A relational post-processing approach for forms recognition

Zhenxing Mao
University of Nevada, Las Vegas

Follow this and additional works at: https://digitalscholarship.unlv.edu/rtds

Repository Citation
https://digitalscholarship.unlv.edu/rtds/1538

This Thesis is brought to you for free and open access by Digital Scholarship@UNLV. It has been accepted for inclusion in UNLV Retrospective Theses & Dissertations by an authorized administrator of Digital Scholarship@UNLV. For more information, please contact digitalscholarship@unlv.edu.
A RELATIONAL POST PROCESSING APPROACH
FOR FORMS RECOGNITION

by

Zhenxing Mao

Bachelor of Science
Hangzhou University
1996

A thesis submitted in partial fulfillment
of the requirements for the

Master of Science Degree
Department of Computer Science
Howard R. Hughes College of Engineering

Graduate College
University of Nevada, Las Vegas
August 2003
Thesis Approval
The Graduate College
University of Nevada, Las Vegas

May 9, 2003

The Thesis prepared by

Zhenxing Mao

Entitled

A Relational Post Processing Approach For Forms Recognition

is approved in partial fulfillment of the requirements for the degree of

Master of Science

Examination Committee Chair

Dean of the Graduate College

Examination Committee Member

Graduate College Faculty Representative
ABSTRACT

A Relational Post Processing Approach
For Forms Recognition

by

Zhenxing Mao

Dr. Kazem Taghva, Examination Committee Chair
Professor of Computer Science
University of Nevada, Las Vegas

Optical Character Recognition (OCR) is used to convert paper documents into
electronic form. Unfortunately the technology is not perfect and the output can be
erroneous. Conversion then is generally augmented by manual error detection and
correction procedures which can be very costly.

One approach to minimizing cost is to apply an OCR post processing system that will
reduce the amount of manual correction required. The post processor takes advantage of
knowledge associated with a particular project.

In this thesis, we look into the feasibility of using integrity constraints to detect and
correct errors in forms recognition. The general idea is to construct a database of form
values that can be used to direct recognition and consequently, make automatic correction.
# TABLE OF CONTENTS

ABSTRACT ............................................................................................................................ iii  
LIST OF TABLES ................................................................................................................... v  
ACKNOWLEDGEMENTS ................................................................................................... vi  
CHAPTER 1 INTRODUCTION ............................................................................................ 1  
CHAPTER 2 BACKGROUND ............................................................................................. 5  
   Relational Database Model ............................................................................................. 5  
   ER Model and UML ............................................................................................................. 6  
   XML Schema ...................................................................................................................... 10  
   XML Schema for This Thesis ................................................................................  14  
   OCR and its Post Processing ...................................................................................  18  
CHAPTER 3 METHODOLOGY ......................................................................................... 20  
   OCR Processing .................................................................................................................. 20  
   OCR Post Processing in ORACLE .................................................................................... 24  
CHAPTER 4 CONCLUSION AND FURTHER RESEARCH ........................................... 37  
BIBLIOGRAPHY .................................................................................................................. 39  
VITA ....................................................................................................................................... 41
LIST OF TABLES

3.1: Static Information of Patient Peter Mark in the Non-relational Tables .................... 29
3.2: Static Information of Patient Peter Mark in the Relational Table .............................. 29
3.3: Dynamic Information for Patient Peter Mark ............................................................... 31
3.4: Modified Dynamic Information for Patient Peter Mark ............................................ 35
3.5: The Word Accuracy Comparison Statistics ................................................................. 36
ACKNOWLEDGEMENTS

I would like to thank all of the staffs at the Information Science Research Institute for their supports about my project as it progressed. Special thanks go to Jeffery Coombs for his consistent help during all phases of my project, and Ron Young for his help in OCR processing. I would like to especially thank Dr. Kazem Taghva for advising this work and Dr. Tom Nartker, Dr. Wolfgang Bein, and Dr. Emma Regentova for serving on the committee.
CHAPTER 1

INTRODUCTION

We are entering the information explosion era. But much of the world’s information is still captured in printed or handwritten hard copies. How to convert the information in paper formats to electronic versions promptly and accurately becomes a key issue in the research of computer science. The health care industry is one of the most information intensive industries in the world [13]. However, it has been one of the most backward industries to adopt technology to manage these huge quantities of health care information. The “medical chart” has already served as the repository and the assemblage of medical information for over a century [17].

With the tremendous advances of computer technology, such as speed, power, and capacity over the past years, it becomes feasible to store medical data electronically, which could be much more accessible, more useful, more timely, and easier to administrate the patient health information. Electronic data storage has greater long-term utility than its paper predecessor to the patient, clinician, and other parties aimed to improve health care in general. The task of modeling and designing electronic medical records is rather sophisticated. The electronic medical records need to be medico-legally acceptable, comprehensive, secure, portable, and transferable. Clinicians, technicians, researchers, patients, educators, and administrators may need to create, modify, and query these medical records for different purposes. The Good Electronic Health Records
(GEHR) project [13] has been working on electronic medical records representation for around fifteen years.

Our project is specific to occupational medicine which represents a concentration within a much larger arena [17]. Further, the focus of our research centers around the conversion of existing medical charts to electronic forms. The goal for this project is to design a system specific to occupational medicine that takes as input hard copy medical data and produces correct, queryable medical information [17]. The research objective is to interface some technologies and construct some intermediate processes to produce this electronic medical information.

Patient medical records are the forms filled out by different kinds of people including patients themselves, clinicians, technicians, and administrators. The central focus is patient health. Some example records are patient occupational treatment records, patient medicine referral records. These records contain some common information like patient names (last name, first name, middle initial), social security number etc. In addition, every form has its own unique data related to specific requirements of each form.

