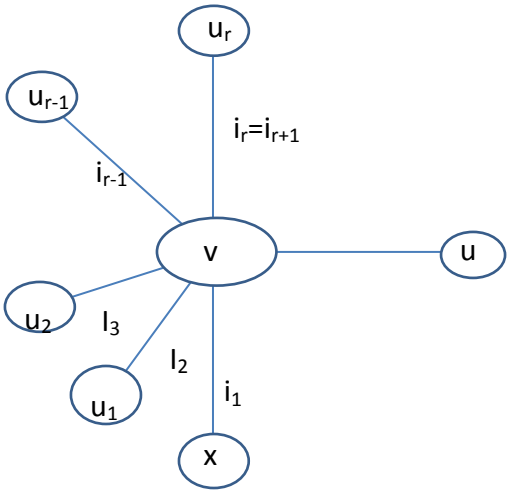

Figure 3.21 Graph of claim 4

By Claim 1, there is a colour$i0 = b(v)$ that is available for v. By Lemma 4.4, the connected components $H1$ of $Gb[i0, ir]$ and $H2$ of $Gg[i0, ir]$ containing the vertex v are cycles, that is, $H1$ is a cycle $(vur-1)$ $P1(urv)$ and $H2$ is a cycle $(vur-1)P2(utv)$, where both$P1 : ur-1 \star\longrightarrow ur$ and $P2 : ur-1 \star\longrightarrow u$ tare paths. However, the edges of $P1$ and $P2$ have the same colours with respect to b and g (either $i0$ or ir). This is not possible, since $P1$ ends in ur while $P2$ ends in a different vertex ut. This contradiction proves the theorem.

Example .We show that $c'(G) = 4$ for the Petersen graph. Indeed, by Vizing' theorem, $c'(G) = 3$ or 4. Suppose 3 colours suffice. Let C: $v1 \longrightarrow \ldots \longrightarrow v5 \longrightarrow v1$ be the outer cycle and C' :$u1 \longrightarrow \ldots \longrightarrow u5 \longrightarrow u1$ the inner cycle of G such that $viui \in EG$ for all i.

Observe that every vertex is adjacent to all colours 1, 2, 3. Now C uses one colour (say 1) once and the other two twice. This can be done uniquely (up to permutations):

However, this means that 1 cannot be a colour of any edge in C'. Since C' needs three colours, the claim follows.

Edge Colouring Problem.Vizing'stheorem (nor its present proof) does not offer any characterization for the graphs, for which $c'(G) = D(G) + 1$. In fact, it is one of the famous open problems of graph theory to find such a characterization. The answer is known (only) for some special classes of graphs. By HOLYER (1981), the problem whether $c'(G)$ is $D(G)$ or $D(G) + 1$ is NP-complete.

The proof of Vizing's theorem can be used to obtain a proper colouring of G with at most $D(G)+1$ colours, when the word 'optimal' is forgotten: colour first the edges as well as you can (if nothing better, then arbitrarily in two colours), and use the proof iteratively to improve the colouring until no improvement is possible – then the proof says that the result is a proper colouring.

3.10 Critical graphs

DEFINITION. A k-chromatic graph G is said to be k-critical,

If$c(H) < k$ for all $H \subseteq G$

With $H \neq G$.

In a critical graph an elimination of any edge and of any vertex will reduce the chromatic number: $c(G-e) < c(G)$ and $c(G-v) < c(G)$ for $e \in G$ and $v \in G$. Each $K_n$is n-critical, since in $K_n-(uv)$ the vertices u and v can gain the same colour.

Example .The graph $K2 = P2$ is the only 2-critical graph. The 3-critical graphs are exactly the odd cycles C_{2n+1} for $n \geq 1$, since a 3-chromatic G is not bipartite, and thus must have a cycle of odd length.

Theorem 3.5.*If G is k-critical for $k \geq 2$, then it is connected, and $d(G) \geq k - 1$.*

Proof. Note that for any graph G with the connected components $G1, G2, \ldots, Gm$,

$c(G) = \max\{c(Gi) \mid i \in [1,m]\}$. Connectivity claim follows from this observation.

Let then G be k-critical, but $d(G) = dG(v) \leq k - 2$ for $v \in G$. Since G is critical, there is a proper $(k - 1)$-colouring of $G-v$. Now v is adjacent to only $d(G) < k - 1$ vertices. But there are k colours, and hence there is an available colour i for v. If we recolour v by i, then a proper $(k-1)$-colouring is obtained for G; a contradiction. ⊓⊔

The case (iii) of the next theorem is due to Szekeres And Wilf (1968).

Theorem 3.6;.*Let G be any graph with k = c(G).*

(i) G has a k-critical subgraph H.

(ii) G has at least k vertices of degree $\geq k - 1$.

(iii) $k \leq 1 + \max_{H \subseteq G} d(H)$.

Proof. For (i), we observe that a k-critical subgraph $H \subseteq G$ is obtained by removing vertices and edges from G as long as the chromatic number remains k.

For (ii), let $H \subseteq G$ be k-critical. By Theorem 4.10, $dH(v) \geq k - 1$ for every $v \in H$.

Of course, also $dG(v) \geq k - 1$ for every $v \in H$. The claim follows, because, clearly,

Every k-critical graph H must have at least k vertices.

For (iii), let $H \subseteq G$ be k-critical. By Theorem 4.10, $c(G) - 1 \leq d(H)$, which proves this claim.

Lemma 3.6.*Let v be a cut vertex of a connected graph G, and let Ai, for i* $\in [1,m]$*, be the connected components of G−v. Denote Gi= G[Ai* $\cup$ *{v}]. Then* $c(G) = \max\{c(Gi) \mid i \in [1,m]\}$. *In particular, a critical graph does not have cut vertices.*

Proof; Suppose each Gihas a proper k-colouring ai. By Lemma 4.5, we may take $ai(v) = 1$ for all i. These k-colourings give a k-colouring of G. ⊓⊔

Brooks' theorem

For *edge* colourings we have Vizing's theorem, but no such strong results are known for vertex colouring.

