A Constraint Satisfaction Approach to Geospatial Reasoning

Martin Michalowski and **Craig A. Knoblock**
University of Southern California
Information Sciences Institute,
4676 Admiralty Way
Marina del Rey, CA 90292 USA
{martinm, knoblock}@isi.edu

Abstract

The large number of data sources on the Internet can be used to augment and verify the accuracy of geospatial sources, such as gazetteers and annotated satellite imagery. Data sources such as satellite imagery, maps, gazetteers and vector data have been traditionally used in *geographic information systems* (GIS), but nontraditional geospatial data, such as online phone books and property records are more difficult to relate to imagery. In this paper, we present a novel approach to combining extracted information from imagery, road vector data, and online data sources. We represent the problem of identifying buildings in satellite images as a *constraint satisfaction problem* (CSP) and use constraint programming to solve it. We apply this technique to real-world data sources in El Segundo, CA and our experimental evaluation shows how this approach can accurately identify buildings when provided with both traditional and nontraditional data sources.

Introduction

The ability to reason over geospatial entities using publicly available information is greatly enhanced by the abundance of geospatial data sources on the Internet. Traditional data sources such as satellite imagery, maps, gazetteers and vector data have long been used in *geographic information systems* (GIS). However, incorporating non-traditional sources such as phone books and property tax sites brings to light integration issues that have not previously been dealt with. For example, it is not clear how phone book information (i.e. street name and building number) could be combined with road vector data[1] to label buildings found in a satellite image.

However, combining traditional and non-traditional data sources provides the ability to verify the accuracy of geospatial databases such as gazetteers and augment these gazetteers with additional information brought in from non-traditional data sources. For example, we can imagine a scenario where different data sources are used to populate a geospatial database for a given area. Data can be retrieved and integrated from multiple sources, both traditional and non-traditional. The resulting integrated data can be stored in a standard format, such as the Gazetteer Content Standard (Hill 2002) proposed by the Alexandria Digital Library (ADL)[2] or the Web Gazetteer service (WFS-G) Standard proposed by OpenGIS[3] and made available to the public. If this process could be automated, the creation and maintenance of public gazetteers would become much easier.

In this article, we present a constraint satisfaction approach to relating online data sources with imagery. We motivate the research by showing the importance of accurate geospatial databases in a real-world scenario. Then, we present our problem-solving approach by introducing publicly available sources and the information they provide. Next, we introduce the *constraint satisfaction problem* (CSP) formulation used to solve the problem. We evaluate our approach using both synthetically generated problems and a real-world example which show that our approach can accurately identify buildings on a satellite image. Finally, we present related work and conclude by discussing possible enhancements to the system and other future work.

Motivating Example

To illustrate the importance of geospatial data integration, consider the inadvertent bombing of the Chinese Embassy in Belgrade. On 7 May 1999, B-2 bombers dropped 5 GPS-guided bombs on what had been incorrectly identified as the headquarters of the Yugoslav Federal Directorate for Supply and Procurement (FDSP). A CIA intelligence analyst had correctly determined that the address of the FDSP headquarters was Bulevar Umetnosti 2, but the analyst then used a flawed procedure to identify the geographic coordinates of that address. The results were tragic, especially in light of the fact that the data was available in the telephone book to determine that the target was in fact the Chinese Embassy and not the FDSP headquarters (Pickering 1999).

In the analysis of the tragedy, the US has acknowledged that the database containing the address of the Chinese Embassy was out of date and if it had been current then this tragedy would not have occurred (Pickering 1999). While

[1] Vector data is used to identify the geographical coordinates of an address. It represents the streets and address ranges of street segments.

[2] http://www.alexandria.ucsb.edu

[3] https://portal.opengeospatial.org/specs

this is certainly true and the US will no doubt maintain careful records of the embassies throughout the world, the underlying problem still remains: it is extraordinarily difficult to both determine the location of addresses and identify buildings in less industrialized parts of the world. If the building had been an office building instead of an embassy, then an up-to-date database of embassies would not have prevented a similar tragedy.

We believe that the approach proposed in this article can help in averting a situation such as this. Our approach can be used to identify all of the buildings in a given area of the world using online data such as a phone book data source. This information would be used to augment and update a geospatial database, such as a gazetteer. This would help keep geospatial sources current and lessen the chances of a reoccurrence of the above scenario. Such an approach would not only benefit the intelligence community but also any individuals or organizations with the aim of building or maintaining geospatial information sources. Of course, the accuracy of the resulting source will be limited by the accuracy of the input data.

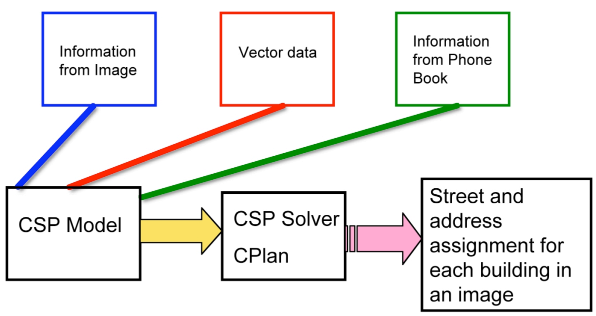

Figure 1: Problem Solving Framework

Problem Solving Approach

Our approach uses a *constraint satisfaction problem* (CSP) approach for assigning labels (street name and address) to buildings in a given satellite image. The general framework of the system can be seen in Figure 1. It is comprised of three main components; a set of publicly available information sources, the CSP model, and the CSP solver. The intuition behind this framework is as follows: begin by gathering all of the publicly available data, such as satellite imagery, vector data, and a phone book. Then incorporate this data into a CSP model that is created using the CSP formulation explained below. After creating a new problem instance, the CSP model is passed to the CSP solver. The CSP solver returns all of the possible solutions to the problem. Finally merge the solutions to create a set of possible street and address assignments for each building and return this set as the final solution.