Scanning technology, image processing, forms recognitions, Optical Character Recognition (OCR) were used to convert hard copy pages to electronic forms. Scanners read texts or illustrations printed on paper and translate the information into a form the computer can use [9]. Scanners digitize hard copy pages. By sensing variations in light intensity, scanners represent patterns on the input page as analog signal—dividing image into a grid of boxes and representing each box with either a zero or a one, depending on whether the box is filled in. (For color and gray scaling, the same principle applies, but each box is then represented by up to 24 bits [9].) Close attention needs to be paid to the
characteristics of the hard copy and to the scanning process so that flaws in the image are minimized [17]. The initial processes in the system are image processing and forms classification. Each form in the patient’s file belongs to a specific category such as Audio, EKG’s etc. The system also extracts the forms logical layout. These layouts have natural tree structures which are modeled using XML Schema for incoming form identification. After form identification, the user-filled data must be extracted for recognition by OCR. OCR typically refers to the recognition of machine printed characters which may or may not be a component of a particular form. Forms recognition is broader. It usually includes several recognition modules: handprint recognition, optical mark recognition, barcode recognition and possibly handwritten recognition [17]. Unfortunately this technology is not perfect to perform this conversion. Its application is limited and the output can be erroneous. To date, OCR technology works fairly well with all typed text documents. But even for text documents there could be up to 25 misspellings in a typical converted typed paper with 99% OCR accuracy [11], not to mention other formats like handwritten documents, forms and graphics.

In this thesis, we look into the feasibility of using integrity constraints to detect and correct errors in forms recognition. Efficient techniques have been devised for detecting strings that do not appear in a given word list, dictionary or lexicon. But correcting a misspelled string is a much harder problem [6]. Our approach relies on the approximation with respect to relational database referential integrity. The general idea is to construct a database of form values that can be used to direct recognition and consequently, make some automatic correction. The purpose of this thesis is not to improve the OCR recognition accuracy from the technological point of view; rather, it
tests the feasibility of using a logical, relational post processing approach for forms recognized by OCR technology. Our method is based on the assumption that some information for a specific person will repeat logically. The methods presented here is a dynamic, semantic based string matching algorithm that use the integral constraints of rational database model to automatically detect the possible errors and correct them. The knowledge of typical OCR errors, the scalar words distribution in certain fields, and date approximation that are context based are incorporated to improve this process as well. It is hoped to correct the misspellings.

Chapter 2 provides an introduction to the concepts and applications in this thesis of relational database, ER diagram, UML, XML, OCR and its post-processing which are the basic constructs of the whole project. Chapter 3 describes our experiments and covers related results. Some modifications and statistic will be presented. Finally, chapter 4 states the conclusion of this study and offer prospects for future research.
CHAPTER 2

BACKGROUND

This chapter describes all the theoretical and practical constructs in our project. These works have already been completed by the staffs at ISRI.

Relational Database Model

All the available data in this project are in the paper forms. These forms are not relational schemas by the design according to the requirements of relational database design—full information with minimal repetition and true consistency. The forms may have a lot of redundancy and inconsistency. The very redundancy is one of the prerequisites to adopt rational post processing. Therefore, a special treatment of conceptual database design based on the existing discursive information is needed. This differs a lot with conventional top down database design, which increases a lot of work and difficulties.

A good database does not just happen: the structure of its content must be designed carefully. Because the database is the source from which information is generated, its design is the subject of detailed study in the modern data environment. The relational database design was formally introduced by E. F. Codd in 1970 and has evolved since then [12]. Relational data models are composed of sets and relations. A set is a collection of objects while a relation is any subset of the Cartesian product of any...
specified sets. The relational database design represents data in the form of two-dimensional tables as relations. Each table represents some real world entity such as person, place, thing, or event about which information is collected [14]. The organization of data into relational tables/schemas is known as the logical view of the database which presents data to the user and the programmer.

Relational database theory is established on a set of constraints. Entity integrity constraint states that all entries are unique, and no primary key value can be null. A key is an attribute or combination of attributes that uniquely identifies each entity in a table. Also it is minimal. The primary key is a selected key to uniquely identify all other attributes values in any given row. It can not contain null value. The foreign key is an attribute or combination of attributes in one table whose values must either match the primary key in another table or be null [12]. A tuple is an entry representing a row in a relational table. A domain constraint determines the range of permissible values in the columns in a table. A referential integrity constraint is specified between two relations and is used to maintain consistency among tuples between of the two relations. The referential integrity constraint says that every non-null foreign key value must reference an existing primary key value [12].

**ER Model and UML**

Data modeling is an integral part of any database project. The Entity relationship (ER) model has played a key role in the fields of relational database modeling since its birth in 1976 by P. P. Chen [1]. An ER diagram is a graphical representation of elements, relationships, and constraints that make up a given design [7]. The ER model is based on
a semantic perception of the real world consisting of a set of basic objects called entities and relationships among these objects. The purpose of this model is to facilitate database design by allowing the specification of an enterprise schema, which represents the overall logical structure of a database. The ER model is extremely useful in mapping the meanings and interactions of real-world enterprises onto a conceptual schema [9]. It achieves a high degree of data independence.

The ER model has three basic components, entity sets, relationship sets, and attributes. An entity is an object or event in the real world that is distinguishable from all other objects and events. For instance, each company is an entity. A relationship is an association among several entities. For example, we can define a relationship between a professor and a student as a professor teaches a student. The role of an entity in a relationship expresses the subject and the object. Information about an entity can be expressed by a set of attribute-value pairs associated with the entity [3]. Entities are described using attributes, and every attribute specifies a particular property for that entity. An attribute of an entity set is a function that maps from the entity set into a value set. The relationship may also have attributes which don’t belong to any side of entities which are related to this particular relationship.

