AN ASSIGNMENT ALGORITHM FOR NURSES USING HL7

ERDEM OKUR

JUNE 2015

AN ASSIGNMENT ALGORITHM FOR NURSES USING HL7

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF

NATURAL AND APPLIED SCIENCES OF

IZMIR UNIVERSITY OF ECONOMICS

BY

ERDEM OKUR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

IN THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

JUNE 2015

M.S. THESIS EXAMINATION RESULT FORM

Approval of the Graduate School of Natural and Applied Sciences

Prof. Dr. Cüneyt Güzeliş
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Turhan Tunalı
Head of Department

We have read the thesis entitled **"A Smart Assginment Algorithm for Nurses Using HL7 "** completed by **Erdem Okur** under supervision of **Assoc. Prof. Dr. M. Süleyman Ünlütürk** and **Asst. Prof. Dr. Gökhan Kılıç** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Asst. Prof. Dr. Gökhan Kılıç
Co-Supervisor

Assoc. Prof. Dr. M. Süleyman Ünlütürk
Supervisor

Examining Committee Members **Date:** 11.06.2015

Assoc. Prof. Dr. M. Süleyman Ünlütürk
Dept. of Software Engineering, IUE

Prof. Dr.Mehmet Kuntalp
Dept. of Electric and Electronics Engineering, DEU

Assoc. Prof. Dr. Hüseyin Akcan
Dept. of Software Engineering, IUE

Asst. Prof. Dr. Gökhan Kılıç
Dept. of Civil Engineering, IUE

Asst. Prof. Dr. Kaan Kurtel
Dept. of Software Engineering, IUE

ABSTRACT

AN ASSIGNMENT ALGORITHM FOR NURSES USING HL7

Erdem Okur
M.S. in Intelligent Computing Systems with Thesis
Graduate School of Natural and Applied Sciences
Supervisor: Assoc. Prof. Dr. M. Süleyman Ünlütürk
Co-Supervisor: Asst. Prof. Dr. Gökhan Kılıç
June 2015

Everyday an increasing number of patients come to the hospitals for several reasons. Some of them does check-ups, others have different or maybe serious problems. Depending on their situation, patients sometimes must stay at the hospital for few days or more. That is where nurse scheduling problem comes in. Nurses are responsible for the patients that stay at the hospital all day long. Of course, from the hospital's point of view, there is a workload for these patients and it has to be split among the nurses equally. This is done by nurse manager or head-nurse manually. Unfortunately manual distribution does not always create equal distributions for the nurses. In some cases they share the rooms where patients stay but some of these rooms have only one patient whereas the others may have up to 4 or 6 depending on the hospital. Moreover, the patients may have different conditions. In order to make distribution process automated and equal an algorithm is proposed in this thesis. This algorithm will distribute patients to nurses using the number of beds occupied and the patient acuity level as parameters.

Keywords: Nurse Scheduling, Nurse Rostering, health-care, Health Level 7, HL7, ADT.

ÖZ

HEMSIRELER ICIN SAGLIK SEVIYESI 7 KULLANAN ATAMA ALGORITMASI

Erdem Okur
Akıllı Hesaplama Sistemleri Tezli, Yüksek Lisans
Fen Bilimleri Enstitüsü
Tez Danışmanı: Doç. Dr. M. Süleyman Ünlütürk
İkinci Tez Danışmanı: Yrd. Doç. Dr. Gökhan Kılıç
Haziran 2015

Her gün artan sayılarla bir çok hasta hastanelere geliyor. Bazıları kontrolleri için, bazılarıysa daha farklı ve belki de daha ciddi problemlerle. Durumlarına göre bazen hastalar, bir ya da bir kaç gün hastanede kalmak zorunda kalabiliyorlar. Hemşire Çizelgesi Problemi tam da bu nokta da ortaya çıkıyor. Hemşireler, hastanede kalan hastalardan bütün gün sorumlu. Tabi ki hastane tarafında bu hastalar bir iş yükü oluşturmakta ve bu yük hemşirelere eşit olarak dağıtılmalı. Bu işlem baş hemşire tarafından gerçekleştiriliyor. Ne yazık ki böyle bir dağıtım, hemşireler arasında her zaman eşit iş yükü sağlayamıyor. Bazı durumlarda hastaların bulunduğu odalar paylaştırılıyor ancak bu odaların bazılarında tek hasta varken, bazılarındaysa 4 ila 6 hasta olabiliyor. Dahası bu hastaların durumları ve ihtiyaçları birbirinden farklı. Bu tezde iş yükü dağılımını otomatik ve daha eşit hale getirmek için tasarlanan bir algoritma sunuluyor. Bu algoritma, hastaları bulundukları yatak bilgisi ve ihtiyaç durumlarını göz önünde bulundurarak hemşirelere dağıtacaktır.

Anahtar Kelimeler: Hemşire Çizelgeleri, bakım, sağlık seviyesi 7, hastane bilgi sistemleri .

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

We live in a world, where we change as we progress. From ancient times to present, we create new things and destroyed some of the old things. This is same for our habits and lives. Today we have so much services we created, which are "must-have", helper or sometimes just for fun. For instance, parking lots, hospitals and cinemas are all daily services we used in need. As we can see, these services are bond to our lives. We need them and plus we expect them to have some quality. Quality is comparing our expectations with performance. Good quality depends on several things like, if the service staff is well trained on the subject. Even if the staff is well trained, they have to be managed correctly in order to get service well done. Especially for the services where a manager needs to assign or schedule staff for some task or tasks. This thesis is about service and staff that has critical importance for our lives, hospitals and nurses.

Hospitals have specialized staff and equipment for patient treatment. They must have an emergency department and may have different specialized centers. Also hospitals have service areas with large number of beds and wards for intensive care and long-term care. A teaching hospital is a type of hospital used in this thesis as an explaining example. In teaching hospitals patient treatments and medical students and nurse trainings are done together. Teaching hospitals may have different names such as; Clinical Schools, Research Hospital or University Hospital depending on local definitions. Nurses have varying and important roles in hospitals. These roles sometimes start from performing tasks such as personal care or social care to more specialized tasks such as blood sampling and recording vital signs of a patient. Also nurses may have specialties on different areas like emergency nursing, dental nursing, cardiac nursing and surgical nursing etc. and if a nurse has a specialty in an area he/she usually works in that area of the hospital.

Nursing is a continuous job. Everyday increasing number of people comes and goes to the hospitals with several different cases. Depending on patient's case, patient may have to stay at the hospital's ward or special rooms. For instance; after a surgery or waiting for some tests etc. Also, each day newborn number gets higher and life expectancy becomes longer. However, with decreasing number of nurses due to number of schools and so [1], combined with the rising demand for nurses comes from the situations written before, brings Nurse Scheduling Problem forth.

This thesis proposes an algorithm for a part of the NSP. The proposed algorithm will assign nurses to patients in hospital's ward using patient's acuity levels. The next chapter is about Nurse Scheduling Problem in detail and related works on the subject.

Chapter 2

Nurse Scheduling Problem (NSP)

Nurse Scheduling Problem (NSP) , that is also called Nurse Rostering Problem (NRP), is generally speaking about the schedule of the nurse's shifts in hospital. It is a scheduling or timetabling problem with lot of constraints from different areas. The constraints include budget, hospital environment, nurse number, ward number, and patient numbers. With all these different constraints at hand it is very hard to solve a NSP. In order to be more specific we must first understand what a rostering/scheduling problem is.

Rostering is a scheduling in a workplace for employers. Workplaces have different tasks for different employers. These tasks depend on the work type and they have to be completed in some time interval mostly. In short, tasks at a workplace are the workload for employers that work there. It is crucial for employees to get efficient results for quality of their work from the employers and creating a valid schedule for them is critical for that matter. A rostering problem this kind is non-deterministic polynomial time (NP)-complete in its general form [2], since its difficulty to find a solution rises exponentially in polynomial time, an effective and acceptable solution cannot be found [3],[4].

NRP that features nurses are even more complicated than their general ones. The constraints that effect to create valid timetables for nurses vary greatly and include some parameters depending on the type of the constraint. These constraints may be divided into two categories like soft and hard constraints [5] according to their importance level. Hard constraints are critical and must be satisfied, whereas some of the soft constraints may not be satisfied. The quality of the schedule depends on the number of constraints satisfied.

There are many published algorithms for NSP. However, every hospital has its own constraints and infra-structures, because of that NSP may have some unique characteristics. Therefore a general solution which is acceptable is hard to achieve. Moreover, some constraints may have to be satisfied before others and this creates some kind of phased structure for problem. For instance, number of nurses and depending on to that enough budgets needed to be calculated. Next, day and night shifts can be set, if hospital has any. Of course these bring high qualified or less qualified nurse issue. Some of the wards in a hospital (Intensive Care etc.) require high qualified nurses whereas, other wards may require less qualified nurses managed by the high qualified nurses (Head Nurse kind). This is just a small part of NSP. Therefore, budget is about administers and can become an important part of the problem. On the other hand, deciding about day and night shifts is another part of NSP and so on. For further details on the subject, you may refer to Aickeln and Dowsland [6].

Now that NSP is explained, it is time to convey information about approaches to solve NSPs. A general approach on the NSP for today's hospitals is either a meta-heuristic algorithm or an algorithm that solves the problem phase by phase. In the next sub-section, a brief explanation on the initial approaches and an extensive look for today's approaches with some samples will be given.

2.1 Related Works

Approaches on the NSP changed as the years past. This is because the characteristics of the NSP have changed from hospital to hospital. Therefore some of the approaches become ineffective for unique NSP. A detailed explanation on these approaches can be found in Bradley and Martin [7], Hung [8] and Sitompul and Randhawa [9].

There are new approaches on the subject though. Meta-heuristic algorithms are one of them. Genetic Algorithms (GA) are meta-heuristics which widely used on this subject. GA's basically uses some features of natural evolution and survival of the fittest concept. They are stochastic meta-heuristics which starts with randomly created population of the solutions. The better solutions or fittest ones have more possibility to be selected as parents to create next generation. Also in every generation there is a small chance for mutation of the old solutions. These generations continue until some stopping criteria is achieved. Besides, there are concepts like penalty and repair functions for GAs. Briefly, penalty functions try to steer generations into feasible solution searching and repair functions try to repair or fix some infeasible solutions with potential to be feasible. GAs are originated back to 1970's to Holland [10] and his students. However evolutionary computing idea dates further back and for an extensive review on the subject, refer to Fogel [11].

One of the good examples of meta-heuristic algorithm approach is Aickeln and Dowsland [12]. This approach is created from their previous one [6], which show a direct GA implementation is not able to handle the problem. The problem specific information must be added into the solution. Sub-populations that co-evolving with real population and modified fitness functions created by them and still could not handle small changes in problem specifications. In order to improve the solution they changed it to be an indirect GA. This means that, individuals in a population are not representing direct encoding of a solution. Solutions can be reached by a separate decoder heuristic which uses available nurses as constraint guides. The advantage here is making GA canonical. If any of the constraints are changed or new constraints came, the GA stays unchanged. It is because constraint handling is done by decoder heuristic. It is found out that, indirect GA approach is better than previous evolutionary approaches.

Another meta-heuristic approach on NSP is using Tabu Search. Dowsland [13], used this approach and find relatively good solutions on a particular hospital.

However, its data was less good on the hospitals with different characteristics. For detailed result comparison with Tabu Search and indirect GA approach refer again Aickeln and Dowsland experiments [12].

