Contextual analysis of machine printed addresses

Peter B. Cullen, Tin Kam Ho, Jonathan J. Hull
Michal Prussak and Sargur N. Srihari

Center of Excellence for Document Analysis and Recognition
State University of New York at Buffalo
226 Bell Hall, Buffalo, New York 14260

ABSTRACT

The assignment of a nine digit ZIP Code (ZIP+4 Code) to the digital image of a machine printed address block is a problem of central importance in automated mail sorting. This problem is especially difficult since most addresses do not contain ZIP+4 Codes and often the information that must be read to match an address to one of the 28 million entries in the ZIP+4 file is either erroneous, incomplete or missing altogether.

This paper discusses a system for interpreting a machine printed address and assigning a ZIP+4 Code that uses a constraint satisfaction approach. Words in an address block are first segmented and parsed to assign probable semantic categories. Word images are then recognized by a combination of digit, character and word recognition algorithms. The control structure uses a constraint satisfaction problem solving approach to match the recognition results to an entry in the ZIP+4 file. It is shown how this technique can both determine correct responses as well as compensate for incomplete or erroneous information.

Experimental results demonstrate the success of this system. In a recent test on over 1000 machine printed address blocks, the ZIP+4 encode rate was over 73 percent. This compares to the success rate of current postal OCRs which is about 45 percent. Additionally, the word recognition algorithm recognizes over 92 percent of the input images (over 98 percent in the top 10 choices).

1 Introduction

Assigning nine-digit ZIP Codes (ZIP+4 Codes) to mailpieces is of central importance in automated mail processing. Most addresses only contain the traditional five-digit ZIP Code. This information can only specify a single city or a large portion of a city. However, the ZIP+4 Code specifies a small portion of a street. Thus, the goal of this automation effort is to find ZIP+4 Codes for those mailpieces that don't contain them. This can be done by recognizing other information on the mailpiece and then using this information to look up the ZIP+4 Code in the ZIP+4 file.

For example, the address block shown in Figure 1* does not contain a ZIP+4 Code. However, recognizing that the city word is "DALLAS", the ZIP Code is "75240", and the street address is "5000 QUORUM DR", we can look in the ZIP+4 file for a single entry that matches this address. In this case there is an entry that matches all of this information and the ZIP+4 code is 75240-7506.

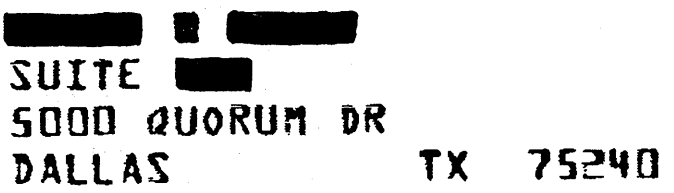

Figure 1: Sample Machine Printed Address Block

*Portions of addresses in this paper have been modified or blocked out to preserve confidentiality

Machine Vision Applications in Character Recognition and Industrial Inspection,
SPIE Vol. 1661, San Jose, CA, Feb. 10-12, 1992, pp. 257-268.

The range of addresses that correspond to each ZIP+4 code are described by a single record in the national ZIP+4 file. The United States Postal Service (USPS) designed the ZIP+4 Code so that mail can be automatically sorted by an Optical Character Reader (OCR) down to the level of individual carriers and sections of their routes. A further desire of USPS automation efforts is to sort the mail into carrier walk sequence. This is the order that a carrier visits the customers on his/her route. It should be noted that both of these goals depend on the ability of OCRs to read and assign ZIP+4 Codes.

Current postal OCRs can, to a certain degree, read ZIP+4 Codes if they appear on a mailpiece. However, the ZIP+4 Code has not been fully accepted by the American public and only a small amount of mail bears the ZIP+4 Code. To counter this problem, USPS has been deploying OCRs that can read several lines of text in an address and lookup the proper ZIP+4 Code. However, this strategy has met with limited success because of processing speed requirements for the OCRs (about ten pieces per second) and the limits that this places on the recognition techniques that can be used.

This paper discusses a research effort that seeks to improve the ability of OCRs to assign ZIP+4 Codes to addresses that may only contain a five-digit ZIP Code or no ZIP Code at all. This is done by applying advanced word recognition techniques[1] that avoid various recognition problems as well as a control structure based on the constraint satisfaction paradigm[2] which exploits the contextual information found in the address block.

An additional goal of this system is to overcome various uncertainties that can be found when interpreting an image. Only half of the address block images tested by our system can be assigned a ZIP+4 Code just by matching the address information to the ZIP+4 file, assuming we have perfect word recognition. The other half require extra processing to find the ZIP+4 Code. This extra processing is designed to overcome three kinds of uncertainties. First, an incomplete image may be input. For example, in the case of an address block, the street suffix may be left out. Second, errors may be present in the image, such as misspellings or an incorrect street number. Finally, recognition errors may have occurred. Each of these uncertainties are illustrated in Figure 2. The city name has been obscured during image acquisition, the patron has actually made an error in the ZIP Code, and word recognition was unable to perform correctly on the suffix due to poor image quality.