The ER model can be represented by the Unified Modeling Language (UML). UML is an object modeling methodology and is becoming increasingly popular in software design and engineering. The UML model is independent of implementation languages and has proven to be valuable for data modeling [4]. Although it was developed mainly for software design, a major part of software design involves designing the database that will be accessed by the software module.
In UML class diagrams, a class is shown as a rectangular square box that includes three sections: the top part gives the class name; the middle part includes the attributes for individual objects of the class; and the bottom part includes operations that the object of this class can run. No operation is defined in ER diagram. Relationship types are called associations in UML terminology, and relationship instances are called links [12]. A binary relationship type (association) is represented as a line connecting the entity types (participating classes). A relationship attribute (link) is placed in a box that is connected to the association’s line by a dashed line. The number of relationship connection used to specify relationship constraints is called multiplicity. It is specified in the form of min..max, and an asterisk (*) indicates no limit on participation.

We illustrate with an example of an ER diagram represented by UML as in Figure 2.1. In UML, the first letter of each word is the entity name is upper case. The name of the entity shown in the top section of the box is usually a singular noun [2]. Each relationship type is shown as a line connecting the associated entity types, labeled with the name of the relationship. A relationship is normally named using a verb or verb phrase. The first letter of each word in the relationship name is shown in upper case. A relationship is only marked in one way, which normally means that the name of relationship only makes sense in one direction. Once the relationship name is chosen, an arrow symbol is placed beside the name indicating the correct direction to interpret the relationship name. The middle section of the rectangle lists the name of the attributes associated with an entity. The names of the primary key attributes can be labeled with the tag {PK}. In UML, the name of an attribute is displayed with the first letter in lower
case, if the name has more than one word, with the first letter of each subsequent word in upper case.

A huge conceptual relational database model can be designed for the whole project. In this thesis, only 5 entities and partial attributes of these entities are selected to conduct this experiment. There are corresponding 4 different forms matching this diagram. They are Occupational and Non-occupational Treatment Records, Medicine Referral Record and Accident Report. All the selected attributes in each form have some relationships which are used for cross reference. This small diagram is shown as follows in Figure 2.1. A small difference with the original diagram is due to the collection size and simplicity. The core part of this diagram is the entity Personal and OtherInfo with their relationship because they are the main focus of this thesis.
Figure 2.1: ER Diagram for This Thesis.

XML Schema

XML is a markup, meta-language (a language for describing other languages). It helps developers create their own tags to provide functionality not available in HTML. Below are the features in XML but not in HTML:

1. XML document contains large number of tags chosen by the author.
2. Every opening tag of XML must have a matching closing tag.
3. XML document has a root element—the element that contains all other element.
XML started in 1996 and has been a W3C recommendation since February 1998. XML Schema is defined in the W3C's XML Schema Working Group Working Draft published on May 6, 1999 [25]. It took the best parts of SGML (Standard Generalized Markup Language), but it is more simple and easy to use.

XML is extensible, platform-independent, fully Unicode compliant, and it supports internationalization and localization. XML is a set of rules for designing text formats that let us structure our data. XML is referred to as a ‘write once, publish anywhere’ language, with facilities, such as stylesheets that allow the same XML document to be published in different ways using a variety of formats and media. An XML document consists of elements, attributes, entity references, comments, CDATA sections, and processing instructions [15] [20]. Elements identify the nature of the content they surround delimited by angle brackets. Some elements may be empty like <applause></applause>, which means no content. Attributes are name-value pairs that occur inside start-tags after the element name. All attributes values must be quoted. For example, <sex gender= "M"> is a sex element with the attribute gender having the value M. Attributes can not contain any child information, and they are always simple types. Every entity must have a unique name is simply referenced by name. Entity references begin with the ampersand and end with a semicolon. For example, &lt; entity inserts a literal < into a document. Comments are enclosed in <! - and -> tags and can contain any data except the literal character '-'. The XML document begins with a processing instructions (PIs) : <?xml version = "1.0”?> [21]. Its presence explicitly identifies the document as an XML document though it is not required. A CDATA section instructs the XML parser to ignore markup characters (< and &, for example). Between the start...
of the section, <![CDATA[ and the end of the section, ]]>], all character data are passed directly to the application, without interpretation [22].

DTD (Document type definition) is a set of rules for structuring an XML document. It is a formal grammar that specifies a legal XML document, based on the tags used in the document and its attributes. DTDs have a number of limitations. DTDs are written in non-XML syntax; they have no support for namespaces, and they only offer extremely limited data types. The W3C XML Schema specifies the kinds of objects allowed in an XML document, as well as how the objects and their properties are to be organized and the types of values that can be assigned to the object attributes. An XML Schema overcomes these limitations and is much more expressive than DTDs.

One big improvement of XML Schema over DTD is using namespace. Namespace allow element names and relationships in an XML document to be qualified to avoid name collisions for elements that have the same name but are defined in different vocabularies. XML allows us to define a new document format by combining and reusing other formats. We want to have a namespace and a local name for every tag or attribute name in order to fix the above problem. If a name tag has no prefix, it belongs to default namespace. Each schema starts with namespace declaration. We can define namespaces as one of the following:

1. http://www.w3.org/2001/XMLSchema

The names in this namespace are understood by all schema XML processors. These are not names occurring in the document instance.
   A small number of special names that are defined in the XML Schema specification
   but are used in the instance document.