It is discussed that NSP's may be accepted as phased structures. Different from meta-heuristic, solving phases one by one is another approach for NSP. A close example for this type of approach and motivation of this thesis can be found in Punnakitikashem, Rosenberger and Behan [14]. In their characteristic NSP, they called whole problem as Nurse Planning and divided it into 4 phases. First phase is Budgeting. Nurses will be hired as permanent staff from some agency and this is decided by financial planners. Second phase is Nurse Scheduling. A nurse manager forecasts the number of patient that may enter the hospital over next several hours. Depending on that number, nurses and their professions -if they have any-, a schedule is created with day and night shifts that approximately 8-12 hours in a day. Third phase is Nurse Rescheduling. Before 90 minutes from every shift a nurse supervisor (nurse manager or head-nurse) checks the schedule and if any shortage of nurse -due to some unforeseen situation- exists for the upcoming shift, it is solved in this phase. Lastly, the fourth phase is Nurse Assignment. Again a nurse manager assigns every nurse to a task at the beginning of a shift but this time tasks are patients. This is where the real workload on nurses calculated. Workload depends on time required for a nurse to care a patient over a time period. Of course caring time periods are different for patients and if a nurse has to spend too much time to care a patient, it creates Excess Workload. Overworking nurses may need assistance of others. This leads nurse shortage or ineffective care. Therefore workload balance is important to achieve.

Creating balanced workloads for nurses is really hard task. First of all the patients they care have very different conditions. Most of the time nurses are assigned to patients by dividing the patient number to nurses or dividing the number of beds in the rooms in hospital wards. By that, every nurse has same number of beds or patients to look out. Punnakitikashem et. al. solved this final phase of nurse planning using a stochastic model and different assumptions varies from conditions of the patients to nurse shifts [14].

In the next section, a proposed algorithm will be explained on the nurse assignment part of a NSP. This algorithm utilizes hospital information systems as data source. It uses these data to look for bed occupancy in the hospital wards and patient's acuity to assign the nurses with their workloads balanced.

Chapter 3

The Proposed Algorithm

The proposed algorithm for nurse assignment basically works as the following: System communicates with Hospital Information System (HIS) using Health Level 7 (HL7) messaging standard in order to get initial data. After that, it seeks the information for patients and their acuity levels and also beds occupied by them. In that time, the patient's acuity level points out the condition of patient. This acuity level directly affects the amount of time that a nurse requires to properly care for patient. To give a quick example; unoccupied beds require no staff and an occupied bed require a staff naturally but according to the acuity level of the patient that time may be substantial. Also workload balance depends on this constraint too. Assigning nurses to patients fairly according to their conditions, give nurses an equal care time which is actually their true workload. In the next sub-sections, Admission-Discharge-Transfer (ADT) systems, Health Level 7 (HL7) standards and how bed occupancy data is taken from ADT based information systems using HL7 messaging standard will be explained. At the last sub-section, the proposed algorithm will be given in detail with the help of these initial explanations.

3.1 Admission-Discharge-Transfer (ADT)

An admission, discharge and transfer (ADT) system is one of the four types of core business systems. The others are financial, acuity and scheduling. These are all used in health care facilities for different purposes. For instance; financial payment, quality improvements and some new practices in the facility, if they are proven to be beneficial, are some of them [15]. In short, an ADT is a software application which tracks patients from the time of their arrival at the hospital until departure by transfer, discharge or death.

ADT systems hold various data. Most of the data are very valuable because they are usually information about the patients such as a medical record number, age or contact information. Because of these specialties about ADT, they are usually the foundation for other types of health care information systems. Besides, it can be used to share patient information with other systems [15], if it is required. Moreover, a facility may use ADT as an alarm system, which is active upon patient's arrival [16]. With the help of this kind of feature, history of a patient which includes infectious disease or maybe important information like heart conditions can be learned immediately after patient is recorded to the hospital.

In this thesis, a Hospital Information System (HIS) used as a data source for the nurse assignment algorithm. This HIS is assumed to utilize an ADT database as foundation. A HIS is a general system that provides common information about patient's health history. It keeps this data in secure and usually has a control method about form who and when a data can be reached in different circumstances. Generally speaking, HIS controls hospitals, different documents, patient information, medical histories, prescriptions and operations. Also many other things can be included to this such as laboratory results. Every ward in a hospital may have specific software for them and a database like ADT based ones can be used in them. A HIS will interconnect all together. At the time these different systems share information HL7 standards comes to forward.

In Figure 3.1, you can see a design sample for HIS. It is minimal and basic

Figure 3.1: A HIS design sample. Dots mean more ares can be added.

but, more can be added. It shows how a HIS come together with other systems and handles communication.

3.2 Health Level 7 (HL7) Standards

Health Level 7 International is a not-for-profit organization, which is founded in 1987. HL7 name comes from the Open System Interconnection (OSI) model. Level 7 means application level in OSI model. Basically, HL7 is a set of standards for transfer of clinical and sometimes administrative data between the applications used by various health care facilities. Facilities use different software even in themselves and usually these are contained as a HIS. However, HIS do not have to be same at every hospital or facility but, patients may be. Therefore, reaching some critical data and/or transfer it to another component or HIS is crucial. This is where HL7 standards come in handy [17].

HL7 International provides number of different flexible standards or guidelines. These are actually set of rules to make data transfer, share and processing in a uniform. When such a consistency is achieved, it is easier for the hospitals to share information. The following standards are the primary ones for HL7. They are most commonly used and implemented [17].

- Version 2.x Messaging Standard
- Version 3 Messaging Standard
- Clinical Document Architecture (CDA)
- Continuity of Care Document (CCD)
- Structured Product Labeling (SPL)
- Clinical Control Object Work-group (CCOW)

The following sub-sections will describe each standard briefly except Version 2.x. Version 2.x is used on the proposed algorithm. The data from HIS in a hospital communicates with the proposed algorithm to share its patient data and bed occupancy data utilizing this Version 2.x messaging HL7 standard. Thus, Version 2.x standard will be given in more detail.

3.2.1 Version 2.x Messaging Standard

Version 2.x is a basic messaging standard for supporting hospital work-flows. Originally it is created in 1989 and it is known as Pipehat too [18]. From the time it is started, version 2 is updated regularly. That is why it is called Version 2.x. There were several updates and these resulted in 2.1, 2.2, 2.3, 2.3.1, 2.4, 2.5, 2.5.1, 2.6, 2.7 and 2.8. All of these updates are backwards compatible. Thus, if you used a message based on 2.1, it can be understood by another one which based on 2.6. Also, this means that, the message is not important between the versions, but how you sent it is.

HL7 Version 2.x messages use a unique syntax that is human-readable (ASCII). A HL7 message is created by one or more segments. These segments include one more fields (also known as composites). Below, you can see an instance of a HL7 message.

```
MSH|^~\&|EPIC|EPICADT|SMS|SMSADT|199912271408|CHARRIS|ADT^A04|1817457|D|2.5|

PID||0493575^^^2^ID 1|454721||DOE^JOHN^^^^|DOE^JOHN^^^^|19489293|M||B|254 MYSTREET

AVE^^MYTOWN^OH^44123^USA||(216)123—4567|||M|NON|400003403~1129086|

NK1||ROE^MARIE^^^^|SPO||(216)123-4567||EC|||||||||||||||||||||||||

PV1||O|168 ~219~C~PMA^^^^^^^^^^^||||277^ALLEN MYLASTNAME^BONNIE^^^^||||||||||

||2688684|||||||||||||||||||||||||199912271498|||||||002376853
```

Figure 3.2: A HL7 message example [19]

Segments in the message are separated by carriage return character which is "\r" or 0D in hexadecimal. That means each line of the sample above is a segment of that message and it is same for each HL7 message. In the each line, a pipe character "|" can be seen every now and then. This character separates the fields in the segment of a message. If there is a case in which a field has a sub-field, it is separated by "^" characters normally. All these dividing characters are called delimiters or delimiter characters [19]. In the next sub-sections HL7 Version 2.x message components will be explained.

3.2.1.1 Segments

HL7 segments are the main parts of a message. They contain one specific category of information. These categories may be patient information or visit data etc. Usually the first field in the segment is its name. This name is always three characters long. In the figure 2, there are 4 segments: MSH, PID, NK1 and PV1. These segments may change and will change depending on the HL7 message type.

(The message type can be learned from the message's MSH segment. It is mostly the ninth field of the MSH.) Meanings of segments from the sample are given below.

- MSH (Message Header): It contains information about the message itself. This information may consist of sender, receiver, type of the message and date it was sent. Every HL7 message has MSH as the first segment.
- PID (Patient Information): It includes a general information about patient, such as name, ID and her/his address.
- NK1 (Next of Kin): Contact information for the patient's relatives.
- PV1 (Patient Visit): It is the information, which is about the patient's time at the hospital. It may have the time that patient stayed and who was the doctor responsible from her/him.

Segments in the sample given in figure 2 are basic segments. They can exist in several different messages and MSH must be in every one of them. However, there are more than 120 segments for HL7 messages.

3.2.1.2 Fields

A field is a part of the segment. It can be a string or a number or it can contain other fields. If a field contains another one, second field is called sub-field. Sub-fields are separated by "^" character. However, in some cases even sub-fields may have sub-sub-fields and they are separated by ampersand character. Moreover, sub-sub-fields must be either string or number. In the sample in figure 2, it can be clearly seen that, patient name and its address fields exist in the PID segment. Another important thing that comes to the eye in the sample is a delimiter that occurs as series. It is right after the name of the patient. There are 4 " ^" characters as series. It means there are 6 sub-fields in that field and only two of them are defined. Besides, some fields may include two double-quotes. This is a sign for a field, which has information that is present but null.

3.2.1.3 Delimiter Characters

Delimiter characters are the separators of HL7 messages as it is discussed before. By default, ”\r” character separates segments, ”|” separates fields, ”ˆ” character separates sub-fields and ampersand character separates sub-sub-fields. Moreover, ”˜” character separates repeating fields and ”\” character is used for escape sequence. There is an important feature in the HL7 messages about delimiter characters. It is allowed to re-define them. Although it gives flexibility to the sender, it is hard to be read due to probable difference from the receiver. Of course, there is a solution to that problem. Every message starts with MSH segment. If a message contains different delimiter characters, MSH segment informs the receiver. Right after the MSH name a series of delimiter characters are given in an order. The order goes from segments to the sub-sub-fields.

Escape sequence character is really important, if there are any delimiter characters is used in the message as a part of the message. In such cases, one can use the figure below for escape sequences.

Table 3.1: Common escape sequences for the special characters. Additional sequences can be found in the reference link [20].

Character	**Escape Sequence**
&	\T\
$\wedge$	\S\
$\vert$	\F\
$\sim$	\R\
\	\E\
Hexadecimal Character xx	\Xxx..\

3.2.2 Version 3 (V3) Messaging Standard

Since it is created back in 1997, V3 actually aims the same goals as Version 2.x. Main difference is, it tries to improve Version 2 process and the outcomes of it. It is mostly based on the Reference Information Model (RIM). In short, RIM is for reducing the implementation costs and further standardization. Although the goals of both versions have much in common, V3 is very different from Version

2.x. It is planned to include low level communication protocols for encoding. Furthermore, base data type is planned to be XML, instead of the simple ASCII text as in the Version 2. There are some difficulties about V3 [21], so version 2.x still used widely.

3.2.3 Clinical Document Architecture (CDA)

CDA is an exchange standard for clinical documents. This one specifies the encoding, structure and semantics of the documents. CDA is ANSI-certified and it is based on the HL7 Version 3 data type. Plus that means; it is based on the RIM too.

According to CDA standard a clinical document should have the following six characteristics:

- Persistence
- Context
- Stewardship
- Wholeness
- Potential for authentication
- Human Readability

Some examples of clinical documents as follows: Discharge Summary, Admission, Pathology Report. CDA uses XML but more it allows the usage of non-XML body for different implementations. This body can be pdf, word, or jpeg and so on.

