SUBSTITUTE RESOURCE ALLOCATION POLICIES FOR PREVENTING RESOURCE SHORTAGES IN HEALTHCARE APPLICATIONS

A Dissertation Presented

By

Nooshin Valibeig

to

The Department of Mechanical and Industrial Engineering

in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy

In the field of

Industrial Engineering

Northeastern University
Boston, Massachusetts

April 2018

ABSTRACT

Allocating limited resources to requests at the time of the request is a common problem in various domains such as bed management, blood transfusion, water resource allocation, and transportation assignment. One of the main challenges in resource allocation is the limited capacity of resources and the high cost of capacity expansion, which leads to resource shortages and failures in satisfying requests in an appropriate period of time and with an appropriate level of quality.

Substitute resource allocation is one solution to resource shortages. In substitute resource allocation an alternative, but most of the time non-preferred, resource type is assigned to the request. The request type and the complexity assignment rules are two main factors which are used for choosing the alternative resource type. In many systems, the quality of the substitute allocation is not as desirable as that of the preferred allocation, but use of substitute allocations can help to prevent unsatisfied requests or to decrease the time to respond to the requests.

This dissertation examines multiple facets of substitute resource allocation in healthcare systems. First, we develop a novel decision mining methodology to identify the rate of substitute resource allocation using historical data from multiple databases. Next, we demonstrate the effect of applying new substitute resource allocation policies for a blood management problem using a Monte Carlo simulation model. Specifically, we examine two resource allocation policies accounting for substitute resource allocation in

the assignment of limited blood inventory, including multiple blood types, in an isolated environment. A two-stage stochastic optimization model is developed to determine the optimal make-up of the initial blood bank in these environments, as defined by the level of inventory for each blood type. Finally, in Chapter 4, we analyze a generalized online assignment problem, inspired by the blood management setting, and characterize the problem features for which the proposed algorithm performs optimally with results matching those from the corresponding offline assignment problem. For those problems not conforming to these features an upper-bound on the performance of the algorithm is derived.

To my mom and my dad

ACKNOWLEDGEMENT

Writing this dissertation was a journey I will cherish my whole life. It was not an individual experience, rather it took place in a social context which includes several people who have been closely with me in this journey, whom I want to thank sincerely.

First of all, I would like to express my deepest gratitude to my Ph.D. advisor, Prof. Jacqueline Griffin for her excellent guidance, constant encouragement, and valuable advice during the past years. Her patience, kindness, and inspiration, have made working with her such a valuable experience for me. I could not imagine having a better advisor and mentor for my graduate study. I would also like to thank my committee members, Prof. Ozlem Ergun and Prof. Muhammad Noor E Alam for their valuable comments and inputs to my research.

I would also like to sincerely thank my friend and my great inspiration, Prof. Arezoo M. Ardekani, where without her help and support, I would have never started this journey. Also, I would like to thank Prof. Shantia Yarahmadian, who has devoted his valuable time to answer my uncountable questions. I am truly grateful to my friend, Prof. Maryam Tabatabaei Motlagh who has emotionally supported me throughout these years.

I am forever thankful, and have never feel so lucky in life to have Fatemeh Gandomi as my best friend, to which she is always there for me, listening to my endless complaints yet, still never fails to understand me, and continue making me laugh even at difficult times.

To Aida Ehyaei and Mohammad Hajian, thank you for the kind friendship you showed me in times of need. And to my officemates, Rana Azghandi, Ann Suhaimi, and Vahab Vahdatzad thank you for making my graduate student life more bearable, and often, fun.

But this dissertation will not be complete, without the continuous support from my immediate family. I am deeply grateful for having a wonderful mother, Ferdos Saberi, who has always been my role model, an influential and independent woman figure for me to look upon. To my loving father, Morteza Valibeig, you are the best teacher I have ever had, your teaching on moral values and ethics has inspired me to keep being good not only to myself, but also to others as well. To my siblings, Neda, Nima and Naazi, who are constant support for me, no matter how far they are, they are always with me through all the strange, and weird experiences I have been. All of you are the beacon of light for me, when it is all dark in the tunnel of life.

Last but not least, I would like to thank the one person, who is not only the love of my life, but my soulmate, my crying shoulder, and most importantly, my husband, Ali Moghaddas, for his undying love, supporting words, and for making my life full of love and joy. Ali Jounam, I was able to bear all the burden, because you have given me strength to hold on, and I am very fortunate to have you on my side.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1 -
Introduction

Allocation of limited resources to requests is a common problem in various domains such as bed management, blood transfusion, water resource allocation, and transportation assignment. Decisions about resource allocation are made daily in most organizations and resource allocation policies can be made and implemented for strategic, tactical or operational level decision making. One of the main challenges in resource allocation is the limited capacity of resources, and correspondingly the high cost of capacity expansion, which can lead to resource shortages and failures to satisfy requests in a timely fashion while ensuring the appropriate quality. Correspondingly, making optimal decisions in resource allocation can lead to significant improvements in system performance.

In cases where resource allocation decisions are made at an operational level, decisions to allocate resources to requests must often be made at the time of the request without exact knowledge of future requests. Such an allocation problem is called an online

resource allocation problem. In online resource allocation problems, the decision maker must make the best decision at the time of the request in order to satisfy the current request while accounting for the ability to be prepared to satisfy upcoming requests. When the requested resource is not available, a substitute resource allocation may be made in which an alternative, but in most cases non-preferred, resource type is assigned to the request. A well-defined substitute resource allocation policy can prevent shortages and unmet requests. The features of the request and complexity assignment rules are two main factors that must be accounted for in choosing the alternative resource type. In many systems, the quality of the substitute allocation is not as desirable as that of the preferred allocation, but can help to prevent unsatisfied requests or to decrease the time to respond to the request.

This dissertation analytically examines the practice of substitute resource allocation and analyzes the performance of policies for online resource allocation problems which allow for substitute resource allocation.

1-1. Research Objectives

Resource allocation is a critical operational decision-making process in many health care settings such as, hospital patient transfers, bed assignment, outpatient room assignment, organ transplant and blood transfusion. The Institute of Medicine (IOM), recommends six aims for improving healthcare quality including efficient and equitable care. Specifically, these recommendations focus on avoiding waste while ensuring the same quality of care among populations with various characteristics (Hughes 2008). This dissertation focuses on healthcare resource allocation at the operational level with the purpose of characterizing

historical resources allocation decisions by implementing a data driven approach and providing insights into how resource allocation decision making can be improved to lead to better future outcomes.

The first objective of this research is to develop methods for characterizing historical practices of substitute resource allocation in a healthcare setting by analyzing and mining multiple historical databases. The second objective is to develop an online substitute resource allocation policy which accounts for equitability in the distribution of resources, in addition to classic measurements of effectiveness and efficiency. This substitute resource allocation policy, which can be employed in real-time, is defined to balance the requirements to satisfy current requests, while also ensuring that resources are available for meeting future demand. The final objective, is to comprehensively analyze and characterize the performance of these allocation policies under a variety of settings.

1-2. Summary of Research Contributions

The contributions of this work are summarized as follows:

(1) We provide a polynomial time algorithm that allows for uncovering and discovering resource allocation decisions regarding choices among a preferred resource and substitute resources within historical databases. The effects of the decisions are stored in two heterogeneous spatio-temporal databases, but not the decisions themselves. This algorithm is demonstrated to be robust to missing data on various system sizes in accuracy, precision, and recall, and follows a logical

trend with regard to time-distorted simulated data. For the bed assignment application, in which time lags between occurring and recording events are expected to be small, this algorithm performs well.

(2) We develop an online blood assignment policy, maximum assignment value (MAV), which aims to prevent blood shortages in isolated environments. This policy defines rules for the assignment of a substitute resource when the preferred resource, in this case defined by the blood type, is not available. The MAV algorithm is simple to implement and is especially appropriate for time sensitive decision-making in resource-deprived settings. A two-stage stochastic model is used to determine the optimal distribution of blood types in the initial blood inventory.

Applying the MAV policy and an inventory greedy assignment (GA) policy on Monte Carlo simulation model, the performance is measured in the average number of red blood cells (RBC) in shortage per capita and the average number of patients facing shortage per capita. The MAV policy significantly overperforms the GA policy in both performance measures even when volume of the blood bank, the distribution of blood types, and the population size changes, demonstrating the robustness of this approach. Also, we demonstrate that strategic use of the optimal blood type distributions in the blood bank, rather than that observed in the population, results in better performance.

(3) In addition to comparison with alternate online policies, we analyze the performance of the MAV algorithm for online resource allocation compared to the optimal offline allocation which assumes a priori knowledge of all requests. We characterize the circumstances and settings under which the MAV algorithm performs as well as the optimal allocation and we define an upper bound on the performance of the MAV algorithm in relation to the underlying problem features.

1-3. Outline

The remainder of this dissertation examines multiple facets of substitute resource allocation in healthcare systems. Chapter 2 describes a newly developed algorithm to identify past decisions made by hospital managers choosing between preferred or substitute resource allocation by harnessing historical data from multiple databases. The analysis of the performance of this algorithm applied to hospital patient flow and census data to determine the bed manager's decision to assign patients to hospital wards is presented.

Chapter 3 presents an algorithm, MAV, which defines rules for substitute assignment of red blood cells to requests in order to prevent shortages in an isolated environment. The effectiveness of the two policies, GA and MAV, at decreasing the unmet requests for blood is examined. To compare these policies, a Monte Carlo simulation model, simulating the features of a primary healthcare center in South Sudan, is developed and the results for the average number of red blood cells (RBC) in shortage per capita and the average number of patients facing shortage per capita are analyzed. A two-stage

stochastic model is used to determine the optimal distribution of blood types in the blood bank.

Chapter 4 presents a generalized online resource allocation problem based on the blood transfusion problem and analyzes the performance of the MAV allocation policy for the goal of minimizing unmet requests. The conditions of the resource allocation problem under which the MAV allocation policy is optimal as compared with the optimal offline resource allocation, is identified. The optimality of MAV under these conditions is proven and an upper bound for the MAV policy performance is presented for problem instances with alternate characteristics.

Chapter 2 -

Uncovering Hidden Resource Allocation Decisions: An Application in Hospital Bed Assignment

In this chapter, we extend the concept of rule-based entity resolution, in the context of resource allocation, to link two heterogeneous spatio-temporal databases and extract information related to non-automated decisions. This problem is motivated by a hospital bed assignment problem and the solution is applied to uncover bed managers' decisions to assign patients to inpatient wards.

2-1. Introduction

Efficient and effective decision making is a key driver to performance in all organizations. Thus analyzing past decisions, their corresponding effects on performance, and procedures for reaching decisions can facilitate development of new and improved decision making policies (Glymour, Madigan et al. 1997). To support this process, decision mining, or

decision point analysis, applies data mining tools to the study of historical decision making data. The goal of decision mining is to extract the rules determining under which circumstances an alternative choice is selected (Rozinat and Aalst 2006) and to suggest rules or policies to support better future decisions (Smirnov, Pashkin et al. 2007).

A key to decision mining is the availability of data from an organization's information systems. Typically, information systems, which are designed to record the evolving state of the system in support of operations and billing, do not provide data regarding two critical phases of the decision making process, (i) problem identification and (ii) alternative development, and instead focus solely on the final solution (Mintzberg, Raisinghani et al. 1976). Often multiple independent databases, with varying structures, are generated to represent various aspects of an organization (Kalashnikov, Mehrotra et al. 2005, Zhao and Ram 2005). Mining these multiple databases, with different semantics and data models, can provide important decision support information (Tomasic, Raschid et al. 1996, Li and Clifton 2000).

The objective of this study is to uncover the decisions made by bed managers in allocating beds in hospital wards to patient requests. The ultimate goal of this research is to use this information to help bed managers to make better decisions in the future. Patients from the emergency department, transferred patients and scheduled patients, depending on their medical condition, request a bed in a specific ward of the hospital. Based on the state of the hospital and bed availability, the bed manager assigns a bed in the requested ward (preferred resource) or assigns the patient to a bed in an overflow ward (substitute resource)

(Griffin, Keskinocak et al. 2012). The decision to assign an overflow ward bed to a patient and the choice of the overflow ward is made by the bed manager. In most hospital information systems, the outcome of the decision, or the occupied ward, is specified but the requested ward and the type of allocation, preferred or substitute, is not recorded. To support better bed assignments in the future through a retrospective analysis, it is necessary to uncover the past decisions made by bed managers.

Similar problems exist in other applications where shared scarce resources of different types are available and upon the unavailability of one resource type, alternate types of resources can be assigned to the requests. For example, in a car rental company, at the time of pick-up, if the requested car is not available the employee may decide to assign a car of a similar class or a different class, often at a higher cost, to the customer's request. Knowledge of the requested class and assigned car allows for analyzing the past decisions made by the employees. This analysis has potential to help the car rental managers to develop better allocation policies to improve their services. The allocation of limited storage space to different goods in inventory management and customer choices of substitute brands over their preferred brand in supermarkets are other examples of shared scarce resource allocation that can be analyzed and better understood through decision mining techniques.

To tackle this class of problems, we define a general problem in the context of resource allocation to uncover the relationship between the identification phase and the final decision of the decision making process. In this setting, the decision maker chooses

to allocate a preferred or a substitute resource for each allocation request. It is assumed that the preferred resource is not recorded or represented in the databases. Consequently, without knowledge of the preferred resource, an analysis of past decisions is impossible. We denote resources in resource allocations problem as the spatial features which through time-dependent multi-criteria decision making are assigned to requests for resources (Aerts and Heuvelink 2002). The primary contribution of this study is the development of a polynomial-time decision mining algorithm to extract information from two heterogeneous spatio-temporal databases. In this algorithm we utilize an entity resolution concept to detect which records in the database denote the same real-world entity (Benjelloun, Garcia-Molina et al. 2009, Whang, Menestrina et al. 2009). This algorithm discovers new features of these databases through a procedure of linking two heterogeneous databases, one event-oriented and one snapshot database, and transferring extracted features. The strength of this algorithm is its performance on integrity issues, both in spatial and temporal manners. The algorithm is demonstrated to be robust to missing data and to efficiently handle temporal inaccuracies, defined through the time lags between occurrence and recording events.

In Section 2-2 we present a brief review of past research pertaining to spatio-temporal databases, multiple heterogeneous database mining, and entity resolution. In Section 2-3 we introduce and formulate the problem definition. In Section 2-4, we define and describe the RESOLVE algorithm. In Section 2-5, through a numerical experiment we analyze the performance of the proposed methods and test the robustness of the algorithm to varying types of noise in the data. In Section 2-6, we apply the algorithm on real-world

bed assignment databases and in the final section, we summarize the conclusions of the work and identify areas for future research.

2-2. Literature Review

Many researchers study the causes of emergency department (ED) overcrowding and explore the role of bed managers to prevent ED inpatient boarding and, as a result, decrease waiting times in the ED. Several independent information systems in the hospitals populate heterogeneous databases representing the states and processes of the patients and the beds. Despite the storage of a wealth of data, the information pertaining to bed managers' decisions are not captured in most cases. We study spatio-temporal heterogeneous database mining to uncover past decisions of bed managers. A review of relevant literature in hospital bed management and spatio-temporal heterogeneous database mining, is provided below.

2-2.1. Hospital bed management

ED overcrowding is a critical problem in hospitals. Overcrowding can affect patients' access to necessary care at the proper time (Hoot and Aronsky 2008, Peck, Gaehde et al. 2013). Asplin et al. (2003) study overcrowding and develop a systemic conceptual model. They present the overcrowding as a function of input, throughput, and output components (Asplin, Magid et al. 2003). Hoot and Aronsky (2008) utilize Asplin's conceptual model to study the causes, effects and solutions of overcrowding in the literature. They list hospital bed shortages and inpatient boarding as two of the main causes of overcrowding

and characterize the solutions in three categories: increased resources, demand management and operations research (Hoot and Aronsky 2008). Inpatient boarding occurs when a patient requests admission to an inpatient unit but stays in the ED because there is no capacity available for the patient in the inpatient unit. Some of the recommended interventions to decrease overcrowding are increasing hospital capacity, improving bed utilization, changing elective admission policies, sending patients to overflow wards (Morris, Boyle et al. 2012), transferring patients to inpatient hallways (Viccellio, Santora et al. 2009), and increasing the number of beds in the intensive care unit (ICU) (Olshaker and Rathlev 2006). When a patient requests a bed in a specific ward but occupies a bed in another ward, the occupied ward is denoted as the overflow ward and the patient as an overflow patient (Griffin, Keskinocak et al. 2012).

The aim of most of the interventions to decrease overcrowding is the transfer of boarding patients to inpatient wards. A low capacity of inpatient beds is a significant limitation to moving patients (Asplin, Magid et al. 2003). Beds are critical limited resources in hospitals (Getoor and Machanavajjhala 2012, Hall 2012, Spinella, Dunne et al. 2012, Schmidt, Geisler et al. 2013) and large financial resources are required to equip beds to facilitate care for patients (Litvak and Bisognano 2011, Vancroonenburg, De Causmaecker et al. 2012). A bed manager's challenge is to keep enough beds available for emergency admissions to decrease patient boarding and to assign elective patients to beds to increase the occupancy (Boaden, Proudlove et al. 1999, Proudlove, Gordon et al. 2003) while decreasing the rate of elective admission cancelation (Schmidt, Geisler et al. 2013). One of the solutions to address this challenge is to send patients to overflow wards. Once

an ED patient, or their corresponding physician, requests a bed in a specific ward and there is no bed available in the specified ward, the patient may board in the ED until a bed becomes available in that ward or the patient may occupy a bed in another inpatient ward, referred to as an overflow ward. This approach decreases patient boarding and allows more patients to be admitted to the ED and decreases ED overcrowding. However, this approach may also affect the performance of the inpatient wards. Patients occupying an overflow bed may affect coordination (Teow, El-Darzi et al. 2012) and efficiency (Xie, Chou et al. 2014) of the care process. In some cases, occupying an overflow ward increases the patient's length of stay and causes poor patient outcomes (Getoor and Machanavajjhala 2012, Hall 2012, Spinella, Dunne et al. 2012, Teow, El-Darzi et al. 2012).

Correspondingly, analyzing the effects of overflow assignments may help bed managers to design better decision policies for future implementation. To conduct such an analysis, new methods are needed to uncover past decisions of bed managers to send patients to overflow units. This information is not stored in most hospital information systems databases. Thus, in this study we introduce a general resource allocation problem and develop an algorithm to extract non-recorded data in two heterogeneous databases which can be applied to address the bed management application.

2-2.2. Spatio-temporal heterogeneous database mining

While many researchers have studied multiple heterogeneous database mining, the mining of heterogeneous spatio-temporal databases has not been explored. We provide an

overview of current research pertaining to spatio-temporal databases, multiple heterogeneous database mining, and entity resolution below.

2-2.2.1. Spatio-temporal databases

Information systems gather and store many types of data in various databases. Two common types of databases are spatial and temporal databases (Nandal 2013). Temporal databases represent time-related features or attributes (Snodgrass 1986, Chaudhuri 1988). Spatial databases represent geometric, geographic, or space related data features such as size, shape, or location (Güting 1994). Databases representing changes of spatial features over time are known as spatio-temporal databases (Erwig, Güting et al. 1998). Spatio-temporal databases represent the changing nature of the real world in various application domains including transportation, monitoring, and environmental systems (Gutiérrez, Navarro et al. 2005). In this study, we focus on two data models of spatio-temporal databases, (i) snapshot and (ii) event-oriented. In a snapshot data model, a temporal sequence of the spatial state of a system is represented at fixed time intervals. In an event-oriented data model, every event and its components, including time and place of the event occurrence, is represented (Pelekis, Theodoulidis et al. 2004).

Recent spatio-temporal studies focus primarily on pattern recognition in spatio-temporal databases (Celik 2014, Obulesu and Reddy 2014, Turdukulov, Calderon Romero et al. 2014, Zhou, Matteson et al. 2015). Norén et al. (2010) identify interesting temporal patterns according to the incidence of medical events on first prescription of a specific drug (Norén, Hopstadius et al. 2010). High computational cost is one of the challenges of spatio-

temporal data mining. Hean et al. (2007) note the progressive refinement and the mining of spatio-temporal sequential patterns as two methods to decrease computational costs (Bohannon, Fan et al. 2007, Han, Cheng et al. 2007, Proudlove, Boaden et al. 2007, Yin, Han et al. 2007). While spatio-temporal database studies are focused on pattern recognition, scenario analysis, and search in the large databases, there are no studies of information retrieval by linking heterogeneous spatio-temporal databases. In this study, we apply the concept of entity resolution to link two spatio-temporal databases, and to retrieve the desired information for all entities in the databases in polynomial time.

2-2.2.2. Multiple heterogeneous database mining

The process of extracting unavailable information from databases is referred to as data mining or knowledge discovery in databases. The most commonly applied data mining algorithms, as identified by Wu et al. (2008), have a limited focus on the extraction of information from single databases with single data models (Wu, Kumar et al. 2008). But in many systems multiple distributed databases are generated. Applying traditional data mining algorithms on these databases individually provides some information, but mining the global pattern and retrieving information from the multiple databases, with heterogeneous database models, provides the opportunity for greater knowledge discovery (Zhang, Wu et al. 2003).

One of the simplest, although not the cheapest, solutions for mining multiple homogenous databases is to integrate to one single database and apply a data mining algorithm. This solution may be challenging to implement due to (i) the large size of the

integrated database, (ii) the existence of irrelevant information, and (iii) the loss of patterns and information (Wu and Zhang 2003, Zhang, Wu et al. 2003). Database integration cannot be applied in some problems. For example, one of the challenges in mining temporal databases is the finding of similar time series in two or more databases (Loh, Kim et al. 2004). Researchers suggest several methods to overcome these problems in mining multiple homogenous databases. For example, Wu et al. (2003) apply a synthesized weighting shared association rule model that uses the discovered high frequency rules of each database (Wu and Zhang 2003, Zhang, Wu et al. 2003). In another study, Wu et al. (2005) implement database selection and classification in order to reduce the search cost in the databases (Wu, Zhang et al. 2005). Czarnowsky (2006) proposes a simple heuristic approach to reduce database sizes and to apply data mining algorithms on the integrated joined database (Czarnowski and Jędrzejowicz 2006). Zhang et al. (2009) use kernel estimation, a non-linear method, to extract the multi-database global pattern (Zhang, You et al. 2009).