3. Target namespace
   User defined names that are to be used in the instance document.

An XML Schema is the definition of a specific XML structure [20] [23]. An XML
schema uses the W3C XML Schema language to specify how each type of element in the
schema is defined and what data type that element has associated with it. The schema
itself is an XML document, so it can be read by the same tools that read the XML
documents it describes. The XML tags make a document self-describing. The W3C
XML Schema gives us a mechanism to define data types and to use these types to define
our attributes and elements [20]. Elements that contain other elements are of type
complexType. Elements that have no subelements or attributes are of type simpleType.
Simple datatypes are defined by derivation of other datatypes, either predefined and
identified by the W3C XML Schema namespace or defined elsewhere in our schema.
The different kinds of restriction that can be applied on a datatype are called facets.
Many facets allow constraints on the length of a value, an enumeration of the possible
value, the minimal and maximal values, its precision and scales, etc. Using references to
elements and attributes definitions enables reuses of element declarations and saves
repeating the declarations inside each element. XML Schema supports three compositors
that can be mixed to allow various combinations. Each of these compositors can have
minOccurs and maxOccurs attributes to define their cardinality. The sequence
compositor defines ordered groups of elements. The choice compositor describes a choice between several possible elements or groups of elements. The W3C XML schema also provides path expression (Xpath) based features for specifying uniqueness constraints and corresponding references constraints that will hold within a certain type. A key might be composed of a sequence of values located at different depths inside an element. XML key declarations are associated with collection of objects rather than types. We can specify a key in XML Schema as candidate key (use the tag unique) or as primary key (use the tag key) [24]. In addition, keyref is used to define a reference to a key or a unique. It defines a referential constraint made up of element and/or attribute fields that refer to a key.

XML Schema for This Thesis

Each entity creates a schema complexType definition containing a child element for each attribute within that entity [10]. The complexType definition includes child elements that define the content model and/or attributes for this element type. Both the child element definitions and attribute definitions specify the type of their content. We present the XML Schema for the Personal and OtherInfo entities and their relationship in sequence. First is the Personal entity:

```xml
<!--definition of Personal entity -->
<element name="Personal">
  <complexType>
    <all>
      <element name="PersonalName" type="PersonalNameType"/>
      <element name="PersonalSex" type="sexType"/>
      <element name="PersonalDOB" type="dateType"/>
    </all>
  </complexType>
</element>
```

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
<attribute name="SSN" type="SSNType" use="required"/>
</complexType>

<complexType name="PersonalNameType">
  <any>
    <element name="lastName" type="nameType"/>
    <element name="firstName" type="nameType"/>
    <element name="MI" type="MIType" minOccurs="0"/>
  </any>
</complexType>

<complexType name="nameType">
  <union>
    <simpleType>
      <restriction base="string">
        <pattern values="[a-z]{20}"/>
      </restriction>
    </simpleType>
    <simpleType>
      <restriction base="string">
        <pattern values="[a-z]{20}"/>
      </restriction>
    </simpleType>
  </union>
</complexType>

<complexType name="MIType">
  <restriction base="string">
    <pattern values="[A-Z]{1}\.(.)"/>
  </restriction>
</complexType>

<complexType name="sexType">
  <restriction base="string">
    <pattern values="Male | Female | M | F"/>
  </restriction>
</complexType>

<complexType name="dateType">
  <union>
    <simpleType>
      <restriction base="date">
      </restriction>
    </simpleType>
    <simpleType>
      <restriction base="integer">
      </restriction>
    </simpleType>
  </union>
</complexType>
Second is the relationship Contains. We need to enforce the relationship between entities which have relationship by key and foreign key constraints.
Third is the entity OtherInfo. The following is the corresponding code:

<! -- definition of OtherInfo entity -->
<element name="OtherInfo" minOccurs="1">
  <complexType>
    <sequence>
      <element name="maritalStatus"
        type="maritalStatusType"/>
      <element name="homePhone" type="homePhoneType"/>
      <element name="companyName" type="companyNameType"/>
      <element name="homeAddress" type="homeAddressType"/>
    </sequence>
    <attribute name="OtherInfodate" type="dateType"
      use="required"/>
  </complexType>
</element>

<simpleType name="maritalStatusType">
  <restriction base="string">
    <pattern values= "Married | Single | Divorce"/>
  </restriction>
</simpleType>

<simpleType name="homePhoneType">
  <restriction base="string"/>
</simpleType>

<simpleType name="companyNameType">
  <restriction base="string"/>
</simpleType>

<simpleType name="homeAddressType">
  <restriction base="string"/>
</simpleType>

<key name="OtherInfoPK" <!--OtherInfo Primary Key -->
  <selector xpath="./OtherInfo"/>
  <field xpath="@OtherInfodate"/>
</key>
OCR and its Post Processing

The OCR procedure generally consists of scanning the document to produce images tif files, converting the image to ASCII text, and correcting the errors caused by OCR. The last step is by far the most expensive.

OCR devices typically rely on feature analysis to recognize individual characters within words. Features may include costs of the number of vertical, horizontal, cured, and crossing lines in a character. Errors made by OCR devices tend to be those that confuse characters with similar features such as 0 and D, S and 5, or t and f [6].

OCR devices might emulate human being’s recognition sense to improve its accuracy. A significant portion of ability of humans to read text is due to their extraordinary error recovery power, such as the lexical, syntactic, semantic, pragmatic, discursive language aspects they apply.

The post processing system is designed to detect and correct OCR errors through approximation matching by extracting knowledge from the complete document. It incorporates knowledge specific to the kinds of problems one would encounter OCR direct output. This is a rule-based technique. Rule based techniques are algorithms or heuristic programs that attempt to represent knowledge of spelling error patterns in the
forms of rules for transforming misspellings into valid words. Since the complete
document has already been processed, information on its content can be used to correct it.
More information for correcting the misrecognized occurrence is obtained by having
entire document available. The goal of OCR post-processing methods is to maximize the
probability that words or sentences generated by correcting the OCR output are correct.
CHAPTER 3

METHODOLOGY

To test the feasibility of the relational post processing approach, we did the following experiments. The process had two major steps: OCR processing and OCR post processing.