3.2.4 Continuity of Care Document (CCD)

CCD is again a standard for clinical document exchange. It focuses on the same features of a clinical document as the CDA. However, this time the clinical document is one type only. CCD is specifically designed for Patient Summary clinical document. From a point of view, it is a constraint on the CDA standard. CCD uses XML likewise.

The patient summary contains several important data about patient's health-care. It includes demographic and the other types of information which may lead the health-care practitioner (doctor etc.) to solve a confusing disease case. In short, patient summary is a snapshot in time which shows clinical, demographic and administrative data [22].

3.2.5 Structured Product Labeling (SPL)

SPL is another document exchange standard. It is XML based and defines the content of prescription drug labeling. It has a specific format and can be displayed in a browser. SPL is a well-known standard and has been adopted by the Food and Drug Administration (FDA) in U.S. for exchanging medical information.

SPL documents include product's content of labeling and at the same time contain specific information about the drug. This information is about the ingredients, dosage forms, the package quantity and type of it and some machine readable data.

3.2.6 Clinical Control Object Work-group (CCOW)

CCOW is a standard protocol for synchronizing disparate applications in real-time. Moreover, this synchronization is at the user-interface level. CCOW is based on the process called Context Management. This is the process of using a subject or more of interest to create a virtual link between applications. A subject example may be patient for health-care in this context. With the help of this process, an end-user can see the application as they operate in a cohesive way. Although, context management can be utilized either in CCOW or non-CCOW compliant applications, CCOW standard aims near plug-and*play interoperability and therefore it is more robust and efficient.

Those were the primary standards that HL7 International provided. However, there are more standards and methodologies like above [23].

3.3 Bed Occupancy and Patient Acuity Levels

Bed and patient acuity information are crucial for the proposed algorithm. They reduce constraints on the nurse to patient assignment phase of nurse scheduling. Therefore, reduce the complexity of required algorithm. Patient acuity data will give information about the patient's condition and this gives hints about how much time is needed for that patient's care. This is handled by the proposed algorithm. That creates a balanced workload to the nurses, opposed to assignments done by administrators which are based on a ward's bed number. This number is just a representative of what might be but it is not what it is now. There may be unoccupied beds that an administrator assigned a nurse to it and simply he/she can do nothing. This is where bed occupancy and patient acuity gets into relation. As it is discussed in Chapter 2, generally nurse assignment phase is done like above. Even if the bed occupancy constraint is handled, patient acuity is not used. In order to get a better workload balance and better assignments with more clear algorithm, these constraints will be handled together in the proposed algorithm.

In order to achieve this, first bed occupancy and patient acuity data must be reached. They will be taken from a Hospital Information ADT system. That is a HIS which uses ADT based database for securing the data we need. To be able to get the data needed, an ADT Gateway Interface that utilizes HL7 Version 2 messaging standard is used [24]. ADT Gateway Interface accepts delivery of

HL7 messages using a Local Area Network (LAN). Technically, ADT Gateway Interface is connected to Nurse Call stations and HIS ADT system through LAN. They all exchange messages between them.

HL7 standard messages are discussed to be different for various purposes. For the purposed algorithm and ADT Gateway Interface, specific segments of a HL7 message are required. These can be seen in the table below. All other ADT segments are ignored.

Table 3.2: The segments required for ADT Gateway Interface and the proposed algorithm.

Segment Character Sequence	**Purpose**
MSH	Message Header
EVN	Event Type
PID	Patient ID
PV1	Patient Visit 1
PV2	Patient Visit 2
NK1	Next of Kin 1
NPU	Bed Status
MRG	Merge Patient Info.

There is a feature of HL7 standard messages, which is not discussed yet. In the MSH segment, the type of the message can be found as it is explained in Chapter 3, but there is a Trigger Event for the message exists too. A Trigger Event is set of conditions which cause to transfer of information between system components. There are three reasons that may initiate a trigger event: User Request Based, State Transition and Interaction Based. User request based trigger events are called Environmental too. An example to that may be pressing a button that will collect some specific data from the system for every 24 hours. State transition trigger events can be understood from their names. As an instance to these types of events, opening a new document or canceling one can be given. Most of the trigger events are based on this type. Interaction based trigger events are results of the other interactions such as response to a query.

In Table 3.3, all trigger events that may be used in ADT Gateway Interface are shown. All others again rejected. This is because of ADT Gateway Interface accepts all HL7 messages but examines what is needed and uses them.

Table 3.3: The trigger event values.

Trigger Event String	Description
A01	Admit a patient
A02	Transfer a patient
A03	Discharge a patient
A04	Register a patient
A05	Pre-admit a patient
A06	Transfer an outpatient to inpatient
A07	Transfer an inpatient to outpatient
A08	Update patient information
A09	Patient departing
A10	Patient arriving
A11	Cancel admit
A12	Cancel transfer
A13	Cancel discharge
A17	Swap patients
A20	Nursing/Census application updates
A21	Leave of absence - out
A22	Leave of absence - in
A29	Delete person information
A30	Merge person information
A33	Cancel patient departing
A34	Merge patient information - patient ID only
A35	Merge patient information - account number only
A36	Merge patient information - ID and acc. number
P01	Add and update patient account

All of the trigger events above have some segments provided for them. The Table 3.4 below shows the minimum required and optional message segments. Of course, there are requirements for the fields of the segments given in the Table 3.2 and Table 3.3. These are not included here in order to not to complicate chapter anymore. Further information can be found in Unluturk and Atay [24].

Patient acuity level is taken as High, Medium and Low. This data is decided when the patient admitted at the hospital. The admitting physician assigns a care code to the patient. HL7 keeps this data inside of the Admission Level of Care Code data field. This data field is included in the Patient Visit Additional Information (PV2) segment of the HL7 message. An example PV2 segment and

Table 3.4: The trigger event sepecific segments and needed fields of them.

Trigger Event String	**Description**	**Opt. Segments**
A01 - Admit a patient	MSH, EVN, PID, PV1	PV2, NK1
A02 - Transfer a patient	MSH, EVN, PID, PV1	PV2, NK1
A03 - Discharge a patient	MSH, EVN, PID, PV1	PV2, NK1
A04 - Register a patient	MSH, EVN	
A05 - Pre-admit a patient	MSH, EVN	
A06 - Transfer an outpatient to inpatient	MSH, EVN, PID, PV1	PV2, NK1
A07 - Transfer an inpatient to outpatient	MSH, EVN, PID, PV1	PV2, NK1
A08 - Update patient information	MSH, EVN, PID, PV1	PV2, NK1
A09 - Patient departing	MSH, EVN, PID, PV1	PV2, NK1
A10 - Patient arriving	MSH, EVN, PID, PV1	PV2, NK1
A11 - Cancel admit	MSH, EVN, PID, PV1	PV2, NK1
A12 - Cancel transfer	MSH, EVN, PID, PV1	PV2, NK1
A13 - Cancel discharge	MSH, EVN, PID, PV1	PV2, NK1
A17 - Swap patients	MSH, EVN, PID, PV1	PV2, NK1
A20 - Nursing/Census application updates	MSH, EVN, NPU	
A21 - Leave of absence - out	MSH, EVN, PID, PV1	PV2, NK1
A22 - Leave of absence - in	MSH, EVN, PID, PV1	PV2, NK1
A29 - Delete person information	MSH, EVN, PID, PV1	PV2, NK1
A30 - Merge person information	MSH, EVN, MRG	
A33 - Cancel patient departing	MSH, EVN, PID, PV1	PV2, NK1
A34 - Merge patient info. - ID	MSH, EVN, MRG	
A35 - Merge patient info. - acc. number	MSH, EVN, MRG	
A36 - Merge patient info. - ID and acc. no	MSH, EVN, MRG	
P01 - Add and update patient account	MSH, EVN, PID	

the care code is shown in Figure 3.3. After care code is read, it is converted to acuity levels as low, medium or high. There are 6 care codes: Critical (CR), Acute (AC), Chronic (CH), Comatose (CO), Improved (IM) and Moribund (MO). When the care code is read, a piece of code runs. This code gives the acuity levels. If the care code is MO, CR or CO, the acuity level is decided as High. If the care code is AC, Medium is returned as acuity level. Lastly, if the care code is IM or by default, Low acuity level is returned. The proposed algorithm uses Table 3.5 to convert them into corresponding relative weights. Bed data is provided from the Hospital Information ADT system.

The algorithm requires the full set of available patient information for the

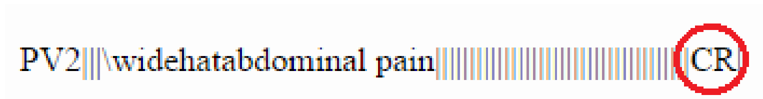

Figure 3.3: An example of PV2 segment in HL7. Red circle shows acuity level.

Table 3.5: The patient acuity levels are converted to the relative weights with the help of this table.

Patient Acuity Levels	Relative Weight
High	10
Medium	5
Low	1

Admit a patient (A01) trigger event. Transfer an inpatient to outpatient (A07) and Patient departing (A09) messages are treated as Discharge (A03). Moreover, it requires the full set of available information in both sets of PID, PV1 and optional PV2 segments for the Swap patients (A17) trigger event.

The next section will explain how the proposed algorithm works with the help of the data it get from previous sections.

3.4 The Proposed Algorithm

The proposed algorithm first starts with the important data. In order to properly work, algorithm needs patient acuity levels and bed occupancy information. Of course, to be able to assign them to patients, nurses must be in the algorithm as staff information too. Process works as follows. Firstly, the staff data is read from Hospital Information ADT system database. If this data is missing, algorithm stops. Otherwise, it is time for the bed occupancy data. If it exists, then the patient acuity levels are converted to the relative weights (Table 3.5). However, if there is no bed data, again algorithm stops. After conversion of the acuity levels, beds are sorted according to these weights in descending order. At the last part for preparation is about workload. All acuity levels are aggregated and an average number is calculated by dividing this aggregated value by the total number of nurses in the hospital (or one of the wards at the hospital depending on where nurse wants to use the algorithm). If any case occurs where a floating-point number is the result, this number is rounded up to the next integer. This result is the average workload for nurses. Upon this calculation, algorithm is ready to run. At the end of the chapter, pseudo code for the algorithm is given in Algorithm 1. Moreover, in Figure 3.4 the flowchart diagram for the proposed algorithm is shown, its purpose is to give a better understanding of the structure and steps of the algorithm. Besides, it provides a clearer and less complicated view of the proposed algorithm.