Lemma 3.7.*For all graphs G,* $c(G) \leq D(G) + 1$. *In fact, there exists a proper colouring* a: $VG \rightarrow [1, D(G) + 1]$ *such that* $a(v) \leq dG(v) + 1$ *for all vertices* $v \in G$.

Proof. We use greedy colouring to prove the claim. Let $VG = \{v1, \ldots, vn\}$ be ordered in some way, and define a: $VG \rightarrow$ N inductively as follows: $a(v1) = 1$, and

$a(vi) = \min\{j \mid a(vt) 6= j$ for all $t < i$ with $vivt \in G\}$.

Then a is proper, and $a(vi) \leq dG(vi) + 1$ for all i. The claim follows from this. ⊓⊔

Although, we always have $c(G) \leq D(G) + 1$, the chromatic number $c(G)$ usually takes much lower values – as seen in the bipartite case. Moreover, the maximum value $D(G) + 1$ is obtained only in two special cases as was shown by Brooks in 1941.

The next proof of Brook's theorem is by LOVÁSZ (1975) as modified by BRYANT (1996).

Lemma 3.8.*Let G be a* 2*-connected graph. Then the following are equivalent:*

(i) G is a complete graph or a cycle.

(ii) For all $u, v \in G$, *if* $uv/\in G$, *then* $\{u, v\}$ *is a separating set.*

(iii) For all $u, v \in G$, *if* $dG(u, v) = 2$, *then* $\{u, v\}$ *is a separating set.*

Since v is not a cut vertex, there exists a $y \in U$ such that $uy \in G$. Hence $dG(x, y) = 2$, and by (iii), $\{x, y\}$ is a separating set. Thus $VG = W1 \cup \{x, y\} \cup U1$, where all paths from $W1$ to $U1$ pass through x or y. Assume that $w \in W1$, and hence that also $u, v \in W1$. (Since $uw, vw \in VG - \{x, y\}$).

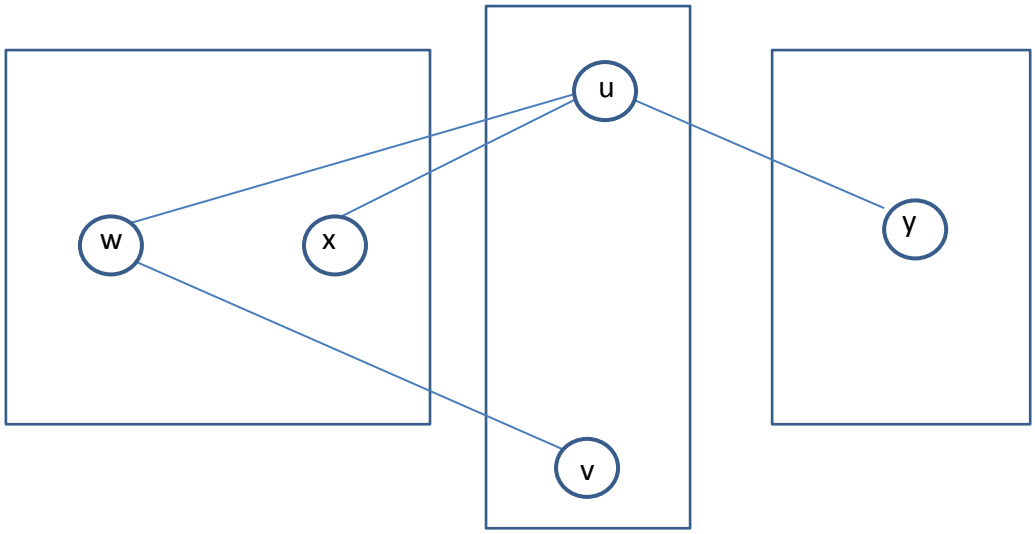

Figure 3.22 A connected graph

There exists a vertex $z \in U1$. Note that $U1 \subseteq W \cup U$. If $z \in W$ (or $z \in U$, respectively), then all paths from z to u must pass through x (or y, respectively), and x (or y, respectively) would be a cut vertex of G. This contradiction, proves the claim. ⊓⊔

Theorem 3.7.*Let G be connected.*

Then $c(G) = D(G) + 1$ *if and only*

if either G is an odd cycle or a complete graph.

Proof.(⇐=) Indeed, $c(C2k+1) = 3$, $D(C2k+1) = 2$, and $c(Kn) = n$, $D(Kn) = n - 1$.

(=⇒) Assume that $k = c(G)$.We may suppose that G is k-critical. Indeed, assume the claim holds for k-critical graphs. Let $k = D(G) + 1$, and let $H \subset G$ be a k-critical proper subgraph. Since $c(H) = k = D(G) + 1 > D(H)$, we must have

$c(H) = D(H) + 1$, and thus H is a complete graph or an odd cycle. Now G is connected, and therefore there exists an edge $uv \in G$ with $u \in H$ and $v \notin H$.

But then $dG(u) > dH(u)$, and $D(G) > D(H)$, since $H = Kn$or $H = Cn$.

Let then G be any k-critical graph for $k \geq 2$. By Lemma 4.6, it is 2-connected. If G is an *even* cycle, then $k = 2 = D(G)$. Suppose now that G is neither complete nor acycle (odd or even).We show that $c(G) \leq D(G)$.

3.11 Colouring planar graphs

A graph is embeddable on a surface Σ if its vertices can be mapped onto distincts points of Σ and its edges onto simple curves of Σ joining the points onto which its end vertices are mapped, so that two edge curves do not intersect except in their common extremity. A face of an embedding ˜G of a graph G is a component of Σ \ ˜G. We denote by F(˜G) the set of faces of ˜G. A graph is planar if it can be embedded in the plane.

Let ˜G be an embedding of a planar graph G. Its numbers of vertices, faces and edges are related by Euler's Formula:

|V(˜G)|+|F(˜G)|−|E(˜G)| = 1+comp(G) where comp(G) is the number of connected components of G.

Proof. We prove of Euler's Formula by induction on the number of edges of G.

If G has no edges, then every vertex is a connected component and the graph has a unique face, the outer one.