Our system attempts to overcome such uncertainties using the constraints provided by the ZIP+4 file. In addition to providing the ZIP+4 Code for a given address, we are able to query this database and ask whether certain pieces of information are compatible. The control structure for our system is then able to overcome such uncertainties by using only information from the address block that is compatible.

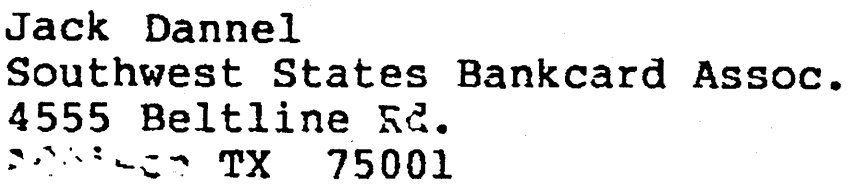

Figure 2: Address Block Image Containing Uncertainties

The remainder of this paper provides an overview of our contextual analysis system, describes the major algorithms used in the system and presents a detailed example of how the system processes a machine printed address. An evaluation of the performance and experimental results are also included.

2 Overview

The flow of control in the contextual analysis system is shown in Figure 3. The digital image of an address block is input to the system.

2.1 Segmentation/Parsing

The first step is segmentation/parsing. This step segments the image into lines of text and then breaks up each line into individual word images. The parser assigns semantic categories to each word. The assignment for a given word is based on structural characteristics of the address block, including the number of lines in the address block, the number

Machine Vision Applications in Character Recognition and Industrial Inspection,
SPIE Vol. 1661, San Jose, CA, Feb. 10-12, 1992, pp. 257-268.

of words in the given word's line, and the number of characters in the given words. This information is then used to look up the most probable category for the given word. Additional processing by the parser compensates for redundant words on the same line and redundant configurations of lines in the same address block.

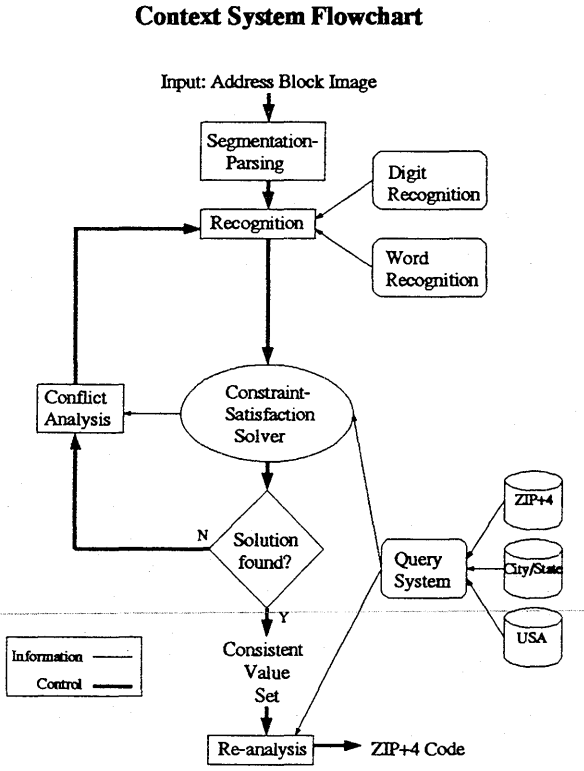

Figure 3: Control Flow Diagram

For example, once the most probable category for each word has been assigned, the parser would check for two words both assigned the ZIP Code category. One of these would be changed to the next most probably category since it is unlikely that an address block would contain two ZIP Codes. The removal of redundant information on a given line is referred to as the horizontal step since it uses the horizontal context of lines in address blocks. The parser also checks for redundant line configurations. Just as it is unlikely to find two ZIP Codes in an address block, it is unlikely to find two city-state-ZIP lines. The removal of redundant line configurations is referred to as the vertical step since it uses the vertical context of an address block.

2.2 Recognition

The next step in the system is recognition. Depending on the semantic category assigned by the parser, either digit recognition or word recognition is used. Digit recognition is used for ZIP Codes, street numbers, box numbers and so on. The digit recognizer assigns one or more choices (0...9) to each image. Word recognition is used for city words, street name words, street suffixes, etc. The word recognizer is less constrained. It uses the semantic category to choose a lexicon of possible words. That is, there are separate lexicons for city words, street words, suffixes and so on. The word recognizer then ranks the lexicon in descending order in terms of what it thinks is the correct word. Due to processing limitations, the system only uses the top twenty choices.

2.3 Constraint Satisfaction Solver

The parsing and recognition results are then sent to the constraint satisfaction solver. This part of the system attempts to assign a single recognition choice to each word in the address block such that all assigned choices, when taken together, match an entry in the ZIP+4 file. Since each word may have more than one recognition choice this part of the system must use some "intelligence" to select choices that are most compatible. A blind, brute force approach would be to try all possible combinations of assignments for all the words. This would be too time consuming though. Instead, the constraint satisfaction solver initially uses the top recognition choice for each word. Then it compares two words at a time to see if they are compatible. If the choices assigned to each of the words are compatible then a solution has been found. Otherwise the most incompatible word has another recognition choice assigned and the process repeats.