There is one exception of the algorithm which is not shown in the flowchart and pseudo code. This is when all the beds in the BedList are checked for the same nurse. This means there is no available bed for this staff according to his/her workload. If this case occurs, algorithm does following:

- Resort all nurses in the NurseList according to their workload in ascending order. Thus, nurse with relatively less work ends up top of the list.
- Resort beds in BedList according to the patient acuity levels in descending order. With that, beds requiring more time end up at the top.
- **FOR EACH** nurse in the NurseList,
    Assign first bed in the BedList to the first nurse.
    Remove the first bed from the BedList.
    IF BedList is empty,
      THEN break;
    END IF.
  END FOR EACH.
- Assignment process is done. **STOP.**

Again this exception is not shown in Algorithm 1 and Figure 3.4 to prevent it from complicating them any further. In the next chapter, the proposed algorithm is used in a demoware. This demoware shows how algorithm works in real life with a sample hospital ward. The software and how it is made is explained too.

```
Data: Patient Acuity Levels, Bed List and Nurse List
Result: Beds assigned to nurses
Read staff data;
if Staff data exists then
    Read Bed data with patient acuity levels;
    if Bed data exists then
        Sort the collection according to Acuity Levels in descending order;
        Calculate the average workload;
        while There is a nurse in the Nurse List do
            Pick a nurse from the list;
            if Any bed exists in the Bed List then
                if the first bed is "Assigned" in the List then
                    Resort the Bed List in descending acuity levels order.;
                    Mark all the beds "NOT Assigned";
                    Remove the nurse from the Nurse List;
                    go back to beginning of while;
                else
                    Assign the first bed from the sorted Bed List;
                    if Nurse's workload is greater than average workload
                    then
                        Put that bed at the end and mark as "Assigned";
                        go back to beginning of previous if;
                    else
                        Remove the bed from the Bed List;
                        go back to beginning of first if after while;
                    end
                end
            else
                Exit;
            end
        end
        Exit;
    else
        Exit;
    end
else
    Exit;
end
```

Algorithm 1: Pseudo code of the proposed algorithm

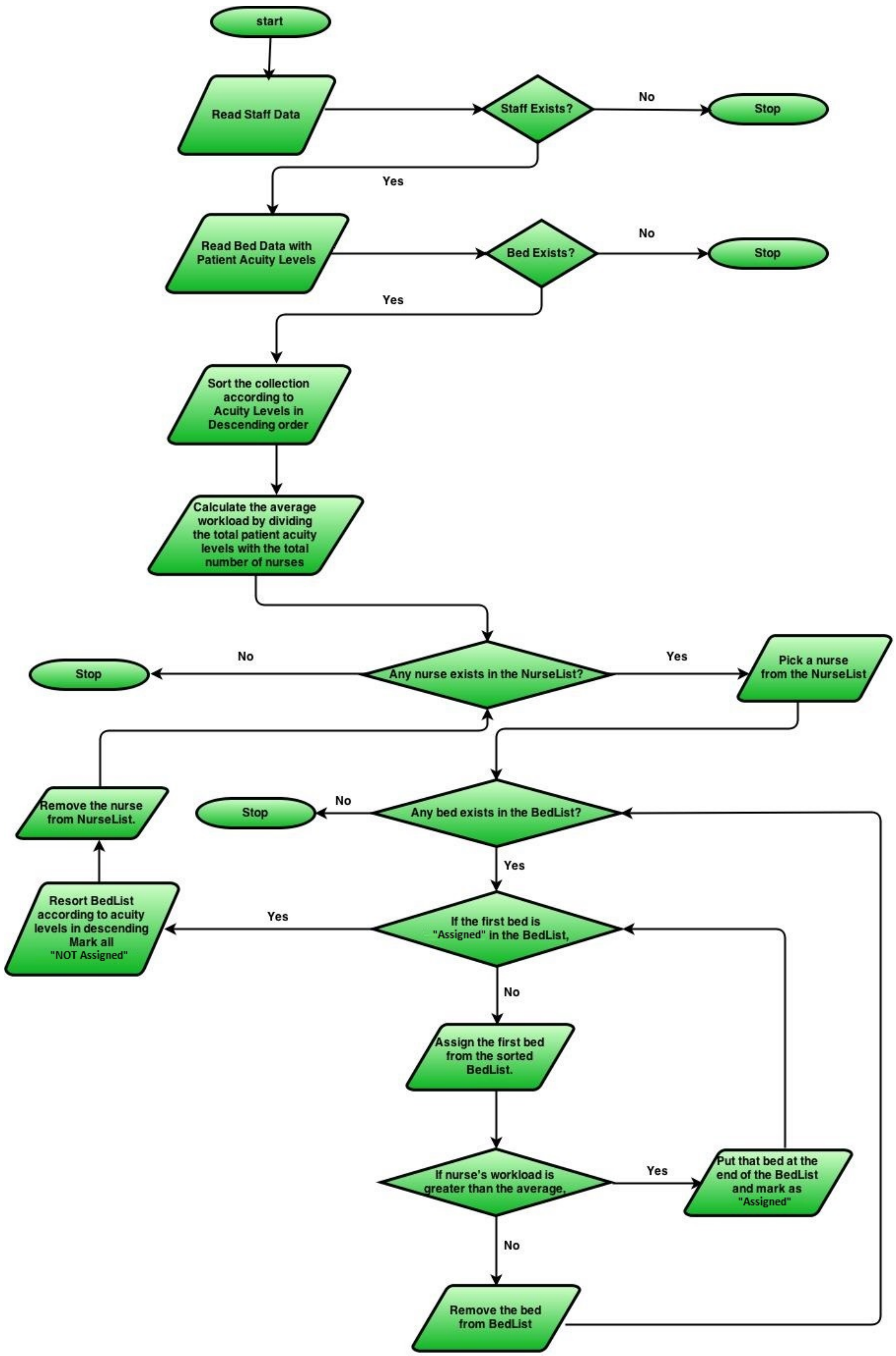

Figure 3.4: Flowchart diagram which shows how algorithm works in a clear way.

Chapter 4

Demoware

The proposed algorithm for assignment of nurses to the patients is given in the last chapter. However, without a proper demonstration, it is less promising. For demonstration purposes, an application is written. It is for Windows 8.1 operating system and created by using Windows Presentation Foundation (WPF). The application is running on a sample hospital ward. This ward has different number of beds, patients and nurses. The proposed algorithm works behind and assigns the nurses to the patients.

This chapter includes 2 sections. One will explain what the WPF is and the other one will give details about the application itself and its parts.

4.1 Windows Presentation Foundation (WPF)

WPF is a presentation system for building Windows applications with visually enhanced user interfaces. At its core, it is a resolution independent and vector based engine which is built for benefiting from the modern graphics hardware. WPF was actually named Avalon before and it is released as a part of .NET Framework 3.0 in 2006 [25]. Because of this, a developer who has experience on the framework can use his/her favorite elements from it in the WPF. Furthermore, WPF includes features from ASP .NET too. With that, a developer can create an application using both mark-up and code-behind. If someone used .NET Framework and ASP .NET before, it is very easy to adapt to WPF, in short. Below there is a picture of an example WPF application.

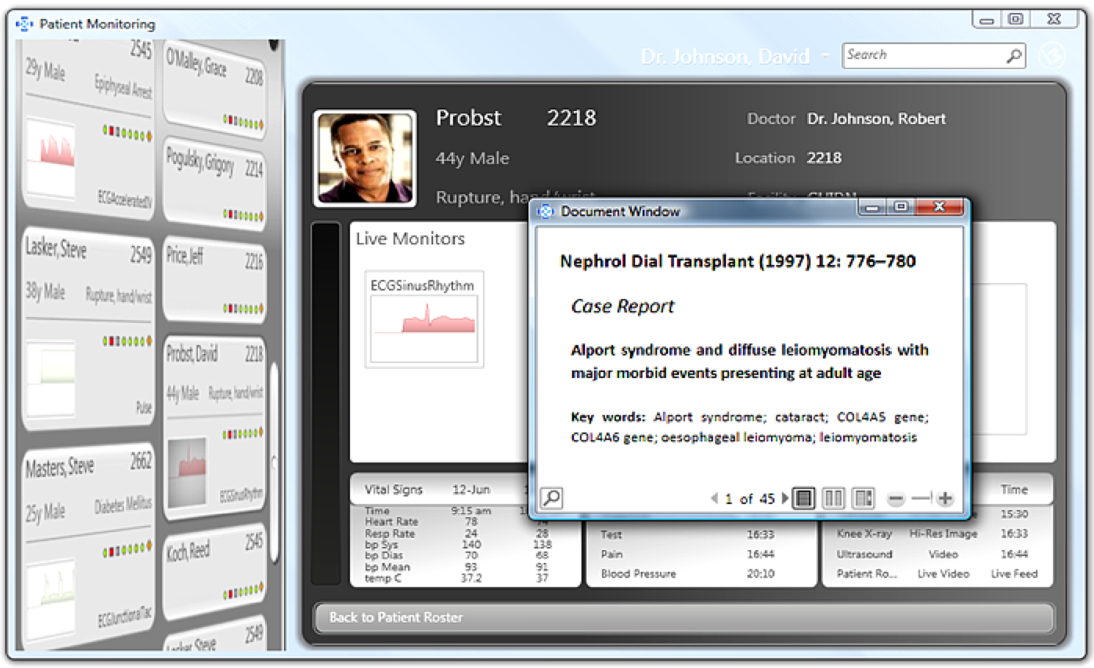

Figure 4.1: A WPF application example [25].

WPF's core is talked above and it extends this core with following features: Extensible Application Markup Language (XAML), controls, data binding, layout, styles, templates and a few more visual enhancements that can be found with detail in Microsoft website [25]. In the following sub-sections the features which are used in the demoware application will be explained briefly.

4.1.1 Markup Language (XAML) and Code-Behind

In WPF, there exists very useful development ways. One of them is having the ability to develop an application using both Markup and Code-behind. Mostly markup (XAML) is used to implement the appearance of an application whereas code-behind is used to give the appearance its behaviour. For instance, when a button will be created, developer places it in the designer and shapes its appearance with the help of the XAML. After visuals are done, developer can define its behaviour writing the code-behind. When someone clicked it, it reflects what developer implemented in XAML and works as the implemented code-behind. In short, it is a way of separating appearance and behaviour. In Figure 4.2, the designer with its XAML and the code-behind can be seen.

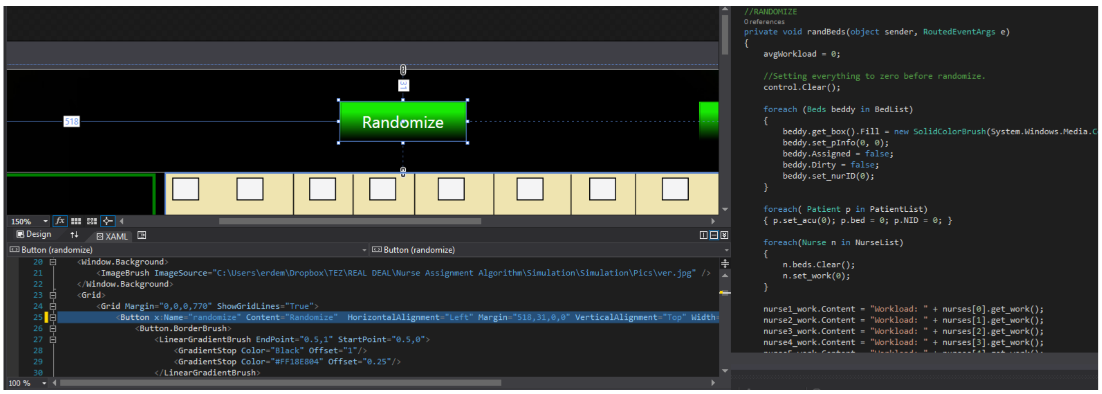

Figure 4.2: A view of XAML implementiation and its result in the designer and its code-behind.

The separation of the appearance and behavior gives certain advantages on design and implementation. These are listed below:

- In traditional approach appearance code and behavior code is tightly coupled. Developer should know how to implement them both together. However, in WPF they are not tightly coupled. A designer can implement XAML for user interface and a developer can implement the code-behind separately. This reduces development and maintenance costs.
- With the advantage explained before, another one come front. Because a designer and a developer can work separately, they can work simultaneously. Therefore development becomes more efficient and fast.
- There are several design tools to get an XAML markup output. Thus, enhanced visual appearance can be achieved for application.
- Globalization and localization of WPF is another advantage that will not

be given in detail in the thesis. Basically it is an advantage for reaching your application to global audience [26].

In the next sub-sections, markup language XAML and code-behind explained further.

4.1.1.1 Extensible Application Markup Language (XAML)

XAML is XML-based language which is used to implement appearance of an application declaratively. With the help of it, developers can create windows dialog boxes, user controls and fill them with different features like shapes, graphics or more control. In the example in Figure 4.3, it can be seen that how a button is created in XAML for this thesis demoware.