OCR Processing

First, we obtained clean forms of the occupational treatment record, non-occupational treatment record, occupational medicine referral, and an accident report as mentioned before and made some copies for each form. Second, some sample data were filled out either typed or handwritten. These data were fictional and not factual for the privacy issues. All the contents in each form were well related, which meant these data follow the logical medical rules. Data of any occurrence for a specific patient might be recorded consistently in different forms for the same person. There were 6 patients total with 38 sample test entries. Of the 38, 20 were typed and the remaining were handwritten. Only useful fields were filled with information after designing the method and knowing what was going to be used. Data in non-occupational and occupational treatment records were treated the same because the information for cross-references was the same. None of fields related to questionnaires were chosen because of their inconsistency.
Third, all the filled documents were then scanned at a resolution of 300 dots per inch into tif files. Fourth, we used ocrc.pl script to produce the jdb files. The usage of this utility was the following: ocrc.pl socketNo: portNo <docid> <page> <imagefile> <zonefile> > <stdout>. The interesting part here was the zone file. The basic idea for the zone file is to use the round box to best capture the character in the square. Page zoning can be done either manually by drawing a box around text to be captured or automatically by an OCR engine [5]. Manual zoning not only necessitates thousands of hours of manpower, it also requires a pre-defined set of zoning rules which we have found to be error prone [19]. On the other hand, automatic zoning is run by the recognition system’s engine and captures all the necessary data for information retrieval though not always 100% accurate. Many experiments have been done, showing that there is no statistically significant difference in the results between manual zoning and auto zoning [18]. However, these results mostly dealt with printed texts rather than handwritten texts. In this experiment, we tried both methods and chose the better results as a base for the following post processing. There are 2 ways to deal with manual zoning: simple manualzone, manualzone with handprint module for handwritten documents. We used zonecapture.pl program to capture the manual zoning files. This program takes a collection of page images and displays them one after another, allowing users to modify or to add boundary boxes specifying the areas of interest on the image. For each file, the coordinates of these boxes (along with a text identifier and zone type information) are saved to a file called a zone file. Here is an example of a zone file:

```
867 967 38 60 text text_MI
567 960 130 67 text text_firstName
248 934 143 92 text text_lastName
883 1037 44 20 label label_MI
```
In addition, we can use some handwritten recognition modules, which can be programmed in the zone file to assist OCR results. We tried all 3 methods. The results for the handwritten documents were rather poor although we made extra efforts. Printed forms achieved good results. Most of the OCR errors in the printed forms are due to the inappropriate locations of particular words—some may intersect with underlines; some meet the boundaries of the form lines.

Fifth, we used ss2pdoc utility to convert all of the jdb files for a document into a pdoc file. The pdoc file is an XML document to describe the captured information by the OCR engine.

Finally, the pdoctr utility was used to convert a pdoc file into txt file. Here is an example of a part of a txt file related to Figure 3.1.

05/23/1998
Name
Social Security No.
Company
Home Telephone No.
Lost
First
M.I.
Bechtwel Nevada?
NONOCCUPATIONAL TREATMENT RECORD
Sex
Marital Status
Date of Birth
Power Company
<table>
<thead>
<tr>
<th>Name</th>
<th>Peter Mark J.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Social Security No.</td>
<td>723 - 54 - 2806</td>
</tr>
<tr>
<td>Company</td>
<td>Power Company</td>
</tr>
<tr>
<td>Home Telephone No.</td>
<td>7023456565</td>
</tr>
<tr>
<td>Date of Birth</td>
<td>05/25/1977</td>
</tr>
</tbody>
</table>

**Nonoccupational Treatment Record**

<table>
<thead>
<tr>
<th>DATE</th>
<th>TIME IN &amp; OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>05/23/1998</td>
<td></td>
</tr>
</tbody>
</table>

*Figure 3.1: An Example of a Medicare Form of Patient Peter Mark.*

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
The whole OCR process can be viewed in Figure 3.2.

![Diagram](image)

Figure 3.2: Brief Overview of OCR Processing.

All the information was captured in the txt files and ready to be further post processed. This is the end of the OCR step. The major part of this thesis is the following post processing.

**OCR Post Processing in ORACLE**

We did the experiment in the Oracle 9i environment using PL/SQL*PLUS. Several tables were built according to our mini ER diagram. The Personal and OtherInfo are relational tables. Key and foreign key constraints must be implemented to enforce the entity and referential integrity. These two relational tables are where the query should get information after the implementation of this small database. Following is the code for OtherInfo tables:
drop table otherInfo cascade constraints;
create table otherInfo(
dt varchar2(30),
maritalStatus varchar2(20),
homePhone varchar2(20),
companyName varchar2(30),
homeAddress varchar2(100),
ID integer,
primary key (dt, ID),
foreign key (ID) references personal);

Since we knew to whom certain data belong beforehand, an artificial key ID was
added to the Personal table to differentiate different patients.

The remaining 3 non-relational tables corresponded to the 4 different forms. As
mentioned before, non-occupational and occupational treatment records were combined
as the same type of table. These electronic tables matched paper forms. We would not
need to do this experiment if the results of OCR for both typed and handwritten
documents were perfect. The following is a code for a referral table:

drop table referral cascade constraints;
create table referral (  
ID integer,
SSN varchar2(20),
lastName varchar2(20),
firstName varchar2(20),
MI varchar2(5),
dt varchar2(30),
companyName varchar2(30),
homeAddress varchar2(100),
handOrPrint integer,
oldOrNew integer );

Besides attributes mentioned in the ER diagram, some other fields were added to
further smooth out this process. The handOrPrint field differentiates whether the form is
handwritten or typed, whereas oldOrNew tells whether there is a record for a certain
person. We assumed that we knew the form type beforehand that what a particular form was and whether it was typed or written. In addition, in our implementation, an exact copy of each above non-relational table was also built to prevent problematic table mutation, which occurred in Oracle when triggers were used. The cause of this problem is that a trigger attempts to modify a table that is in the middle of being modified by the statement which fires it [8]. A copy of each non-relational table avoids this problem.