Suppose now that G is a planar graph on at least one edge and that the result holds for planar graphs with less edges. Let e be an edge of G. Then two cases may occur.

Assume first that e is a bridge (i.e. G \ e has one more component than G). Then e is incident to a unique face in G. So G\ e has as many faces as G. By the induction hypothesis, |V(˜G \ e)|+|F(˜G \ e)|−|E(˜G \ e)| = 1+comp(G\ e). So |V(˜G)|+|F(˜G)|−(|E(˜G)|−1) = 1+comp(G)+1.

Assume now that e is not a bridge. Then G \ e has the same number of components as

G. Then e is incident to two faces in G. Removing e transform these two faces into a single one (their union). So G \ e has as many faces as G. By the induction hypothesis, $|V(\tilde{G} \setminus e)|+|F(\tilde{G} \setminus e)|-|E(\tilde{G} \setminus e)| = 2-\text{comp}(G\setminus e)$. So $|V(\tilde{G})|+(|F(\tilde{G})|)-1-(|E(\tilde{G})|-1) = 1+\text{comp}(G)$.

Corollary. 3.1 If G is a planar graph, then

$|E(G)|\leq 3|V(G)|-6$.

Proof. Let $\tilde{G}$ be an embedding of G. Every face de $\tilde{G}$ contains at least three edges and every edge is in at most two faces. Hence, considering the number N of edge-face incidences, we have $2|E(G)|\geq 3|F(\tilde{G})|$. Putting this inequality into Euler's Formula we obtain $|V(G)|+ 2|E(G)|/3\geq|E(G)|+2$ so $3|V(G)|-6\geq|E(G)|$.

Corollary. 3.2 Every planar graph has a vertex of degree at most 5.

Proof. Let G be a planar graph. By Corollary 8.16, $\Sigma\{d(v) : v \in G\} = 2|E(G)|\leq 6|V(G)|-12$. The minimum degree of G is less or equal to the average degree which is equal to $6|V(G)|-12\ |V(G)|< 6$. Hence there is a vertex of degree less than 6.

Corollary.3.3 Every planar graph is 6-colourable.

Proof. Let G be a planar graph. Every subgraph of G is planar and so has minimum degree at most 5 by Corollary 8.17. Hence G is 5-degenerate. Thus, by Proposition 8.7, $\chi(G) \leq 6$.

Theorem 3.8 Every planar graph is 5-colourable.

Proof. By induction on the number of vertices of G, the result holding trivially if G has one vertex. By Corollary 8.17, there is a vertex v of degree at most 5 in G By the induction hypothesis, the graph $G-v$ is 5-colourable. Let c be a proper 5-colouring of $G-v$. From c, we will construct a proper 5-colouring of G.

Assume first, that one of the colours, say i, is assigned to no neighbours of v. Then one can extend c by setting $c(v) = i$. (Note that this is the case if $d(v) \leq 4$.)

Hence we may assume that that v has five neighbours coloured differently. Let v1, v2, v3, v4 and v5 be these neighbours in counter-clockwise order around v. Free to permute the colours, we may suppose that $c(vi) = i$ for all $1 \leq i \leq 5$.

Let C1,3 be the component of v1 in the subgraph G induced by the vertices coloured 1 or 3. If v3 is not in C1,3, then interchanging the colours 1 and 3 in C1,3 and colouring v with 1, we obtain a proper 5-colouring of G. If v3 ∈C1,3, then there exists a path P linking v1 to v3 in

C1. Together with vv1 and vv3 it forms cycle C which separates v2 and v4. Thus the component C2,4 of v2 in the subgraph of G induced by the vertices coloured 2 and 4 does not contain v4, otherwise an edge of the path joining v2 to v4 inC2,4 would cross an edge of C. Hence on can interchange the colours 2 and 4 in C2,4 and colour v with 2 to obtain a proper 5-colouring of G.

Theorem 3.7 is not best possible: the celebrated Four Colour Theorem by Appel and Haken [1, 2, 3] states that every planar graph is 4-colourable. A simpler proof was presented by Robertson, Sanders, Seymour and Thomas [22, 23]. However it still uses complicated reductions to a huge number of configurations (more than six hundreds) which need to be solved by computer assistance.

3.12 Graph colouring, an integer linear program.

An integer program is a discrete optimization of the form

Minimize (or Maximize) $F(x_1, x_2, \ldots, x_n)$

Subject to a set of m equality constraints

$g_i (x_1, x_2, \ldots, x_n) = b_1$

$g_2 (x_1, x_2, \ldots, x_n) = b_2$

‘ ‘ ‘

‘ ‘ ‘

‘ ‘ ‘

$g_m (x_1, x_2, \ldots , x_n) = b_m$

And K inequality constraints

$h_1(x_1, x_2, \ldots , x_n) \leq r_1$

$h_2 (x_1, x_2, \ldots , x_n) \geq r_2$

‘ ‘ ‘

‘ ‘ ‘

‘ ‘ ‘

$h_k (x_1, x_2, \ldots , x_n) \leq r_k$

In addition, the value of the decision variable $x_1, x_2, \ldots\ldots , x_n$ must be integers. No fractional values of $x_1, x_2, \ldots .. , xn$ are permitted. In an integer program, the objective function F and the constraint function $g_1, g_2, \ldots\ldots, g_m$ and $h_1, h_2, \ldots..,$ hk may be linear or non linear. If we restrict the objective function F and the constraint function $g_1, g_2, \ldots.., g_m$ and $h_1, h_2, \ldots..$ hk to be linear, then we have an integer linear program.

Integer linear programming formalities are much more easily handled in computation than integer (non linear) programming formalities. We seek to an integer programming (IP) formalities of graph colouring which contains non linear constraint. In this formulation, G is the graph we wish to properly colour. The number of vertices in G is denoted by n and the number of edges in G is denoted by e. We use k to represent the

number of colour we wish to use to properly colour G. The value of n and k are known constraints which serve as parameters in the model. The formulation addresses the following two questions;

Given a graph G and a number of colour K, is there a proper K – colouring of G?