If a solution cannot be found, the system proceeds to the section labeled conflict analysis. This part of the system either modifies the recognition results for a word or removes it from consideration altogether. This word is chosen as the most likely candidate that is preventing the system from finding a solution. This step is possible because many addresses contain redundant information. So parts of the address can be removed without affecting the chance that the correct interpretation can be found. Once the modification or removal has taken place the system returns to the constraint satisfaction solver to once again try and find a solution.

2.4 Reanalysis

Once a solution has been found, the system proceeds to the re-analysis step. This is where the ZIP+4 Code is assigned. Each address block might match several ZIP+4 Codes. The most general corresponds to a record that matches a range of houses or buildings on a given street. A finer ZIP+4 Code corresponds to an actual building on a street. Getting even more specific, there could be a ZIP+4 Code for all suites or apartments on a single floor in a building or there could even be a unique ZIP+4 Code for a given office in a building. The reanalysis section attempts to find the most specific ZIP+4 Code possible using the solution found in the constraint satisfaction solver.

2.5 ZIP+4 Access

An extremely important part of the context system is querying the ZIP+4 database. This is accessed from the constraint-satisfaction solver and the reanalysis step. This is a multi-access database system that supports of variety of queries. A given query may consist of all the information given in an address block or may be as simple as whether a given city and ZIP Code are compatible. Most of the queries only require a yes/no answer. Others expect a list of ZIP+4 Codes to be returned. The query system accesses the city/state file, the DSF (Delivery Sequence File) as well as the ZIP+4 file. The city/state file relates ZIP Codes to city and state names. The DSF file contains carrier walk sequence information. Due to the enormity of this information (the national ZIP+4 file contains over 28 million records), it must be compressed. Thus, an additional effort in developing this system has been the compression and access of this data.

3 Algorithm Descriptions

This section describes the major algorithms used in the system with particular emphasis on the constraint satisfaction solver, which uses the results of the other algorithms. The other algorithms covered in this section are used for parsing, word recognition, and database compression.

3.1 Parsing

The parser uses a bottom-up algorithm[3] to classify the lines and words in the address block. There are three major steps in the algorithm that use progressively more contextual information.

3.1.1 Individual Word Parsing

The first step is to classify the type of each word in the address block individually. In this step, the information used is the position of the word in the address block (line and word number), its length, whether it contains letters or digits and more global characteristics of the address block such as the number of lines and the number words on the line of the word being parsed. Discrimination between words that contain only letters and words that contain digits is performed with an algorithm based on template matching character recognition. Such an algorithm uses a large set of templates

Machine Vision Applications in Character Recognition and Industrial Inspection,
SPIE Vol. 1661, San Jose, CA, Feb. 10-12, 1992, pp. 257-268.

whose meaning is known. Then the meaning of the template that most closely matches the input is assigned. In this case, all we are concerned about is whether the template that matched was a digit or a letter.

The above information is used to access a database of frequencies of words of different types occurring at various locations in a set of sample address blocks. For example, the system may be given an address block with three lines, the bottom line containing three words and the last word consisting of 5 digits. To assign a category to the last word on the bottom line, the above information would be used to lookup the most probable category for that word, in this case, the ZIP Code.

3.1.2 Horizontal Parsing

The second (horizontal) step uses the context of words within a line to choose a configuration of word types for a whole line, given a list of choices for each word from the first step. This step is accomplished by using frequencies of occurrence of various configurations of words on various lines together with the confidences of word types computed in the first step. This step of the algorithm works by considering all possible configurations of words in the given line. This promotes more likely category assignments based on the context of the given line.

3.1.3 Vertical Parsing

The third (vertical) step uses the context between the lines of the address block in order to choose the best combination of line types for the address block. In this step the results of the horizontal parser for each line are collected and each choice is labeled by the kind of information it contains. If a line contains any street words the line is labeled a street line; a line with city, state, and ZIP Code is labeled city-state-ZIP line. These labelings are referred to as line classifications. The purpose of this step is to select from each line all choices that have the same line classification and in this way to select the best configuration of line classifications in the address block.

3.1.4 Keyword Recognition

After the vertical step of the algorithm is performed, keyword recognition is applied to all the word images. Keyword recognition is simply a recognition procedure that looks for words in the address block which are contained in a small lexicon of keywords. Example keywords are: PO, BOX, RURAL, APT, etc. The keyword recognition procedure makes use of a template matching character recognition algorithm. If any keywords are recognized in the address block, then the output of the vertical step is re-ranked to reflect the identified keywords.