Multiple heterogeneous database mining, in which databases have different underlying data models, is more challenging than multiple homogenous database mining. In multiple heterogeneous database mining two main steps are considered, (i) linking the databases and (ii) transferring the information, a costly process. To address this heterogeneous database mining, Li and Clifton (2000) propose SEMINT, learning metadata, to identify attribute connections in databases (Li and Clifton 2000). For each resource class in the data mining process, Mastroianni et al. (2003) develop a metadata structure (Mastroianni, Talia et al. 2003). Yin and Han (2005) develop regression-based

methods to link databases and transfer information. (Yin and Han 2005). Though Mehenni and Moussaoui (2012) follow Yin and Han's method to identify links between databases, they apply support vector regression for the classification (Mehenni and Moussaoui 2012). When statistical methods and classification are not a good fit for unsupervised learning, a rule-based method to identify the relationship among features can be developed to retrieve information. In this study, we propose applying an entity resolution method to extract the desired information by linking two heterogeneous spatio-temporal databases.

2-2.2.3 Entity resolution.

Entity resolution, or the matching and resolving of records in structured and unstructured databases which refer to the same real-world entity (Benjelloun, Garcia-Molina et al. 2009, Whang, Menestrina et al. 2009), deals with missing information and differences between records referring to the same entity. Developing a fast and accurate entity resolution approach is a significant challenge in data processing fields (Benjelloun, Garcia-Molina et al. 2009, Getoor and Machanavajjhala 2012).

Approaches in entity resolution (ER) include the rule-based method, pair-wise classification, clustering approaches, and richer forms of probabilistic inference (Getoor and Machanavajjhala 2012). Pair-wise classification, the traditional ER method, is based on the assumption of similarity of records which refer to the same entity (Brizan and Tansel 2015, Chinnaswamy, Gopalakrishnan et al. 2015, Li, Li et al. 2015, Pelekis, Frentzos et al. 2015, Wang, Lee et al. 2015). Pairwise ER is best used for joining databases with the same

set of entities (Getoor and Machanavajjhala 2012). The clustering approach is used when each entity is referred to by more than one record (Whang, Marmaros et al. 2013). Records in the same, or different, clusters will refer to the same, or different, entities, respectively (Adewumi, Budlender et al. 2012, Adewumi, Budlender et al. 2012, Beliën and Forcé 2012, Dufourq, Olusanya et al. 2012, Getoor and Machanavajjhala 2012, Henkelman and Rakhorst 2012, Mehenni and Moussaoui 2012, Morris, Boyle et al. 2012, On, Lee et al. 2012, Peck, Benneyan et al. 2012, Pidcoke, Aden et al. 2012, Shi, Chou et al. 2012, Spinella, Dunne et al. 2012, Vancroonenburg, De Causmaecker et al. 2012, Guo, Sun et al. 2014). The key assumption of these current methods is that similarity of records indicates that they refer to the same entity. When this assumption does not hold, in many cases, domain knowledge can be used to develop rules to match records of database and real-world entities (Brizan and Tansel 2015, Chinnaswamy, Gopalakrishnan et al. 2015, Li, Li et al. 2015, Pelekis, Frentzos et al. 2015, Wang, Lee et al. 2015). While some ER studies focus on matching heterogeneous text databases, they do not explore matching heterogeneous spatio-temporal databases. We extend the study of Zhao & Ram (2005) and Li et al. (2015) to utilize rule-based entity resolution by focusing on spatio-temporal databases on the domain of resource allocation to associate entities of different databases and extract new information.

This is the first study to our knowledge that addresses application of entity resolution in multiple heterogeneous database mining on spatio-temporal databases. Unlike previous studies, outlined above, we extract information related to decisions made by decision makers by applying the concept of a rule-based method of entity resolution to

identify the links between the databases. We then extract new features and information about resource allocation decisions.

2-3. Problem Statement

In this section, we first introduce a model of a general resource allocation system. We then present two different databases representing the system's information. Finally, we describe the relationship between the two databases via the definition of a relational algorithm.

Consider a system that processes different types of tasks. As seen in Figure 2-1, each task type requests to seize a preferred resource and the system allocates an appropriate resource to each task to get processed. In this system, each task goes through the events of requesting, seizing and releasing a resource. Allocations may be of two types, preferred or substitute. In *preferred allocation*, the requested resource is allocated to the task. Allocating resources other than the preferred resource to a task is considered a *substitute allocation* and the allocated resource is a substitute resource. We assume that the system favors preferred allocation, and that substitute allocation negatively affects the performance of the system. The decision of preferred or substitute allocation is made based on the state of the system, including the availability of resources.

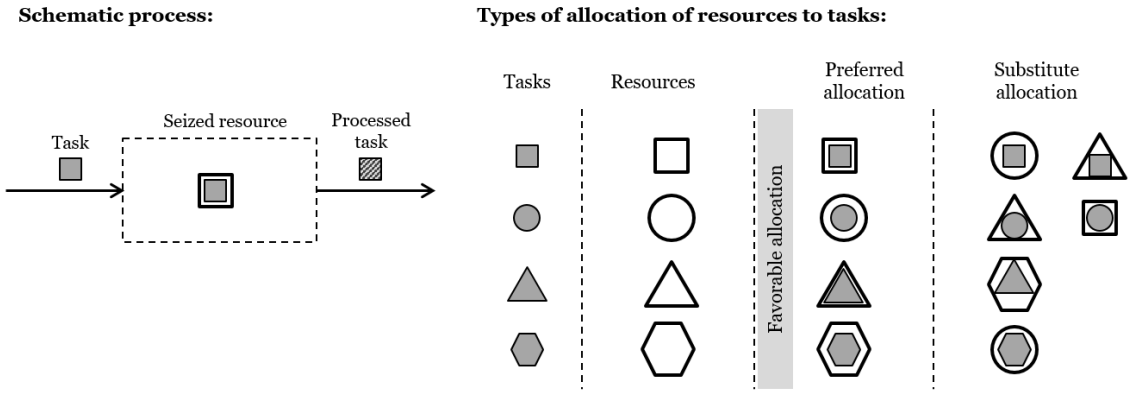

Figure 2-1- Schematic process. The system allocates an appropriate, preferred or substitute, resource to the task for processing. Preferred allocation is favored over substitute allocation.

Considering resource types as spatial features, two heterogeneous spatio-temporal databases from this system are available: an event-oriented database and a snapshot database. The event-oriented database represents task flows and records (i) requesting, (ii) seizing, and (iii) releasing data for each task. The snapshot database records the state of the system at consecutive equal duration time intervals. We refer to these time intervals as *time blocks*. The snapshot data model provides the number of requested and number of allocated resources for each resource type in each time block. The type of allocation, preferred or substitute, is not recorded in the databases. Correspondingly, we wish to extract the type of allocation, preferred or substitute, for each task.

We outline a model and description of the event-oriented and snapshot databases in Sections 3.1 and 3.2, respectively. For these models, assume there are N types of tasks and M types of resources. The total number of units of resource $j, j = \{1, 2, \dots, M\}$ is limited to A_j and refers to the capacity of the resource type j.

2-3.1 Event-oriented database

In the event-oriented database, for each task the time and corresponding allocated resource of each event is recorded and stored. The schematic description of this database and its features is depicted in Figure 2-2. This database for each task i, includes data pertaining to each requesting, seizing and releasing event. The features of each requesting event are the request time for a resource (R_qT), the requested resource (RR_i), which is one of the M types of the resources, and the assigned time (AT_i). The features of each seizing event are the type of seized resource (SR_i) and the seize time (ST_i). Finally the only feature of a releasing event is the release time (RT_i). Note that the released resource is the same as the seized resource and assigned resource. The preferred resource or requested resource of task i (RR_i) is not recorded in the database. Therefore, the type of allocation, preferred or substitute, is not known. We refer to this event-oriented database as the Flow database and define it as follows: $Flow(i, R_qT, RR, AT, ST, SR, RT)$. The key of this database is i, or the taskID. The Flow database is sorted such that the primary sorting key is ascending request time and the secondary sorting key is descending assigned time. As mentioned above, the feature RR, requested resource, is null for all tasks.

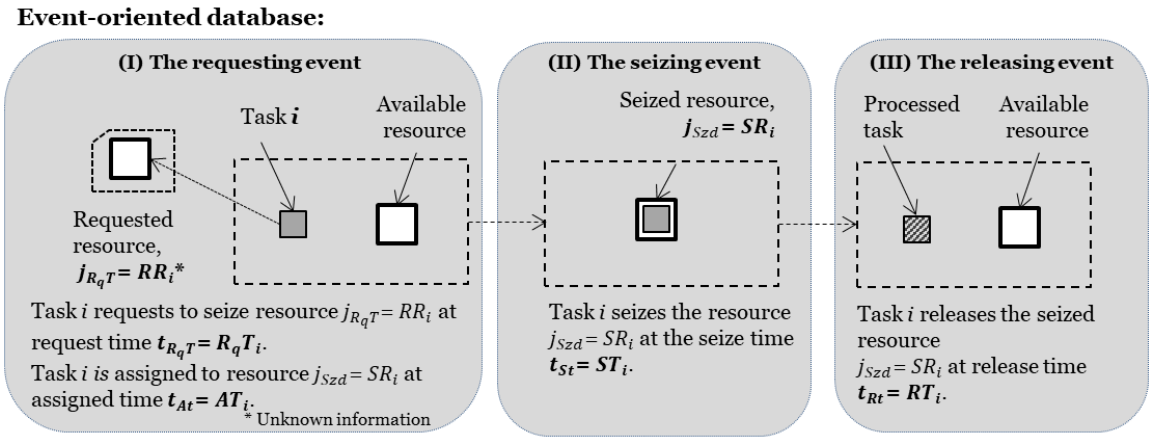

Figure 2-2- Event-oriented database represents the data of a task requesting, seizing and releasing a resource

2-3.2. Snapshot database

The snapshot database represents the state of the system at every time block. The schematic illustration of this database and its features is shown in Figure 2-3. In each time block t for each resource type j, the database tracks the number of requests (NR_{tj}), the number of assigned units (NA_{tj}), and the number of seized units (NS_{tj}). The number of requested units is represented in the database, but the type of allocation is not included in the snapshot database. We refer to this database as the Census database. The Census database is defined as follows: $Census(t, j, NR, NA, NS)$. This is a compound key, (t, j), time and resource type, database.

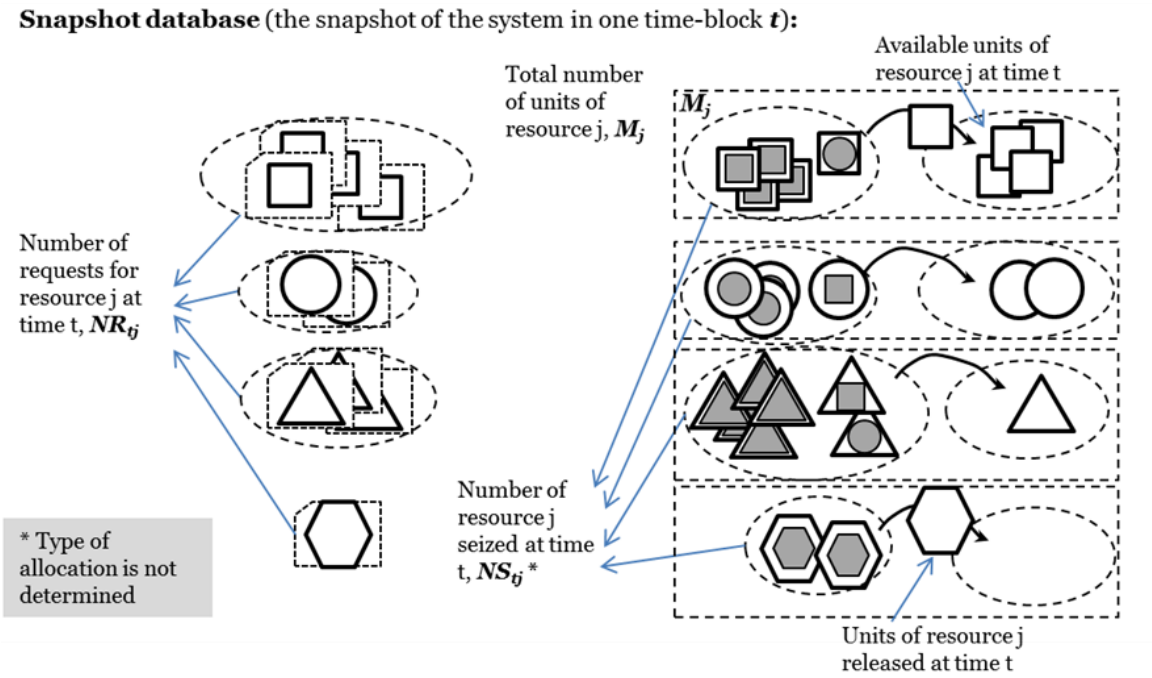

Figure 2-3- Snapshot database represents the number of requests and seized units for each resource at each time block t. The number of seized resources may change when a resource is assigned or released.

2-3.3. Relational algorithm

Although in real world, the Census and Flow databases are generated heterogeneously and the data stored in these databases are recorded by different information systems, theoretically these two databases are strongly interdependent as the Census database can be generated from the Flow database. The relationship between the databases is described as follows. When a request is placed in the Flow database, the number of requests for the requested resource at the time of request increases in the Census database. Similarly, at the assignment time of a resource in the Flow database, the number of assigned units for the assigned resource in the Census database increases. For all time blocks in the interval between the seized and release times of a resource in the Flow database, the number of

seized units of that resource increments. Algorithm 2-1, the Relational algorithm, demonstrates how the Flow database can be used to generate the Census database.

Algorithm 2-1- The Relational algorithm for generating snapshot model database from event-oriented database

Input: Flow database
Initialization (for simulated data):
$\boldsymbol{NR_{tj} = 0, NA_{tj} = 0, NS_{tj} = 0} \quad \forall\, \boldsymbol{t, j}$
Relational algorithm for generating Census database:
for i in Flow database:
$\boldsymbol{t_{R_q t} = R_q T_i}$
$\boldsymbol{t_{St} = ST_i}$
$\boldsymbol{t_{At} = AT_i}$
$\boldsymbol{t_{Rt} = RT_i}$
$\boldsymbol{j_{R_q t} = RR_i}$
$\boldsymbol{j_{Szd} = SR_i}$
$\boldsymbol{NR_{t_{R_q t}\, j_{R_q t}} = NR_{t_{R_q t}\, j_{R_q t}} + 1}$
$\boldsymbol{NA_{t_{AT}\, j_{Szd}} = NA_{t_{AT}\, j_{Szd}} + 1}$
for $\boldsymbol{t_{St} \le t < t_{Rt}}$:
$\boldsymbol{NS_{t\, j_{Szd}} = NS_{t\, j_{Szd}} + 1}$

To generate the Census database from the Flow database the initial state of the system is needed. With real-world data, the state of the system at the starting time ($t = 0$) is used to generate the Census database. When simulating data (see Section 5.1), to set the initial state of the system we assume that for all pairs of t and j, the values of NR_{tj}, NA_{tj}, and NS_{tj} are equal to zero. By using this assumption a warm up period should be considered until the steady state of the system is reached. In the described problem, the Flow database does not provide data concerning the requested resource, RR_i. Therefore, applying the Relational algorithm on the Flow database cannot generate the number of requests for the resources in the Census database.

2-4. Decision Mining Algorithm

We use the underlying relationship described in the Relational algorithms (Alg. 1), to link the heterogeneous Census and Flow databases and to extract information about the requested resource of each task. We define the RESOLVE algorithm which determines the tasks in the Flow database that can be resolved to records in the Census database. Considering the assumptions of the system and applying a defined function named *resolvedFactor*, we identify the allocation type of resolved tasks as preferred or substitute. Also, we determine the requested resources, or a list of potential requested resources, for each resolved task with the substitute allocation. Details of this algorithm are provided below.

2-4.1. The RESOLVE algorithm

The RESOLVE algorithm, resolves records from the Flow database with the Census database. Two event types, d, is defined for each task, requesting event and an assigned event, $d \in \{requesting, assigned\}$. To simplify calling the associated features in the Census database we define number of event d for resource j at time block t in the Census database as, $ND_{t,j}$, this refers to and as number of requests for resource j at time block t, $NR_{t,j}$, and number of assigned units of resource j at time block t, $NA_{t,j}$. Specifically, for each task in the Flow database, for which each has a corresponding requesting event and an assigned event, the function *resolvedFactor* defines the resolution with a particular resource type j during the time interval $[t_1, t_2)$ for the requesting and assigned events in

the Census database. The *resolvedFactor* for requesting events, during time interval $[t_1, t_2)$, for resource type j, is equal to 1 if for all time blocks $t \in [t_1, t_2)$, the number of requests for resource type j is positive. The value of his function is 0, if there exist a time bock $t \in [t_1, t_2)$ and number of requests for resource j at this time block is 0. The *resolvedFactor* for assigned events is defined respectively.

(2-1) $resolvedFactor_{t_1,t_2}^{jd} =$

$$\begin{cases} 1 \;\; if \;\; \forall\, t \; in \; [t_1, t_2), \; ND_{t,j} > 0 \;\; d \in \{requesting, assigned\}, ND_{t,j} \in \{NR_{t,j}, NA_{t,j}\} \\ 0 \;\; if \;\; \exists\, t \; in \; [t_1, t_2), \; ND_{t,j} = 0 \;\; d \in \{requesting, assigned\}, ND_{t,j} \in \{NR_{t,j}, NA_{t,j}\} \end{cases}$$

In the algorithm, first, the *resolvedFactor* for the assigned event of the task is calculated. If the *resolvedFactor* for the assigned event is one, the assignment of the task is resolved such that the task in Flow is resolved to the records in Census. If instead the *resolvedFactor* equals zero, the task pertaining to the Flow database is not resolved to the Census.

For all resolved tasks, the *resolvedFactor* for the requesting event is then calculated. If the *resolvedFactor* for the requesting event equals one, this implies that the task is resolved to the seized resource in Census, and denotes a preferred allocation for the task. If instead the *resolvedFactor* for the requesting event equals zero, this may imply a substitute allocation for the requesting event. Correspondingly, the *resolvedFactor* of the requesting event of the task for all of the other resource types in Census is computed. Among all resources, the ones with a *resolvedFactor* of one are denoted as potential

requested resources of the corresponding task. After resolving each task, Census is updated by applying the *updateCensus* function.

(2-2) $$updateCensus_{t_1,t_2}^{jd} = \left(ND_{t,j} \leftarrow ND_{t,j} - 1\ ,\ \forall\ t\ in\ [t_1, t_2), d \in \{requesting, assigned\}, ND_{tj} \in \{NR_{t,j}, NA_{t,j}\}\right)$$

A detailed definition of the RESOLVE algorithm is provided below (Algorithm 2-2). The output of the algorithm defines each event as resolved or unresolved. Additionally, for each resolved event, the designation of preferred or substitute allocation is provided. For all tasks with substitute allocations, a list of potential requested resources is defined.

Algorithm 2-2- RESOLVE Algorithm: Resolves tasks of the Flow database into the Census database, labels the tasks with an allocation type and develops the list of potential requested resources for substitute allocations.

Resolving tasks in Flow to number of assigned units in Census
input: Flow database, Census database
for tasks in Flow:
- $j = SR_{task}$, $t_1 = AT_{task}$, $t_2 = ST_{task}$
- ***if*** $resolvedFactor_{t_1,t_2}^{jd}$ ***== 1, d = assigned:***
  - ***the task is resolved***
  - $updateCensus_{t_1,t_2}^{jd}$***, d = assigned***
  - $j = SR_{task}$, $t_1 = R_qT_{task}$, $t_2 = AT_{task}$
  - ***if*** $resolvedFactor_{t_1,t_2}^{jd}$ ***== 1, d = requesting:***
    - ***allocation type of the task is preferred allocation***
    - $updateCensus_{t_1,t_2}^{jd}$***, d = requesting***
  - ***else:***
    - ***allocation type of the task is substitute allocation***
    - ***for k in M resource types:***
      - $t_1 = R_qT_{task}$, $t_2 = AT_{task}$
      - ***if*** $resolvedFactor_{t_1,t_2}^{kd}$ ***== 1, d = requesting:***
        - ***add resource k to the list of potential requested resources***
        - $updateCensus_{t_1,t_2}^{kd}$***, d = requesting***
- ***else:***
  - ***the task is not resolved***

output: resolved status, allocation type and potential requested resource list of substitute allocations

2-4.2. The algorithm performance

While the algorithm performs well on simulated data (see Section 2-5), there are two cases in which the output of the algorithm may not provide the accurate allocation type or the requested resource of some tasks. The occurrence of these cases, may cause errors in the corresponding time block as when the case occurs, but has minimal effect on other time blocks.

First, for any task, if both the requesting and assigning events happen in the same time block, the corresponding records of the Census database may not represent this effectively. Therefore, the task cannot be resolved and the algorithm does not provide accurate information about the allocation type of that task. To estimate an upper bound on the effect of this type of error, an analysis of the duration between request and assign times for each task can be calculated and compared with the interval of the time blocks. Reductions in this type of error can be achieved through use of a smaller time block interval.