If such a K – colouring of G exist, what is an example of such colouring

Integer programming formulation of Graph colouring:

Minimise objective function

Subject to:

1. $1 \leq X_i \leq k; i = 1, 2, \ldots, n$
2. $1 \leq \ Xi - Xj$ for each edge $ij \epsilon\ E(G)$
3. X_1, X_2, ..., X_n are integer – valued.

The above formulation contains n integer variables x, x_2, …,x_n , each representing one of the n vertices of G. A feasible solution {x1, x2, …,xn} gives a proper K – colouring of G, if indeed one does exist. Three constraints define precisely what it means for a graph to exhibit a proper k-colouring . Constraint 1 maintains that each variable xi must receive a value between I and k. That is to say, each vertex of G must be coloured using a coloured using a colour from I to K. Constraint 2 further illustrates the definition of a proper colouring of G. For each edge of G, the vertices corresponding to that edge must be coloured using different colours. Finally, constraint 3 requires that the values of the variable be x_1, x_2, …, xn integers.

Obviously this condition must hold since our k colours are numbered as integers 1, 2, …, k. If we obtain a solution {x_1, x_2, …, x_n} that satisfies all of the above constraints for a particular graph G given k, then G has a proper k-colouring and {x_1, x_2, …, x_n} is such a colouring.

CHAPTER 4

DATA COLLECTION AND ANALYSIS

In this chapter the data used for the generation of the nurse schedule was considered and how graph colouring was used to group the various types of nurses, taking into consideration the hospital hard rules and the soft nurses' preferences.

4.1 NURSES DATA

There are fifteen nurses in the female and paediatrics ward. To solve the nurse schedule problem, all the fifteen nurses were considered. The types of nurses in the ward are Staff Nurse (SN), Principal Enrolled Nurse (PEN), Rotation Nurse (RN), Enrolled Nurse (EN), Senior Ward Assistant (SWA) and Health Extension Workers (HEW)

They were named as SN1, SN2, SN3, SN4, PEN, RN1, RN2,EN1, EN2, SWA, HEW1, HEW2, HEW3, HEW4, HEW5. The table below show the names of the nurses

Table 4.1 Nurses data

Names of nurses	Rank	Ward
Marian AsankomanOwoo	SN1	W1
Mariama Ahmed	SN2	W1
Yvonne O. Berko	SN3	W1
VeneciaBoateng	SN4	W1
Cecilia Millicent Aman	PEN	W1
Isaac AntwiMireku	RN1	W1
QuinzyAmoh	RN2	W1
Constance Koramah	EN1	W1
Monica Bady	EN2	W1
Cecilia Appiah	SWA	W1
Obed Atta Yeboah	HEW1	W1
Doris Owusu	HEW2	W1
Janet Appiah	HEW3	W1
FaustinaAcheampong	HEW4	W1
Sandra A. Garbrah	HEW5	W1

4.2 APPLYING GRAPH THEORY FOR THE NURSE SCHEDULE PROBLEM

To solve most scheduling problem Graph Theory was used. In scheduling problem Graph coloring in Graph theory can use to avoid conflicts and to allocate resources effectively.

In this problem when considering the constraints, nurses with different skills levels can be in same shift. But only the junior or senior nurses can't be in same shift. And also

there are some nurses, who don't like to work together. So they must put in to different groups. Sometimes hospital management also wants to put some nurses in to different shifts to create high quality roster. Also when creating the roster patients requirements also have to be considered. They also may ask to for some nurses to be put in to different shifts.

Considering the different skills levels the nurses were put into groups. The Staff nurses were put in one group, group A and Rotation nurses were also put in another group, group C. Like that by considering nurse requirements, hospital management requirements, patients requirements, etc; nurses were grouped as following table 4.1 given below.

Table 4.2 Group of conflicting nurses.

Group	Nurses
A	SN1, SN2, SN4
B	SN4, EN2, EN3
C	RN1, RN2
D	EN2, HEW3
E	HEW2,HEW3,HEW4
F	RN1, EN1
G	EN1, EN2

By using above data a matrix was created for the nurses. It is a 15 X 15 matrix. In that matrix nurse's names take as i and j .Then ij-th entry is put according to the above table 4.1, If any two nurses are in same group, then ij-th entry is put as '1' otherwise put it as

‘0’. Following Figure 5.1 shows the relevant matrix.

Table 4.3 Adjacency matrix (conflict matrix) of nurse

	SN1	SN2	SN3	SN4	PEN	RN1	RN2	EN1	EN 2	SWA	HEW1	HEW2	HEW3	HEW4	HEW5
SN1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
SN2	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
SN3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SN4	1	1	0	0	0	0	0	1	1	0	0	0	0	0	0
PEM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RN1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
RN2	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
EN1	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0
EN2	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0
SWA	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
HEW1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0
HEW2	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0
HEW3	0	0	0	0	0	0	0	0	1	0	1	1	0	1	0
HEW4	0	0	0	0	0	0	0	0	0	0	1	1	1	0	
HEW5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Using above matrix, a graph was constructed by taking nurses as vertices. If two nurses are in same group according to the above table 4.1 corresponding vertices are combine using edges.