3.1.5 Parsing Results

The result of these steps is a ranked list of vertical configurations for the words in the address block. Each vertical configuration may have a ranked list of horizontal configurations for each line. This way, the system can attempt to process the address block using several different configurations from the parser. Figure 4 shows an example address block and the top two vertical configurations as output by the parser. Each configuration contains the line number, the line classification and the individual category assignments for each word. For example, in the top configuration, the first choice for the street line incorrectly assigns the street suffix category (ss) to the last word. It should in fact be a secondary number (cn). This is the case in the second choice for the street line. By using more than a single choice from the parser the system is much more likely to develop a correct interpretation for the address block.

The parsing system was tested on 748 images of address blocks. It correctly identified the category for each word in 70 percent of the images. It should be noted however that the control structure is flexible enough to overcome certain parsing errors and still assign a ZIP+4 Code.

3.2 Word Recognition

Word images are recognized by an algorithm that integrates isolated character recognition with word shape analysis[4]. The algorithm is designed to recognize word images of a wide range of font types and qualities that are typically seen in address block images. A lexicon and a word image are input to the algorithm, which produces a ranking of the lexicon.

```
1 city-state-zip cw sw z5
1 city-state-zip cw sw z9
2 street sn sp st ss
2 street sn sp st cn
2 street sn sp st st
2 street sn st st ss
3 recipient nn nn nn
3 recipient nn nn ow

1 city-state-zip cw sw z5
1 city-state-zip cw sw z9
2 po-box pw pw pi pn
3 recipient nn nn nn
3 recipient nn nn ow
```

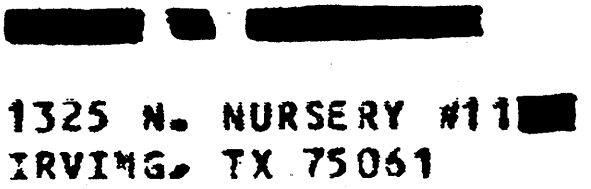

Figure 4: Address Block and Parse Results

3.2.1 Word Recognition Algorithm

The algorithm consists of a set of serial filters and a set of parallel classifiers as illustrated in Figure 5. The filters reduce the input lexicon based on reliable shape information extracted from the image. The parallel classifiers produce independent rankings of the lexicon based on different features and contextual analyses. These rankings are combined to generate a final decision.

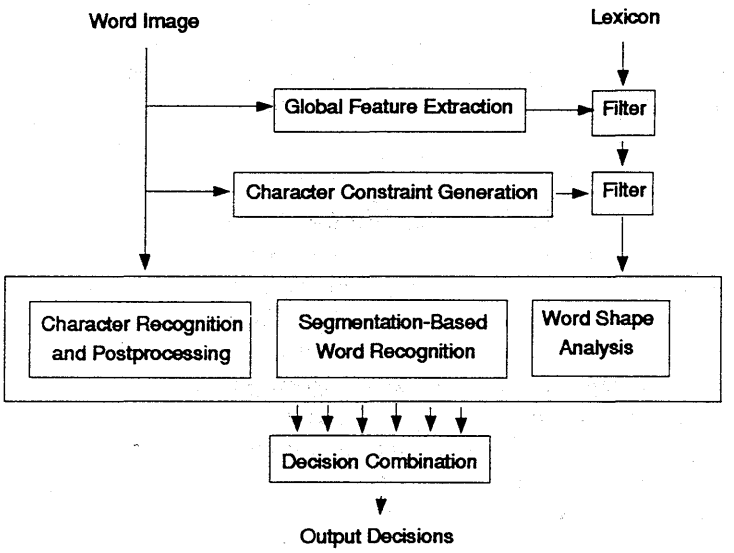

Figure 5: The word recognition algorithm.

The first filter makes use of some global features computed from an input image, including a range estimate of the word length, and an estimate of the word case. Word prototypes in the lexicon are retained if they match the computed length and case. The second filter makes use of a set of decisions on the identities of characters segmented from the word image. The decisions are from several character classifiers that include a template matcher, a pixel based Bayesian classifier, and a feature based nearest neighbor classifier. The decisions are assigned a confidence score based on the

Machine Vision Applications in Character Recognition and Industrial Inspection,
SPIE Vol. 1661, San Jose, CA, Feb. 10-12, 1992, pp. 257-268.

agreement among the several classifiers. The most reliable ones are used to construct some constraints, which are then used to filter the lexicon.

The parallel classifiers are based on three different approaches, including (1) character recognition methods (2) segmentation-based word recognition, and (3) word shape analysis methods. In the character recognition approach, segmented characters are first recognized individually and then post-processed with the contextual constraints given in the lexicon. In the segmentation based recognition approach, features of segmented characters are extracted and matched under the constraints of the word context. In the word shape analysis approach, features of the word as a whole unit are extracted and matched. A control strategy is designed which selectively activates appropriate classifiers and combines their decisions.

A decision combination strategy combines the rankings produced by the collection of classifiers and derives a consensus ranking. It first determines a candidate set of words from these rankings by taking the union of a small number of top decisions from each ranking. The words in the candidate set are then re-ranked by summing the ranks assigned by the individual classifiers.