In the second case, the algorithm may inaccurately designate tasks as alternate and preferred resources when two tasks are assigned to and seize the same resource type and for which both request times are in the same time block. Specifically, the error may occur when the true allocation type of one of the tasks is a substitute allocation while the allocation type of the other task is a preferred allocation. If the task with the substitute allocation is earlier than the task the with preferred allocation in the Flow database, the algorithm may label the allocation type of the first task as a preferred allocation and the

allocation type of the second task as a substitute allocation, incorrectly. The likelihood of such an occurrence is dependent on the underlying system features, but is expected to occur infrequently in practice. Similar to the first type of error, a decrease in time block duration will decrease the likelihood of this occurrence.

Despite these possibilities for misclassification, the RESOLVE algorithm performs well in experiments with simulated data. A description of these experiments is provided in the next section.

2-5. Tests on Simulated Data

In the following section, we demonstrate the effectiveness of the RESOLVE algorithm via a numerical experiment. Specifically, we apply the algorithm to simulated datasets corresponding to bed assignment systems. To test the robustness of the method to system parameters, we vary the number of resource types and the characteristics of the resources when simulating the corresponding Flow and Census databases (Section 5-2). Since missing data and corrupted data are common challenges in real-world datasets, which can affect the effectiveness of the algorithm, we also test the impact of these features in Section 2-5.3.

2-5.1. Generating simulated databases

In generating the simulated databases, we allow the system size to vary between 4 and 50 resource types. Additionally, we assume that the capacity exceeds the expected requests

for each resource and that there are consistent underlying rules, referred to as *the substitute set*, defining which resources may be substituted for each other.

Assumptions for the simulation, as inspired by assumptions in bed assignment applications and analysis of corresponding datasets, include (i) Poisson arrivals of requests, (ii) a Normal distribution for time between request and assignment of a resource, (iii) a Normal distribution for time between assignment and seizure of a resource, and (iv) a Normal distribution for time between seizure and release. Details of the probability distributions are included in Table 2-1. Parameters for these probability distributions are varied to create 5 different system settings. For each system setting, 30 pairs of databases, including Flow and Census, are generated. Warm-up periods are defined to ensure that the databases record steady state activity. The substitute subset for each resource is randomly generated.

Table 2-1- Detailed information about the simulation model input

Variables	Distribution	Mean	Standard deviation
Capacity of each resource, M_j	Constant	Min(1.33*$\frac{RT_i - ST_i}{R_qT_{i_j} - R_qT_{i-1_j}}$, 48)	
The time interval between two (Hours)sequential requests for each resource, $(R_qT_{i_j} - R_qT_{i-1_j})$ (Hours)	Exponential	U[2, 38]	
The time interval between request and assignment of a resource* $(AT_i - R_qT_i)$ (Hours)	Normal	U[0, 3]	Mean/|N(3,1)|
The time interval between assignment and seizure of a resource* $(ST_i - AT_i)$ (Hours)	Normal	U[0, 9]	Mean/|N(3,1)|
The length of time a task seizes a resource $(RT_i - ST_i)$ (Hours)	Normal	U[0, 300]	Mean/|N(3,1)|
Warm up period (Hours)	Constant	Max($(R_qT_{i_j} - R_qT_{i-1_j})$* M_j)	

* With respect to the seized resource type

Based on the system setting parameters, Flow and Census databases are simulated and populated simultaneously using the Relational algorithm (Alg. 1). If at the request time for a resource, all the capacity of the preferred resource is seized, a resource type is chosen, from the substitute subset with equal probability among types, and if there is available capacity a substitute allocation is made. If the selected resource has no available capacity another resource type from the substitute subset is chosen in the same manner. In the unlikely case that all the resource types in the substitute subset are fully seized, a resource type with available capacity, not in the substitute set, is chosen. The simulation time span is one year and each time block in the Census is one hour in length. The number of tasks per database varies between 5,000 and 45,000, depending on the size of the corresponding system.

2-5.2 Testing the algorithm

By simulating the databases, the true preferred resource for each task is known and can be used to evaluate the performance of the RESOLVE algorithm in identifying the type of allocation. For each of the generated databases, we calculate the accuracy, recall, and precision of the algorithm in identifying the allocation type. In this case, accuracy denotes the percentage of tasks with a correctly identified allocation type, preferred or substitute. Recall is the percentage of substitute allocations correctly identified by the algorithm. Precision is the percentage of correct substitute allocations identified by the algorithm. The calculations of these performance measures are:

(2-3) $$\text{Accuracy} = \frac{\text{number of tasks with correctly identified allocation type}}{\text{total number of tasks}} * 100\%$$

(2-4) $$\text{Recall} = \frac{\text{number of tasks correctly identified as substitute allocation}}{\text{total number of tasks with substitute allocations}} * 100\%$$

(2-5) $$\text{Precision} = \frac{\text{number of tasks correctly identified as substitute allocation}}{\text{total number of tasks identified as substitute allocations}} * 100\%$$

Since all datasets are independently generated and normality assumptions are valid, a t-test is applied to assess the performance of the algorithm on these three performance metrics. With a 95% confidence level, the accuracy and recall of the algorithm is greater than 97.4% and 80.1%, respectively, in the simulated databases. Additionally, the accuracy and recall of the algorithm do not change, or correlate, with the system size. The precision of the algorithm is 100% for any number of resource types.

Unlike the designation of the allocation type, or accuracy, the effectiveness of the algorithm to correctly identify the specific requested resources, for substitute allocations, is correlated with the system size, or number of resource types. Identification may include an exact definition or definition as part of the list of potential requested resources. As the size of the system increases the percentage of correctly identified requested resources decreases and the percentage of correctly identified potential requested resources increases (Figure 2-4). With 95% confidence, the average percentage of correctly identified requested resources and the percentage of correctly identified potential requested resources are statistically different for different sized systems. With 95% confidence, regardless of

the system size the sum of these two performance indicators is greater than 99.6% in the simulated datasets.

Increasing the system size, the substitute subset list becomes longer and more tasks share their request time in the same time block. It is due to both of these reasons that less exact definitions and more potential definitions of the requested resources occur as the system size increases. As mentioned above, for all system sizes, the precision of the algorithm is 100% and all allocation types identified as substitute are correct in the simulated data. Therefore, the performance of the algorithm in identifying the requested resource of substitute allocations are not correlated with the performance of algorithm in identifying the allocation type. Regardless of the number of resources, whether through identification of the exact requested resource or the potential requested resource, with 95% confidence, for less than 0.2% of tasks, the requested resource is not defined by the algorithm.

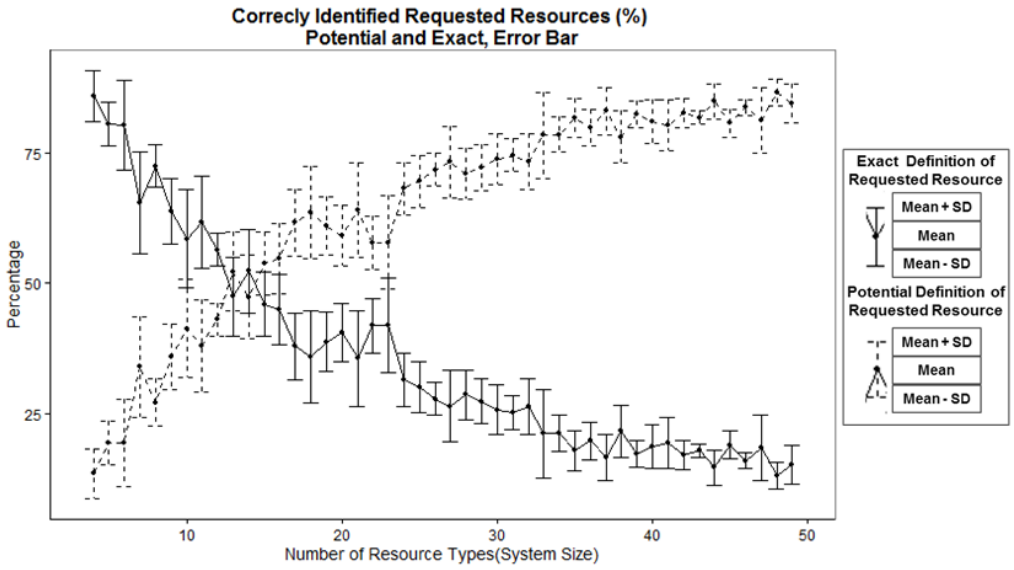

Figure 2-4- Percentage of correctly identified requested resource (one resource or a list of potential resources) in simulated databases with varying system sizes and settings

2-5.3. Missing and corrupted databases

In addition to testing the effectiveness in relation to system size, we also test the effectiveness when there are data inaccuracies, as is often found in real world data. Hence, we analyze the effect of missing and corrupted data on the performance of the algorithm. To simulate missing data, we randomly remove with equal probability selected records from the Flow database while protecting the Census database from any changes and examine the effect of removing increasing amounts of data on the performance of the algorithm.

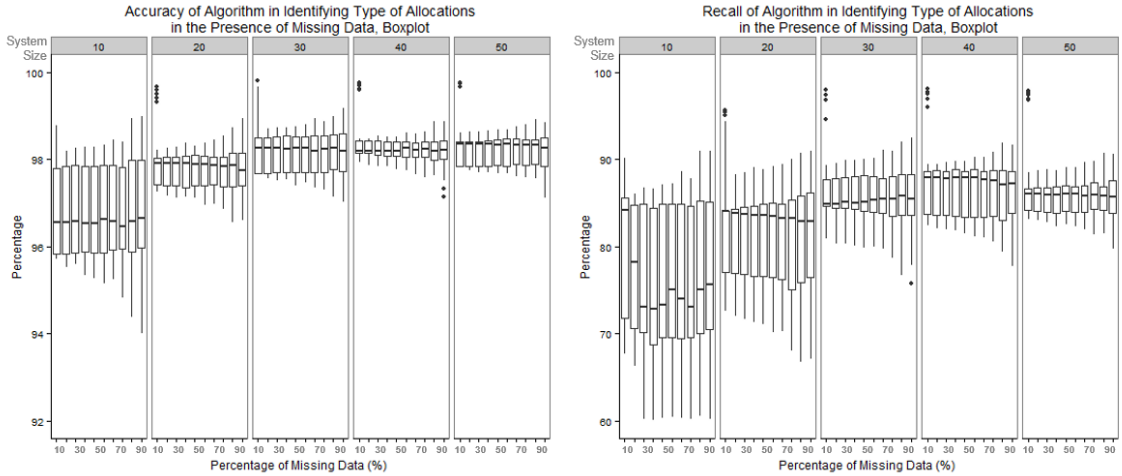

Figure 2-5- Accuracy and recall of algorithm in identifying the type of allocations in the presence of missing data for different sized systems

Five different system sizes are selected with 10, 20, 30, 40 and 50 resource types. For each system size, five different system settings, as defined by the substitute set, are considered. For each setting, there are 30 replications of simulations and from each a fixed percentage of records in the Flow database are removed. The results of applying the algorithm on

databases with different percentages of missing data are shown in Figure 2-5. With the percentage of data removed varying from 10% to 90%, no significant change is observed in the average accuracy, recall, or precision of the substitute allocations identified by the algorithm. Figure 2-6 demonstrates that there is no change in the mean percentage of correctly identified exact requested resources and an increase in the mean rate of identifying potential requested resources with increasing levels of missing data. Regardless of the percentage of missing data, a t-test confirms with 95% confidence that more than 99.6% of requested resources, exact or potential, are identified.

Time inconsistency across databases is a common data corruption problem in information systems, particularly in healthcare organizations. Therefore, we also examine the effect of time lags between valid time and transaction time of events in analyzing the robustness of the algorithm. Valid time refers to the time an event happens and transaction time refers to the time that the event is recorded. Among the transaction times stored in the database, the request time and assigned time are used in the algorithm to identify the allocation type. Therefore, to generate corrupted data, after randomly selecting 20% of tasks, we corrupt the selected data by adding a *time lag* to the request time of the task. In a separate analysis we add the time lag to the assigned time. To evaluate the impact of the size of the time lag, the value is generated randomly using a Normal distribution with means of 15, 30, 45, and 60 minutes.

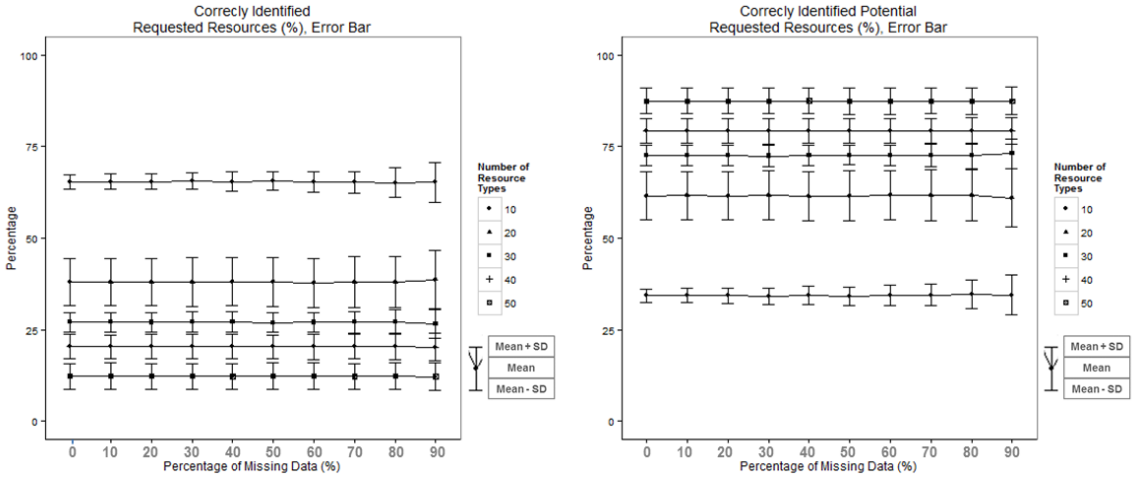

Figure 2-6- Effects of several percentage of missing data on identifying the requested resource in substitute allocations

The effects of the time lag vary depending on where the time lag is applied. A t-test with the confidence of 95% reveals that adding a time lag to the request time of randomly selected tasks does not change the performance of the algorithm. The trend is similar for accuracy and recall.

Unlike lags in request time, for all system sizes, accuracy and recall decrease as assigned time lags increase (Figure 2-7). Despite this behavior, precision of the outcome remains at 100%. Regardless of the size of the system, the mean accuracy falls from 97.4% with no time lag to approximately 93.9% with a one hour time lag. While the accuracy and recall of the algorithm are not robust to the time lag between occurrence and recording the assigned time, the change is not drastic, especially for smaller time lags which are more prevalent.

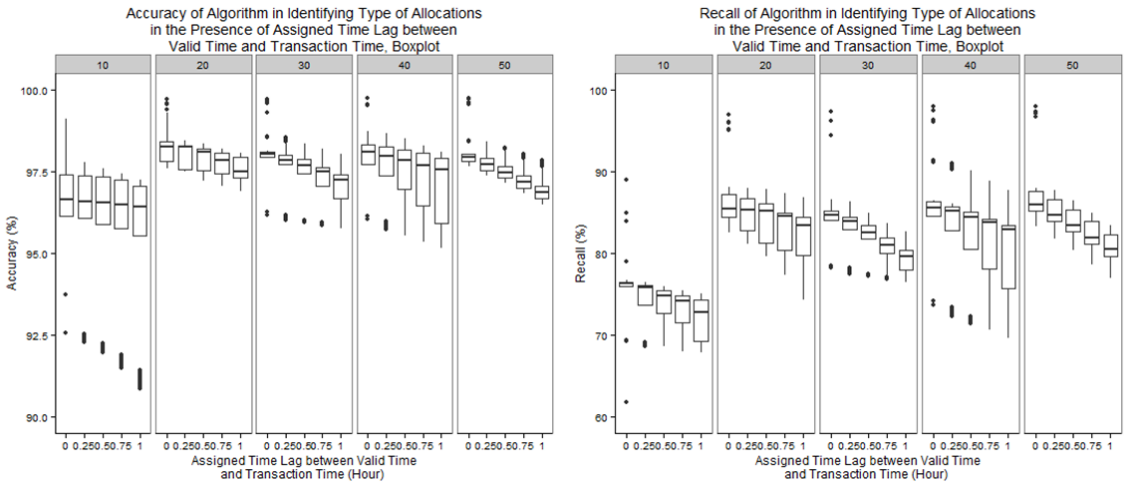

Figure 2-7- Effects of several time lag between transaction time and valid time of a proportion of requests on determining the type of allocation, on different sizes of the system

By increasing the size of the time lag, no change is observed in the percentage of correctly identified requested resources, exact or potential (Figure 2-8). At all system sizes, a t-test with 95% confidence confirms that the effectiveness of the algorithm in identifying the requested resources, exact or potential, is 99.6%, 99.5%, 99.3%, 99.2% and 99.2% when the assigned time lag mean is 0, 15, 30, 45, and 60 minutes, respectively. Considering time lag in recording assigned time for 20% or less percentage of tasks, the performance of algorithm is acceptable, and can be used to dependably extract information and identify past decisions.

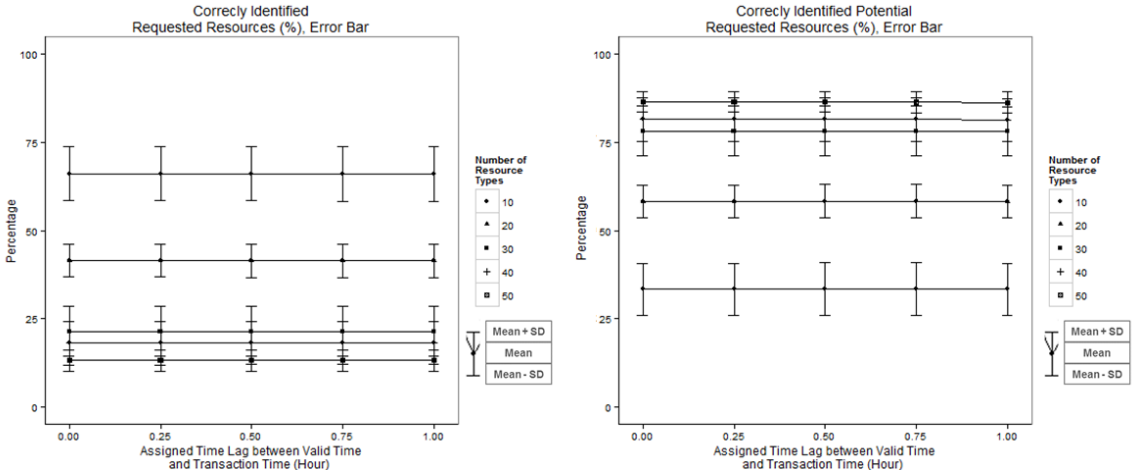

Figure 2-8- Effects of several time lag between transaction time and valid time of a proportion of requests on identifying the requested resource in substitute allocations

2-6. Case Study, Bed Assignment

To demonstrate applicability, we apply the algorithm on a hospital bed assignment problem. The algorithm is applied to historical bed assignment data to identify when a manager chose to assign a patient to an overflow ward, rather than a preferred ward. Specifically, patient flow data from more than 14,500 patients during one year was used. This patient flow data, stored in two separate databases were mapped to construct the event-oriented and snapshot databases after data cleaning. A summary of the fields corresponding to these databases is provided in Table 2-2.

Table 2-2- The equivalent variables in hospital databases with the defined resource allocation problem

	Resource allocation	Bed assignment
	preferred allocation	regular assignment
	substitute allocation	overflow assignment
	substitute resource	overflow ward
Flow Database:		
RQi	request time of task i	request time of patient i
RRi	requested resource of task i	requested ward of patient i
AT_i	assigned time of task i	assigned time of patient i
STi	seize time of task i	occupation time of patient i
SRi	seized resource of task i	occupied ward of patient i
RTi	release time of task i	discharge time of patient i
Census Database:		

NR_{tj}	number of requests for resource j at t^{th} time block	number of requests for ward j at t^{th} time block
NA_{tj}	number of units of resource j assigned to tasks at t^{th} time block	number of beds in ward j assigned to patients at t^{th} time block
NS_{tj}	number of units of resource j seized at t^{th} time block	number of beds in ward j occupied at t^{th} time block

The Flow, or event-oriented, database represents depersonalized data of 14,606 patients, and the Census database provide snapshots of the hospital state for 8,664 consecutive one hour time blocks (the snapshot data of 4 days is not available). For 158 patients, 1.1% of patients in the hospital Flow database, one of the requesting, assigning or seizing events was not recorded.

2-6.1. Resolving patients in Flow database to Census database

The results from applying the RESOLVE algorithm on the hospital data are presented in Table 2-3. As discussed in Section 4.2, the algorithm can produce errors in resolving records of the databases when the request and seizure events occur at the same time block. For 24.3% of patients the request time, and the occupation time are in the same time block. As various studies (Hakre, Manak et al. 2013, Malsby III, Quesada et al. 2013, Shi, Dai et al. 2013, Wang, Lee et al. 2015) support, patients are sent to the overflow ward when there is no bed available at the requested ward and the time waiting to be assigned is significant. Therefore, we assume that if both request and occupation events happen at the same time block, this patient is not an overflow patient and the type assignment is a regular assignment. For the 1.1% of the patients whose request, assigned or occupation time was not recorded, the requested ward and type of assignment cannot be determined. A total of 274 patients are not resolved by the algorithm, and 1,159 patients of the Flow database are

resolved as the overflow patients. The remaining 64.8% of patients are labeled as regular assignment, non-overflow, patients.

Table 2-3- The results of applying RESOLVE algorithm on Hospital Flow and Census databases

	Not Available	Not Resolved	T($R_q T$) = T(ST)	Regular Assignment	Overflow Assignment	Total
Number of Patients	158	274	3546	9468	1160	14606
Percentage of Patients	1.1%	1.9%	24.3%	64.8%	7.9%	100%

Among 1,160 patients with overflow determined assignment, for 358 patients the requested ward is determined and for 368 patients potential requested wards are listed. For the other 434 the output of the algorithm does not reveal any requested ward. While the requested ward was not precisely determined for the majority of overflow patients, the results of this analysis provides the hospital administration with valuable estimates about the rate of overflow, patterns in the use of wards for overflow, and temporal trends in overflow assignments. This is information that has not been available in the past and is information that will be valuable for potential analysis and redesign of policies.