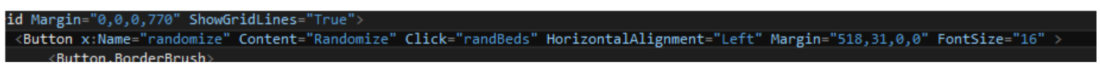

Figure 4.3: XAML code for Randomize button in the demoware.

This XAML defines a button using Button element and set some of its attributes. When the application is run, WPF converts this element with its attributes to instances of WPF classes. In the example, Button element is converted to a Button class and attributes have become properties of that class. Properties have different effects on the object. Content property, for instance, is a string for what will be written on the button in application. The connection between markup and code-behind is achieved with a property too. For this button, it is the Click property. Click property holds the name of the function that will be called, if the button is clicked. In Figure 4.4, visual result of the XAML in the example can be seen on the run.

4.1.1.2 Code-Behind

Code-Behind is basically implementing the way that controls act. It is called code-behind, because the implemented code is associated with the markup, which

Figure 4.4: Randomize button which created from the XAML before.

creates the visuals of the control in front. It is triggered by user interactions and events that come from the controls created by markup. The next figure includes some of the code-behind for the XAML sample from demoware in the XAML example.

```
//RANDOMIZE
1 reference
private void randBeds(object sender, RoutedEventArgs e)
{
    avgWorkload = 0;

    //Setting everything to zero before randomize.
    control.Clear();

    foreach (Beds beddy in BedList)
    {
        beddy.get_box().Fill = new SolidColorBrush(System.Windows.Media.Colors.White);
        beddy.set_pInfo(0, 0);
        beddy.Assigned = false;
        beddy.Dirty = false;
        beddy.set_nurID(0);
    }

    foreach( Patient p in PatientList)
    { p.set_acu(0); p.bed = 0; p.NID = 0; }

    foreach(Nurse n in NurseList)
    {
        n.beds.Clear();
        n.set_work(0);
    }

    nurse1_work.Content = "Workload: " + nurses[0].get_work();
    nurse2_work.Content = "Workload: " + nurses[1].get_work();
    nurse3_work.Content = "Workload: " + nurses[2].get_work();
    nurse4_work.Content = "Workload: " + nurses[3].get_work();
    nurse5_work.Content = "Workload: " + nurses[4].get_work();
    nurse6_work.Content = "Workload: " + nurses[5].get_work();
    Den.Content = "Avarage Workload: 0";

    for (i = 0; i < NURSENUMBER; i++)
    {
        if (i == 0)
        {nurse1_beds.Content = "Beds Assigned: 0"; }

        if (i == 1)
        {nurse2_beds.Content = "Beds Assigned: 0"; }

        if (i == 2)
        {nurse3_beds.Content = "Beds Assigned: 0"; }

        if (i == 3)
        {nurse4_beds.Content = "Beds Assigned: 0"; }

        if (i == 4)
        {nurse5_beds.Content = "Beds Assigned: 0"; }

        if (i == 5)
        {nurse6_beds.Content = "Beds Assigned: 0"; }
    }

    foreach (Patient p in PatientList)
    {
        //Acuity Levels
```

Figure 4.5: Part of the code-behind for Randomize button. It defines the behaviour of button.

It shows how a click event handled for the button. After the click event handler is implemented and when someone clicks that button, this handler takes control and initializes the environment for next run in demoware. This will be explained in detail in the next section.

4.1.1.3 Controls and Input

Control is a category of classes that are hosted in either a window or depending on your project- a page. They have user interface and some behaviour to be implemented. Well known controls are; buttons, dialog boxes, documents, layout, media, menus and selection (check boxes, combo boxes, radio buttons and so on). Controls mostly detect user input. Depending on their behaviour they respond the user input differently. User inputs may be text input, mouse positioning (over, etc.). This concept gives robust user interfaces in WPF due to the separation of their implementation discussed earlier.

4.1.1.4 Data Binding

In WPF, it is already included due to .NET Framework, that databases can be used to view and edit data. Besides, a developer can create objects for the data and communicate or populate them from databases. After that, controls can be used to bind them to user interface. For instance, in the demoware a list view is used to show the data about patients. The data comes from the ADT database and assigned to Patient class and its properties. Then data binding feature binds objects in the code to list view in the markup to show Patient data on the screen.

4.2 Application

The application to test the proposed algorithm is written in WPF using Visual Studio 2013 [27]. The main purpose of the application is to simulate a hospital ward with several rooms that have beds. These beds may have or may not have patients and the patients have different conditions. For the proposed algorithms case, these conditions are acuity levels of the patients. Furthermore, the ward has limited number of nurses and that settles the environment. At last, simple functions change the beds and acuity levels of patients. Therefore initializes the ward. Then, algorithm runs and shows which nurses assigned to which patients based on their beds. For this demo, a hospital ward which has 43 beds and 6 nurses is used. There will be 34 patients who will have their acuity levels and beds randomized on every run.

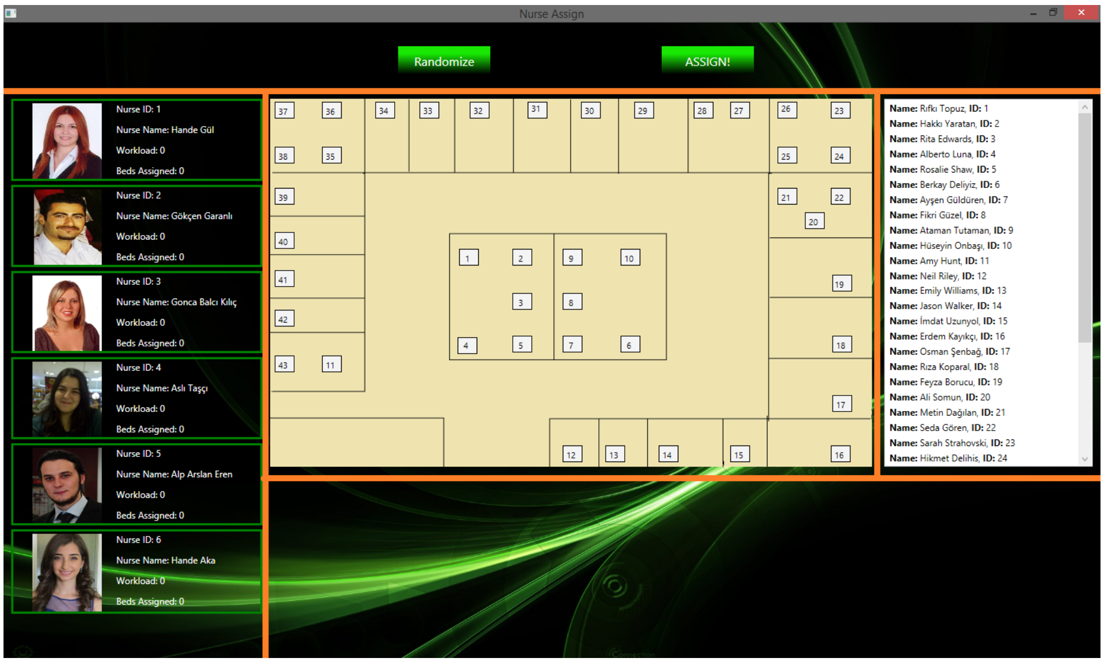

Figure 4.6: Interface of the demoware with lines that indicates the grids.

The main window is separated into five grids (Figure 4.6). In the top grid there are two buttons and a label that shows average workload. Below there is a grid which includes nurses and their properties on the left side of them. A simple sketch that shows the ward and the rooms in it with beds can be seen in middle grid. At the most right, there is the list of patients. The fifth and empty for now grid will have a different function and it will be explained in the next chapter. The grids named in order like: Top Grid, Nurse Grid, Bed Grid and Patient Grid. Last one is called the fifth grid as mentioned before.

In the next subsections the grids will be given briefly. If the grid has a direct effect on how the algorithm works in the application, it will be more detailed.

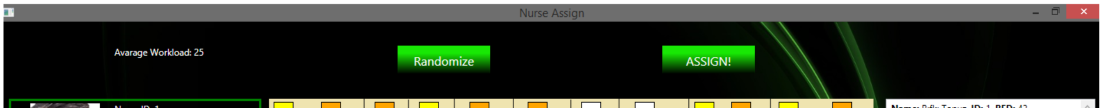

Figure 4.7: Top Grid with its objects inside.

4.2.1 Top Grid

Top Grid is where most functionality of the application exists. Actually there are only 3 objects on that area (Figure 4.7). First of them is Average Workload label. It shows the information about the average workload of the nurses when calculated by the algorithm. It is initialized to 0 at the start. Next, there is a button which named as Randomize. Randomize button is for initializing everything used in the algorithm to their first values. At the start of the application defining properties for Nurses, Patients and Beds is set. After that, when algorithm ran and results are calculated, Randomize can be clicked to set the environment for another run. Moreover, it randomizes the patient acuity levels and beds assigned to create different scenarios. Changes on the other grids can be seen too (Bed Grid beds). Patient, nurse and bed numbers are not changed, because algorithm works independent of their numbers. In order to make scenarios more natural, acuity levels are randomized but have probabilities of happening too. Low level has %60, medium has %30 and high level has %10 of being an acuity level for a patient. The last object is another button. The name of this button is Assign. As it can be understood from its name, when clicked it runs the algorithm. It must be clicked after the Randomize button is clicked. Upon click, all data set from the Randomize button is used to make assign objective using the proposed algorithm. The results are shown in the Nurse Grid and Top Grid's average workload label.

4.2.2 Nurse Grid

In the Nurse Grid there is information about all nurses (Figure 4.8). There are blocks as the nurse number and each block have 5 objects in them. One object for picture of the nurse and the other objects are for the properties of his/her. Labels show the name, id, workload, and the beds they are occupied. After Assign is clicked, results of the assignment can be seen in workload and bed labels. Workload shows how much workload is on the nurse at the given assignment.

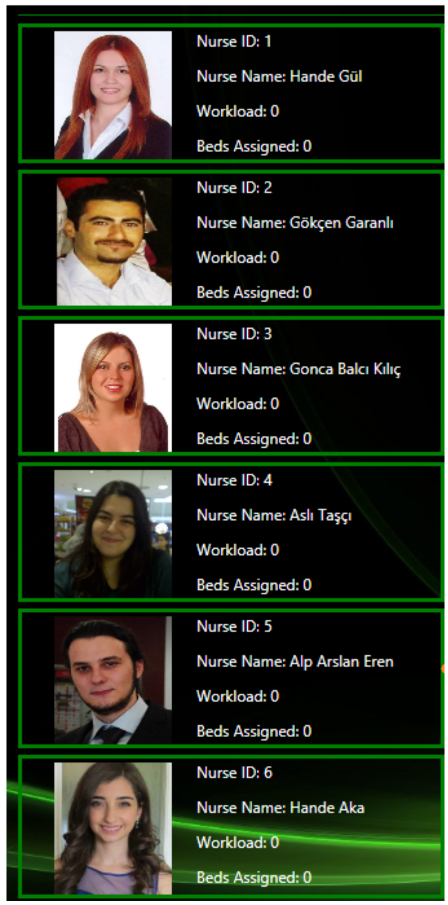

Figure 4.8: Nurse Grid with its objects.

Average workload in the Top Grid can be compared to this workload value in order to learn if the nurse has overloaded or not. Beds show to which patients the nurse will care. It is given as bed ids rather patient ids in purpose. Because of every patient changes their beds randomly but in Bed Grid which beds are occupied can be seen with ids, giving assignments as bed ids makes it easy to understand. Nurse Grid starts like there is no bed or workload data. After Assign is clicked, results are replaced with the old data of the beds and workload.