3.2.2 Word Recognition Performance

The recognition algorithm has been developed using a collection of machine-printed postal words obtained from live mail. They were scanned on a postal optical character reader at roughly 200 pixels per inch and binarized. Figure 6 shows some example images in our data set. As can be seen in these examples, the image quality is very unstable and the characters can be severely broken or touching each other. The unrestricted font variation also can cause substantial failures in a conventional text recognition system. In an experiment with a lexicon of 1995 words and 1055 test images from this database, the algorithm achieved a recognition rate of 93% at the top choice and 98% in the top 10 choices.

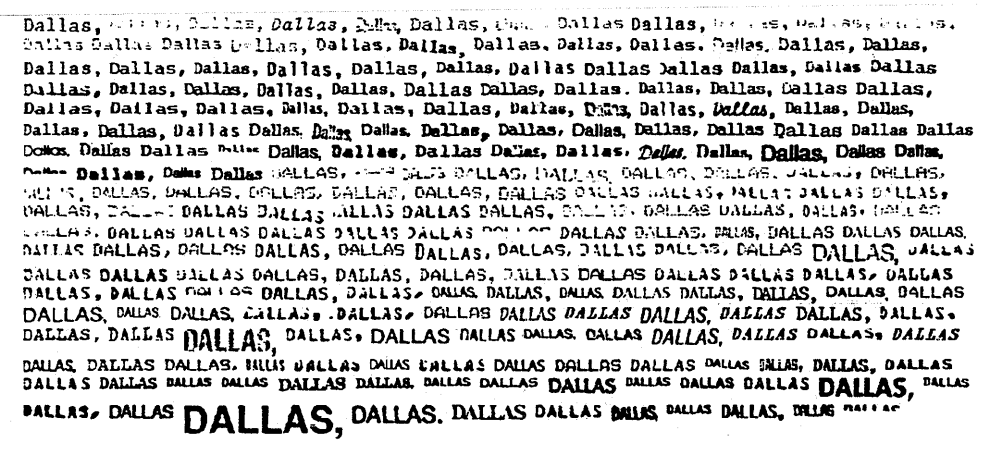

Figure 6: Examples of image degradation and font style variations included in the test database.

3.3 Database Compression and Access

The context system is currently working with a subset of the national ZIP+4 database which includes ZIP Codes 75000-75399. This subset contains over three hundred thousand ZIP+4 records and occupies over 40 megabytes (40 million bytes) in its uncompressed form. Figure 7 shows some examples of the data found in the ZIP+4 database. From left to right the figure shows the ZIP Code, ZIP+4 record type, first and second street words, street suffix, lower bound on the street number range, upper bound on the street number range, street number odd/even code, lower bound of the addon range and upper bound of the addon range. Since the control structure needs to be able to make rapid, versatile queries of the data, a simple linear search would be unsatisfactory. This type of search would look at one record after another and on average would need to examine half the records in the database. The objectives in designing the compression and access software were to: allow flexible queries, have fast access to the database and reduce its size.

Machine Vision Applications in Character Recognition and Industrial Inspection,
SPIE Vol. 1661, San Jose, CA, Feb. 10-12, 1992, pp. 257-268.

```
75248 10 HIDDEN GLEN DR 17200 17298 E 1336 1336
75248 10 HIDDEN GLEN DR 17201 17299 O 1335 1335
```

Figure 7: Sample ZIP+4 records

To accomplish these objectives we have implemented a two-level solution. To reduce the size of the database, we first compressed the actual ZIP+4 file (also called the flat file). We also removed information in this file which was not pertinent to our system. The second step was to build data structures on top of the compressed flat file. The data structures allow efficient access to the records in the database that match any specified fields.

3.3.1 Compressed Flat File

Two different methods were used to compress the flat file. For data fields which have a limited domain, we can list all the possible values for that field and assign a number to each. Then we only need to store that number which takes much less space. For example, there are only 142 legal street suffixes each of which is from two to four letters long. Each letter requires a byte. A single byte, however, can be used to store a number between 0 and 255. By assigning a number to each suffix we can encode every suffix using only one byte instead of four.

Strings from the ZIP+4 file were stored in a character heap. A heap is a large array of characters used to store strings in which no string appears more than once. This method of compression takes advantage of the redundancy in the database to reduce the overall record size. For example, many records have the same information in the street name field (as shown in the ZIP+4 example, both records have a street name of "HIDDEN GLEN"), but this data need only be stored once.

Using these two methods of compression, we were able to reduce the record size from 132 bytes used in the flat file down to 36 bytes. As the records are compressed, they are stored in an array, and the index for each record in this array can be treated as a unique key for the record. This key number is then used by other parts of the database to quickly retrieve individual records.

3.3.2 Data Structures for Flat File Access

Our system makes use of two data structures which allow access to the compressed flat file. For fields in which a record has a range of values, we use sorted arrays of range information. Each entry in these arrays contains a beginning and ending number for the range. Associated with each entry is a list of keys that correspond to all the records that contain the data in the specified range. As an example, an array for primary address numbers (street numbers) may have an entry whose beginning number is 100 and whose ending number is 198. There would be a list of keys attached to this entry which contained the key for every record that has a primary address between 100 and 198.