2-7. Conclusion and Future Work

Decision making is a learning process and decisions are made based only on the information that is available. Variations in the input information can be the difference between the right decision and the wrong one (Siemens 2005). Thus, organizations need reliable information to analyze their performance, to evaluate decisions and to predict key variables for making better decisions in the future. Unfortunately, information systems often are structured to result in distributed databases which on their own do not provide

enough information to precisely identify past decisions made in the organizations. Multiple database mining is one solution to extract information about these man-made decisions.

In our study, we use the concept of rule-based entity resolution, in the context of resource allocation, to link two heterogeneous spatio-temporal databases and to extract information about non-automated decisions. The problem is inspired by a hospital bed assignment application and the need to uncover information about bed managers' decisions to send patients to overflow wards. To our knowledge, this is the first application of rule-based entity resolution to link two heterogeneous spatio-temporal databases in order to extract information and uncover past decisions.

Using this rule-based entity resolution approach, we define a polynomial-time algorithm to allow for the uncovering of these hidden and unrecorded decisions. Specifically, the algorithm identifies each resource allocation as preferred or substitute. For those designated as a substitute allocation, an exact definition of the requested resource type or a list of potential requested resources is provided.

The algorithm's accuracy, recall, and precision are tested on a variety of simulated databases. The algorithm is robust to missing data on various system sizes, and follows a logical trend in performance with regard to time distortions in the data. In addition to dependably identifying preferred and substitute allocations, the algorithm identifies the requested resource, exact or potential, for a high percentage of the tasks regardless of the size of the system. However, there is a negative statistical correlation between the percentage of correctly identified requested resources and the system size.

Correspondingly, there is a positive statistical correlation between the percentage of correctly identified potential requested resources and the system size.

Additionally, when examining the effects of inaccuracies in the spatial aspects of the database, we find that the amount missing data has no statistically significant effect on the correct identification of allocation type and the requested resource, exact or potential, for substitute allocations. But when analyzing the effect of inaccuracies in the temporal aspects, the time lag between transaction and valid time of assignment is found to negatively affect the effectiveness of algorithm in identifying the type of allocation. These effects are enhanced by increasing the time lag.

For application of the algorithm on hospital databases, in which there exists data missing and time lags are expected to be small, this algorithm performs well. For example, when the average assigned time lag is 15 minutes for 20% of the data, accuracy and recall measures are above 95% and 75% respectively and the precision is 100%. As the system size increases, accuracy of the algorithm changes very slightly but the recall decreases. If exact identification is required, then this algorithm works well on systems with smaller numbers of resources. As the complexity of the system, and the system size increases, the algorithm performs well in identifying the correct resource within the list of potential resources, but not the exact. Since one of the purposes of this the methodology is to allow for studying past decision making, where previously no information was available, the results of this algorithm and the resulting information are valuable in conducting these analyses. This is supported by the results of the analysis of historical hospital data.

In this study we focused on one crucial assumption, that the system favors preferred allocation, in developing the rule-based algorithm. In practice, sometimes the underlying rules are slightly different or more complex. For future work we suggest considering more complex assumptions such as incorporating the limited capacity of each resource. Additionally, integration of machine learning methods may allow for updates and redefinitions of these rules. This may lead to better results for the time-distorted simulated data. We also suggest development of new data mining techniques to extract the substitute patterns from the output of the algorithm. This pattern can be used to uncover the decision policies to choose a substitute resource for a substitute allocation.

Chapter 3 -

Analysis of Proactive Red Blood Cell Assignment Policies: A Simulation Model of Transfusion in an Isolated Environment

In this chapter, we examine the effectiveness of two blood assignment (resource allocation) policies, greedy assignment (GA) and maximum transfusion value (MTV), for decreasing the unmet requests for blood. To compare these policies, a Monte Carlo simulation model, simulating the features of a primary healthcare center in South Sudan, is developed and the results for the average number of red blood cells (RBC) in shortage per capita and the average number of patients facing shortage per capita are analyzed. A two-stage stochastic model is used to determine the optimal distribution of blood types in the blood bank.

3-1. Introduction

Blood transfusion is a critical process in medical care. It is often vital to patient recovery and, in some cases, prevention of patient fatalities. Correspondingly, the task of managing blood supplies, is critical for reduction of morbidity and mortality in a population. Additionally, the perishable nature of blood and life threatening risks associated with inappropriate blood product transfusion makes blood management challenging (Li, Dong et al. 2008). One of the key tasks of blood management corresponds to the critical decision to assign available compatible blood resources to patients' requests. In order to provide a safe and reliable blood assignment, ABO and Rh compatibility of the patient's blood with the assigned blood units should be tested and accounted for. An ABO incompatible blood transfusion can have dire consequences including intravascular haemolysis, haemolytic transfusion reactions, and haemolytic disease (Klein and Anstee 2008). Correspondingly, beyond a simple need for blood availability, it is necessary to have precisely compatible supplies for each patient to gain the benefits of transfusion.

One of the key considerations in blood management is the shelf life of blood and blood products. Thus a key challenge in blood banking is the need to keep enough blood units of all blood types to appropriately respond to demand while, simultaneously, minimizing the stock of outdated blood and blood products due to cost and ethical concerns (Van Dijk, Haijema et al. 2009). As a result of uncertainties about future demand, identifying the appropriate levels of each blood type to store in a blood bank can be difficult. Correspondingly, seasonal and acute blood shortages are critical problems in US

hospitals and blood banking systems (McCarthy 2007, Erickson, Champion et al. 2008, Klein and Anstee 2008).

Although blood shortages can not be eliminated, their frequency can be minimized (Van Dijk, Haijema et al. 2009). A variety of policies and products have been proposed to address this challenge. Joseph et al. (2009) note that not all blood transfusions, in a particular teaching hospital, were neccesary and that it is possible to conserve blood by controling transfusions to medical patients (Joseph, Hendry et al. 2009). McCary (2007) suggests prioritizing patients with legitimate hemotherapy needs for blood transfusions and communicating with hospital stakeholders at the time of blood shortages (McCarthy 2007). Gould et al.(2002) suggest using human polymerized hemoglobin as a blood substitute in massive blood transfusion (Gould, Moore et al. 2002). Alternatively, Bhangu et al.(2013) propose using salvaged blood in combat related injuries as a successful solution in gunshot wounds and cavity injuries (Bhangu, Nepogodiev et al. 2013). Despite these developments in blood conservation strategies, blood shortages continue to occur regularly and there exists an opportunity to improve the policies and practices in blood transfusion.

This need for new policy developments is particularly critical in some isolated African rural areas, in which hemorrhage is one of the major causes of maternal death (Buor and Bream 2004) and blood availability is extremely important to save lives. Confounding this issue is the fact that blood management can be significantly more difficult in rural isolated environments than in developed areas. Due to the high prevalence of anemia, mostly caused by malaria or pregnancy-related challenges, blood transfusion

has become one of the most effective medical practices to reduce the number of lives lost (Fleming 1997, Erhabor and Adias 2012). Studies claim that lack of blood for transfusion is the cause of 26% maternal haemorrhage deaths in sub-Saharan Africa (Buor and Bream 2004). Bates et al. (2008) note that almost 80% of blood used for transfusion is provided by replacement donors, either family members or paid donors (Bates, Chapotera et al. 2008). The cost and time to collect and screen blood of replacement donors is high but essential due to the high risk of transfusion of unscreened blood in transmission of HIV, HBV, and HCV infections (Allain, Owusu-Ofori et al. 2004). Since the risk and the cost of replacement donors and blood shortages is higher in this environment, it is essential to develop an appropriate blood assignment policy to preserve enough blood units to respond to all requests.

The World Health Organization (WHO) provides several resolutions for African member states to help them minimizing the risk of blood transfusion while providing adequate amounts of safe blood to satisfy the population's need for blood. According to WHO guidelines, 10 units of blood per 1000 population should be collected in a year. In 2010, on average only 4.3 units of blood per 1000 individuals in the population were collected for WHO African regions (WorldHealthOrganization 2014). Thus, there is significant shortfall of blood units to treat patients.

Timeliness of blood availability is critical since there is a high mortality risk among children who visit health centers with severe anaemia during first two and a half hour of their visit. Hospitals in three East African countries, typically transfuse an inadequate

volume of blood to children with anaemia. Although this policy preserves blood to serve more children with severe conditions, it also leads to a high re-transfusion rate in these hospitals (Kiguli, Maitland et al. 2015). Bates et. al (2008) after studying research and reports on obstetric haemorrhage and maternal deaths in Sub-Saharan Africa concluded that blood shortages is a major cause of maternal death and their main suggestion, beside blood donation, is to manage blood systems more efficiently (Bates, Chapotera et al. 2008).

The need for new policy developments is also particularly critical in combat environments, in which massive blood loss by military members and civilians is a major cause of death (Henkelman and Rakhorst 2012). Specifically, in this setting similar to the isolated environments, demand for each blood type is unpredictable and the timing and quantity of blood stock replenishment is unreliable. Although for blood bank facilities or hospitals a robust blood supply chain network minimizes the risk of unavailability of blood, a dramatic increase in request for blood leads to a serious risk of the inability to meet the requests. Specifically, the risk is high in the aftermath of events on islands where sending blood after mass casualty may not by logistically possible. Such incidents happened in Puerto Rico and the Virgin Islands after Irma and Maria hurricanes in 2017.

Beyond a study of past practices, researchers examine new approaches in blood management practices and test their solutions via computer simulations. Van Dijk et al. (2009) suggest a stochastic dynamic programming and simulated model to find a nearly optimal order-up-to level for blood products (Van Dijk, Haijema et al. 2009). Asllani et al. (2014) develop a simulation-based decision model for blood products inventory

management. They test the performance of the approach under scenarios in which demand is uncertain and uncontrollable (Asllani, Culler et al. 2014). Simonetti et al. (2014) suggest a first-in first-out (FIFO) blood management strategy to address average storage time. The use of this strategy is supported by results of a simulation based on the Center for Medicare and Medicaid Service's blood use data (Simonetti, Forshee et al. 2014).

Most of these aforementioned studies primarily focus on the identification of high-level policies for inventory management or actions to be taken when facing a blood shortage. These studies fail to provide a strategy for management of inventory at the transfusion level, which would allow for the proactive prevention of shortages. Correspondingly, the goal of this study is to explore how simple blood assignment policies, which account for blood type compatibility, can result in the prevention of blood shortages. More specifically, via a simulation model, we analyze the effect of new policies of blood management which account for the compatibility of blood type substitution with the goal of minimizing future shortages and preventing unmet demand.

3-2. Materials and Methods

Employing a reliable blood assignment policy is crucial to prevent loss of lives in rural isolated environments. With particular consideration for children with severe anaemia and pregnant women in labor, medical teams at health center locations must be able to respond within 2 hours of the request for blood (Bates, Chapotera et al. 2008, Kiguli, Maitland et al. 2015), while ensuring they are properly prepared for additional emergency situations in the future. Due to the need to make quick decisions, it is imperative that a medical team

has a simple and consistent blood assignment protocol that can be quickly implemented. In this study, we examine two such easily implementable policies for blood assignment in rural isolated environments. We compare the performance of these two policies, as defined by the prevention of lives lost and blood shortages, via a simulation of a resource-constrained location in South Sudan. While the first policy only considers the current available inventory of each blood type, the second policy considers the effects of a current assignment decision on the likelihood of fulfilling future blood requests.

3-3. Blood assignment policies

Blood assignment is the process of assigning appropriate blood units of a compatible blood type to a blood request. As shown in Figure 3-1, the set of compatible assignments can be described as an asymmetric matrix in which each blood type can be assigned to one, some, or all other blood types. For example, blood type O^- is the universal donor and is compatible to be assigned to any blood type. Conversely, all blood types are compatible to be assigned to AB^+, the universal receiver, while AB^+, is only compatible to be assigned to AB^+. Using this concept, Adewumi et al. (2012) define a transfusion value (TV) for each blood type. Considering one blood type as the requested blood type, the transfusion value is the ratio of the number of blood types compatible to be assigned to this requested blood type and the total number of compatible transfusions between all blood types, which is 27 (Adewumi, Budlender et al. 2012). Applying this definition, O^- has the lowest TV, at $\frac{1}{27}$. Likewise, the TV of AB^+ is $\frac{8}{27}$, the largest TV among all blood types (Figure 3-1).

Compatible blood type

Requested blood type	O^+	A^+	B^+	AB^+	O^-	A^-	B^-	AB^-	TV
O^+	✓				✓				$\frac{2}{27}$
A^+	✓	✓			✓	✓			$\frac{4}{27}$
B^+	✓		✓		✓		✓		$\frac{4}{27}$
AB^+	✓	✓	✓	✓	✓	✓	✓	✓	$\frac{8}{27}$
O^-					✓				$\frac{1}{27}$
A^-					✓	✓			$\frac{2}{27}$
B^-					✓		✓		$\frac{2}{27}$
AB^-					✓	✓	✓	✓	$\frac{4}{27}$

Figure 3-1- Transfusion value (TV) of each blood type, shows the ratio of number of compatible blood types for transfusion and the total number of compatible transfusions among all blood types to the specified blood type

Two policies for blood assignment are examined in this study. The first, and simplest policy, is denoted as the real-time greedy assignment (GA) policy. To implement the real-time GA policy, among all compatible blood types including the requested type, the type with highest inventory is chosen. While there is value in assigning blood type with higher inventory level, this policy can lead to possible shortages among certain blood types in periods of high demand.

To address the effects of blood assignment decisions on the ability to meet future requests, an alternate real-time blood assignment policy is proposed, the maximum transfusion value (MTV) policy. The MTV policy aims to decrease future blood shortages by maximizing the TV of each assignment while simultaneously minimizing current blood shortages by fully satisfying the present request. Correspondingly, for each blood request, among those compatible blood types with inventory level higher than the request size, the

blood type with the highest TV is chosen. If the inventory level of all the compatible blood types is lower than the request size, the available compatible blood type with the highest TV is chosen. Among the compatible blood types, the maximum TV always belongs to the requested type.

To compare and contrast the effects of the two policies, we develop a Monte Carlo simulation model, which utilizes repeated random sampling to simulate blood requests by patients, and correspondingly predicts the likelihood of outcomes, such as unmet requests. In this model a primary health care center (PHCC) with the capability of blood transfusion is simulated. The simulation model is calibrated to the characteristics of those seen in South Sudan, one of the countries in Africa with high maternal morbidity rate (WHO 2015). At the beginning of the simulation, an initial blood bank, denoted as the *population-based blood bank*, is defined. The distribution of the inventory by blood type is equivalent to the blood type distribution as occurs among the South Sudan population blood type distribution (contributors 2018). It is assumed that the blood bank is not replenished throughout the simulation duration. Thus, the available units in the initial blood bank are the only resources available to fulfill patients' requests for red blood cells (RBCs). For each patient, request size and request type refers to the number of units of RBCs requested and the patient's blood type, respectively. The length of the simulation is assumed to be 42 days, corresponding to the shelf life of RBCs.

The occurrence of patients requesting RBC transfusion is modeled with a Poisson distribution. For each patient, the request size is assumed to follow a Normal distribution.

The likelihood of each blood type, and correspondingly the request type of each patient, follows the same empirical distribution that defines the likelihood of blood types in the South Sudan population. The parameters for these statistical distributions are provided in Table 3-1.

Table 3-1- Statistical distributions used in the Monte Carlo simulation model

Variables	Distribution	Mean	Standard deviation
Population-based blood bank	Constant	X¶ $*$ $\begin{cases} O^+: 0.480, A^+: 0.277, B^+: 0.152, AB^+: 0.028, \\ O^-: 0.035, A^-: 0.018, B^-: 0.008, AB^-: 0.002 \end{cases}$	
Number of patients requesting for RBC (Month)	Poisson	21.43	
Request size	Folded Normal	6	2
Request type	Empirical	CDF=$\begin{cases} O^+: 0.480, A^+: 0.757, B^+: 0909, AB^+: 0.937, \\ O^-: 0.972, A^-: 0.990, B^-: 0.998, AB^-: 1.000 \end{cases}$	

¶ X = blood bank size

In South Sudan, each PHCC serves a population of 15,000 to 50,000 residents (Macharia, Ouma et al. 2017). To ensure a robust analysis with type I and type II errors, of 0.01 and 0.05 respectively, the *"base case"* simulation includes 20,000 iterations for a population of 15,000 residents. The total number of RBC units in the initial blood bank is referred to as the *blood bank size*. The blood bank size is sufficient to respond to the expected patient requests for RBC and is determined by the product of the average request size and expected number of patients requesting RBC during the simulation duration.

In each iteration, patients requesting RBCs are generated for the entire simulation duration, 42 days. Then for each patient, the request type and the request size are generated according to the statistical distributions defined above. Requests are fulfilled

chronologically by following one of the two assignment policies, GA or MTV. Neither policy presumes exact knowledge of future requests when an assignment decision is made.

To assess the effectiveness of the policies, two performance measures are calculated to describe the unsatisfied requests for each policy and iteration with respect to (i) the RBC units and (ii) the number of patients affected. First, we calculate the ratio of the accumulated number of requested RBC units that are not satisfied and the population size for each iteration. The average of this ratio over all iterations is denoted as *the average number of RBC units in shortage per capita*. Additionally, we calculate the ratio of the total number of patients whose requests for blood are not fully satisfied, due to unavailability of compatible blood type units, and the population size. The average of this ratio over all iterations is labeled as *the average number of patients facing shortage per capita.* These two performance measures are calculated to compare the performance of MTV and GA policies.

In the following section, the performance measures from applying MTV and GA policies in the simulated iterations are presented. Additionally, the performance of these policies are further analyzed through additional experiments.

3-4. Results

Performance measures resulting from applying GA and MTV policies on the base case simulated model is represented as follows. The simulated model and the results of applying policies are implemented using Python 2.7 on an Intel i7 CPU laptop with a 2.5 GHz

processor and 12 GB RAM. To test the effectiveness of the blood assignment policies, under a variety of settings assumption for the base case simulation, such as the size and distribution of initial blood bank and the population size, are changed and the corresponding performance measures are provided.

Base case: Initial results for the base case, which correspond to the assumptions above, are presented in Table 3-2. Via a paired t-test applied to the outcomes of the simulation, we conclude that there is strong evidence that, on average, MTV policy performs better than GA policy on both performance measures (p-value = $6.952 * 10^{-300}$).

Table 3-2- Performance measures of applying MTV and GA policies on the base case simulated data, using population-based blood bank

Performance measure	Policy	Mean	Standard deviation
The average number of RBC units in shortage per capita	MTV	15.891	14.884
	GA	16.239	15.029
The average number of patients facing shortage per capita	MTV	3.222	2.727
	GA	3.382	2.780

A further analysis of the relative performance of these two policies is conducted via a sensitivity analysis. Specifically, we explore how changes to the (i) initial blood bank distribution, (ii) initial blood bank size, and (iii) the size of the population in the simulation affects the performance in relation to the base case, presented above.

3-4.1. Alternative blood type distributions

The blood type distribution in the initial blood bank may not always be exactly the same as the requests, as it is considered in the base case. Assuming the size of the initial blood

bank is fixed to meet the total expected requests, varying the distribution of the initial blood bank, by altering the frequency of each blood type, may improve the performance of blood assignment practices. Since the distribution of type and size of the requests is known, stochastic programming is suggested to generate the optimal blood bank distribution (Jabbarzadeh, Fahimnia et al. 2014, Gunpinar and Centeno 2015). Additional simulations are conducted to assess how sensitive the performance of these two policies is to this optimal blood bank.

As expected, the optimal blood bank with the goal of minimizing the number of RBC units in shortage, without any limitation on the quantity of each blood type, contains only O^-. Since providing enough units of O^- to meet the total expected requests is not feasible in practice, we add a constraint to limit the quantity of each blood type in the optimal blood bank to beat most 10% more than the quantity of that type in the population-based blood bank. Since the quantity of each blood type in the blood bank is an integer number, this upper bound is rounded to the nearest integer. If for a blood type, the rounded upper bound is the same as the quantity of that blood type in the population-based blood bank, the upper bound for the mentioned type is incremented by one unit.

To model the problem of finding the optimal blood bank distribution, a two-stage stochastic model with recourse is implemented in order to account for the uncertain nature of the requests. In a two-stage stochastic programming model, the first-stage decision is made prior to learning information about future events. The second-stage decision are limited by the first-stage decisions but are made after information becomes available (King

and Wallace 2012). In this stochastic model at the first-stage the optimal blood bank, defined by the quantity of each blood type in the bank, is chosen such that the assignment of RBC units to requests minimizes unsatisfied requests. This mathematical model for the second-stage corresponds to the offline blood assignment problem in which compatible available blood units in the blood bank are assigned to requests and each request is fulfilled by only one blood type. In an offline blood assignment problem, the request type and size of all patients, including future requests are known at the moment of the decision.

As input to the model we have multiple scenarios with each scenario being defined as the set of patient requests that occur. The optimal solution to the model defines the optimal blood bank distribution, B_j, that should be chosen (in the first-stage) without having information about which scenario will occur. The second-stage decisions about allocation of RBC units to requests, x_{jt_s}, is made after information about the scenario is realized. These two decisions in turn determine that expected number of unmet requests, $\Sigma_{t_s=1}^{T_s} z_{t_s}$. While the first stage decision about the blood bank distribution is not included in the objective function, it effects the second stage decision variables , z_{t_s},which are included in the objective function. The indices, parameters and variables of the model are presented in Tables 3-3, 3-4, and 3-5.