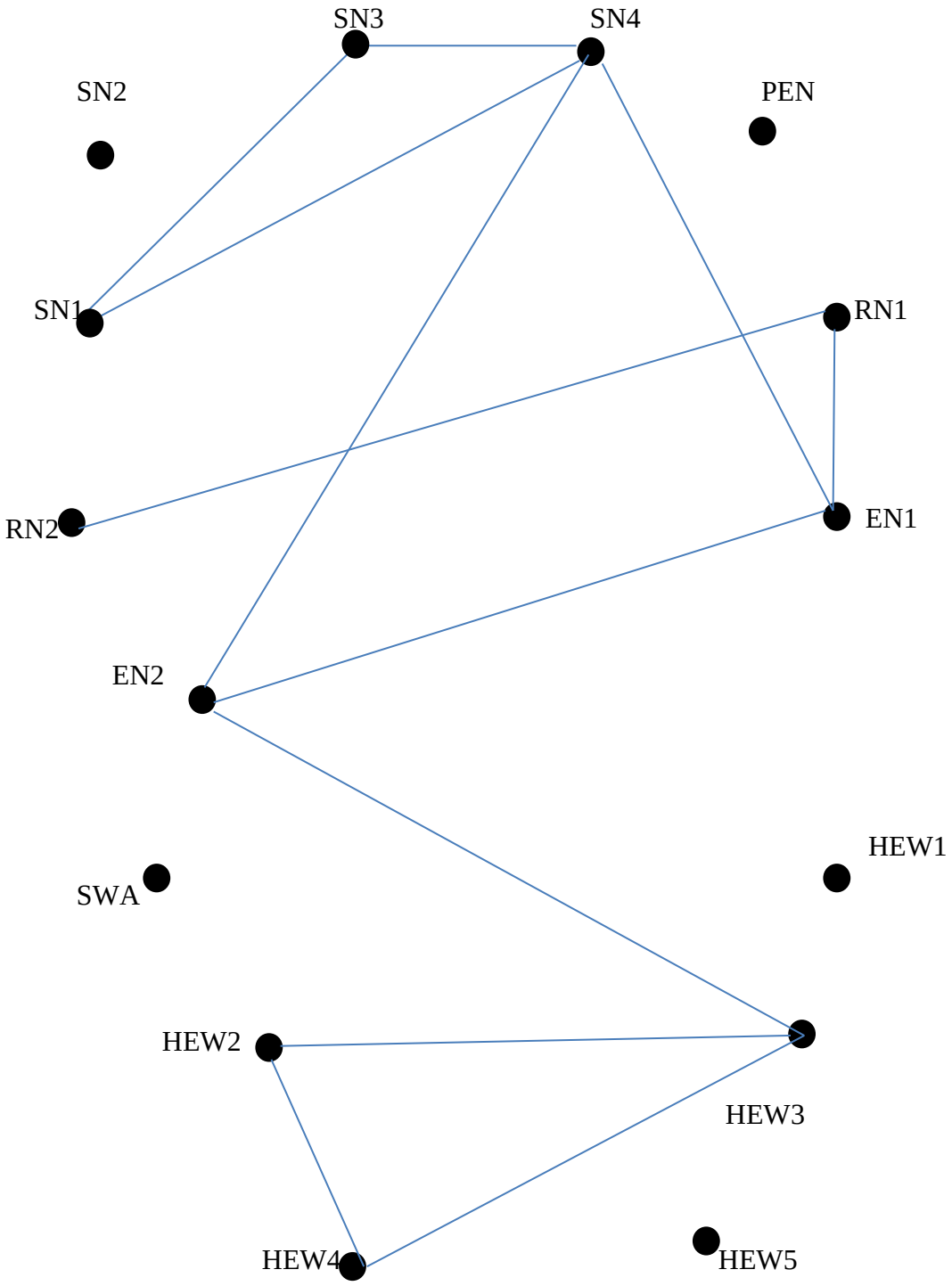

Figure 4.1 A conflicting graph of nurse

As the next part vertices in graph were colored. If two vertices are adjacent, they were coloured using two different colors. Below is figure 4.2 showing the resulting graph after colouring.