We assumed that each person had a unique SSN, one lastName, firstName etc. Since all the documents for a specific patient were compiled at the same directory, we could compare all basic personal information entries for a certain patient and then select the best. In this thesis, the characteristics of relational database—referential integrity were used to cross reference a patient record. A unique primary key determines the rest of the fields in a relational table according to the functional dependence theory, the basis for the relational database. Therefore, for a specific primary key like ID for a specific person, rest of the table’s static information was determined so that data can be cross-referenced to check its accuracy. Our approach was to count the majority of occurrences of a certain field in all printed forms. For example, if one ID had 7 occurrences of last names in printed forms, 3 were ‘William’; 2 were ‘Wllllam’; 1 was 'Ulilliianl'; and 1 was ‘Williani’. Then this algorithm would assume ‘William’ to be the correct lastName because of its number of occurrences. This approach was based on two important assumptions. First, the results of OCR in typed forms were most correct and reliable, which was the case with the OCR engine. Second, the results of handwritten forms were false and not consistent, therefore, not to be counted in the algorithm for the relational database. This algorithm is dynamic since it depends on the number of certain fields in
printed forms. With the increase of other forms for a certain patient, the result for this patient at a certain fields would change. The following is a portion of a utility function called numberInPersonal which calculates the number of occurrences in a certain field by given ID and returns that number for comparisons for future use in triggers.

```sql
create or replace function numberInPersonal
(  type_in in integer, variable_in in varchar2, content_in in varchar2)
return integer
is
cursor firstcur is
select * from copy_treatment where handOrPrint = 1;
firstrec firstcur%ROWTYPE;
//we can replace the above with othercursor from other
//copy of tables.
cnt number:=0;
begin
for firstrec in firstcur
loop
exit when firstcur%NOTFOUND;
if firstrec.ID = type_in
then
if upper(variable_in) = 'SSN'
//we can replace here with other names of attributes
then if firstrec.SSN = content_in
then
cnt:=cnt +1;
end if;
end if;
//similar code will be added here with other names of
//attributes in copy_treatment table.
...
...
end loop;
return cnt;
end;
```

Based on the aforementioned conditions, we would create some triggers to put related data to the relational database tables. Each trigger corresponded to each non relational
table. These triggers would be fired after the insertion of data to each individual table.

The following is the code for trigger of insert_referral to insert only static information in Personal:

```sql
create or replace trigger insert_referral
after insert on referral
for each row
declare cursor my_cur is
  select * from personalInfo;
my_rec my_cur%ROWTYPE;
begin
  if :new.oldOrNew = 1
  then insert into personalInfo
    (ID, SSN, lastName, firstName, MI )
    values (:new.ID,:new.SSN, :new.lastName, :new.firstName,:new.MI);
  else
    for my_rec in my_cur
    loop
      exit when my_cur%NOTFOUND;
      if numberInPersonal (:new.ID, 'SSN',:new.SSN) >
      numberInPersonal (my_rec.ID,'SSN',my_rec.SSN)
    then
      update personalInfo set SSN=:new.SSN
      where ID=:new.ID;
    end if;
    //
    ...
    ...
    end loop;
  end if;
end;
```

The following 2 tables (Table 3.1 and Table 3.2) show an example of the static information in both non-relational and relational tables if we insert the data into the database for patient Peter Mark.
Table 3.1: Static Information of Patient Peter Mark in the Non-relational Tables.

<table>
<thead>
<tr>
<th>SSN</th>
<th>Lastname</th>
<th>firstname</th>
<th>MI</th>
<th>DOB</th>
<th>sex</th>
</tr>
</thead>
<tbody>
<tr>
<td>723542156</td>
<td>^^er</td>
<td>;</td>
<td>blank</td>
<td>05^25/^AAAA^</td>
<td>M</td>
</tr>
<tr>
<td>723542806</td>
<td>Peter</td>
<td>Mark</td>
<td>i</td>
<td>05/25/1977</td>
<td>M</td>
</tr>
<tr>
<td>^3^42901</td>
<td>RHY</td>
<td>MGA</td>
<td>^</td>
<td></td>
<td></td>
</tr>
<tr>
<td>723542806</td>
<td>Peter</td>
<td>Mark</td>
<td>i</td>
<td>05/25/1977</td>
<td>M</td>
</tr>
<tr>
<td>723542806</td>
<td>Peter</td>
<td>Mark</td>
<td>i</td>
<td>05/25/1977</td>
<td>M</td>
</tr>
<tr>
<td>^35428 6</td>
<td>^24';</td>
<td>Mrk</td>
<td>^</td>
<td>15^N\Y7;</td>
<td>M</td>
</tr>
</tbody>
</table>

Table 3.2: Static Information of Patient Peter Mark in the Relational Table.

<table>
<thead>
<tr>
<th>SSN</th>
<th>lastName</th>
<th>firstName</th>
<th>MI</th>
<th>DOB</th>
<th>sex</th>
</tr>
</thead>
<tbody>
<tr>
<td>723542806</td>
<td>Peter</td>
<td>Mark</td>
<td>i</td>
<td>05/25/1977</td>
<td>M</td>
</tr>
</tbody>
</table>

In this experiment, there are 214 common fields related to Personal. Of which 76 are from handwritten, 138 from typed. Using the above relational integrity constraints, the word accuracy rates in the Personal table increase from 20/76 to 73/76 for handwritten, and 120/138 to 135/138 for typed, respectively. So the total is from 140/214 to 208/214. Using relational post processing approach achieved a great improvement here for Personal’ static information.