Table 3-3- Indices for model

s	$request\ scenario, s = 1, 2, \dots, S$
j	$blood\ types, j \in J = \{O^-, A^-, B^-, AB^-, O^+, A^+, B^+, AB^+\}$
t_s	$sequence\ of\ requests\ in\ scenario\ s, t_s = 1, 2, \dots, T_s$

Table 3-4- Parameters for model

d_{t_s}	$size\ \ of\ request\ t\ in\ \ scenario\ s$

r_{t_s}	$type\ of request\ \ t\ in\ scenario\ s$
E_{ij}	$compatibility\ matrix\ to\ assign\ RBC\ type\ j\ to\ request\ type\ i, i = r_{t_s}$
D	$initial\ blood\ bank\ size$
B_j^0	$upper\ bound\ on\ quantity\ of\ blood\ type\ j\ \ in\ optimal\ blood\ bank$
M	$big\ number$
p_s	$probability\ of\ occurance\ of\ scenario\ s$

Table 3-5- Decision variables

I_{jt_s}	$inventory\ of\ RBC\ type\ j\ after\ request\ t\ in\ scenario\ s$
x_{jt_s}	$number\ of\ units\ of\ RBC\ type\ j\ assigned\ to\ request\ t\ in\ scenario\ s$
y_{jt_s}	$conditional\ varibale\ indicates\ if\ the\ blood\ type\ j\ is\ assigned\ to\ request\ \ t\ in\ scenario\ s$
z_{t_s}	$number\ of\ units\ of\ unfulfilled\ request\ after\ request\ t\ in\ scenario\ s$
B_j	$number\ of\ units\ of\ RBC\ type\ j\ in\ optimal\ blood\ bank$

The two-stage stochastic mixed integer programming model of this problem is formulated as follow:

(3-1) $min \quad \Sigma_{s\in S}\, p_s\, \Sigma_{t_s=1}^{T_s} z_{t_s}$

(3-2) *s.t.* $\Sigma_{j\in J} E_{ij} x_{jt_s} + z_{t_s} = d_{t_s} \qquad s = 1,2,\dots,S\ , t_s = 1,2,\dots,T_s$

(3-3) $x_{jt_s} \le M y_{jt_s} \qquad s = 1,2,\dots,S\ , t_s = 1,2,\dots,T_s, j \in J$

(3-4) $\Sigma_{j\in J}\ y_{jt_s} \le 1 \qquad s = 1,2,\dots,S\ , t_s = 1,2,\dots,T_s$

(3-5) $\Sigma_{j\in J}\, B_j \le D$

(3-6) $B_j \le B_j^0 \qquad j \in J$

(3-7) $B_j - x_{j1_s} = I_{j1_s} \qquad s = 1,2,\dots,S\ , j \in J$

(3-8) $I_{j(t-1)_s} - x_{jt_s} = I_{jt_s} \qquad s = 1,2,\dots,S\ , t_s = 2,3,\dots,T_s, j \in J$

(3-9) $x_{jt_s} \in \mathbb{N} \qquad s = 1,2,\dots,S\ , t_s = 1,2,\dots,T_s, j \in J$

(3-10) $y_{jt_s} \in \{0,1\} \qquad s = 1,2,\dots,S\ , t_s = 1,2,\dots,T_s, j \in J$

(3-11) $I_{jt_s} \geq 0$ $s = 1, 2, \dots, S\ , t_s = 1, 2, \dots, T_s, j \in J$

(3-12) $z_{t_s} \geq 0$ $s = 1, 2, \dots, S\ , t_s = 1, 2, \dots, T_s$

(3-13) $B_j \geq 0$ $j \in J$

The objective function (3-1) seeks to minimize the expected unsatisfied requests across all scenarios. Constraint (3-2) assigns blood units of compatible blood type to fulfill the requests. Constraint (3-3) and (3-4) ensure that each request is fulfilled by only one blood type. Constraint (3-5) limits the total number of units of blood in the optimal blood bank to the initial blood bank size, D, which is equal to expected number of requests. Constraint (3-6) bounds the number of units of each blood type in the initial blood bank to an upper bound, B_j^0. This upper bound ensures that, for each blood type, collecting the specified amount of blood in the optimum blood bank is practical. Constraint (3-7) updates the inventory for the first request and assignment. In Constraint (3-8) the inventory is updated for all other requests. Constraint (3-9) indicates that all assignments are nonnegative integers. Constraint (3-10) defines the conditional variable as a binary variable. Constraints (3-11) -(3-13) state the nonnegativity of all other decision variables.

Computational study: The proposed model is implemented using Python 2.7 and Gurobi 6.5 on an Intel i7 CPU laptop with a 2.5 GHz processor and 12 GB RAM. The output of the stochastic model is the optimal blood bank. Defining the scenarios via the base case simulation, the optimal blood bank converges after the inclusion of 100 scenarios, such that additional scenarios does not change the optimal blood bank. The optimal blood bank is compared with population-based blood bank for the base case in Table 3-6.

Table 3-6- Distribution of optimum blood bank and initial blood bank for the base case

Blood type	O^+	A^+	B^+	AB^+	O^-	A^-	B^-	AB^-	Total
Population-based blood bank	87	50	27	5	6	3	2	0	180
Optimal blood bank	95	47	24	0	7	4	3	0	180

Similar to the previous section we use simulation model with 20,000 iterations to compare the effectiveness of the MTV and GA policy under the two blood bank distributions. By applying the paired t-test on the output of the simulation model, the results reveal that the two performance measures are significantly smaller by applying the MTV policy as compared to GA policy. Also, the paired t-test shows that both MTV and GA policy perform significantly better with the optimal blood bank rather than the population-based blood bank.

Table 3-7- Performance measures of applying MTV and GA policies on the base case simulated data, using optimal blood bank

Performance measure	Policy	Mean	Standard deviation
The average number of RBC units in shortage per capita	MTV	13.584	14.154
	GA	13.885	14.281
The average number of patients facing shortage per capita	MTV	2.795	2.631
	GA	2.959	2.681

3-4.2. Initial blood bank size

To examine the effect of the initial blood bank inventory, Monte Carlo simulations with newly generated iterations are performed. For each simulation, the initial blood bank sizes are either 30%, 20%, or 10% greater or less than the expected requests during the simulation duration.

The distribution of blood types in the population-based blood bank remains the same. Additionally, the optimal blood bank is computed via stochastic model with the change in the blood bank size. The frequency of blood types of the population-based and optimal blood banks are presented in Table 3-8. It can be seen, for all initial blood bank sizes, the frequency of O^+, O^-, A^- and B^- is higher and the frequency of A^+, B^+ and AB^+ is lower in the optimal blood bank. The frequency of AB^- is zero in both the population-based and optimal blood banks.

Table 3-8- Frequency of blood types in population and optimal blood bank. The optimal blood bank is computed considering various initial blood bank size

Blood bank size	Bank	O^+	A^+	B^+	AB^+	O^-	A^-	B^-	AB^-
100%	Population-based blood bank	0.48	0.28	0.15	0.03	0.03	0.02	0.01	0
70%	Optimal blood bank	0.53	0.27	0.12	0	0.04	0.02	0.02	0
80%	Optimal blood bank	0.53	0.26	0.13	0	0.04	0.03	0.01	0
90%	Optimal blood bank	0.53	0.26	0.14	0	0.04	0.02	0.01	0
100%	Optimal blood bank	0.53	0.26	0.13	0	0.04	0.02	0.02	0
110%	Optimal blood bank	0.53	0.26	0.13	0	0.04	0.03	0.01	0
120%	Optimal blood bank	0.53	0.26	0.14	0	0.04	0.02	0.01	0
130%	Optimal blood bank	0.53	0.26	0.14	0	0.04	0.02	0.01	0

The impact of the initial inventory levels and the inventory distribution on performance measures is provided in Figures 3-2 and 3-3. Applying paired t-test we conclude that for both performance measures the MTV policy performs significantly better than the GA policy and both MTV and GA policies perform significantly better with optimal blood bank rather than population-based blood bank. As the initial blood bank size increases, the average number of RBC units in shortage per capita and the average number of patients facing shortage per capita decrease, for both policies.

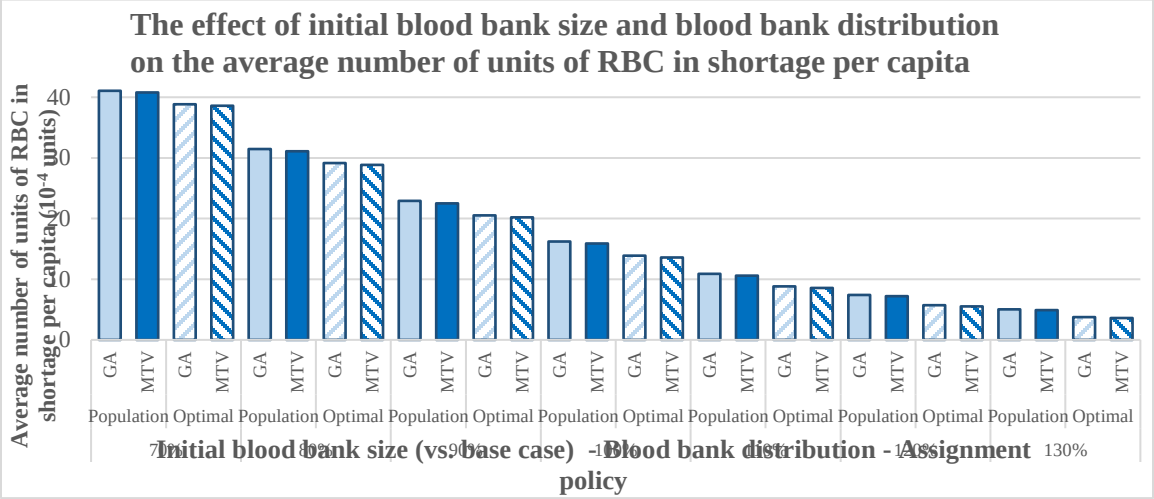

Figure 3-2- The effect of varying initial blood bank size (vs. base case), blood type distribution, and assignment policies on the number of units of RBC in shortage per capita

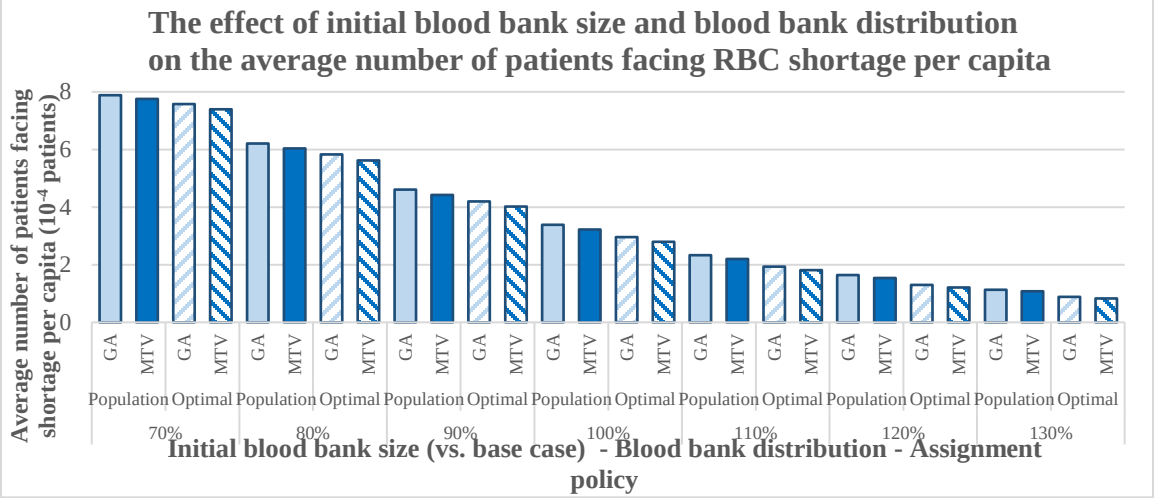

Figure 3-3- The effect of varying the initial blood bank size (vs. base case) and blood type distribution, and assignment policies on the number of patients facing shortage per capita

3-4.3. The population size

Finally, additional simulations are analyzed to characterize the effects of changing the population size, while simultaneously changing the blood bank size to meet the total expected requests. As mentioned above, in South Sudan each PHCC serves populations of 15,000 to 50,000 residents. Table 3-9 provides the distribution of blood types for

population-based and optimal blood banks for the studied population sizes. The distribution of blood types in the optimal blood bank when changing the population size is very similar to the results when altering the initial blood bank size.

Table 3-9- Frequency of blood types in population-based and optimal blood bank. The optimal blood bank is computed for different population sizes

Population	Bank	O+	A+	B+	AB+	O-	A-	B-	AB-
15000	Population-based blood bank	0.48	0.28	0.15	0.03	0.03	0.02	0.01	0
10000	Optimal blood bank	0.53	0.27	0.12	0	0.04	0.02	0.02	0
15000	Optimal blood bank	0.53	0.26	0.13	0	0.04	0.02	0.02	0
20000	Optimal blood bank	0.53	0.26	0.14	0	0.04	0.02	0.01	0
30000	Optimal blood bank	0.53	0.26	0.14	0	0.04	0.02	0.01	0
40000	Optimal blood bank	0.53	0.26	0.14	0	0.04	0.02	0.01	0
50000	Optimal blood bank	0.53	0.26	0.14	0	0.04	0.02	0.01	0

Performance measures of applying MTV and GA policies on the simulated data after changing the population size and blood bank distribution are presented in Figures 3-4 and 3-5. Despite other disadvantages, increasing the size of the population served by the PHCC, the performance measures of applying MTV and GA policies is improved. As expected, optimal blood bank results are significantly better than the population-based blood bank, also the MTV policy performs significantly better than the GA policy in both performance measures.

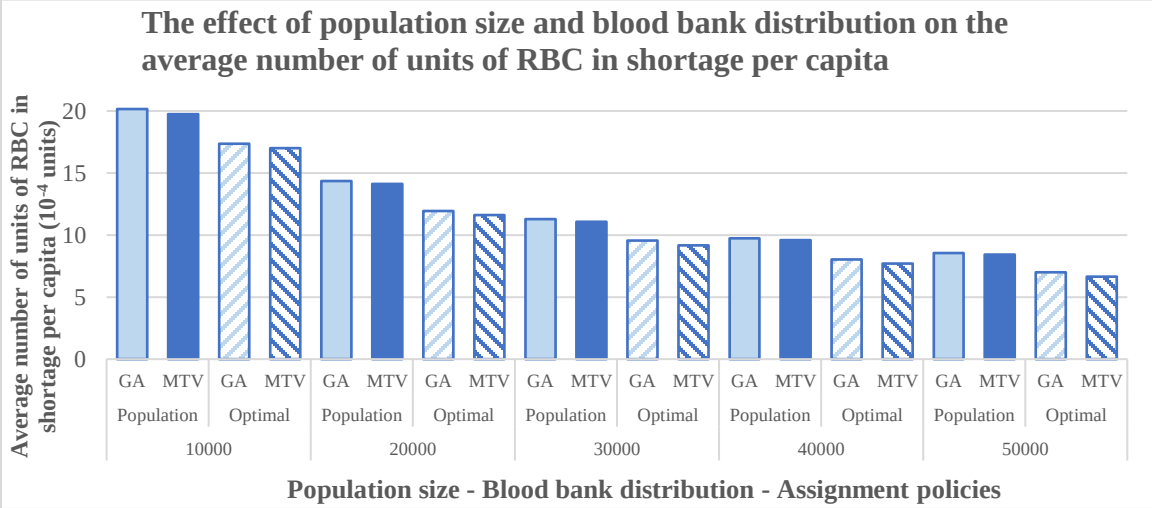

Figure 3-4- The effect of changing population size, initial blood bank distribution, and assignment policies on the number of units of RBC in shortage per capita

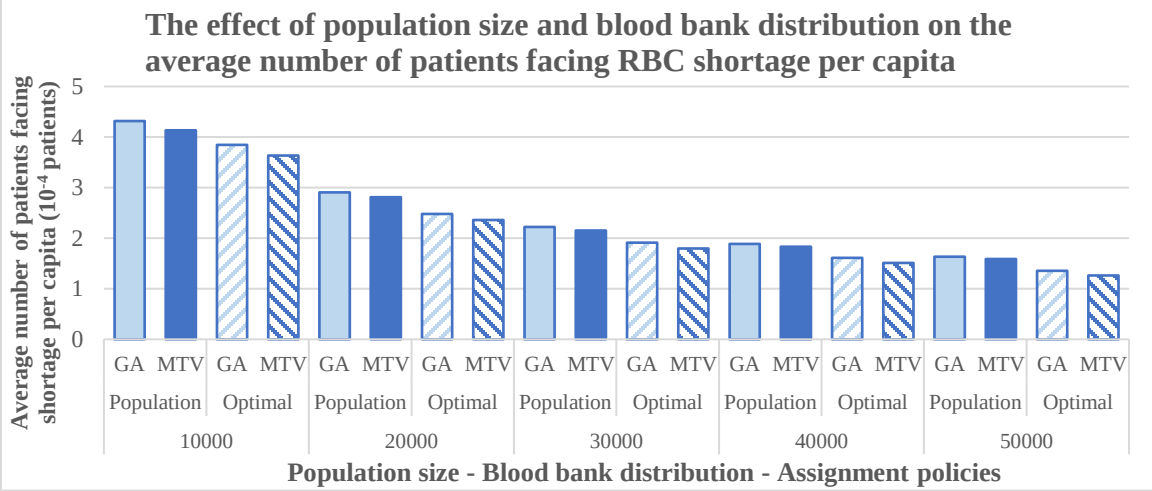

Figure 3-5-The effect of changing population size, initial blood bank distribution, and assignment policies on the number of patients facing shortage per capita

3-5. Discussion

Blood shortages are a significant problem, particularly in isolated environments, and can directly lead to loss of lives. In such environments, resupply of blood may not be readily available and thus the limited starting blood bank supplies are the primary resource for addressing requests. Past researchers have examined inventory management policies, but

minimal attention has been given to determining how to assign blood units as requests are made. In this study, we examine the efficiency of two policies in reducing the occurrence of blood shortages in an isolated environment. In the GA policy all efforts are made to assign the compatible blood type with the highest level of available inventory to the request. The MTV policy seeks to minimize future shortages by assignment through a proactive blood type substitution approach.

Specifically, the MTV policy performs significantly better in comparison to the GA policy in both performance measures. In the base case, the distribution of blood types in the population-based blood bank is assumed to be the same as that found among the population. As seen above, the optimal blood bank, computed by the described two-stage stochastic optimization model performs better, for both policies, with respect to the average number of RBC units in shortage per capita and the average number of patients facing shortage per capita. The O^- blood type is the global donor, O^- can be assigned to requests for any blood type. Correspondingly, higher proportions of O^- in stock increases the ability to respond to blood requests and prevent unmet requests. Though higher volumes of O^- in the blood bank are beneficial it is not necessarily easily achievable since is not the most prevalent among donors and the cost of acquiring large amounts of O^- units is great. Therefore, the quantity of each blood type in the optimization model is limited to an upper bound which limits the quantity of each blood type to be at most the greater value of 10% of the base case or a one-unit increase. The optimal blood banks computed via the stochastic optimization model for the altering the sizes of blood bank and population, follow similar trends. In the optimal blood bank, the quantity of O^-, A^-, $B^-$and O^+ is

increased to the upper bound, and the quantity of A^+ and B^+ is decreased to meet the blood bank size limitation. The optimal blood bank does not contain any units of AB^- or AB^+. The performance improvement of optimal blood bank as compared to the population-based blood bank is made greater by applying the MTV policy rather than the GA policy.

One of the primary factors demonstrated to improve performance measures is the initial blood bank size. As is expected, the greater the number of blood units available, the smaller the average number of RBC units in shortage per capita and the average number of patients facing shortage per capita. While this trend is true for both policies, decisions to stock more blood in the initial blood bank does have disadvantages including higher costs, challenges for storing larger quantities, and ethical concerns of having wasted, unused RBC. For the two performance measures, the improvement of these measures of MTV policy to that of GA policy increases by increasing the initial blood bank size and by using the optimal blood bank distribution.

The population size serving by PHCC also affects the performance of the policies. As population sizes increase, in proportion to the blood bank size, the average number of RBC units in shortage per capita decreases. Besides disadvantages of serving higher population size by a specific PHCC, having more units of blood from each type, provides additional flexibility in meeting requests if a shortage occurs for a particular blood type. The MTV policy performance improvement is higher on larger population sizes and by optimal blood bank. These findings support the value of having centralized systems for

blood banking and distribution for multiple PHCCs assuming that timely delivery of blood units can be achieved.

Even with various changes to the population size, blood bank size, and distribution of blood types in the blood bank, the findings are consistent with (i) the MTV policy significantly resulting in better performance measure than the GA policy and (ii) for any of the two assignment policies the optimal blood bank provides significantly better performance than the population-based blood bank. Additionally, the underlying system parameters of the blood bank size and the size of population that is served by PHCC are demonstrated to affect performance regardless of which of the two policies are used.

These results support the need for consideration of developing proactive blood assignment policies and using an optimal distribution in blood banks, while considering feasibility limitations, in environments where requests may not be satisfied. The MTV policy is simple to implement and is demonstrated to produce better results than the GA policy. Following the results here, future research is needed to further examine alternate blood assignment policies that may perform better than the proactive blood type substitution policy, MTV. Finally, this work can be applied on additional instances in which there may be a large number of requests in a short period of time, such as the aftermath of a high-casualty event.

Chapter 4 -

Characterizing Optimality of the MAV Resource Allocation Policy for the Generalized Online Resource Allocation Problem

In this chapter we analyze the performance of the resource allocation policy suggested in Chapter 3 for a generalized resource allocation problem. We characterize the features of problem for which the allocation policy provides an optimal solution to the online resource allocation problem, with equivalent performance to that found with the offline version of the problem. Also, we analyze the performance of applying this allocation policy when these characteristics are violated and provide an upper-bound pertaining to the worst-case performance.