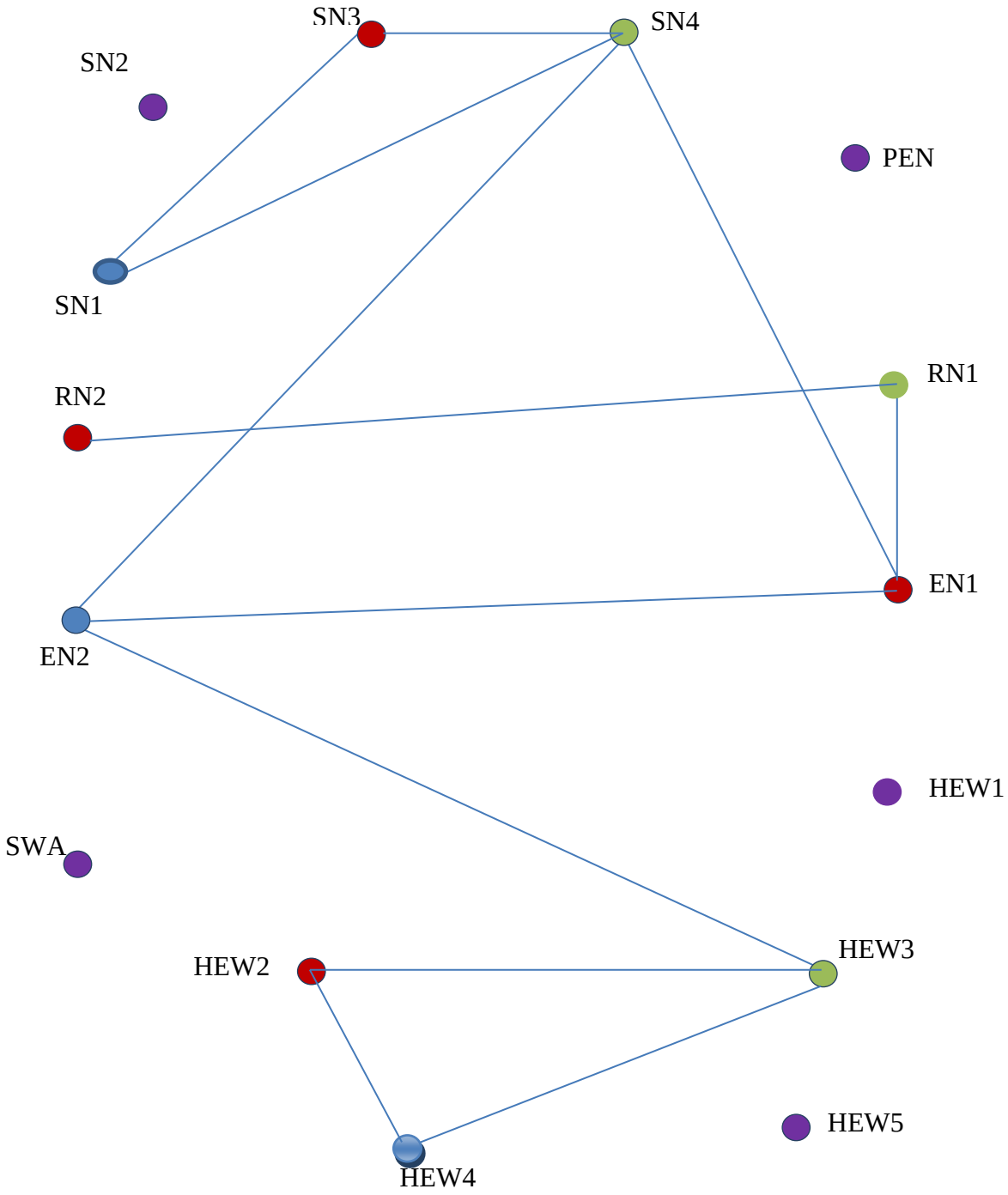

Figure 4.5 A coloured conflict graph of nurses

Using vertices colours again nurses were grouped. In grouping, nurses with same colour were put in to one group. Normally in graph colouring if any vertices were not adjacent with other vertices, as an example here SN2 and PEN were to be coloured using existing colour. That is using another vertices colour in the graph that has already been coloured. But in this study those are also coloured using a new colour and were put into a different group called bench group (B). Following table 4.3 shows the group details after applying the graph colouring techniques.

Table 4.4 Nurse's group after applying graph colouring

Group	**Nurses**
G1	SN3, RN2, EN1, HEW2
G2	SN4, RN1, HEW3
G3	SN1, EN2, HEW4
Bench (B)	SN2, PEN, SWA, HEW1

According to the above table 4.2 Nurses in G1, G2 and G3 could be in same shift only if they are in same group. This means SN3, RN2, EN1 and HEW2 can be in same shift and can work together because they are all members of same group G1. But SN4 and EN1 can't work together because they are in different groups G1 and G2. To be presented in the same shift, nurses must be in same group otherwise some constraints may be violated. But there is a special group; Bench group or group B. Members in group B has special facility. They can work together with any group. There is no any restriction. So members in group B can work in any shift with anyone.

4.3 A NURSE SCHEDULE

Below table 4.3is a part of nurse-roster that shows the continuation of the same shift. Columns are the days of the month and the rows are the names of the nurses. In this chart in front of the name head nurse has specify the ranks of the nurses whether the individual is a senior or junior nurse. ‘1’ is denoting morning shift, '2’ for afternoon shift, ‘3’ for night shift and ‘4’ for off day.

The allocation below follows a format, which takes in into consideration some of the hard constraints of the hospital and soft nurses preferences. Some of which are:

I. Only the principal nurse is entitled to Holiday off duty.

II. Principal enrolled nurse is scheduled for only morning shifts and has day off duties on both Saturday and Sundays

III. Every nurse is entitled to at least one day-off in the week where there is no night shift

IV. Every nurse is entitled to four days off after three consecutive night shift

V. Every night shift is taken continuously for three consecutive days

VI. The minimum number of nurses for morning shift should not be less than three. That is $A \geq 3$

VII The number of nurses for both Afternoon and Night should be at least two. That is

$A \geq 2$ and $N \geq 2$

VIII. Morning and afternoon shift alternate for the other remaining days.

IX. Every nurse is entitled to only one shift a day.

CHAPTER 5

RESULTS, CONCLUSION AND RECOMMENDATION

This chapter deals with the results, conclusion and recommendation.

5.1 RESULTS

The result of the study was divided into two major parts. In the first the nurses were divided into shift groups using graph theory. This was done after colouring the conflict graph thereby removing the various conflicting nurses. In the graph, vertices of the same colour were grouped as one. This indicates that, the vertices which were used to represent the nurses with the same colour can be together without creating any conflict.

The shift groups were then used to create the nurses scheduling timetable in the second part.

The following Table 5.1 shows the results of a portion of the scheduling table generated. In this table the first row represent the days of the month and in the first column the names of the nurse were listed. In the schedule ‘A’ represent morning shift, ‘M’ represent afternoon shift , ‘N’ represent night shift and ‘M’ represent off day. In second table 5.2 also show how text characters were also used to represent the various schedules. In this table ‘M’ represent morning shift, ‘A’ represent afternoon shift, ‘N’ represent night shift and ‘OFF’ represent off day.

Table 5.1 Schedule table using matlab

JUASO DISTRICT HOSPITAL – ASANTE AKIM SOUTH

NURSES DUTY ROSTER

WARD: FEMALE / PAEDIATRICS

Name		1	2	3	4	5	6	7	8	9	10	11
1	Marian Asankoman Owoo	3	3	3	3	4	4	4	1	2	1	2
2	Mariama Ahmed	1	2	1	4	3	3	3	3	4	4	4
3	Yvonne O. Berko	1	2	1	4	3	3	3	3	4	4	4
4	Venecia Boateng	3	3	3	3	4	4	4	1	2	1	2
5	Isaac Antwi Mireku	2	1	2	1	2	1	2	4	3	3	3
6	QuinzyAmoh	3	3	3	3	4	4	4	1	2	1	2
7	Constance Koramah	2	1	2	1	2	1	2	4	3	3	3
8	Monica Bady	4	4	4	2	1	2	1	2	1	2	1
9	Cecilia Appiah	2	1	2	1	2	1	2	4	3	3	3
10	Obed Atta Yeboah	1	2	1	4	3	3	3	3	4	4	4
11	Doris Owusu	4	4	4	2	1	2	1	2	1	2	1
12	Janet Appiah	3	3	3	3	4	4	4	1	2	1	2
13	Faustina Acheampong	4	4	4	2	1	2	1	2	1	2	1
14	Sandra A. Garbrah	4	4	4	2	1	2	1	2	1	2	1