4.2.3 Beds Grid

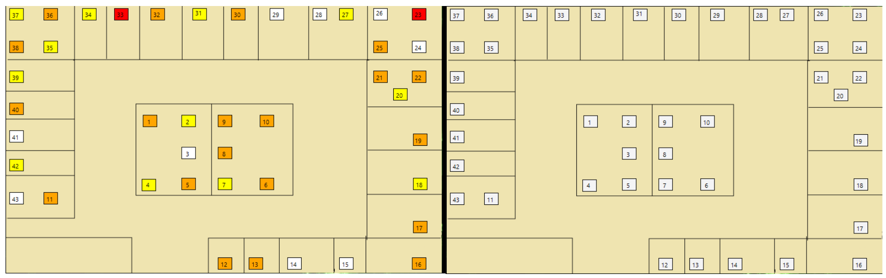

Figure 4.9: Beds Grid. On the right side the beds are occupied with the patients. On the left, all the beds are unoccupied.

Beds Grid represents the hospital ward rooms and the beds in it as a sketch. Boxes in the rooms are the beds and their ids placed on the boxes. When simulating application is started beds will be white as in the left side of the Figure 4.9. White colour means that the beds are unoccupied. After the Randomize is clicked, all beds but unoccupied ones will be red, orange or yellow. The initializing process at the Randomize click event gives different random beds and acuity levels to patients. It is talked that the acuity levels are represented as low, medium or high and their respective weights are 1, 5 and 10. With that information, the yellow bed means the bed has an acuity level 1 patient in it. Orange has level 5 and the red has level 10 patient in it. After Assign is clicked, ids on the beds can be compared to assigned beds to the nurses. So that, which nurse cares which level patient can be understood.

4.2.4 Patient Grid

Patient Grid has the lowest functionality among the others. When the demo is started, data created about patients are shown in a list view in this grid. List view includes patient's name and id property. If there is need for the other properties, it is easy to add them in XAML too. Patients are not displayed like nurses due to their number. It is easier to represent 38 patient objects in scrollable list view (Figure 4.10).

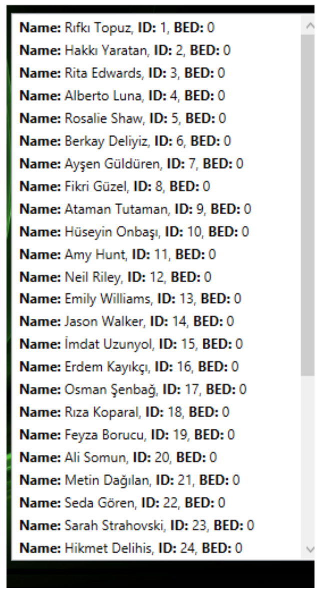

Figure 4.10: Patient Grid which displays patients in a list view.

In the next chapter, varying results from the demoware on the sample hospital is discussed. Moreover, a general Genetic Algorithm for the subject is created for comparison between proposed algorithm's deterministic approach and a meta-heuristic one. Results and the comparison is given also.

Chapter 5

Experimental Results

This section discusses the outcomes of the proposed algorithm on the sample ward and its comparison to meta-heuristic approach. It includes outcome discussion on several runs for the ward and a snapshot for a random result among them. We conduct experiments on a sample ward, which has 43 beds, 6 nurses and 38 patients. The numbers were taken from an existing service in a hospital in Turkey. The ward's Head-nurse assigns each nurse to patients or rooms depending on the situation at her hand. For privacy purposes, all names used in this thesis are fictional.

The main purpose of the algorithm is assigning the nurses to patients with equal workload. Aside from the NSP final phase it is also important making this process faster than an individual and with fewer errors too. At the end contributing to the solution of NSP by solving the final phase of assigning nurses to patients is the real goal. Below in Figure 5.1, a snapshot of one of the runs on this hospital ward can be seen.

Each experiment is conducted 100 times with random patient acuity levels and bed assignments. A result of a run accepted as the comparison of average workload to assigned nurse's workload. Results are examined by considering these:

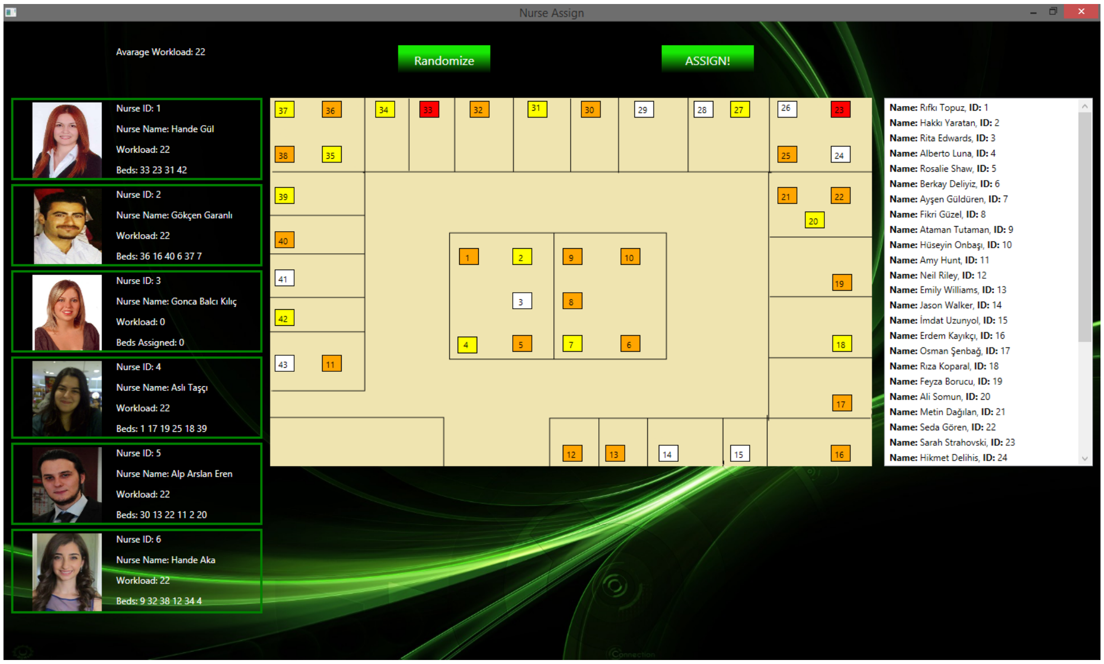

Figure 5.1: A snapshot of a result from a run.

- All occupied beds have to be assigned.
- A result of a run depends on the workload and can be equal, unequal or near equal.
- An equal workload balance is accepted either if all nurses have same workload or at least 5 nurses have same workload but the other one is not overloading or not 4 unit under of the average workload.
- An unequal workload balance is accepted when all nurses have different workloads from average workload (overloaded or underloaded).
- A near equal workload balance is accepted when 20% of the nurses are under the average workload but other are equal to average. If there is one overloaded nurse, the result is unequal.

The difference between nurse workloads happens because of the acuity levels. Acuity levels have just 3 values and their average may not allow an equal distribution. For example, if a case occurs, when average overload is calculated a number that can't be divided by 5 or 10 (like 23) and there is one patient left to be assigned with acuity level 5, it is almost obligatory to assign this patient to a nurse with a workload among others. However, if all nurses already close to the average workload like 2-3 units, the nurse who will be assigned to this last patient will be overloaded. Because of this any overloading nurse is not accepted as an equal result but with the same reason again approximately 15% percent of the nurses who has a workload under average is accepted as equal, when all other nurses have average workload. A long term solution to this will be discussed in the next chapter.

After running the proposed algorithm 100 times with these terms, results are presented in Table 5.1. 82% of the results were equal and in 31% of the equal results, all nurses had same average workload. 11% of the results were near equal. 7% of them were unequal results. Of course, this demo only proves the algorithm works on the sample ward with different conditions of patients.

Table 5.1: Results of the proposed algorithm's 100 runs on the sample ward.

Test Run	Average Workload	Nurse Workloads After Assignments	Equal	Near Equal	Unequal
1	30	30 30 30 30 26 30	x	-	-
2	25	25 25 25 25 25 25	x	-	-
3	25	20 25 25 25 25 25	x	-	-
4	22	22 22 22 22 22 22	x	-	-
5	23	20 21 23 23 23 23	-	x	-
6	30	26 30 30 30 30 30	x	-	-
7	25	25 25 25 25 25 25	x	-	-
8	29	30 25 30 26 29 30	-	-	x
9	25	25 25 25 25 25 25	x	-	-
10	22	22 22 22 22 22 22	x	-	-
11	21	21 21 21 21 17 21	x	-	-
12	25	25 25 25 25 25 22	x	-	-
13	22	22 22 22 22 22 22	x	-	-

Test Run	Average Workload	Nurse Workloads After Assignments	Equal	Near Equal	Unequal
14	22	22 22 22 22 22 22	x	-	-
15	23	25 22 23 23 23 23	-	x	-
16	26	26 26 26 26 22 26	x	-	-
17	24	24 25 25 24 24 20	-	-	x
18	30	30 30 30 30 30 30	x	-	-
19	25	25 20 25 25 25 25	x	-	-
20	26	26 26 26 26 26 26	x	-	-
21	26	26 26 26 26 22 26	x	-	-
22	27	25 27 27 27 27 27	x	-	-
23	22	22 22 22 22 21 22	x	-	-
24	25	25 25 25 25 25 25	x	-	-
25	25	25 25 25 25 25 25	x	-	-
26	28	30 25 28 30 26 28	-	-	x
27	25	25 25 25 25 25 24	x	-	-
28	25	25 24 25 25 25 25	x	-	-
29	21	21 21 21 19 21 21	x	-	-
30	26	26 26 26 26 25 26	x	-	-
31	25	25 25 25 25 24 25	x	-	-
32	25	21 25 25 25 25 25	x	-	-
33	26	26 26 26 26 24 26	x	-	-
34	25	25 25 25 25 25 24	x	-	-
35	27	25 27 27 27 25 27	-	x	-
36	22	22 22 22 22 22 22	x	-	-
37	26	26 26 26 26 21 26	x	-	-
38	26	22 26 26 26 26 26	x	-	-
39	25	25 25 25 25 25 25	x	-	-
40	20	20 20 20 15 20 20	x	-	-
41	25	25 25 25 25 25 24	x	-	-
42	21	21 20 21 21 21 21	x	-	-
43	22	22 22 19 22 22 22	x	-	-

Test Run	Average Workload	Nurse Workloads After Assignments	Equal	Near Equal	Unequal
44	20	20 20 20 18 20 20	x	-	-
45	22	18 22 22 22 22 22	x	-	-
46	30	30 30 30 30 30 26	x	-	-
47	25	20 25 25 25 25 25	x	-	-
48	20	20 20 20 20 20 20	x	-	-
49	29	29 29 26 30 30 25	-	-	x
50	22	22 22 17 22 22 22	x	-	-
51	23	23 23 23 23 22 20	-	x	-
52	27	27 27 27 27 25 25	-	x	-
53	27	25 27 27 27 27 26	-	x	-
54	25	25 25 25 25 25 25	x	-	-
55	25	25 25 25 25 20 25	x	-	-
56	27	27 27 25 27 27 25	-	x	-
57	26	21 26 26 26 26 26	x	-	-
58	26	26 26 26 26 26 26	x	-	-
59	26	26 26 26 24 26 26	x	-	-
60	22	22 22 22 22 22 22	x	-	-
61	25	21 25 25 25 25 25	x	-	-
62	27	25 27 27 25 27 30	-	-	x
63	22	22 22 21 22 22 22	x	-	-
64	25	25 25 25 25 25 22	x	-	-
65	30	30 30 30 30 25 30	x	-	-
66	27	27 27 27 27 25 27	x	-	-
67	30	30 26 30 30 30 30	x	-	-
68	23	20 23 21 23 23 23	-	x	-
69	26	26 26 26 26 26 21	x	-	-
70	25	25 24 25 25 25 25	x	-	-
71	25	25 25 25 25 25 25	x	-	-
72	25	25 25 25 25 25 25	x	-	-