4-1- Introduction

The resource allocation problem pursues the best allocation of a limited number of resources of various types to requests for each type of resource in order to optimize a specified objective subject to constraints on the availability of resources (Yin and Wang 2006). This problem has been well studied (Debreu 1951, Hurwicz 1973) in various domains such as politics (Romer and Rosenthal 1978), welfare economics (Arrow 1962), healthcare (Mooney, Russell et al. 1980) and intra-household management (Thomas 1990). In many industries including healthcare, the major challenges in the resource allocation problem include limited resources and growing demand for these resources.

The resource allocation problem can further be distinguished as focusing on either offline or online allocations. In the offline resource allocation problem, all requests and their features are known prior to the allocation of resources. In contrast, in the online resource allocation problem allocation decisions are made over time and allocations to requests must be made without exact knowledge of future demand. The offline resource allocation problem is shown to be NP-Complete (Wang and Liu 2005). Correspondingly, a variety of different methods are employed for formulating and solving this problem including linear programming, dynamic programming, branch-and-bound solution approaches, and genetic algorithms (Yin and Wang 2006).

This chapter is motivated by the online blood assignment problem that occurs in isolated environments, as presented in Chapter 3. Blood assignment is the process of

assigning adequate units of a compatible blood type to a blood request. Since the risk and the cost of unsatisfied requests for blood is higher in isolated environments, it is essential to develop an appropriate online blood assignment policy to preserve enough blood units to respond to all requests. In the previous chapter, two online resource allocation policies are presented and through comparing the policies it is found that one performs significantly better. Although the performance of the policies is compared through a simulation model focuses on the blood assignment setting in the previous chapter, the performance of the policy for the generalized assignment problem for different settings is not.

In this chapter we define a generalized online resource allocation problem in alignment with the problem discussed in Chapter 3. For this problem, we propose an online allocation policy based on the concepts of MTV, referred to as the MAV policy. Dissimilar from the blood resource allocation problem, in this generalized problem the structure of the compatibility of resources is not limited to those found in the blood transfusion application. Since the MTV policy performs well on the blood assignment problem, we specify characteristics of the online resource allocation problem such that this allocation policy provides an optimal allocation. An online resource allocation is designated as optimal if the objective function value of the online allocation is equal to the optimal objective function value in the offline resource allocation problem (Asadpour, Wang et al. 2016). We perform further analysis of the suggested allocation policy, while violating some these characteristics.

The remainder of this chapter is organized as follows. Section 4-2 formulates the generalized resource allocation problem. Section 4-3 presents an overview of research focusing on variants of the generalized online resource allocation problem. Section 4-4 introduces an online allocation policy, MAV. Section 4-5 defines characteristics of the allocation problem under which the output of implementing the MAV policy is optimal. Section 4-6 discusses the performance of the allocation policy when these characteristics are eliminated. Finally, Section 4-7 presents a summary of the work.

4-2- Problem definition

The online resource allocation problem examined here is inspired by the blood assignment problem in which requests are submitted sequentially and must be responded to at the time of the request. Compatible resources are assigned to the requests, and if there are no compatible resources available to satisfy the request, the request is unmet. In this chapter we present and analyze the performance of an algorithm for a generalized online resource allocation with the objective of minimizing shortages, or unmet requests.

Suppose there is a set of n types of resources, $S = \{s_1, s_2, \dots, s_n\}$, and for each resource type, s_j, there is a limited and known initial inventory, I_j^0. Requests for these resources are made sequentially and each request is defined by the timing within the sequence of requests, r^t. Without loss of generality, we assume that one request is made per unit of time $t \in \{1, 2, \dots, T\}$ in a fixed time horizon T. Correspondingly, when $t = 1$, the first request, r^1, is made and the second request, r^2, occurs at $t = 2$. Each request at

time t is for one unit of resource type $s_j \in S$, denoted as the request type, $r^t = s_j$. The probability distribution of the frequency of request types for each time are independent and identically distributed (I.I.D.) according to a discrete probability distribution with the probability of the request being for resource type s_j defined as $p_j = p(r^t = s_j, s_j \in S)$.

A request type s_j can be satisfied by assigning either (i) one unit of resource type s_j from the inventory or (ii) one unit of a compatible resource type s_k, such that $k \neq j$, from the available inventory. Inventory of all resources update after each time period. If resource type $s_j \in S$ is assigned to the request at time t, r^t, then the inventory of resource type s_j is decreased by one unit, $I_j^t = I_j^{t-1} - 1$, and the inventory of all other resource types remain the same, $I_k^t = I_k^{t-1}$ for all resource types, s_k, where $k \neq j$. It is assumed the inventory for all resource types do not replenish during the time horizon, T.

The complexity matrix, M, represents the ability of a resource of type s_k to satisfy a request type s_j. The matrix M is a $n \times n$, binary matrix and element m_{jk} is 1 if resource type s_k is able to satisfy request type s_j. Likewise, m_{jk} is zero if resource type s_k is not compatible to satisfy the request type s_j. The indices of rows in the matrix represent the request type and the indices of columns represent the resource types which are assigned to the requests.

(4-1)
$$M = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 1 & 1 & \ddots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 1 & 0 & \dots & 1 \end{bmatrix}$$

If the assigned resource type is not the same as the request type, this allocation is denoted as a substitute allocation. If the resource type of the assigned unit is the same as the request type, the allocation is denoted as a regular allocation.

Since one unit is requested at each period of time, the total number of units requested in the time horizon, T, is equal to T. In order to disregard the cases in which requests are not satisfied due to insufficient total inventory, we assume that the sum of the initial inventory across all resource types is equal to T. Thus:

(4-2) $$\Sigma_{j\in S}\, I_j^0 = T$$

Additionally, we assume that there is no mismatch in the expected number of requests and the total initial inventory for each resource type.

(4-3) $$I_j^0 = p_j * T$$

If the full sequence of requests is known, the total requests for resource type s_j, is defined to be R_j. The total remaining inventory at time T, for resource type s_j is I_j^T. The number of total unmet requests at time T is denoted by Z and is defined as the sum of unmet requests for each resource type s_j at time T, z_j^T:

(4-4) $$Z = \Sigma_{j\leq n} z_j^T = \Sigma_{j\leq n\,,R_j>I_j^0}\,(R_j - I_j^0)$$

The goal of the resource allocation problem is to minimize the unmet requests and two variants of this problem are examined. The first problem variant focuses on solving the offline allocation problem which assumes all the requests are known a priori and the allocations are made to minimize the unmet requests. The total unmet requests in the optimal solution of this offline allocation problem is denoted as the optimum unmet requests, Z^*. Unlike the offline allocation problem, in practice, resource allocation decisions must be made without full knowledge of future requests. This corresponds to the online resource allocation problem which deals with uncertainty of future requests, while assigning resources to the current requests. The optimum solution to the offline allocation problem is a lower bound on the best solution for the corresponding online allocation problem. As was shown in the previous chapter, the total number of unmet requests in the online allocation problem, Z^+, depends on the allocation policy. If the number of unmet requests in the online allocation policy, Z^+, is equal to the optimum unmet requests in the offline problem, Z^*, the solution is referred to as optimal allocation.

In Section 4-4 we introduce an allocation policy to decrease the expected total number of unmet requests for the online resource allocation problem with stochastic requests (similar to the policy from the previous chapter). We then define system characteristics for which the number of unmet requests when applying this policy in the online setting is guaranteed to result in the same performance as in the corresponding optimal offline allocation.

4-3- Related research

There is an extensive amount of research focused on designing and analyzing online resource allocation problems. In this section we present similar online resource allocation problems to the problem described in Section 4-2 and common solution approaches for these online resource allocation problems. Specifically, we present a summary of research pertaining to the online-bin packing problem, the online variable-sized bin packing problem, and the online multiple knapsack problem. We also provide a brief review of research in the graph theory domain focused on the online resource allocation problem.

4-3.1. Online bin packing problem

In the online bin packing problem, items arrive one at a time and must be assigned to a bin without knowledge of upcoming items. Item sizes are in the range (0, 1] and the capacity of each bin is 1. The goal of the online bin packing problem is to minimize the number of utilized bins. Ullman first studied this problem and since then many researchers have developed effective allocation algorithms for addressing this problem (Klein 2016). Johnson et. al (1974) and Johnson (1974) study the performance of Best Fit, First Fit and Revised First Fit algorithms (Johnson 1974, Johnson, Demers et al. 1974). Seiden (2001) provides a performance analysis of the HARMONIC, REFINED HARMONIC, MODIFIED HARMONIC, MODIFIED HARMONIC2 and HARMONIC+1 algorithms (Seiden 2001) for the bin packing problem.

4-3.2. Online variable-sized bin packing problem

Taking into consideration different types of bins with specified but varying capacities, the variable-sized online bin packing problem is defined. In this problem, items can be packed into bins of different sizes. In this problem, bin sizes are specified to be less than or equal to 1 and access to an infinite number of bins from each size is provided. The goal of the variable-sized bin packing problem is to minimize the total size of bins used for packing. In other words, if the cost of bins is proportionally related to the size of bins then the goal of the problem is to minimize the cost of packing (Friesen and Langston 1986). In the online variable-sized bin packing problem decisions are made about whether or not to open a new bin and decisions are made about the size of the new bins (Kinnerseley and Langston 1988). The VARIABLE HARMONIC algorithm is proposed by Csirik (1989) based on the HARMONIC algorithm. Similar to the HARMONIC algorithm, this algorithm uses bounded space and Seiden shows that this algorithm is an optimum bounded space online algorithm (Seiden 2000). Seiden (2003) proposes the VRH1 and the VRH2 algorithms based on the HARMONIC and REFINED HARMONIC algorithms, and improves the upper-bound for this problem (Seiden, Van Stee et al. 2003).

4-3.3. Online multiple knapsack problem

With the consideration of limiting the number of available resources from various types, the multiple knapsack problem is studied. In the multiple knapsack problem items with various weights and profits associated with them are packed in different capacity knapsacks and the goal is to maximize the profit of packed items accounting for the limited capacity

of the knapsacks (Chekuri and Khanna 2005). Chakrabarty et al. (2008) denotes the ratio of profit to weight as the efficiency for each item and defines an efficiency threshold for each knapsack. The proposed solution approach states that if the efficiency of the item is larger than the specified threshold and enough capacity is available, the item is packed into the knapsack. (Chakrabarty, Zhou et al. 2008). In the knapsack problem the weight of items is assumed to be very small compared to the capacity of the knapsack (Chakrabarty, Zhou et al. 2008).

The online bin packing problems and online knapsack problems share similarities with the generalized online allocation problem defined in Section 4-2. First, there are requests that may or may not be compatible with a particular resource, such as a bin or knapsack. Additionally, decisions about allocations of requests are made in an online fashion and a decision in one period will affect the effectiveness of meeting future requests. Also, the goal of minimizing unused space in bins or knapsacks can be seen as equivalent to goal of minimizing unmet demand in the generalized allocation problem presented above.

While the presented research areas share similarities, there are also differences between these problems and the generalized resource allocation problem. In the online bin packing problem items (requests) have different sizes (types) but there are no differences between bins (resources). All the bins have the capacity of 1, and all the items can be packed in any bin with available capacity larger than the item size. Also, there is no limit on the number of available bins and the goal is to utilize the minimum possible number of

resources while satisfying all requests. In the online variable-sized bin packing problem different-size bins can be assigned to items with various sizes. Thus, bins can be packed with any item with a smaller size than the capacity of the bin. Again, the goal is to minimize the total capacity of utilized resources. Like the online bin packing problem, there is no limit on the number of available bins of each size and one bin can be packed with more than one item. In the multiple knapsack problem, knapsacks (resources) are utilized to pack the items with different profit values and each item is feasible to be packed in any knapsack the goal is to maximize the profit of the knapsacks. Although this problem limits the number resources, there is a choice of not packing an item into any knapsack despite the availability of capacity upon item arrival. Although in these problems different types of requests are satisfied by allocating resources, the specifications and goals of these problems are different than our defined problem.

4-3.4. Applications of graph theory in online resource allocation problem

Many studies have explored the application of graph theory to resource allocation problems in various domains, including online advertising and cloud distribution.(Stoffel 2005)

The Adwords problem is one class of problem in which keywords in online advertising are assigned to advertisement buyers (bidders) with the goal of maximizing profits. This problem is represented with a bipartite graph with keywords and bidders being modeled with distinct sets of nodes. Karp et al. (1990) define a RANKING algorithm for the bipartite matching in this problem in which resources are assigned to a random ranking, and the resource with the highest rank is assigned to the request. The authors claim that the

RANKING algorithm performs as well as greedy algorithm and they also modified the RANKING algorithm to address random refusals or EARLY refusals (Karp, Vazirani et al. 1990). Karande et al. (2011) prove the performance guarantee of the RANKING algorithm by providing an upper-bound and lower-bound for this algorithm performance on problems with unknown distribution models (Karande, Mehta et al. 2011). Devanur et al. (2011) introduce a bid to budget ratio to bound their greedy algorithm and solve the online bipartite b-matching problem (Devanur, Jain et al. 2011). Vee et al. (2010) formulate the online allocation problem with forecasts. In this problem there is a knowledge of some random group of requests from the future. They suggest a two-phase solution including an offline compact allocation in the first phase and the online allocation in the second phase (Vee, Vassilvitskii et al. 2010). Although in this online allocation problem resources are limited and non-reusable similar to the problem defined in 4-2, in this class of problems one resource can be assigned to more than one request, and any resource is able to satisfy the requests, although potentially with a poor profit. Thus there is no focus on unmet requests, as occurs in the blood allocation setting. In applications of online matching in cloud distribution, cloud resources with various features are allocated to requests over time. Constraints on requests pertaining to geographical location, network distance band, and cost ranges are accounted for in the allocation of virtual machines to requests (Hao, Kodialam et al. 2017). The online adversary (Beloglazov and Buyya 2012), random weight assignment (Sahai 2011), and primal-dual algorithm (Shi, Zhang et al. 2014) are the most common methods in online cloud distribution problem. Unlike the allocation problem

defined in this chapter, in the cloud distribution problem resources are reusable and requests wait for a finite or infinite time to be satisfied.

Although the focus of most studies on online assignment problems focus on generalized versions, some researchers have examined the problem as it applies to specific graph structures. Correspondingly, they focus on the development of algorithms for solving the online assignment problem for network graphs with specific features. In assignment problems one of the graph features that is often examined is the transitivity closure between resource types. Lin et al. (2001) suggest transitive closure graphs to represent non-slicing floorplans and applied an algorithm based on the features of these graphs to allocate the modules of a circuit into a chip to optimize time and area. By analyzing the experimental results they claim that transitive closure graph modeling is an effective solution in area and wireless optimization (Lin and Chang 2001). Goel et al. (2005) study the performance of a greedy online algorithm for a routing and load balancing problem. They prove that the algorithm is $O(\log_n) - fair$ and $O(\log_m) - balanced$ where n is number of jobs and n is number of machines (Goel, Meyerson et al. 2005). Similarly, we examine the relationship between algorithm performance and graph network structure, and specifically the transitive closure property among resource types for the generalized assignment problem below.

4-4. Allocation policy

For the online assignment problem, the use of different online allocation policies can lead to significantly different results, with respect to the total unmet demand (as shown in

Chapter 3). These allocation policies define how to assign a compatible resource to each request without exact knowledge of future requests with the goal of achieving the optimal system performance. In this section we introduce and generalize one of these studied polices from the previous chapter. The policy, named maximum assignment value (MAV) policy, is designed to fulfill requests while maximizing the capability of system to fulfill future unknown requests.

As mentioned earlier, in the online blood assignment problem, Adewumi et al. (2012) define the transfusion value (TV) for blood types. Considering one blood type as the requested blood type, the transfusion value is the ratio of the number of blood types compatible to be assigned to the requested blood type and the total number of compatible transfusions between all blood types (Adewumi, Budlender et al. 2012). Inspired by this concept, for the generalized online resource allocation problem, we define the "assignment value". Assignment value, AV_{s_j}, is a feature for each resource type s_j and represents the number of resource types that are compatible to be assigned to requests of this type. The compatibility matrix M defined in Section 4-1 is used to calculate AV_{s_j} for all resource types. For resource type s_j the j^{th} row of M represents the compatibility of resource types to fulfill request type s_j. The sum over the values in this row defines AV_{s_j}, $AV_{s_j} = \Sigma_{k=1}^{n} m_{jk}$. In the MAV policy, for a request the compatible resource type, s_j, with the largest AV_{s_j} and sufficient inventory is assigned. If no compatible resources has sufficient inventory, the request is not fulfilled and one unit of unmet request is recorded. An example of the allocation of resources using the MAV policy is presented in Section 4-4.1.

4-4.1. Example of MAV allocation rules for a case of four resource types

An example of the MAV online resource allocation algorithm when there are four resource types, $S = \{s_1, s_2, s_3, s_4\}$, is presented below. The compatibility matrix, M_1 and AV_{s_j} for all $s_j \in S$ are provided in (4-5).

(4-5)
$$M_1 = \begin{matrix} \\ 1 \\ 2 \\ 3 \\ 4 \end{matrix} \begin{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \end{matrix} \begin{matrix} AV \\ 1 \\ 2 \\ 3 \\ 4 \end{matrix}$$

Before any requests are made, at time 0 the initial inventory is defined as $I_j^0, j \in \{1, 2, 3, 4\}$. At each time $t \in \{1, 2, \dots, T\}$ a request is made, a resource is assigned, and the inventory of each resource type is updated. The update of inventory levels at each time t as defined by the MAV policy are dependent of the request type r^t and the inventory levels at time t. These updates based on request type are presented in (4-6), (4-7), (4-8) and (4-9).

(4-6)
$$r^t = s_4 : \begin{cases} I_4^t = I_4^{t-1} - 1, if\ (I_4^{t-1} > 0) \\ I_3^t = I_3^{t-1} - 1, if\ (I_3^{t-1} > 0, I_4^{t-1} = 0) \\ I_2^t = I_2^{t-1} - 1, if\ (I_2^{t-1} > 0, I_4^{t-1} = I_3^{t-1} = 0) \\ I_1^t = I_1^{t-1} - 1, if\ (I_4^{t-1} = I_3^{t-1} = I_2^{t-1} = 0) \end{cases}$$

(4-7)
$$r^t = s_3 : \begin{cases} I_3^t = I_3^{t-1} - 1, if\ (I_3^{t-1} > 0) \\ I_2^t = I_2^{t-1} - 1, if\ (I_2^{t-1} > 0, I_3^{t-1} = 0) \\ I_1^t = I_1^{t-1} - 1, if\ (I_1^{t-1} > 0,\ I_3^{t-1} = I_2^{t-1} = 0) \end{cases}$$

(4-8)
$$r^t = s_2 : \begin{cases} I_2^t = I_2^{t-1} - 1, if\ (I_2^{t-1} > 0) \\ I_1^t = I_1^{t-1} - 1, if\ (I_1^{t-1} > 0,\ I_2^{t-1} = 0) \end{cases}$$

(4-9) $$r^t = s_1 : \ \{I_1^t = \ I_1^{t-1} - 1, \ if \ \ I_1^{t-1} > 0$$

In some problem incidences of this case it is not possible to meet all the requests with a compatible resource and unmet requests are inevitable. It is notable that despite the sequence of requests, in this example the number of unmet requests resulting from applying the MAV policy, Z^+, is the same as the optimum number of unmet requests in the offline allocation, Z^*. In the next section we provide details of the instance characteristics that results in the MAV allocation being an optimal online allocation.

4-5- Performance analysis

The proposed online allocation policy, MAV, is described in Section 4-4, and in this section, the performance of this algorithm is analyzed. The circumstances under which the MAV policy results in an optimal allocation policy is characterized.

4-5.1. Analysis of cases with optimality

The MAV policy results in an optimal solution to the generalized resource allocation problem in Section 4-2 if the underlying problem instance has specific conditions. These conditions are established in definitions I, II, III and IV, and the optimality of MAV policy under these definitions is proven in Theorem 1.

Definition I. If there is a request type s_j and resource type s_k can satisfy the request type s_j, then resource type s_k is compatible with resource type s_j.

Definition II. Any resource type s_j is compatible with itself.

Definition III. The assignment value of every resources is unique. In other words, for any two resource types s_j and s_k, such that $s_j, s_k \in S, j \neq k$, the assignment value of resource type s_j is not equal to the assignment value of resource s_k $AV_{s_j} \neq AV_{s_k}$.

Definition IV. If resource type s_j is compatible with resource type s_k, then resource type s_k is not compatible with resource type s_j. This is denoted as the orientation of compatibility between resource types.

Theorem 1. Suppose that for some instance of the generalized resource allocation problem presented in Section 4-2 the Definitions I, II, III and IV are satisfied. Then the MAV allocation policy is an optimal policy.

In support of proving Theorem 1, we define Lemmas 1 and 2. In Lemma 1, resource types are ordered to establish a compatibility rule based on the uniqueness of the assignment values.

Lemma 1. Suppose that for some instance of the generalized resource allocation problem presented in Section 4-2 the Definitions I, II, III and IV are satisfied. If (i) the assignment value of resource type s_i, is equal to i, $AV_{s_i} = i$, and (ii) the assignment value of resource type s_j, is equal to j, $AV_{s_j} = j$, and $i < j$, then resource type s_i is compatible with resource type s_j.

Proof of Lemma 1. Assume there are *n* types of resources and n types of requests for these resources. According to Definition II, request type s_i can always be satisfied by resource type s_i. Therefore, the minimum possible value of AV_{s_i} is one. Likewise, the maximum possible AV_{s_i} is n, in cases that all of the n resource types are compatible to be assigned to requests for a specific resource type. Since, there are n types of resources and there is a unique assignment value corresponding to each resource type, the maximum possible assignment value is n, and the minimum possible assignment value is one, the assignment values for resource types must be a sequence of natural numbers from 1 through n. Without loss of generalization we define the index of each resource type to be the same as the assignment value corresponding to that resource type. For example, the resource type with an assignment value equal to 3 is labeled as s_3. Consequently, s_1 is the resource type for which its request can only be satisfied by s_1. And s_n is the resource type for which all the n resource types are compatible to satisfy its requests.