The second relational table called otherInfo, contained some other personal information other than name, SSN, sex and DOB, such as maritalStatus, homePhone, companyName and homeAddress. To become a relational table, we added ID and date as its primary keys. Unlike the information in the Personal table, this information may change from time to time. We still needed the utility function called numberInOther which returns the number of a certain field by a given ID and a date. The only difference between numberInPersonal and numberInOther is that numberInOther add one more
input which is date used for comparison. Here we also added one more utility function called dateSearch. It takes certain IDs and dates as inputs and then searches through OtherInfo table. It returns true if this date exists in OtherInfo table by given ID and date and vice versa. We entered the otherInfo entry by taking the date which does not exist when a specific person is given. The code for insert_referral at this time follows:

```
create or replace trigger insert_referral
after insert on referral
for each row
declare cursor my_cur is
  select * from personalInfo;
my_rec my_cur%ROWTYPE;
cursor you_cur is
  select * from OtherInfo;
you_rec you_cur%ROWTYPE;
begin
  if :new.oldOrNew = 1
    then insert into personalInfo
      (ID, SSN, lastName, firstName, MI) values
      (:new.ID, :new.SSN, :new.lastName, :new.firstName, :new.MI);
      insert into otherInfo (dt, companyName, homeAddress, ID)
      values (:new.dt, companyValue, :new.homeAddress, :new.ID);
  else if :new.handOrPrint = 0
    then insert into otherInfo(dt, companyName, homeAddress, ID)
      values (:new.dt, companyValue, :new.homeAddress, :new.ID);
  else
    for my_rec in my_cur
      loop
        exit when my_cur%NOTFOUND;
        if numberInPersonal (:new.ID, 'SSN', :new.SSN) >
          numberInPersonal (my_rec.ID, 'SSN', my_rec.SSN)
        then
          update personalInfo set SSN=:new.SSN where ID=:new.ID;
        end if;
      ...
    ...
  end loop;
  if datesearch (:new.ID, :new.dt) = 0
    then
      insert into otherInfo(dt, companyName, homeAddress, ID)
```
values (:new.dt,companyValue,:new.homeAddress,:new.ID);
else
for you_rec in you_cur
loop
exit when you_cur%NOTFOUND;
if numberInOther
(:new.ID,:new.dt,'companyName',:new.companyName)
> numberInOther
(you_rec.ID,you_rec.dt,'companyName',you_rec.companyName)
then
update otherInfo set companyName = :new.companyName where
ID=:new.ID and dt=:new.dt;
end if;
... 
... 
... 
end loop;
end if;
end if;
end if;
end if;
end;

The following is a portion of some dynamic information in both non-relational and relational tables for patient Peter Mark.

Table 3.3: Dynamic Information for Patient Peter Mark.

<table>
<thead>
<tr>
<th>CurrentDate</th>
<th>maritalStatus</th>
<th>homePhone</th>
<th>companyName</th>
<th>homeAddress</th>
</tr>
</thead>
<tbody>
<tr>
<td>09/21/1999</td>
<td>Single</td>
<td>7021435278</td>
<td>Instrument Inc.</td>
<td></td>
</tr>
<tr>
<td>05/23/1998</td>
<td>Single</td>
<td>7023456565</td>
<td>Power Company</td>
<td></td>
</tr>
<tr>
<td>11/13/2000</td>
<td>Single</td>
<td>7025935278</td>
<td>Instrument Inc.</td>
<td>4325 Spencer Rd. Las Vegas, NV89119</td>
</tr>
<tr>
<td>05/23/1998</td>
<td>Single</td>
<td>7023456565</td>
<td>Power Company</td>
<td></td>
</tr>
</tbody>
</table>
However, the technique for the Personal table did not work well for the otherInfo table for several reasons. It reduces the entry items from 38 to 36. The only 2 entry differences are those that have the same date for a certain patient in typed forms.

1. The currentDate information in the handprint forms captured by OCR is so poor that no single date is correct. According to the relational database theory, there is no way to refer integrity if the key is wrong. In this experiment, the improvement rate using the above technique is zero.

2. Even if we can recognize the currentDate information correctly somehow, the improvement rates increase 16/152. The reason is that a patient may not have many documents on the same day.

3. Even if we can have the correct currentDate information and some documents for same patients on the same day, the power of this technique is limited because different forms captured some different information for otherInfo table. For instance, the Referral form has only homeAddress and companyName information while the Treatment form has maritalStatus, homePhone and companyName.

These are the results before using other constraints. Therefore, other constraints such as domain constraint besides the entity constraints and referential integrity should be considered to improve the results. We noticed that maritalStatus has only two/three possible values. It is generally believed that few errors tend to occur in the first meaningful letter of a word according to the first position rule [6]. By the observable facts that first effective letter is more likely to be correct, we determined the content of
this field by checking the first effective letter and first letter approximation matching techniques. The accuracy rose from 17/31 to 30/31.

In addition, the companyName field also has limited size. Based on its first effective letter, the letter approximation techniques, the length of words, an arbitrary decision will be made to the content of the companyName field. The word accuracy rate increased from 20/38 to 30/38. Thus, the above two fields are more likely to be accurate than before. It is a context-based approach which only considers some possible OCR errors with the help of human judgment. It is not practical to assume that homePhone and homeAddress only have a few limited entries. The only logical way to be further assumed is that a specific person at a given date within a reasonable time span, his/her homePhone and homeAddress are more likely to be the same. The reason behind this rule is that when a patient gets medical treatments, it is more likely that there will be several different but related types of forms to be filled out within a reasonable period of time. However, this rule can only be applied to the printed forms since handwritten documents can not recognize the date field correctly. The results for this approach are not encouraging with homePhone’s improvements from 20/31 to 21/31 and no improvement for homeAddress. The sample documents collection size would also be a factor related to failure to apply this approach besides the key factor of unrecognized handwritten data in date field. The final algorithm for this thesis follows:

If first effective letter of maritalStatus equals S
    Then assign maritalStatus to Single
Else
    Assign maritalStatus to married
End if
If first effective letter of companyName equals I
    Then assign companyName to Instrument Company
End if
If first effective letter of companyName equals G or C
Then assign companyName to Computer Company
End if
If first effective letter of companyName equals T
If length of companyName is longer than 22
Then assign companyName to Tel Communications Inc
Else
Assign companyName to Technical Company
End if
End if
// similar codes are added here
...
...
...
If incoming data is New
// this patient information is new to the database
Then
Insert into Personal of related values
Insert into OtherInfo of related values
Else// this patient has some record in this database
If incoming data is handwritten
// we need to keep all this information since we can
// not recognize the handwritten documents
Then
Insert into OtherInfo of related values
Else
Put the most occurrence of a certain field of this
patient in Personal table
If datesearch of incoming data returns 0
// This patient has no record at that day
Then //approximation method
If the absolute value of date difference between
incoming data and data in the OtherInfo table for
a certain patient is less than 30 days
Then
Put the most occurrence of homeAddress or homePhone
of this patient in OtherInfo table
Else
Insert into OtherInfo of related values
End if
Else
Put the most occurrence of a certain field in
OtherInfo table
End if
End if;
End;
The following is an example of dynamic information of patient Peter Mark after the addition of constraints.

Table 3.4: Modified Dynamic Information for Patient Peter Mark.

<table>
<thead>
<tr>
<th>CurrentDate</th>
<th>maritalStatus</th>
<th>homePhone</th>
<th>companyName</th>
<th>homeAddress</th>
</tr>
</thead>
<tbody>
<tr>
<td>09/21/1999</td>
<td>Single</td>
<td>7021435278</td>
<td>Instrument Inc.</td>
<td></td>
</tr>
<tr>
<td>05/23/1998</td>
<td>Single</td>
<td>7023456565</td>
<td>Power Company</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Las Vegas, NV 891119</td>
</tr>
</tbody>
</table>

We presented the Table 3.5 with all the comparable statistics in this thesis.
Table 3.5: The Word Accuracy Comparison Statistics.

<table>
<thead>
<tr>
<th>Type of fields</th>
<th>Number of fields</th>
<th>Correct No. without relational post processing</th>
<th>Correct No. with relational post processing</th>
<th>Correctness improvement percentage of relational post processing over non¹</th>
<th>Correct No. with added constraints</th>
<th>Correctness improvement percentage of added constraints over non²</th>
</tr>
</thead>
<tbody>
<tr>
<td>Typed static data</td>
<td>138</td>
<td>120</td>
<td>135</td>
<td>15/138</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Handwritten static data</td>
<td>76</td>
<td>20</td>
<td>73</td>
<td>53/76</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Total static data</td>
<td>214</td>
<td>140</td>
<td>208</td>
<td>68/214</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Typed dynamic data</td>
<td>98</td>
<td>78</td>
<td>80</td>
<td>2/98</td>
<td>90</td>
<td>12/98</td>
</tr>
<tr>
<td>Handwritten dynamic data</td>
<td>59</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>15</td>
<td>15/59</td>
</tr>
<tr>
<td>Total dynamic data</td>
<td>157</td>
<td>78</td>
<td>80</td>
<td>2/157</td>
<td>105</td>
<td>27/157</td>
</tr>
<tr>
<td>Total data</td>
<td>371</td>
<td>218</td>
<td>288</td>
<td>70/371</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

1. We define the correctness improvement percentage of relational post processing over non as difference between correct number with relational post processing and that without the relational post processing, and then divide by the number of corresponding field.

2. We define the correctness improvement percentage of added constraints over non as difference between correct number with added constraints and that without the relational post processing, and then divide by the number of corresponding field.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
CHAPTER 4

CONCLUSION AND FURTHER RESEARCH

The design of a relational database provides a simple, yet rigorously defined, concept of how users perceive data. It is the core of this thesis and the whole project. We overviewed all the necessary components of this project, briefly introduced some previous work done by other ISRI staffs and students which are the preliminary for this thesis, described the idea of this thesis, illustrated the process, and presented the results and statistics.

Relational post processing seems great at the very first beginning. We need to use the database to capture, store, track, access, query all the information on medical charts. As we can foresee, the accuracy of some information obtained from the patient’s file which does not change over time can be greatly improved by the introduction of relational database’s constraints to the OCR technology in the post processing phase, especially for some harder problems like handwritten texts. This greatly reduces the cost and time for people to correct all these OCR errors afterwards if the OCR technology is used. However, the accuracy of other dynamic information on patients can not be improved significantly due to the misrecognized data in the key field because of the poor performance of handwritten recognition and some characteristics of dynamic information.

The constructs of relational database might provide a way to improve the OCR output. This is a kind of reverse engineering. The essence and preliminary condition of this
approach are to determine the key and to ensure the accuracy of the key. Without a big improvement in the handwritten module of the OCR technology, it is extremely hard to take this approach. Our conclusion is that only logically related and repeated data can adopt this relational database constraints approach to post process the document text output from an OCR engine. The requirements would improve documents retrievability.

Future research plans are to consider other approaches to post processing the OCR output, and implement the physical model of this project.
BIBLIOGRAPHY


VITA

Graduate College
University Of Nevada, Las Vegas

Zhenxing Mao

Local Address:
1600 E. University Ave #130
Las Vegas, NV 89119

Degrees:
Bachelor of Science, Tourism Economics, 1996
Hangzhou University, P.R. China

Master of Science, Hotel Administration, 2001
University of Nevada, Las Vegas

Thesis Title
A Relational Post Processing Approach For Forms Recognition

Thesis Examination Committee:
Chairperson, Kazem Taghva, Ph. D.
Committee Member, Thomas A. Nartker, Ph. D.
Committee Member, Wolfgang W. Bein, Ph. D.
Graduate Faculty Representative, Emma Regentova, Ph. D.