Table 5.2 Schedule table

JUASO DISTRICT HOSPITAL – ASANTE AKIM SOUTH

NURSES DUTY ROSTER

WARD: FEMALE / PAEDIATRICS

Name		1	2	3	4	5	6	7	8	9	10	11
1	Marian Asankoman Owoo	N	N	N	N	OFF	OFF	OFF	M	A	M	A
2	Mariama Ahmed	M	A	M	OFF	N	N	N	N	OFF	OFF	OFF
3	Yvonne O. Berko	M	A	M	OFF	N	N	N	N	OFF	OFF	OFF
4	Venecia Boateng	N	N	N	N	OFF	OFF	OFF	M	A	M	A
5	Isaac Antwi Mireku	A	M	A	M	A	M	A	OFF	N	N	N
6	QuinzyAmoh	N	N	N	N	OFF	OFF	OFF	M	A	M	A
7	Constance Koramah	A	M	A	M	A	M	A	OFF	N	N	N
8	Monica Bady	OFF	OFF	OFF	A	M	A	M	A	M	A	M
9	Cecilia Appiah	A	M	A	M	A	M	A	OFF	N	N	N
10	Obed Atta Yeboah	M	A	M	OFF	N	N	N	N	OFF	OFF	OFF
11	Doris Owusu	OFF	OFF	OFF	A	M	A	M	A	M	A	M
12	Janet Appiah	N	N	N	N	OFF	OFF	OFF	M	A	M	A
13	Faustina Acheampong	OFF	OFF	OFF	A	M	A	M	A	M	A	M
14	Sandra A. Garbrah	OFF	OFF	OFF	A	M	A	M	A	M	A	M

5.2 CONCLUSION

Most of the hospitals in Ghana manually go through hard time in preparing the scheduling timetable. This indicates that the nurse schedule problem is a real world problem. Instead of consuming time in generating the roster, hospital officers can spare more time and effort on patients and other medical duties since the nurse scheduling problem is a very complex problem with so many constrains. In solving the nurse schedule problem, the soft constraints were not the only constraints considered. All hard constraints were also considered. Results therefore show a feasible solution to the problem. This means that the study ends with a solution to be used in the real world.

5.3 RECOMMENDATION

In the development of the system, it was observed that the technique in solving some to the constraints were insufficient to cope with real-life problems. Particularly, soft constraints get a holiday off and weekend off. Also the program software could pick up the groups after taking away the conflicting schedules.

A future direction of development is to improve upon the program software taking into consideration some of the constraints stated above.

REFERENCES

1. Amponsah S. K., Agyeman E. and Okrah K. G., (2004) Graph Colouring, an Approach to Nurses Scheduling, Case Study: EJura District Hospital, Ashanti Region, Ghana
2. Appel K and Haken W. (1979) Every Planar map is four colourable. Illinois Journal of math, Vol 21:429 – 567, 1979.
3. Arthur J.L., Ravindran A., (1981). A multiple objective nurse scheduling model. AIIE Transactions 13 (1), PP 55–60.
4. BakiKoyuneu and MahnutSecir (2004) student time table by using graph colouring algorithm.
5. Berrada I., Ferland J.A., Michelon P. (1996). A multi-objective approach to nurse scheduling with both hard and soft constraints.Socio-Economic Planning Science 30 (3), PP 183–193.
6. Burke E. K., Elliman D. G and Weare R (1995) A university Timetabling system based on Graph Colouring and constraint manipulation.
7. Burns R.N., (1978). Manpower scheduling with variable demands and alternate weekends off. INFOR 16 (2), 101–111.
8. Cheng B. M. W., Lee J. H. M and Wu J. C. K (2003) A constraint-based Nurse Rostering system using a Redundant modeling Approach.
9. Culberson J. and Ian Gent (2001). Frolen development in graph colouring.
10. Culberson J.& Gent(2001) Frozen development in graph colouring, Theoretical Computer Science. 265, 1-2, PP,227-264

11. Dowsland K. A., (1998). Nurse scheduling with tabu search and strategic oscillation. European Journal of Operational Research 106, PP393–407.
12. Duffy K. R, O'Connell N. and Sapothnikov A., (2006) Complexity analysis of a decentralized graph colouring of algorithm. Ireland Mathematics Istitude
13. Ellie D'Hondt, (2008). Quantum algorithms for graph colouring problems.
14. Ferland J.A., Berrada, I., Nabli I., Ahiod, B., Michelon P., Gascon V., Gagne E. (2001). Generalized assignment type goal programming problem: Application to nurse scheduling. Journal of Heuristics 7, PP 391–413.
15. Griesmer H., (1993). Self-scheduling turned us into a winning team. Management Decisions 56 (12), PP 21–23.
16. Hedetniemi S. T., David P. Jacobs, Pradip K. Srimani (2003) Linear Time Self – Stabilizing Coloring
17. Howell J.P., (1998). Cyclical scheduling of nursing personnel. Hospital JAHA 40, PP 77–85.
18. Jaumard B., Semet, F., Vovor, T., (1998). A generalized linear programming model for nurse scheduling. European Journal of Operational Research 107, 1–18.
19. Johnson M., graph colouring CS103B Handout # 20.
20. JuhosIstvan, Attila Toth, Jano I. van Hemert, (2004). Binary Merge Model Representation of the Graph Colouring Problem
21. Kumara B.T.G.S. Perera, A.A.I. (2011).Automated system for nurse scheduling using Graph Colouring. Indian Journal of Computer Science and Engineering (IJCSE) Vol. 2 No. 3 PP 476 – 485.

22. Kundu S., Mahato M., Mahanty B. and Acharyya S. (2008). "Comparative Performance of Simulated Annealing and Genetic Algorithm in Solving Nursing Scheduling Problem," Proceedings of the International Multiconference of Engineers and Computer Scientists, Hong Kong, PP. 96.
23. Marx D. (2004). Graph colouring problems and their applications in scheduling
24. Miller H.E., Pierskalla, W.P., Rath, G.J. (1976). Nurse scheduling using mathematical programming. Operations Research 24 (5), PP 857–870.
25. Murphey A. R., Pardolos P. M. , Resende M. G. C. (1999) Frequency Assignment Problems. Handbook of combinatorial optimization.
26. NogaAlon, (1993). Restricted colouring of graphs. London math Soc. Lecture Notes series 187.
27. Nolte A. and Schrader W., (2002). An application of simulated annealing to the 3 colouring problem.
28. Rosen K. H (1999). Discrete Mathematics and its application 4th edition, 1999. PP 438 – 517
29. Silver M., Sakuta T., Su H. C., Dolins S. B. and Oshea M. J.,(2001) "Case Study: How to Apply Data Mining Techniques in a Healthcare Data Warehouse," Journal of Healthcare Information Information Management, Vol. 15, No. 2, pp. 155-164
30. TeroHarju (2011) Lecture Notes on Graph Theory, Department of Mathematics, University of Turku, Finland.
31. Villiers P., " Clinical Data Warehouse Functionality," SAS Institute Inc., New Caledonia.