All the n resource types are compatible with resource type s_n and n-1 of the resource types are compatible with resource type s_{n-1}, according to Definition IV. Since resource type s_{n-1} is compatible with resource type s_n, resource type s_n is the only resource type which is not compatible with resource type s_{n-1}. We use this logic to prove Lemma 1 by induction. For resource type s_n, all resource types s_i, $i \leq n$, are compatible with resource type s_n. Assume Lemma 1 is satisfied for all resource types s_k, then all resource types s_i, $i \leq k$, are compatible with resource type s_k. We want to show that for resource type s_m, $m < k$, all resource type s_j, $j \leq m$ are compatible with resource type

s_m. According to the naming conventions described above there are m resource types compatible with resource type s_m. As stated in the induction assumption, resource type s_m is compatible with all resource types s_k, $k > m$. According to Definition IV, these resource types s_k, $k > m$ are not compatible with resource type s_m, therefore all the m resources type s_l, $l \leq m$ are compatible with resource type s_m. □

Lemma 2. Suppose that for some instance of the generalized resource allocation problem presented in Section 4-2 the Definitions I, II, III and IV are satisfied and that for all resource types s_k, the assignment value is equal to k, $AV_{s_k} = k$. Applying the MAV resource allocation policy, if a request for any resource type s_i is not satisfied at time t, $z_i^t > 0$, for all resource types s_j, such that $j < i$, and the inventory of the resource type s_j must be equal to zero at time t, $I_j^t = 0$.

Proof of Lemma 2. Assume to the contrary that Lemma 2 is not satisfied and by applying the MAV resource allocation policy and some request type s_i is not met by time t, ($z_i^t > 0$) while the inventory of resource type s_j is not equal to zero. Since resource type s_j is compatible with resource type s_i and the inventory of the resource type s_j is positive, $I_j^t > 0$, the total requests is equal to the total inventory at time zero, and the objective is to minimize the unmet requests, resource type s_j must be assigned to request type s_i and request for type s_i must be satisfied at time t. Therefore no unmet request can have occurred for resource s_i at time t. □

Here we deploy these definitions and lemmas to prove the Theorem 1.

Proof of Theorem 1. Assume to the contrary that MAV is not an optimal policy and the quantity of unmet requests resulting from applying MAV policy, Z^+, is larger than the total unmet requests of the optimal offline policy, Z^*. Therefore, we have $Z^+ > Z^*$.

Correspondingly, there must be at least one request at some time t for request type s_m that is satisfied in the optimal offline allocation policy but cannot be satisfied by the MAV policy and thus $z_m^{+t} > 0$. According to Lemma 2 for all resource types s_i, and $i \leq m$, inventory of resource type s_i is empty, $I_i^t = 0$. Since in the optimal offline allocation policy, the request type s_m is satisfied, there should exist at least one resource type s_j and $j \leq m$ with at least one unit of inventory available at time t, $I_j^{*t} > 0$. This means there must be least one request type s_r, at time t_0, before time t, $t_0 < t$, and this request is satisfied by resource type $s_l, 1 \leq l \leq m$, by applying the MAV policy while it is satisfied by some resource type $s_k, k > m$ in the optimal offline policy. In other words, at time t_0, inventory of resource type s_k, $l \leq m < k$ is nonzero, and while by applying MAV policy resource type s_l is assigned. According to Lemma 1, $l < k$, implies that $AV_{s_l} < AV_{s_k}$. If resource type s_k is available at time t_0 and is compatible to be assigned to request of time t_0, based on the definition of the MAV policy, resource type s_k must be assigned at time t_0 while resource type s_l is assigned. This assignment which is made at time t_0 contradicts the MAV assignment policy. Thus, the MAV is optimal and $Z^+ = Z'$. □

4-6 Discussion

Although using the MAV policy in instances of the online resource allocation problems for which all 4 definitions are satisfied results in an optimal number of unmet requests, when any of these definitions are violated, MAV does not necessarily lead to the optimal results. Specifically, we discuss the effects of violating Definitions III and IV and how the performance of the MAV allocation policy is affected by this violation.

4-6.1. Violating Definition IV while satisfying Definitions I, II and III

First, we study the performance of the MAV algorithm applied to a problem instance of the generalized assignment problem is which Definitions I, II and III are satisfied while Definition IV, or the orientation of compatibility between resource types, is not satisfied. Since Definition III is satisfied and assignment values are unique the naming convention used for definition of resource types is assumed to be the same as was presented in the proof of Lemma 1, without loss of generality. An example of a problem instance in which the compatibility between resource types in Definition IV is not satisfied is presented below.

4-6.1.1. Example of MAV allocation rules for a case of four resource types without orientation of compatibility between resource types assumption

An example of MAV online resource allocation problem with four resource types, $S = \{s_1, s_2, s_3, s_4\}$, is provided as follow. Compatibility matrix, M_2 and assignment values for

all resource types are presented in Formula (4-10), Note that the compatibility between the resource types are not oriented.

(4-10)
$$M_2 = \begin{matrix} \\ \\ \\ \\ \end{matrix} \begin{matrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \end{matrix} \begin{matrix} AV \\ 1 \\ 2 \\ 3 \\ 4 \end{matrix}$$

In this case, the allocation preference, as defined by the MAV policy, for satisfying request type s_2 is to assign resource type s_4 upon availability over assigning resource type s_2. It is notable that resource type s_1 is not compatible with resource type s_2. The allocation rules based on the MAV policy are provided in Formulas (4-11), (4-12), (4-13) and (4-14).

(4-11)
$$r^t = s_4 : \begin{cases} I_4^t = I_4^{t-1} - 1, if\ (I_4^{t-1} > 0) \\ I_3^t = I_3^{t-1} - 1, if\ (I_3^{t-1} > 0,\ I_4^{t-1} = 0) \\ I_2^t = I_2^{t-1} - 1, if\ (I_2^{t-1} > 0,\ I_4^{t-1} = I_3^{t-1} = 0) \\ I_1^t = I_1^{t-1} - 1, if\ (I_1^{t-1} > 0,\ I_4^{t-1} = I_3^{t-1} = I_2^{t-1} = 0) \end{cases}$$

(4-12)
$$r^t = s_3 : \begin{cases} I_3^t = I_3^{t-1} - 1, if\ (I_3^{t-1} > 0) \\ I_2^t = I_2^{t-1} - 1, if\ (I_2^{t-1} > 0,\ I_3^{t-1} = 0) \\ I_1^t = I_1^{t-1} - 1, if\ (I_1^{t-1} > 0,\ I_3^{t-1} = I_2^{t-1} = 0) \end{cases}$$

(4-13)
$$r^t = s_2 : \begin{cases} I_4^t = I_4^{t-1} - 1, if(I_4^{t-1} > 0) \\ I_2^t = I_2^{t-1} - 1, if(I_2^{t-1} > 0,\ I_4^{t-1} = 0) \end{cases}$$

(4-14)
$$r^t = s_1 : \{I_1^t = I_1^{t-1} - 1, if\ (I_1^{t-1} > 0)$$

The sequence of request types may affect the number of unmet requests. Assume $R_2 > I_2^0$, $R_3 > I_3^0$ and $R_4 > I_4^0$, according to Formula (4-2), $R_1 < I_1^0$. To reach the same number of unmet requests in the online allocation and optimal offline resource allocation, in case of unavailability of resource type s_3, request type s_3 would need to be satisfied by resource type s_1. In the MAV allocation policy upon unavailability of resource type s_3 and availability of resource type s_2, request type s_3 is instead satisfied by resource type s_2. This leads to a greater number of unmet requests than is found in the optimal solution to the offline assignment problem. Thus, we demonstrate that the MAV policy is not guaranteed to be optimal if Definition IV is not satisfied.

4-6.1.2. Analysis of non-optimal cases with violating Definition IV

Applying MAV online allocation policy does not necessarily lead to optimal results when all definitions except Definition IV are satisfied. Violating Definition IV means there exist at least one pair of resource types that are compatible with each other. Assuming there is only one pair of resource types s_i and s_j, $i < j$, resource type s_i is compatible with resource type s_j, and at the same time, resource type s_j is compatible with resource type s_i. If the assignment values of resource types are unique and the naming of resource types follows the same rule as described in Lemma 1, without loss of generality we can define $AV_{s_i} = i$. Since resource type s_j is compatible with resource type s_i, $i < j$, and resource type s_i is compatible with itself (Definition II) there should exist one resource type s_k, $k < i$ such that resource type s_k is not compatible with resource type s_i. Therefore Lemma 1 is not valid in the case of violation of Definition IV. It is notable that, $i < j$, then $AV_{s_i} < AV_{s_j}$,

therefore, in MAV resource allocation policy, in case of availability of resource type s_j, $I_j^t > 0$, request type s_i is satisfied by resource type s_j. Lemma 1 is not satisfied in this case and the MAV allocation may not be optimal.

Theorem 2. Assume that only resource types s_i and s_j, $i < j$, are compatible with each other and resource type s_k, $k < i$, is not compatible with resource type s_i. The upper-bound for the difference between the optimal number of unmet requests in the offline problem, Z'^*, and the unmet requests resulting from use of the MAV allocation policy, Z', is smaller than I_k^0.

Proof of Theorem 2. Applying the MAV allocation policy may not be optimal if at time t_0 there is a request type s_l, $r^{t_0} = s_l$, $l > i > k$, and the inventory of resource type s_l is empty, $I'^{t_0}_l = 0$, while the inventory of resource type s_i is positive, $I'^{t_0}_i > 0$, and has the largest assignment value. Following MAV allocation policy, resource type s_i is assigned to request type s_l. Assume at time t, $t > t_0$, there is a request for resource type s_i, $r^t = s_i$, while the inventory of resource type s_i is zero, $I'^t_i = 0$, and the only resource type s_m, $m \leq i$ with positive inventory is resource type s_k. Since resource type s_k is not compatible with resource type s_i, this leads to unmet requests for resource type s_i. In this case, at time T, the inventory of resource type s_k is positive, $I'^T_k > 0$, while there are some unmet requests for resource type s_i, $z'^T_i > 0$. In the optimal offline resource allocation, the requests are known a priori, specifically in this case the requests are satisfied in a way such that the

extra requests for resource type s_i are satisfied by resource types other than resource type s_k.

As shown above, the allocation of resource type s_k to requests type s_i causes the difference between the unmet demand resulting from the MAV policy and the optimal offline solution. The upper-bound, UB, is defined for the difference between the number of unmet requests of resource type, s_i, in the optimal offline resource allocation and the MAV allocation policy, $Z' - Z'^{*} = z_i'^{T} - z_i'^{*T}$. The maximum difference between unmet requests at time T for resource type s_i by applying the two allocation policies is equal to the inventory of resource type s_k at time T in MAV allocation policy, $z_i'^{T} - z_i'^{*T} = I'^{T}_{k} - I'^{T*}_{k}$. The upper-bound on this value be defined as $Max(I_k'^{T}) = I_k^0 - R_k$. Since the number of total requests for resource type s_k, R_k, is stochastic and nonnegative, instead we conclude that I_k^0 is an upper-bound for the difference between the unmet requests of the optimal offline resource allocation problem and the MAV policy for any instance of the generalized problem (presented in Section 4-2) in which Definition I, II, and III are satisfied, but Definition IV is not satisfied. □

4-6.2. Violating Definition III while satisfying Definitions I, II and IV

Here, we study the consequences of violating Definition *III*, or the uniqueness of the assignment values. As mentioned in the proof of Lemma 1, the maximum possible assignment value is equal to n and the minimum possible assignment value is one. Violating uniqueness of assignment values means that the assignment value for at least two

resource types are equal. Since in Lemma 1 we named the resource types after their assignment values, in the instances where two or more resource types have equal assignment values, an alternate naming convention must be used. In the new naming convention, the resource types are sorted based on their assignment values increasingly. Since the assignment values are not unique, those resource types with equal assignment values are ordered increasingly based on the size of their initial inventory. We define the index of each resources type by their rank in this sorted list. Following this rule, although assignment values are not unique, the names of resource types are unique.

Since the assignment values of some of the resources types are not unique, there exists a decision point at which the MAV policy does not distinguish between the assignment of multiple resource types. Therefore, an extra rule is needed to make a consistent MAV allocation. In the *modified MAV* allocation policy, the compatible resource type with the largest assignment value, adequate inventory, and among those with equal assignment value the one with the largest initial inventory is selected to satisfy the request. For example, to choose among resource types s_i, s_j and s_k, $i < j < k$, with equal assignment values, $AV_{s_i} = AV_{s_j} = AV_{s_k}$, if the inventory of resource type s_k is positive, resource type s_k is chosen, otherwise consecutively resource types s_j and s_i are chosen assuming they have nonzero inventory levels. The following example presents the performance of the modified MAV allocation policy when Definition III is not satisfied.

4-6.2.1. Example of MAV allocation rules for a case of four resource types and nonunique assignment values

An example of the application of the modified MAV policy in the online resource allocation problem with four resource types, $S = \{s_1, s_2, s_3, s_4\}$, is provided as follows. Compatibility matrix, M_3 and assignment values of resource types are presented in Formula (4-15).

(4-15)
$$M_3 = \begin{matrix} \\ s_1 \\ s_2 \\ s_3 \\ s_4 \end{matrix} \begin{matrix} \begin{matrix} s_1 & s_2 & s_3 & s_4 \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \end{matrix} \begin{matrix} AV \\ 1 \\ 2 \\ 2 \\ 4 \end{matrix}$$

This setting is initiated similarly to the example presented in 4-4.1. As shown in the sequences (4-16), (4-17), (4-18) and (4-19), in a situation when the inventory of resource type s_4 is zero upon arrival of a request for this resource type, following the modified MAV policy, assigning resource type s_3 is prioritized to resource type s_2.

(4-16)
$$r^t = s_4 : \begin{cases} I_4^t = I_4^{t-1} - 1, if\ (I_4^{t-1} > 0) \\ I_3^t = I_3^{t-1} - 1, if\ (I_3^{t-1} > 0,\ I_4^{t-1} = 0) \\ I_2^t = I_2^{t-1} - 1, if\ (I_2^{t-1} > 0,\ I_4^{t-1} = I_3^{t-1} = 0) \\ I_1^t = I_1^{t-1} - 1, if\ (I_1^{t-1} > 0,\ I_4^{t-1} = I_3^{t-1} = I_2^{t-1} = 0) \end{cases}$$

(4-17)
$$r^t = s_3 : \begin{cases} I_3^t = I_3^{t-1} - 1, if\ (I_3^{t-1} > 0) \\ I_1^t = I_1^{t-1} - 1, if\ (I_1^{t-1} > 0,\ I_3^{t-1} = 0) \end{cases}$$

(4-18) $$r^t = s_2 : \begin{cases} I_2^t = I_2^{t-1} - 1, if\ (I_2^{t-1} > 0) \\ I_1^t = I_1^{t-1} - 1,\ if\ (I_1^{t-1} > 0,\ I_2^{t-1} = 0) \end{cases}$$

(4-19) $$r^t = s_1 : \{I_1^t = I_1^{t-1} - 1, if\ (I_1^{t-1} > 0)$$

Following these rules, the number of unmet requests depends on the sequence of the arrival of requests. Assuming $R_4 > I_4^0$, if resource type s_3 is available some requests for resource type s_4 are satisfied by assigning resource type s_3, or resource type s_2 if available. In the case of unavailability of both resource types s_3 and s_2, resource type s_1 meets requests for type s_4. Depending on the sequence of events, when implementing the modified MAV policy, the resulting allocation may lead to unmet requests for types s_1 and s_3. Thus, it is seen that the number of unmet requests depends on the sequences of request arrivals.

4-6.2.2. Analysis of non-optimal cases with Definition III violated

Here, we study problem instances for which Definition III is violated, such that the assignment values are not unique and Lemma 1 is not satisfied. Still the minimum possible assignment value is 1 and the maximum possible assignment value is n. There is no limitation on the assignment values, and as a result various combination of compatibility between resource types are feasible. Applying the modified MAV policy on such instances of the generalized allocation problem does not follow a pattern. Therefore, we examine specific instances of this problem in which the assignment values are unique except for one pair of resource types. This problem can be formed by eliminating compatibility between

two or more resource types in the instances of the generalized problem with which all 4 Definitions are satisfied.

Assume that there are resource types s_i and s_j, such that $i < j$, and resource type s_i is not compatible to resource type s_j. By applying the modified MAV policy, at time t, if there is a request type s_j, while the inventories of all resource types except resource type s_i are zero, $I_k^t = 0,\ \forall k \neq i,\ I_i^t > 0$, then the request type s_j cannot be satisfied. The maximum difference between the number of unmet requests of the optimal offline policy, Z''^*, and the modified MAV policy, Z'' is calculated as, $Z'' - Z''^* = z_j''^T - z_j''^{*T} \leq I''^T_i - I''^{*T}_i$. As a result, I_i^0 is an upper-bound for this specific case.

If instead resource types s_i, s_j and s_k, are such that $i < j < k$, and both resource types s_i and s_j are not compatible with resource type s_k, then, $Z'' - Z''^* = z_k''^T - z_K''^{*T} \leq I''^T_i + I_j''^T - (I_i''^{*T} + I_j''^{*T})$, and $I_i^0 + I_j^0$ is an upper-bound for the difference between the performance of the modified MAV policy and the offline optimal solution. Finally, assume resource types s_i, s_j and s_k, are such that $i < j < k$, and resource types s_i is not compatible with neither resource type s_j nor resource type s_k, then $Z'' - Z''^* = z_j''^T + z_k''^T - (z_j''^{*T} + z_K''^{*T}) \leq I_i^T$, and the upper bound is I_i^0. Thus, it is seen that the upper bound on the difference in performance between the modified MAV and the optimum solution varies based on the features of the data instance when Definition III is not satisfied.

4-7- Conclusion

We have defined a generalized online resource allocation problem based on the blood transfusion problem with the goal of minimizing unmet requests. Inspired by an approach in the blood transfusion literature we have suggested an online resource allocation policy named maximum assignment value, MAV. The MAV allocation policy and the optimal offline allocation policy results in an equal number of unmet requests when the generalized resource allocation problem satisfies some specific conditions, Definitions I-IV. We have provided these conditions and proved the optimality of the MAV allocation policy under these conditions. Two of the main circumstances leading MAV policy to be optimal is the uniqueness of assignment values and the orientation of compatibility between resource types. We have analyzed violations of these two conditions and their effects on the performance of MAV allocation policy, or the modified MAV allocation policy.

Providing improved bounds on the performance of the MAV policy, or modified MAV policy, for problem instances in with Definitions I, II, III, and IV are not satisfied is suggested for future work. Also in the allocation problem the request sizes are assumed to be for one unit, while in real world, the size of the request is stochastic. Providing improved algorithms, building on MAV policy, that accounts for the size of the requests is also recommended.

Chapter 5 -
Conclusions and Future Work

In this dissertation we focus on substitute resource allocation policies in resource allocation problems in health care settings. Our goal is to help in providing efficient and equitable care to improve quality and healthcare delivery in accordance with the aims of the Institute of Medicine. In this study we focus on online resource allocation problem in which decisions about the allocation of limited resources must be made without perfect knowledge about the future requests. We examine the problem of allocating resources at the operational level, in which decisions about the type and size of resources to be assigned must be made upon arrival of the requests while considering the ability to satisfy upcoming requests.

One approach to preserving the ability to meet future requests is the use of substitute resource allocation in which an alternative resource type is assigned to the

request rather than the requested resource type. Substitute resource allocation also prevents unsatisfied requests when the requested resource type is not available. Different approaches to decision making about how and when to deploy substitute resource allocations affect the performance of systems in satisfying requests. We study three aspects of substitute resource allocation in this dissertation. First, we uncover information about past substitute resource allocation policies through mining heterogenous spatio-temporal databases of a hospital. Next, we suggest two online resource allocation policies with two different approaches for substitute resource allocations in blood assignment policies, and compare them through a Monte Carlo simulation model. Finally, we characterize the performance of one of these policies as it relates to the underlying problem instance characteristics.

In Chapter 2 we provide a polynomial algorithm to uncover the historical resource allocation decisions choosing among a regular or substitute resource assignment in a hospital bed assignment application. The output of a resource allocation system is recorded on the two heterogenous spatio-temporal databases, snapshot and event-oriented. In these databases, the effects of decisions are stored but not the decisions themselves. We apply and test the performance of a decision-mining algorithm while incorporating a variety of different noise features in the data. Using tests of simulated data, we demonstrate the algorithm to be robust to missing data on various system sizes in accuracy, precision, and recall. The problem is motivated by hospital bed assignment problem and for this problem, in which time lags between occurring and recording events are expected to be small, this algorithm performs well.

In Chapter 3, we define and analyze the performance of blood assignment policies in an isolated environment to minimize the unsatisfied requests for red blood cells (RBC). We propose two online assignment policies with different approaches to deploy substitute resource allocation. In the first policy, greedy assignment (GA) assigns the compatible blood type with the highest inventory level to the request for RBC, while in the second policy maximum transfusion value (MTV), in the case of the availability of enough inventory, the blood type with the highest transfusion value is chosen to be assigned to the request. In the latter approach the system is more flexible in satisfying future requests upon unavailability of requested blood type. Applying these policies on a Monte Carlo simulation model, simulating a primary healthcare center in South Sudan, the performance is measured with respect to the average number of RBC units in shortage per capita and the average number of patients facing shortage per capita. The MTV policy performs significantly better than the GA policy in both performance measures even when volume of the blood bank, the distribution of blood types, and the population size changes.