Test Run	Average Workload	Nurse Workloads After Assignments	Equal	Near Equal	Unequal
73	25	25 25 22 25 25 25	x	-	-
74	27	27 27 25 27 27 25	-	x	-
75	20	18 20 20 20 20 20	x	-	-
76	25	25 25 25 25 25 23	x	-	-
77	27	27 25 26 25 27 27	-	-	x
78	25	25 25 25 25 21 25	x	-	-
79	27	25 27 27 27 27 26	-	x	-
80	25	25 25 25 25 22 25	x	-	-
81	27	25 27 27 27 27 26	-	x	-
82	22	22 22 22 22 19 22	x	-	-
83	23	23 23 20 23 23 23	x	-	-
84	25	20 25 25 25 25 25	x	-	-
85	25	25 25 25 25 25 25	x	-	-
86	20	20 20 20 20 20 20	x	-	-
87	26	26 26 26 26 25 26	x	-	-
88	21	21 21 21 20 21 21	x	-	-
89	26	26 26 26 26 26 26	x	-	-
90	21	21 21 21 21 21 20	x	-	-
91	30	30 30 30 30 30 30	x	-	-
92	26	26 26 26 23 26 26	x	-	-
93	23	25 23 20 21 23 23	-	-	x
94	26	26 26 26 26 26 26	x	-	-
95	25	25 25 25 24 25 25	x	-	-
96	25	25 25 25 25 25 25	x	-	-
97	23	20 23 23 23 23 23	x	-	-
98	26	26 26 26 25 26 26	x	-	-
99	31	31 27 31 31 31 31	x	-	-
100	26	26 26 26 26 26 26	x	-	-

For an extensive demo of the algorithm, different datasets which have varying numbers of nurses, patients and average workloads are needed. In order to do that, similar problems are researched and Bin Packing Problem (BPP) datasets are found to be used on the algorithm. These datasets are created by Armin Scholl and Robert Klein in 90's and can be found online in their web-site as ".BPP" files [28]. Basically in BPP, you have a number of items which have different weights and you have infinite number of bins with same capacities (capacities can be change depending on the type of problem). Your job is to find the minimum number of bins needed for putting all the items in the bins. You cannot exceed capacity of a bin. The BPP datasets can be used on this thesis problem easily. The items and their weights can be used as patients and their acuity levels, bins and capacity can be used as nurses and average workload. Normally, datasets give the minimum bin number as a solution to that data-set. However, in our problem we already know the number of nurses. Therefore, the minimum bin number is used as a proof and assigned as the nurse number directly. Minimum bin number reflects that you must use at least that many bins in order to put your items into them without exceeding certain capacity. Thus, if proposed algorithm can assign all items to minimum number of bins without exceeding the capacity, it means that it has found the optimal solution for that data-set. Furthermore, a genetic algorithm with general aspects is created and ran on the data-sets as a comparison method between our deterministic nurse assignment algorithm and a meta-heuristic genetic algorithm. Genetic Algorithms search solution space to find an optimal or at least a feasible solution [10]. Because of that, a comparison with proposed algorithm on the same data-sets may be a point of view for how well the proposed algorithm operates. The next sub-section describes our GA in detail.

5.1 Genetic Algorithm

Our genetic algorithm coded for the comparison has general aspects. GA's consist of different features and it is covered thoroughly as it is needed for the GA coded in the thesis. First part to explain is encoding of the problem. For GA, a problem must be represented as a chromosome. After trying various representations, the one in Figure 5.2 is used. The chromosomes are created from the nurse ids that are assigned to patients in the order of the patient ids.

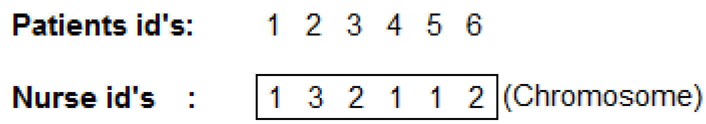

Figure 5.2: Representation of the problem for GA. Boxed numbers in the Nurse id line shows the chromosome for 3 nurse 6 patients assignment problem.

For every patient in the problem, there must be a nurse assigned. So, in the Figure 5.2, how a chromosome is created from the patient list sorted according to their ids with assigned nurse ids to every one of them, is shown. This makes it easier to be used in crossover operator. Selection is the part where chromosomes in the population are selected to be in crossover operator and pass their genes to the next generation. This is done by a general method called Roulette Wheel Selection [29]. This method is like a roulette wheel game and it gives every chromosome a chance to be selected for crossover where this chance is increasing with better fitness. Crossover is the part where selected chromosomes change their parts or genes depending on the operator. There are options to do that such as; single point, two point or uniform. In our GA single point crossover is used. Single point crossover is done by selecting a random point on the selected parent chromosomes and replacing the parts beyond that point between them. Mutation is another part that steers the population of chromosomes, besides crossover. It has a little chance of occurring on a chromosome and how it happens may change depending on the problem. For this one, two points on a chromosome is selected and their values are replaced with each other. Also, depending on the best fitness in the population it has one more way to happen. If the best fitness stays same for more than 150 generations, mutation happens such a way that it changes one point on the chromosome with a random nurse id which is not used in the chromosome before, if it exists, of course. Lastly, calculating the fitness of a solution is the most important part for a GA. Fitness function differs for every problem and it shows the algorithm how a chromosome or solution is better than the others. For this problem, fitness is calculated from the number of nurses unused and the overload amount of the nurses. It is a double number where integer part shows unused nurses and floating part shows how much the average workload exceeded for all nurses. After the algorithm is coded, parameter tuning is done on the code. In order to get lesser computer load and accurate results, different values are tried on the GA parameters. So, some of the variables set to appropriate values. Population size is set to 500; iteration is set to 40000. Crossover and mutation probabilities are decided to be 0,85 and 0,15 respectively. In Algorithm 2, our GA is given.

```
Data: Maximum bin number, item number and weights and the capacity
Result: Best solution with items assigned to bins
function geneticAlgorithm()
Read dataset from BPP file;
Population ← {};
for i=0 to populationSize do
|   chromosome ← randomChromosome();
|   fitness(chromosome);
|   add chromosome to Population;
end
for i=0 to iteration do
|   selectedChromosomes ← rouletteWheelSelection(Population);
|   for each chromosome c in selectedChromosomes do
|   |   r ← random(0,100);
|   |   if r <crossoverRate then
|   |   |   children ← crossover(c);
|   |   end
|   end
|   for each chromosome c in children do
|   |   r ← random(0,100);
|   |   if r <mutationRate then
|   |   |   mutation(c);
|   |   end
|   |   fitness(c);
|   end
|   add children to Population;
|   evaluate(Population);
end
bestSolution ← best solution in Population;
end function
```

Algorithm 2: Genetic Algorithm that will be used with BPP datasets.

5.2 Experimental Results on the BPP datasets

Experiments for the proposed algorithm and the GA created for comparison is done on a computer which has an Intel i5 2450m processor and 8GBs of ram. In the Table 5.3, the results of experiments on the both algorithms can be seen side by side. The first column includes the names of the data-sets. This is also important, because the names represent the values in the data-set. It is given in the "NxCyWz_A..T.BPP" format. Explanations about this format are below:

- "N" letter indicates the number of items. "x" shows the 4 values (e.g.: 1, 2, 3, 4) it can be; 50, 100, 200 or 500.
- "C" letter indicates the capacity of the data-set. "y" shows the 3 values (e.g.: 1, 2, 3) of it; 100, 120, 150.
- "W" letter indicates the weight range of the items. "z" number can be either 1, 2 or 4. It represents weight range as; [1,100], [20,100], [30,100] respectively.

Also Table 5.2 shows these explanations clarified:

Table 5.2: Conversion table for the data-set name format that indicates the variables in the data-set.

Letter/Values	1	2	3	4
x	50	100	200	500
y	100	120	150	-
z	[1,100]	[20,100]	-	[30,100]

Depending on these, if the name of the data-set file is `"N1C3W4_A.BPP"`, it means it has 50 items with weights ranging from 30 to 100 and capacity of the bins is 150. The letter "A" for naming the data-sets with same features, if there are any. Second column of the table shows a range from worst possible fitness to best one for that data-set. The minimum bin number of the data-set which is the nurse number of our problem is used as a basis in the fitness function as discussed above. The minimum bin number is the first line in the data-set file. Since the worst case scenario of a data-set is none of the nurses is used but one, the fitness of a solution ranges from 0 to nurse number minus one. 0 means that all the nurses are used and there is no nurse with overloading workloads, thus this is an optimal solution. This range will give perspective on how well an algorithm works on the data-set. Third column shows the fitness for the proposed algorithm's solution on that data-set. After the proposed algorithm finds a solution, same fitness function for the GA is run on the solution to get its fitness. The last column shows the GA's average fitness on that data-set. It is an average number, because GA searches solution space and it can give different solutions for every run with limited generations. The fitness of GA represents the average of 10 results for the same data-set.

Table 5.3: Results of the proposed algorithm and GA's run on the different BPP data-sets.

Data-set file	Fitness Range (0 is optimal)	Proposed Algorithm's fitness	Genetic Algorithm's fitness
"N1C1W1_A.BPP"	[-25,0]	0	-0,00008
"N1C1W2_A.BPP"	[-29,0]	0	-0,000001
"N1C1W4_A.BPP"	[-35,0]	0	-0,00234
"N1C2W1_A.BPP"	[-21,0]	0	-0,000001
"N1C2W2_A.BPP"	[-24,0]	0	-0,00224
"N1C2W4_A.BPP"	[-29,0]	0	-0,01624
"N1C3W1_A.BPP"	[-16,0]	-0,00029	-0,00018
"N1C3W2_A.BPP"	[-19,0]	0	-0,00019
"N1C3W4_A.BPP"	[-21,0]	-0,00059	-0,00355
"N2C1W1_A.BPP"	[-48,0]	0	-0,00899
"N2C1W2_A.BPP"	[-64,0]	0	-0,00876
"N2C1W4_A.BPP"	[-73,0]	0	-0,00026
"N2C2W1_A.BPP"	[-42,0]	0	-0,00453
"N2C2W2_A.BPP"	[-52,0]	0	-0,00204
"N2C2W4_A.BPP"	[-57,0]	0	-0,01988
"N2C3W1_A.BPP"	[-35,0]	0	-0,01665
"N2C3W2_A.BPP"	[-41,0]	-0,00031	-0,01125
"N2C3W4_A.BPP"	[-43,0]	-0,00345	-0,02888
"N3C1W1_A.BPP"	[-105,0]	-0,00019	-0,00151
"N4C1W1_A.BPP"	[-240,0]	-0,00099	-0.08462

From the table, it can be seen that the results for different data-sets are also promising. The proposed algorithm found optimal solution in the 70% of the tests. Other results are very close to the optimal too and can be adjusted manually in need, for real cases. GA results may seem very close to optimal but not an optimal solution. However it is explained that, those results are average results from 10 runs on the same data-set. GA is a meta-heuristic and depending on various variables it can give very close but different results for every run. Both algorithms perform well on the data-sets, but for real cases proposed algorithm is more acceptable. The nurse assignment phase of the NSP is done in every shift of nurses. Therefore it is essential for the head-nurse that, these assignments are done in advance at the start of a shift. The proposed algorithm finds a solution in 5ms even with the big values of nurse and patients (240 nurses, 500 patients). On the other hand, GA does search in the solution space and must work generation by generation due to its nature. It takes several hours to get a solution after 40000 generations for larger data-sets, which is not acceptable assignment time for a shift.