Since these range arrays are sorted, a binary search can be performed which is substantially faster than a linear one.

All other types of data are stored in binary trees. Each node in the binary tree contains the data for that node and a list of keys corresponding to records that contain that data. Since these are binary trees they can be searched using a binary search.

Figure 8 illustrates the compression and access mechanism that we have implemented. It shows representations of the uncompressed and compressed ZIP+4 file. In addition the figure shows an example entry in the range array which points to all records in the compressed file that match that range. When the ZIP+4 file is queried on an even street number between 17200 and 17298, it will match the entry shown in the range array and retrieve all the keys that point to the compressed ZIP+4 file.

3.3.3 Query Implementation

The general strategy for querying the database is to: identify the fields in the current query, find the list of keys using the appropriate data structure, and intersect all the lists of keys. If the query only needs a yes or no answer, yes is returned if the intersection is not empty, no is returned otherwise. If actual ZIP+4 Codes are requested, the list of keys that resulted from the intersection is used to index the array of compressed records and the ZIP+4 codes are returned.

Machine Vision Applications in Character Recognition and Industrial Inspection,
SPIE Vol. 1661, San Jose, CA, Feb. 10-12, 1992, pp. 257-268.

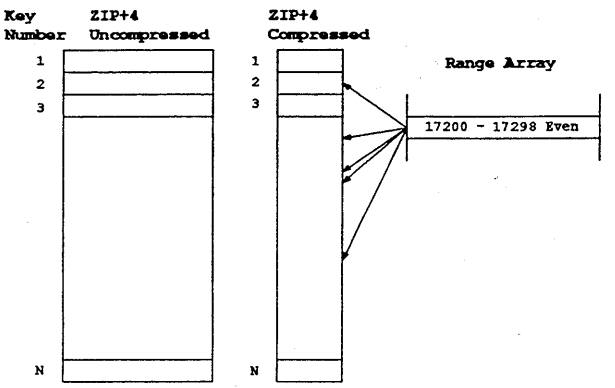

Figure 8: ZIP+4 Compression and Access

3.4 Constraint Satisfaction

The constraint satisfaction solver in our system is based on the constraint satisfaction paradigm. This paradigm is used to solve constraint satisfaction problems (CSPs). A constraint satisfaction problem consists of n variables, X1...Xn, each having an associated domain of possible values. Each variable can be assigned any value in its domain but *constraints* are imposed on the legal combinations of values that two or more variables can take on. A solution to the CSP consists of a set of values, one for each variable such that the constraints over all variables are satisfied.

The classic example of a CSP is the N-Queens problem. The goal is to place N Queens on an N by N chess board such that no two queens are attacking each other horizontally, vertically or diagonally. In terms of a CSP, the N variables are the positions of the N queens on the chess board and the constraints are all pairwise: no two queens are attacking each other. A standard approach to this problem is to place only one queen in each column, removing the vertical constraint. A brute-force solution, called backtracking, is to start each queen in the first row. Then, starting with the second column, for each queen that is in conflict with another, move it to a lower row. If a row can't be found for some queen where it is not in conflict then backtrack to one of the previous queens and move it to a lower row. This is where the algorithm gets its name. This search is brute-force because it doesn't consider how good or bad a given assignment is, only whether it is in conflict or not.

Informed backtracking[5] is a more intelligent backtracking algorithm. It differs from backtracking in the following way: when a variable is in conflict and must be assigned a new value, instead of simply choosing the next value in the domain, it considers all values in the domain and determines which one would have the fewest conflicts. This value is then assigned and the process continues until a solution is found. This algorithm is much less likely to backtrack which is a major problem with the backtracking algorithm. This type of algorithm is easily applied to interpreting postal images as well as other image interpretation problems that are similarly constrained.

3.4.1 Constraint Satisfaction Solver

The constraint satisfaction paradigm maps very easily to postal image interpretation. Each word in the address block image is considered to be a variable in the CSP. The domains for each variable are provided by the recognition routines. Finally the constraints are imposed by the ZIP+4 file. In the N-Queens problem, it is easy to visualize the constraints. It may not be so easy to visualize the constraints provided by the ZIP+4 file, however. These constraints are accessed using the query system previously discussed. Figure 9 illustrates a graph of the various fields in the ZIP+4 that can be queried. Notice that every field is linked to every other field. Thus, every piece of information provides a constraint on every other piece of information. For example, we may be interested in the constraints between the ZIP Code (z5) and the first street word (st1). A query in this case might take the form of the ZIP Code being "75248" and the street word being "HIDDEN". In this case the query would return "yes" since this is a legal combination of values. That is, these two values satisfy the constraints imposed by the ZIP+4 file.

The constraint satisfaction solver uses the informed backtracking algorithm to assign a value (recognition choice) to

Machine Vision Applications in Character Recognition and Industrial Inspection,
SPIE Vol. 1661, San Jose, CA, Feb. 10-12, 1992, pp. 257-268.