The distribution of blood types in the initial blood inventory which is provided to the isolated environment may affect performance measures and influences the capability to satisfy the requests for RBC. A two-stage stochastic optimization model is developed to determine the optimal distribution of blood types in the initial blood inventory. Regardless of the assignment policy, using the optimal distribution of the blood types compared to that of the observed population, results in significantly better assignment performances.

In Chapter 4, we analyze an online resource allocation policy accounting for substitute resource allocation, MAV. A generalized resource allocation problem is defined and the MAV algorithm is proven to perform optimally when the problem instance follows specified characteristics. When these characteristics are violated the MAV allocation policy may not be optimal. Moreover, the behavior of the MAV allocation policy under the violation of two of the main characteristics of this problem is studied.

There are numbers of future research directions to extend this research. In the 2nd chapter we uncovered bed assignment decisions in connection of hospital databases. Connecting the inpatient databases including the uncovered decisions with the emergency department (ED) databases can be used to develop a predictive emergency patient admission model. This predictive model allows for better decisions to improve the patient flow from the ED to inpatient wards and specifically can decrease the number of patients boarding in the ED.

In the Chapter 3, the assignment policies are tested via simulation of an isolated environment. Other applications of these methods can include blood assignment policies in mass casualty events, in which the demand rises in short amount of time and there is no opportunity to replenish the inventory in short term. Also, we suggest testing these blood assignment policies on blood center serving various sizes of population, considering both deterministic and stochastic blood inventory replenishment.

The generalized resource allocation problem in Chapter 4, simplifies the problem to requests of size one while in real world the number of units requested can vary among

requests. Refining the MAV policy to account for the request size is suggested for future work. In the blood assignment problem, which was the motivation for this problem, the uniqueness of assignment values is not satisfied. Providing a better bound on the performance of the MAV policy or modified MAV policy for problem instances in which the uniqueness of assignment value is not satisfied is recommended.

REFERENCES

Adewumi, A., et al. (2012). Optimizing the assignment of blood in a blood banking system: some initial results. 2012 IEEE Congress on Evolutionary Computation, IEEE.

Adewumi, A., et al. (2012). Optimizing the assignment of blood in a blood banking system: some initial results. Evolutionary Computation (CEC), 2012 IEEE Congress on, IEEE.

Aerts, J. C. and G. B. Heuvelink (2002). "Using simulated annealing for resource allocation." International Journal of Geographical Information Science **16**(6): 571-587.

Allain, J. P., et al. (2004). "Blood transfusion in sub-Saharan Africa." Transfusion Alternatives in Transfusion Medicine **6**(1): 16-23.

Arrow, K. (1962). Economic welfare and the allocation of resources for invention. The rate and direction of inventive activity: Economic and social factors, Princeton University Press**:** 609-626.

Asadpour, A., et al. (2016). "Online Resource Allocation with Limited Flexibility." Available at SSRN: https://ssrn.com/abstract=2655790.

Asllani, A., et al. (2014). "A simulation-based apheresis platelet inventory management model." Transfusion **54**(10pt2): 2730-2735.

Asplin, B. R., et al. (2003). "fi kzv A conceptual model of emergency department crowding." Annals of emergency medicine **42**(2): 173-180.

Bates, I., et al. (2008). "Maternal mortality in sub-Saharan Africa: the contribution of ineffective blood transfusion services." BJOG: An International Journal of Obstetrics & Gynaecology **115**(11): 1331-1339.

Beliën, J. and H. Forcé (2012). "Supply chain management of blood products: A literature review." European Journal of Operational Research **217**(1): 1-16.

Beloglazov, A. and R. Buyya (2012). "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers." Concurrency and Computation: Practice and Experience **24**(13): 1397-1420.

Benjelloun, O., et al. (2009). "Swoosh: a generic approach to entity resolution." The VLDB Journal **18**(1): 255-276.

Bhangu, A., et al. (2013). "Intraoperative cell salvage in a combat support hospital: a prospective proof of concept study." Transfusion **53**(4): 805-810.

Boaden, R., et al. (1999). "An exploratory study of bed management." Journal of management in medicine **13**(4): 234-250.

Bohannon, P., et al. (2007). Conditional functional dependencies for data cleaning. Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on, IEEE.

Brizan, D. G. and A. U. Tansel (2015). "A. Survey of Entity Resolution and Record Linkage Methodologies." Communications of the IIMA **6**(3): 5.

Buor, D. and K. Bream (2004). "An analysis of the determinants of maternal mortality in sub-Saharan Africa." Journal of Women's Health **13**(8): 926-938.

Celik, M. (2014). "Partial spatio-temporal co-occurrence pattern mining." Knowledge and Information Systems: 1-23.

Chakrabarty, D., et al. (2008). Online knapsack problems. Workshop on internet and network economics (WINE).

Chaudhuri, S. (1988). Temporal Relationships in Databases. VLDB.

Chekuri, C. and S. Khanna (2005). "A polynomial time approximation scheme for the multiple knapsack problem." SIAM Journal on Computing **35**(3): 713-728.

Chinnaswamy, A., et al. (2015). "A Study on Automation of Blood Donor Classification and Notification Techniques." International Journal of Applied Engineering Research **10**(7): 18503-18514.

contributors, W. (2018, 6 March 2018 23:54 UTC). "Blood type distribution by country." from https://en.wikipedia.org/w/index.php?title=Blood_type_distribution_by_country&oldid=829155165.

Czarnowski, I. and P. Jędrzejowicz (2006). "Instance reduction approach to machine learning and multi-database mining." Annales Universitatis Mariae Curie-Sklodowska, sectio AI–Informatica **4**(1): 60-71.

Debreu, G. (1951). "The coefficient of resource utilization." Econometrica: Journal of the Econometric Society: 273-292.

Devanur, N. R., et al. (2011). Near optimal online algorithms and fast approximation algorithms for resource allocation problems. Proceedings of the 12th ACM conference on Electronic commerce, ACM.

Dufourq, E., et al. (2012). Studies in metaheuristics for the blood assignment problem. 41st Annual Conference of the Operations Research Society of South Africa.

Erhabor, O. and T. Adias (2012). "The challenges of meeting the future blood transfusion requirements in Sub-saharan Africa: the need for the development of alternatives to allogeneic blood." Transfusion Alternatives in Transfusion Medicine **12**(2): 10.

Erickson, M. L., et al. (2008). "Management of blood shortages in a tertiary care academic medical center: the Yale-New Haven Hospital frozen blood reserve." Transfusion **48**(10): 2252-2263.

Erwig, M., et al. (1998). Abstract and discrete modeling of spatio-temporal data types. Proceedings of the 6th ACM international symposium on Advances in geographic information systems, ACM.

Fleming, A. F. (1997). "HIV and blood transfusion in sub-Saharan Africa." Transfusion science **18**(2): 167-179.

Friesen, D. K. and M. A. Langston (1986). "Variable sized bin packing." SIAM Journal on Computing **15**(1): 222-230.

Getoor, L. and A. Machanavajjhala (2012). "Entity resolution: theory, practice & open challenges." Proceedings of the VLDB Endowment **5**(12): 2018-2019.

Glymour, C., et al. (1997). "Statistical themes and lessons for data mining." Data Mining and Knowledge Discovery **1**(1): 11-28.

Goel, A., et al. (2005). "Approximate majorization and fair online load balancing." ACM Transactions on Algorithms (TALG) **1**(2): 338-349.

Gould, S. A., et al. (2002). "The life-sustaining capacity of human polymerized hemoglobin when red cells might be unavailable." Journal of the American College of Surgeons **195**(4): 445-452.

Griffin, J., et al. (2012). "150: Development Of Patient-bed Assignment Algorithms To Support Bed Management Processes For Improvements In The Rate Of Overflow Assignments And Average Request To Assign Metrics." Critical Care Medicine **40**(12): 1-328.

Gunpinar, S. and G. Centeno (2015). "Stochastic integer programming models for reducing wastages and shortages of blood products at hospitals." Computers & Operations Research **54**: 129-141.

Guo, L., et al. (2014). Using clustering and transitivity to reduce the costs of crowdsourced entity resolution. Proceedings of the 1st International Workshop on Crowd-based Software Development Methods and Technologies, ACM.

Gutiérrez, G. A., et al. (2005). A spatio-temporal access method based on snapshots and events. Proceedings of the 13th annual ACM international workshop on Geographic information systems, ACM.

Güting, R. H. (1994). "An introduction to spatial database systems." The VLDB Journal—The International Journal on Very Large Data Bases **3**(4): 357-399.

Hakre, S., et al. (2013). "Transfusion-transmitted human T-lymphotropic virus Type I infection in a United States military emergency whole blood transfusion recipient in Afghanistan, 2010." Transfusion **53**(10): 2176-2182.

Hall, R. (2012). Bed assignment and bed management. Handbook of Healthcare System Scheduling, Springer**:** 177-200.

Han, J., et al. (2007). "Frequent pattern mining: current status and future directions." Data Mining and Knowledge Discovery **15**(1): 55-86.

Hao, F., et al. (2017). "Online allocation of virtual machines in a distributed cloud." IEEE/ACM Transactions on Networking **25**(1): 238-249.

Henkelman, S. and G. Rakhorst (2012). "Does modern combat still need fresh whole blood transfusions?" Transfusion **52**(10): 2272-2273.

Hoot, N. R. and D. Aronsky (2008). "Systematic review of emergency department crowding: causes, effects, and solutions." Annals of emergency medicine **52**(2): 126-136. e121.

Hughes, R. (2008). Patient safety and quality: An evidence-based handbook for nurses, Agency for Healthcare Research and Quality Rockville, MD.

Hurwicz, L. (1973). "The design of mechanisms for resource allocation." The American Economic Review **63**(2): 1-30.

Jabbarzadeh, A., et al. (2014). "Dynamic supply chain network design for the supply of blood in disasters: a robust model with real world application." Transportation Research Part E: Logistics and Transportation Review **70**: 225-244.

Johnson, D. S. (1974). "Fast algorithms for bin packing." Journal of Computer and System Sciences **8**(3): 272-314.

Johnson, D. S., et al. (1974). "Worst-case performance bounds for simple one-dimensional packing algorithms." SIAM Journal on Computing **3**(4): 299-325.

Joseph, B. G., et al. (2009). "Red blood cell use outside the operating theater: a prospective observational study with modeling of potential blood conservation during severe blood shortages." Transfusion **49**(10): 2060-2069.

Kalashnikov, D. V., et al. (2005). Exploiting Relationships for Domain-Independent Data Cleaning. SDM, SIAM.

Karande, C., et al. (2011). Online bipartite matching with unknown distributions. Proceedings of the forty-third annual ACM symposium on Theory of computing, ACM.

Karp, R. M., et al. (1990). An optimal algorithm for on-line bipartite matching. Proceedings of the twenty-second annual ACM symposium on Theory of computing, ACM.

Kiguli, S., et al. (2015). "Anaemia and blood transfusion in African children presenting to hospital with severe febrile illness." BMC medicine **13**: 21-21.

King, A. J. and S. W. Wallace (2012). Modeling with stochastic programming, Springer Science & Business Media.

Kinnerseley, N. G. and M. A. Langston (1988). "Online variable-sized bin packing." Discrete Applied Mathematics **22**(2): 143-148.

Klein, H. G. and D. J. Anstee (2008). Mollison's blood transfusion in clinical medicine, John Wiley & Sons.

Klein, K.-M. (2016). Robust Bin Packing. Encyclopedia of Algorithms, Springer**:** 1858-1860.

Li, B. N., et al. (2008). "On decision making support in blood bank information systems." Expert Systems with Applications **34**(2): 1522-1532.

Li, L., et al. (2015). "Rule-based method for entity resolution." IEEE Transactions on Knowledge and Data Engineering **27**(1): 250-263.

Li, W.-S. and C. Clifton (2000). "SEMINT: A tool for identifying attribute correspondences in heterogeneous databases using neural networks." Data & Knowledge Engineering **33**(1): 49-84.

Lin, J.-M. and Y.-W. Chang (2001). TCG: a transitive closure graph-based representation for non-slicing floorplans. Design Automation Conference, 2001. Proceedings, IEEE.

Litvak, E. and M. Bisognano (2011). "More patients, less payment: Increasing hospital efficiency in the aftermath of health reform." Health Affairs **30**(1): 76-80.

Loh, W.-K., et al. (2004). "A subsequence matching algorithm that supports normalization transform in time-series databases." Data Mining and Knowledge Discovery **9**(1): 5-28.

Macharia, P. M., et al. (2017). "Spatial accessibility to basic public health services in South Sudan." 2017 **12**(1).

Malsby III, R. F., et al. (2013). "Prehospital blood product transfusion by US Army MEDEVAC during combat operations in Afghanistan: a process improvement initiative." Military medicine **178**(7): 785-791.

Mastroianni, C., et al. (2003). Managing heterogeneous resources in data mining applications on grids using xml-based metadata. Parallel and Distributed Processing Symposium, 2003. Proceedings. International, IEEE.

McCarthy, L. J. (2007). "How do I manage a blood shortage in a transfusion service?" Transfusion **47**(5): 760-762.

Mehenni, T. and A. Moussaoui (2012). "Data mining from multiple heterogeneous relational databases using decision tree classification." Pattern Recognition Letters **33**(13): 1768-1775.

Mintzberg, H., et al. (1976). "The structure of" unstructured" decision processes." Administrative science quarterly: 246-275.

Mooney, G. H., et al. (1980). Choices for health care. Choices for Health Care, Springer**:** 1-9.

Morris, Z. S., et al. (2012). "Emergency department crowding: towards an agenda for evidence-based intervention." Emergency Medicine Journal **29**(6): 460-466.

Nandal, R. (2013). "Spatio-temporal database and its models: a review." IOSR J. Comput. Eng **11**(2): 91-100.

Norén, G. N., et al. (2010). "Temporal pattern discovery in longitudinal electronic patient records." Data Mining and Knowledge Discovery **20**(3): 361-387.

Obulesu, O. and A. R. M. Reddy (2014). "Finding Frequent and Maximal Periodic Patterns in Spatiotemporal Databases for Shifted Instances." Data Mining and Knowledge Engineering **6**(5): 224-232.

Olshaker, J. S. and N. K. Rathlev (2006). "Emergency department overcrowding and ambulance diversion: the impact and potential solutions of extended boarding of admitted patients in the emergency department." The Journal of emergency medicine **30**(3): 351-356.

On, B.-W., et al. (2012). "Scalable clustering methods for the name disambiguation problem." Knowledge and Information Systems **31**(1): 129-151.

Peck, J. S., et al. (2012). "Predicting Emergency Department Inpatient Admissions to Improve Same-day Patient Flow." Academic Emergency Medicine **19**(9): E1045-E1054.

Peck, J. S., et al. (2013). "Generalizability of a simple approach for predicting hospital admission from an emergency department." Academic Emergency Medicine **20**(11): 1156-1163.

Pelekis, N., et al. (2015). "HERMES: A trajectory DB engine for mobility-centric applications." International Journal of Knowledge-Based Organizations (IJKBO) **5**(2): 19-41.

Pelekis, N., et al. (2004). "Literature review of spatio-temporal database models." The Knowledge Engineering Review **19**(03): 235-274.

Pidcoke, H. F., et al. (2012). "Ten-year analysis of transfusion in Operation Iraqi Freedom and Operation Enduring Freedom: increased plasma and platelet use correlates with improved survival." Journal of Trauma and Acute Care Surgery **73**(6): S445-S452.

Proudlove, N., et al. (2007). "Developing bed managers: the why and the how." Journal of nursing management **15**(1): 34-42.

Proudlove, N., et al. (2003). "Can good bed management solve the overcrowding in accident and emergency departments?" Emergency Medicine Journal **20**(2): 149-155.

Romer, T. and H. Rosenthal (1978). "Political resource allocation, controlled agendas, and the status quo." Public choice **33**(4): 27-43.

Rozinat, A. and W. M. P. Aalst (2006). Decision mining in business processes, Beta, Research School for Operations Management and Logistics.

Sahai, A. (2011). "Online Assignment Algorithms for Dynamic Bipartite Graphs." arXiv preprint arXiv:1105.0232.

Schmidt, R., et al. (2013). "Decision support for hospital bed management using adaptable individual length of stay estimations and shared resources." BMC medical informatics and decision making **13**(1): 3.

Seiden, S. S. (2000). An optimal online algorithm for bounded space variable-sized bin packing. International Colloquium on Automata, Languages, and Programming, Springer.

Seiden, S. S. (2001). On the online bin packing problem. International Colloquium on Automata, Languages, and Programming, Springer.

Seiden, S. S., et al. (2003). "New bounds for variable-sized online bin packing." SIAM Journal on Computing **32**(2): 455-469.

Shi, P., et al. (2012). Hospital inpatient operations: Mathematical models and managerial insights, Working paper.

Shi, P., et al. (2013). Patient Flow from Emergency Department to Inpatient Wards: Empirical Observations from a Singaporean Hospital, Working paper. 1.3.

Shi, W., et al. (2014). "An online auction framework for dynamic resource provisioning in cloud computing." ACM SIGMETRICS Performance Evaluation Review **42**(1): 71-83.

Siemens, G. (2005). "Connectivism: A learning theory for the digital age." International journal of instructional technology and distance learning **2**(1): 3-10.

Simonetti, A., et al. (2014). "A stock-and-flow simulation model of the US blood supply." Transfusion **54**(3pt2): 828-838.

Smirnov, A., et al. (2007). Role-based decision mining for multiagent emergency response management. Autonomous Intelligent Systems: Multi-Agents and Data Mining, Springer**:** 178-191.

Snodgrass, R. T. (1986). Temporal databases. IEEE computer, Citeseer.

Spinella, P. C., et al. (2012). "Constant challenges and evolution of US military transfusion medicine and blood operations in combat." Transfusion **52**(5): 1146-1153.

Stoffel, E.-P. (2005). A research framework for graph theory in routing applications, Diplomarbeit/diploma thesis, Institute of Computer Science, LMU, Munich, 2005. http://www. pms. ifi. lmu. de/publikationen/# DA Edgar. Stoffel.

Teow, K. L., et al. (2012). "Intelligent analysis of acute bed overflow in a tertiary hospital in Singapore." Journal of medical systems **36**(3): 1873-1882.

Thomas, D. (1990). "Intra-Household Resource Allocation: An Inferential Approach." The Journal of Human Resources **25**(4): 635-664.

Tomasic, A., et al. (1996). Scaling heterogeneous databases and the design of disco. Distributed Computing Systems, 1996., Proceedings of the 16th International Conference on, IEEE.

Turdukulov, U., et al. (2014). "Visual mining of moving flock patterns in large spatio-temporal data sets using a frequent pattern approach." International Journal of Geographical Information Science **28**(10): 2013-2029.

Van Dijk, N., et al. (2009). "Blood platelet production: a novel approach for practical optimization." Transfusion **49**(3): 411-420.

Vancroonenburg, W., et al. (2012). Patient-to-room assignment planning in a dynamic context. Proceedings of the 9th International Conference on the Practice and Theory of Automated Timetabling, Citeseer.

Vee, E., et al. (2010). Optimal online assignment with forecasts. Proceedings of the 11th ACM conference on Electronic commerce, ACM.

Viccellio, A., et al. (2009). "The association between transfer of emergency department boarders to inpatient hallways and mortality: a 4-year experience." Annals of emergency medicine **54**(4): 487-491.

Wang, W. and X. Liu (2005). List-coloring based channel allocation for open-spectrum wireless networks. Vehicular Technology Conference, 2005. VTC-2005-Fall. 2005 IEEE 62nd, IEEE.

Wang, Y., et al. (2015). Multi-objective optimization for a hospital inpatient flow process via discrete event simulation. Proceedings of the 2015 Winter Simulation Conference, IEEE Press.

Whang, S. E., et al. (2013). "Pay-as-you-go entity resolution." Knowledge and Data Engineering, IEEE Transactions on **25**(5): 1111-1124.

Whang, S. E., et al. (2009). Entity resolution with iterative blocking. Proceedings of the 2009 ACM SIGMOD International Conference on Management of data, ACM.

WHO, U., UNFPA, World Bank Group, and the United Nations Population Division. (2015). Trends in maternal mortality: 1990−2015. Geneva, World Health Organization.

WorldHealthOrganization (2014). Status of Blood Safety in the WHO African Region: Report of the 2010 Survey.

Wu, X., et al. (2008). "Top 10 algorithms in data mining." Knowledge and Information Systems **14**(1): 1-37.

Wu, X., et al. (2005). "Database classification for multi-database mining." Information Systems **30**(1): 71-88.

Wu, X. and S. Zhang (2003). "Synthesizing high-frequency rules from different data sources." IEEE Transactions on Knowledge and Data Engineering **15**(2): 353-367.

Xie, J., et al. (2014). "Bed Allocation to Reduce Overflow." Available at SSRN 2492478.

Yin, P.-Y. and J.-Y. Wang (2006). "Ant colony optimization for the nonlinear resource allocation problem." Applied mathematics and computation **174**(2): 1438-1453.

Yin, P.-Y. and J.-Y. Wang (2006). "A particle swarm optimization approach to the nonlinear resource allocation problem." Applied mathematics and computation **183**(1): 232-242.

Yin, X. and J. Han (2005). Efficient classification from multiple heterogeneous databases. Knowledge Discovery in Databases: PKDD 2005, Springer**:** 404-416.

Yin, X., et al. (2007). Object distinction: Distinguishing objects with identical names. Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on, IEEE.

Zhang, S., et al. (2003). "Multi-database mining." IEEE Computational Intelligence Bulletin **2**(1): 5-13.

Zhang, S., et al. (2009). "Mining globally interesting patterns from multiple databases using kernel estimation." Expert Systems with Applications **36**(8): 10863-10869.

Zhao, H. and S. Ram (2005). "Entity identification for heterogeneous database integration—a multiple classifier system approach and empirical evaluation." Information Systems **30**(2): 119-132.

Zhou, Z., et al. (2015). "A spatio-temporal point process model for ambulance demand." Journal of the American Statistical Association **110**(509): 6-15.