In the next chapter, further applications and research on the subject is discussed.

Chapter 6

Future Work

This chapter discusses about the improvements can be done on the algorithm and changing the demoware into an efficient application. In the sample hospital results, it is shown that the algorithm gives promising results. However, there is an exception to that. Some of the results create unequal workload balance with overloaded nurses. This is due to acuity levels. They are 1, 5 or 10. Because of that, in some cases average overload is calculated such a number, which can't be achieved with the distribution of current acuity levels. It is explained in detail in the previous chapter. The first improvement must be on this subject. An overload counter system may be implemented for long term solution. Therefore, when a nurse becomes overloaded, it will be the last one to get exceptional patients on the next run. Exceptional patients are the ones that can't be assigned due to their acuity levels make all nurses workloads higher than the average. This can solve the unequal workload distribution not on a run but between runs. Besides, GA created for comparison must be improved for that matter. It will help to find optimal solutions for these exceptional cases of the proposed algorithm. Furthermore, there are different situations which are appropriate to be solved by proposed algorithm's approach. For instance, scheduling doctors based on this methodology can be investigated.

Another important future work will be on the demoware. It can be turned into an application which can be part of a nurse call system. Lot of new features can be added that will help nurses. User interface of the demoware is already designed in a way to be improved. It is not a simple click and sees results for demonstration. There are five grids with different purposes. In addition to them a login system for security can be the first step. Only head nurses who are defined to the system can use the system with that. Moreover, other buttons can be added to the main screen. Add Patient or Edit Patient Data like button will be useful for the head nurse in emergency cases. Of course other editable features will give the application some flexibility. These can change data on the application independent from the algorithm. Right now, algorithm stops when it can't reach the data needed. With this separation, algorithm will be ready and updated within the given interval or when the head nurse needs. Besides, these features give an opportunity for unexpected cases like workload change. If a nurse needs to have less workload because of his/her personal problems, the head nurse can change its initial workload by increasing it. Thus, algorithm assigns fewer patients to that nurse.

The fifth grid mentioned in chapter 4 also, gains importance when turning demoware into an application. Detailed patient information can be shown in that grid. When user clicks a patient from the Patient Grid, the fifth grid will display patient's all properties with his/her medical history and what care does he/she need. More, new wireless devices can be introduced the system. These devices will give important specific data about the patient. Some can give it real time and others can give it on the interval basis. Data from the devices may be: blood pressure, heart rate, respiration rate and skin temperature or a ward specific data. Suggested devices for this feature are included in Appendix A.

With all the data from improved aspects of the application, the whiteboard used in nurse desks can be removed too. Currently, nurses write which patient needs what to be done at the current shift on these whiteboards. However, after all new components added and fast assignment is done by application, human based error will be decreased and the whiteboards may change to an electronic one to be updated automatically.

Lastly, the algorithm can be expanded one by one for all phases of Nurse

Scheduling Problem. It can be used to solve the problem starting from the micro problems it includes. First, algorithm may be changed in a way that it creates day and night shifts too. Then it can solve the schedule on a daily basis. Next it can be improved to include monthly schedules. If an approach on the financial phase of the NSP can be combined with this, NSP solving algorithm based on algorithm created in this thesis can be created. Of course, all improvements together need lot of time and budget. Regardless, it can be a relatively efficient and all-in-one system for Nurse Scheduler and Assistant System.

Chapter 7

Conclusion

Patient care in a hospital ward is very important. Nurses are at the center of it. It is crucial to create effective staff assignments for the nurses in the hospital. NSP is tried to be solved to create this assignments. There are different approaches on how to solve it but it is difficult due to NSP is a NP-Complete. Usually solving the problem as a whole with all of its aspects is main stream. However, it has many micro problems in it. They may be financial or nurse specific or other types. Solving the problem phase based approach is easier to maintain these micro problems. This thesis work is to propose an algorithm to solve the last phase of the phase based approach. Assigning nurses to patients equally is very important to create a base to solve this. Currently, different methods are used to assign nurses to patients but they are focused on the nurse conditions or hospital conditions mostly. Furthermore, this is done only by academic, in real life cases head nurses usually assigns nurses to patients. This can create even more unequal workload balance.

The proposed algorithm connects to a Hospital Information ADT system to fetch the bed and patient acuity levels using HL7 messaging standards. Then using the data, algorithm assigns nurses to patients. Moreover, a demoware is created to show how algorithm works on real life. It ran on a real sample hospital ward with 43 beds, 6 nurses and 34 patients where nurses assigned to patients by hand. The patients beds and acuity levels are randomized for every run. In the end, promising results have been seen which reduce administrative errors and make fast and equal workload distributions. Also, a Genetic Algorithm is coded for comparison to proposed algorithm. It shows how well the proposed algorithm performs on different situations besides the sample hospital. For that, Bin Packing Problem data-sets are used in a way to represent our problem. The results of the both algorithms and their comparison shows that proposed algorithm can give promising results under several different conditions and it is much faster than a meta-heuristic one. Therefore, it can be used for real cases with 30% chance of manual adjusting. Even if it seems much, this adjustment can be handled within a few seconds because it is such a small difference from the optimal, see Table 5.3 again. The demoware can be improved with different modules and devices for nurses too.

All in all, the proposed algorithm combined with the demoware proved a good workload balance for the sample ward can be reached. With step by step improvement of algorithm to include all phases of a NSP and same step by step improvement approach on the demoware, a very effective system can be built for patient care. Naturally, more effort and budget is needed to create this but every important system requires a little first step to become alive. This thesis may contribute to one of them.

Appendix A

Suggested Devices

The devices below can measure important data from a patient and can send that with wireless communication to the improved application of the proposed algorithm. More, they can do that in real-time as a stream, if it is needed.

Figure A.1: Bioharness-5 system from Biopac. Works with Bluetooth and measures Heart Rate, ECG and more [30].

Figure A.2: Bionomadix 10/20 EEG cap is for transfering the patient's live EEG data with wireless communication [31].

BIBLIOGRAPHY

[1] "Nursing Shortage." http://www.aacn.nche.edu/media-relations/fact-sheets/nursing-shortage, Accessed: 2015-02-17.

[2] T. Cooper and J. Kingston, "The complexity of timetable construction problems," in *Selected Papers from the First International Conference on Practice and Theory of Automated Timetabling*, (London, UK, UK), pp. 283–295, Springer-Verlag, 1996.

[3] G. White, B. Xie, and S. Zonjic, "Using tabu search with longer-term memory and relaxation to create examination timetables," *European Journal of Operational Research*, vol. 153, pp. 80 – 91, 2004.

[4] S. Yang and S. Jat, "Genetic Algorithms With Guided and Local Search Strategies for University Course Timetabling," *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICSPART C: APPLICATIONS AND REVIEWS*, vol. 41, no. 1, pp. 93 – 106, 2011.

[5] I. Solos, I. Tassopoulos, and G. Beligiannis, "A Generic Two-Phase Stochastic Variable Neighborhood Approach for Effectively Solving the Nurse Rostering Problem," *Algorithms*, vol. 6, no. 2, pp. 278 – 308, 2013.

[6] U. Aickeln and K. Dowsland, "Exploiting problem structure in a genetic algorithm approach to a nurse rostering problem," *Journal of Scheduling*, vol. 3, no. 3, pp. 139 – 153, 2000.

[7] D. Bradley and J. Martin, "Continuous personnel scheduling algorithms: a literature review.," *Journal of the Society for Health Systems*, vol. 2, no. 2, pp. 8 – 23, 1991.

[8] R. Hung, "Hospital Nurse Scheduling," *Journal of Nursing Administration*, vol. 25, pp. 21 – 23, 1995.

[9] D. Sitompul and S. Rhandawa, "Nurse scheduling models: a state-of-the-art review," *Journal of the Society for Health Systems*, vol. 2, no. 1, pp. 62 – 72, 1990.

[10] J. Holland, *Adaptation in Natural and Artificial Systems*. A Bradford Book, reprint edition ed., 1992.

[11] D. Fogel, *Evolutionary Computation: The Fossil Record*. Wiley-IEEE Press, 1 ed., 1998.

[12] U. Aickeln and K. Dowsland, "An indirect genetic algorithm for a nurse-scheduling problem," *Comput. Oper. Res.*, vol. 31, pp. 761–778, Apr. 2004.

[13] U. Aickeln and K. Dowsland, "Nurse Scheduling with Tabu Search and strategic oscillation," *European Journal of Operational Research*, vol. 106, pp. 393 – 407, 1998.

[14] P. Punnakitikashem, J. Rosenberger, and D. Behan, "Stochastic programming for nurse assignment," *Comput. Optim. Appl.*, vol. 40, pp. 321–349, July 2008.

[15] D. McGonigle and K. Mastrian, *Nursing informatics and the foundation of knowledge*. Jones and Bartlett Learning, 2 ed., 2011.

[16] D. Pittet, E. Safran, S. Harbarth, F. Borst, P. Copin, P. Rohner, J. Scherrer, and R. Auckenthaler, "Automatic alerts for methicillin-resistant staphylococcus aureus surveillance and control: Role of a hospital information system," *Infection Control and Hospital Epidemiology*, vol. 17, pp. 496–502, 8 1996.

[17] "Health Level Seven." http://www.hl7.org/, Accessed: 2015-04-20.

[18] "HL7 FAQs - Version 2." http://www.hl7.org/about/FAQs/index.cfm?ref=nav, Accessed: 2015-04-20.

[19] "HL7 Overview." http://www.interfaceware.com/blog/components-of-an-hl7-message/, Accessed: 2015-04-24.

[20] "Delimiter Escape Sequences." http://www.interfaceware.com/blog/delimiter-escape-sequences/, Accessed: 2015-04-24.

[21] "HL7 Version 3.0." http://www.interfaceware.com/blog/hl7-version-3-0/, Accessed: 2015-04-25.

[22] "C32 Patient Summary." http://cdatools.org/infocenter/index.jsp?topic=%2Forg.openhealthtools.mdht.uml.cda.hitsp.doc%2Fintroduction%2FOverview.html, 2010.

[23] "Introduction to HL7 Standards." http://www.hl7.org/implement/standards/index.cfm, Accessed: 2015-04-29.

[24] M. Unluturk and C. Atay, "Admission-discharge-transfer gateway interface: A health level 7 application," in *Proceedings of the International Conference on Applied Computer Science*, (Malta), 2010.

[25] "Introduction to WPF." https://msdn.microsoft.com/en-us/library/aa970268%28v=vs.110%29.aspx, Accessed: 2015-05-05.

[26] "WPF Globalization and Localization Overview." WPFGlobalizationandLocalizationOverview, Accessed: 2015-05-05.

[27] "Visual Studio 2013." https://msdn.microsoft.com/en-us/library/dd831853.aspx, Accessed: 2015-05-14.

[28] "Bin Packing Problem Data-Sets." http://www.wiwi.uni-jena.de/Entscheidung/binpp/bin1dat.htm, Accessed: 2015-06-15.

[29] J. E. Baker, "Reducing bias and inefficiency in the selection algorithm," in *Proceedings of the Second International Conference on Genetic Algorithms on Genetic Algorithms and Their Application*, (Hillsdale, NJ, USA), pp. 14–21, L. Erlbaum Associates Inc., 1987.

[30] "Bioharness-5." http://www.biopac.com/Education.asp?SubCatID=192, Accessed: 2015-05-15.

[31] "Bionomadix 10/20 EEG Cap System." http://www.biopac.com/BioNomadix-electrode-cap-system-touchproof, Accessed: 2015-05-15.