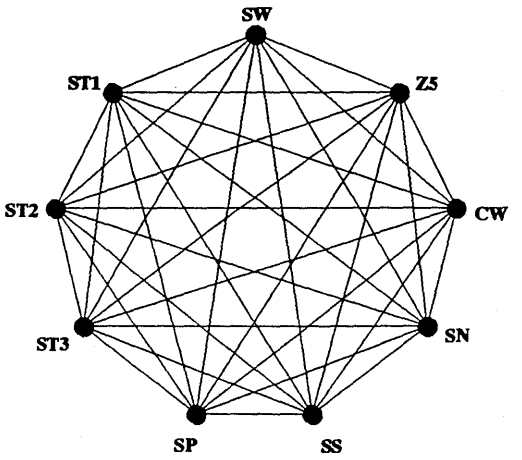

Figure 9: Constraints from the ZIP+4 File

each variable (word) in the address block. An initial assignment for each variable is made using the top recognition choice. If all these values, taken together, do not satisfy the constraints of the ZIP+4 database, the variable whose value has the most conflicts with the other assigned values is repaired. As described in the previous section, this repair process does not blindly choose the next choice from the recognition routine, but considers all choices. Each choice is compared to the values of the other variables and the one with the fewest conflicts (not necessarily zero) is assigned and the process repeats until a solution is found.

3.4.2 Conflict Analysis

This section is closely tied to the constraint satisfaction solver in that is uses information about which variables have been repaired in selecting a variable to be removed or modified, also known as the *culprit*. This section uses a heuristic or rule that says that the most likely candidate to be the culprit is the variable that was repaired first. Depending on the category of the selected culprit, various methods may be tried to improve or modify the recognition results. If nothing can be done to change them, then the culprit will be removed from further processing. A good example of this is when the culprit is the rural route number. Many times this is written as "#2.". This can cause the recognition program to treat the "#" and the "." as digits. A simple modification to the recognition results in this case would be to drop the first or last digit choices from the recognition results.

In some cases, information should be removed. For example, if the person who wrote this address, the patron, made an error in the ZIP Code, we would want to remove that piece of information from further consideration. In this way our system is able to overcome patron errors in the address block.

In the future, conflict analysis will be able to call more time consuming recognition routines that have better performance but take considerably longer to run. In this way, those words that are easily recognized by a simple algorithm can be processed more quickly instead of all words being processed by the same slow technique.

3.4.3 Reanalysis

The reanalysis section is also closely tied to the constraint satisfaction solver. It uses the solution found by constraint satisfaction to lookup the deepest level of ZIP+4 Code for the given solution. The deepest levels of ZIP+4 Coding require some sort of secondary information like suite or apartment number. If this information was recognized on the address block, it is used to identify the corresponding ZIP+4 Codes.

4 System Example

This section illustrates how our system goes about processing an address block. The address block shown in Figure 10 is input to the system. To simplify this example, we will only use the correct parse which can easily be identified in this case.

Machine Vision Applications in Character Recognition and Industrial Inspection,
SPIE Vol. 1661, San Jose, CA, Feb. 10-12, 1992, pp. 257-268.

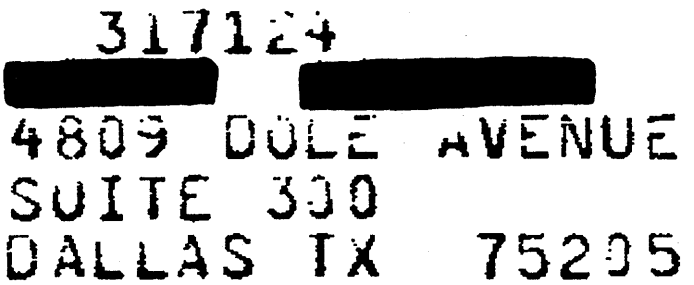

Figure 10: Sample Machine Printed Address Block

The next step is to recognize the words that make up the address. For this address block, we will recognize the city name (cw), the ZIP Code (z5), the street number (sn), the street name (st) and the street suffix (ss). The suite number will be recognized and possibly used later in the reanalysis section. The following are the actual recognition results for the above words as provided by the digit and word recognition algorithms, ranked in descending order of confidence.

cw : DALLAS BARREL LANCASTER FERRIS MABANK WILMER LITTLE MURPHY CELINA

z5 : 75205 75203

sn : 4809 4805 9809 9805 1809 6809 8809 1805 6805 8805

st : BUTE IDLE DOLE GOLL DALE CASS CORE GALE CUBA OSLO GATE CARO CATE COLD IOLA LORE

ss : AVE CIR SQ SPGS GRV KNLS TRL HWY GDNS GRN MDWS RDG TPKE BRK DR PT VLG CTS MNR

The recognition and parse results are then input to the constraint satisfaction solver. The first step is to query the ZIP+4 file using the top recognition choices. The query would take the form of cw:DALLAS, z5:75205, sn:4809, st:BUTE, ss:AVE. The query returns a *no* answer for these values, so the system proceeds to count the number of conflicts each word has with all the other words. For example, the city word value, DALLAS, agrees with all the other values(75205, 4809, BUTE, AVE). However, the street word value, BUTE, is in conflict with the ZIP Code, street number and street suffix, all of which are only in conflict with it. Thus, the street name has the greatest number of conflicts (three) and should be repaired.