32. Wren A. (1996). Scheduling Timetabling and Rostering – a Special Relationship PP 46 – 75

APPENDIX

MATLAB CODES FOR GROUPING THE NURSES

```
function y = myself(x)
g=graph
n = input('number of nurses');
cycle(g,n)
for k=1:n+1,
delete(g,k,k+1)
delete(g,1,n)
end
ndraw(g)

v=input('number of conflict nurses')
for i = 1:v
xi=input('group of conflict nurses');
ui =combntns([xi],2);
add(g,ui)
ndraw(g)
end
label(g)
label(g,1,'EN1')
label(g,2,'RN1')
```

```
label(g,3,'SN4')

label(g,4,'SN3')

label(g,5,'SN1')

label(g,6,'RN2')

label(g,7,'EN2')

label(g,8,'HEW2')

label(g,9,'HEW4')

label(g,10,'HEW3')

ldraw(g)

color(g)

cdraw(g)
```

MATLAB CODES FOR GENERATING THE SCHEDULE TABLE

```
f = figure('Position',[200 200 3000 1500]);

Morning = 1;

Afternoon = 2;

Night = 3;

Offday = 4;

% group of nurses for the various schedules

a = [1 4 6 12];

b = [2 3 10];
```

```
c = [5 7 9];

d = [8 11 13 14];

dat = zeros(14,30);

for w = 1:4

for k = 1:3

dat(a,w)= Night;

dat(b,4+w)= Night;

dat(c,8+w)= Night;

dat(d,12+w)= Night;

dat(a,16+w) = Night;

dat(b,20+w) = Night;

dat(c,24+w) = Night;

dat(d,28+w) = Night;

dat(b,4) = Offday;

dat(c,8) = Offday;

dat(d,12) = Offday;

dat(a,16) = Offday;

dat(b,20) = Offday;

dat(c,24) = Offday;

dat(d,28) = Offday;

dat(a,4+k)= Offday;

dat(b,8+k)= Offday;

dat(c,12+k)= Offday;
```

```
dat(d,16+k)= Offday;

dat(a,20+k)= Offday;

dat(b,24+k)= Offday;

dat(c,28+k)= Offday;

dat(d,k)= Offday;

end

end

y=8:15;

tf=bitget(abs(y),1)~=0;

oddd=y(tf);

% odd = -3 -1 1 3

evenn=y(~tf);

% even = -4 -2 0 2 4

xi=23:30;

tf=bitget(abs(xi),1)~=0;

odd=xi(tf);

% odd = -3 -1 1 3

even=xi(~tf);

% even = -4 -2 0 2 4

dat(a,evenn)=Morning;

dat(a,oddd)=Afternoon;

dat(a,odd)=Morning;

dat(a,even)=Afternoon;
```

```
y=1:3;
tf=bitget(abs(y),1)~=0;
oddi=y(tf);
% odd = -3 -1 1 3
eveni=y(~tf);
% even = -4 -2 0 2 4
yi=12:19;
tf=bitget(abs(yi),1)~=0;
oddii=yi(tf);
% odd = -3 -1 1 3
evenii=yi(~tf);
% even = -4 -2 0 2 4
yii=27:30;
tf=bitget(abs(yii),1)~=0;
oddiii=yii(tf);
% odd = -3 -1 1 3
eveniii=yii(~tf);
% even = -4 -2 0 2 4
dat(b,oddi)=Morning;
dat(b,eveni)=Afternoon;
dat(b,evenii)=Afternoon;
dat(b,oddii)=Morning;
dat(b,oddiii)=Afternoon;
```

```
dat(b,eveniii)=Morning;
q=1:7;
tf=bitget(abs(q),1)~=0;
oddq=q(tf);
% odd = -3 -1 1 3
evenq=q(~tf);
% even = -4 -2 0 2 4
dat(c,oddq)=Afternoon;
dat(c,evenq)=Morning;
qi=16:22;
tf=bitget(abs(qi),1)~=0;
oddqi=qi(tf);
% odd = -3 -1 1 3
evenqi=qi(~tf);
% even = -4 -2 0 2 4
dat(c,oddqi)=Afternoon;
dat(c,evenqi)=Morning;
r=4:11;
tf=bitget(abs(r),1)~=0;
oddr=r(tf);
% odd = -3 -1 1 3
evenr=r(~tf);
% even = -4 -2 0 2 4
```

```
dat(d,oddr)=Morning;

dat(d,evenr)=Afternoon;

ri=20:26;

tf=bitget(abs(ri),1)~=0;

oddri=ri(tf);

% odd = -3 -1 1 3

evenri=ri(~tf);
% even = -4 -2 0 2 4
dat(d,evenri)=Afternoon;
dat(d,oddri)=Morning;
cnames = {'1','2','3','4','5','6','7','8','9','10',...

'11','12','13','14','15','16','17','18','19','20','21',...

'22','23','24','25','28','27','28','29','30'};

rnames = {'Maria ArankomahOwoo','MariamaAhmed','Yvonne O Berko',...

'VeneciaBoateng','Isaac AntwiMireku',...

'QuinzyAmoh','Constance Koramah','Monica Badu','CeciliaAppaiah',...

'Obed Atta Yeboah','Doris Owusu','Janet Appiah',...

'FaustinaAcheampong','Sandra A. Garbrah'};

t = uitable('Parent',f,'Data',dat,'ColumnName',cnames,...

'RowName',rnames,'Position',[2 2 360 100]);

Dat =get(t,'Data');

xlswrite(FileName,Dat);
```