As stated previously, this repair consists of counting the number of conflicts each value in the street word domain has with the other variables. We know that BUTE has three, but by doing this we also discover that DOLE has none. All the other choices have at least two, so we change the street names value to DOLE and query the ZIP+4 file with this set of values: cw:DALLAS, z5:75205, sn:4809, st:*DOLE*, ss:AVE. The result of this query is *yes* and we have found a solution, which is easily seen as the correct one. Now the ZIP+4 Code(s) for this address block can be assigned.

To lookup the ZIP+4 Code(s), the reanalysis section is given the solution found by the constraint satisfaction solver. The ZIP+4 Codes found for this address block are shown in Figure 11. The first is a street range record, which can be assigned to any address on DOLE AVE with an odd street number between 4801 and 4899. The second ZIP+4 Code for this address is a high-rise default, which is meant specifically for the building located at 4809 DOLE AVE. The third record is a high-rise specific which can be assigned to any office or suite at 4809 DOLE AVE with a suite number between 300 and 399. The final ZIP+4 record is a firm record which is meant only for suite 300 at 4809 DOLE AVE. The firm for this address is given in the record as "SOUTHERN LIVING". Since we cannot verify this on the mailpiece, the finest level of sortation our system would output for this piece would be the ZIP+4 Code associated with the third record, 75205-3581.

```
75205 10 DOLE AVE 4801 4899 O 3523 3523
75205 12 DOLE AVE 4809 4809 B 3552 3552
75205 20 DOLE AVE 4809 4809 B 3581 3581 300 399
75205 21 DOLE AVE 4809 4809 B 3596 3596 300 300 SOUTHERN LIVING
```

Figure 11: ZIP+4 records for 4809 DOLE AVE, DALLAS, 75205

Machine Vision Applications in Character Recognition and Industrial Inspection,
SPIE Vol. 1661, San Jose, CA, Feb. 10-12, 1992, pp. 257-268.

5 Experimental Results

The context system was tested on 1013 address block images which had not been previously seen. The output was a sortation assignment for each address block. In some cases the system was not able to resolve the address down to a nine-digit ZIP+4 Code. In such cases the system would output only the five-digit ZIP Code or possibly reject the image altogether.

In the test, our system was able to encode 462 (45.61%) to nine-digits. This compares to the encode rate of a current postal OCR of 165 (16.29%). Both our system and the OCR made errors in about 10% of the nine-digit assignments. An important point to note is that these 1013 images are divided up based on the sortation performance of a current postal OCR. USPS is most interested in developing technology that can correctly process the rejects of a current OCR. With that in mind, the test contains many more images that were rejected by the current OCR. Thus, if our system were to be tested on a more representative mailstream, the projected encode rate for our system would be 73%, while the encode of the OCR would be only 45%.

6 Conclusions

We have shown a system which is able to encode 73% of machine printed address blocks to a nine-digit ZIP+4 Code. The system uses various levels of contextual knowledge in processing the address block. Words from the address are recognized and then matched to the ZIP+4 file using an intelligent backtracking algorithm. The result is a system that can assign ZIP+4 Codes to address blocks even when the ZIP+4 Code is not present.

7 Acknowledgments

This work was supported by the Office of Advanced Technology of the United States Postal Service under contract 104230-84D-0962, Task Order No. 104230-88D-2576. The authors wish to thank Carl O'Connor of the Office of Advanced Technology of the USPS and Gilles Houle and Gerardo Garcia of Arthur D. Little Inc. for their guidance and support of this work.

References

[1] Jonathan J. Hull and Sargur N. Srihari. A computational approach to visual word recognition: Hypothesis generation and testing. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 156–161, June 1986. Miami Beach, Florida.

[2] A. K. Mackworth. Constraint satisfaction. In *Encyclopedia of Artificial Intelligence*, volume 1, pages 205–210. John Wiley and Sons, 1987.

[3] Michal Prussak and Jonathan J. Hull. A multi-level pattern matching method for text image parsing. In *Proceedings of the Seventh IEEE Conference on Artificial Intelligence Applications*, February 1991. Miami, Florida.

[4] J.J. Hull T.K. Ho and S.N. Srihari. Word recognition with multi-level contextual knowledge. In *Proceedings of the first International Conference on Document Analysis and Recognition*, September 30 - October 2 1991. Saint-Malo, France.

[5] S Minton, M D Johnston, A B Philips, and P Laird. Solving large-scale constraint satisfaction and scheduling problems using a heuristic repair method. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, pages 17–24, July 1990. Boston, Massachusetts.

Machine Vision Applications in Character Recognition and Industrial Inspection,
SPIE Vol. 1661, San Jose, CA, Feb. 10-12, 1992, pp. 257-268.