Due to the incompleteness of online data sources, it is possible that the system will generate multiple labels for any given building. Even though this does not guarantee a correct answer for all buildings in the image, the returned solution is still beneficial. Consider our motivating example. If we used our system to provide a set of possible labels for the Chinese Embassy, it could have been deduced that the building mislabeled as the FDSP headquarters (the Chinese Embassy) wasn't at the address in question. Because there was only one address in the phone book for the street Tresnjin Cvet, and the mislabeled building was the only one that could have been on Tresnjin Cvet, our approach would not assign Bulevar Umetnosti as a potential street for this building.

The approach we present is a novel way to use both explicit and implicit information in publicly available data sources. The key challenge lies in combining this information and using it to label buildings in satellite imagery with a high degree of accuracy. Using a constraint satisfaction framework allows us to address the integration issue by generating a CSP model that allows all of the information to be plugged in easily. Finally, leveraging common properties of streets and addresses in the world allows us to provide solutions that could not be deduced from any individual source but require the combination of data from multiple sources.

In the following sections, we describe each component of our system, how it integrates with other components, and any assumptions that are made for the given system component. In the Future Work section, we discuss a potential extension to our system which would associate a probability with each assignment, something that would provide a more informative solution.

Publicly Available Information

As mentioned earlier, our approach uses publicly available data to identify buildings. For the sake of clarity, we will only discuss three sources that are important to the reasoning process: satellite imagery, phone book, and vector data sources. However, our approach is not limited to these sources. We envision having the capability to include any nontraditional data source into the system by simply modeling it as a constraint.

A satellite imagery source returns an image of an area we would like to label. Using this image, we can extract the following information: all of the buildings present in the image (building identification is a separate research topic and we are assuming that we have a tool at our disposal which can be used to identify the buildings in an image (Lin & Nevatia 1998)), on which street(s) each building can potentially be located, the order in which a building occurs on each of its potential streets (this is important for one of the constraints used in the system), and on which side of the street the building lies.

The vector data source provides the street information. This source returns a line file with street information for the given area. We assume the vector data has been aligned with the imagery and we address this issue in (Chen *et al.* 2003; Chen, Shahabi, & Knoblock 2004). The vector information is used in conjunction with the building information extracted from the image to provide potential street and ordering information for each building. Additionally, this source could provide information on which side of the street even numbers lie and in which direction numbers are ascending for north-south and east-west running streets.[4] However, this additional information is optional and its role is described in the CSP Formulation section.

Finally, the phone book source provides all of the known buildings for every street in the satellite image. This information can be further divided into two groups, even and odd numbers for a given street. This information is used in one of the CSP constraints presented below. It is worth noting that we are not making the assumption that the phone book provides complete information (it is not uncommon to find phone books which are incomplete). Rather we assume that the information is correct, meaning an address for a given street that is listed in the phone book corresponds to an actual building on that street. A more complete phone book source further constrains, the problem which leads to a more precise solution (the set of potential addresses for a given building is reduced).

CSP Formulation

Once all of the information from the public data sources has been gathered, we generate a CSP model of the problem. This model is instantiated with all known values for the CSP variables and this instance of the problem is passed to the solver discussed in a later section. Below we define the CSP formulation for the information obtained from online sources and the variables and constraints used in the CSP.

Source Information: The satellite image and vector data information is represented by:

$$\langle \Sigma, B, north\_south, on\_street, side, order \rangle$$

In this formulation, Σ represents the set of street names, denoted by $\{\sigma_1, ..., \sigma_n\}$, obtained from the vector data (where *n* is the total number of streets). B provides the set of buildings in $\{\beta_1, ..., \beta_m\}$. This corresponds to the buildings extracted from the satellite image and serves as the set that needs to be labeled. The predicate $north\_south(\sigma_i)$ indicates if a street σ_i runs north-south. This predicate is set using the information obtained from the vector data and is used to indicate the direction of all the streets in set Σ.

The predicate $on\_street(\beta_i, \sigma_i)$ indicates that building β_i is on street σ_j. Our system sets this predicate for a given building and street based on the potential streets a building can be on. This can be deduced from the satellite image. The function $side : B \times \Sigma \rightarrow \{N, S, E, W\}$[5] indicates the side of the street σ_j that β_i is on. This information is obtained from the satellite image and is important since address assignments depend on which side of a street a building lies on. Finally, the function $order : B \times \Sigma \rightarrow \mathbb{N}$ gives the ordering of the buildings which are $on\_street(\sigma_i)$ from north to south (or west to east). This is important for establishing the order of buildings on a street relative to one another. This information is used in the ordering constraint described below.

[4]Without the loss of generality, we categorize all streets as being either north-south or east-west running streets.

[5]In our implementation, we use a binary representation where N and W are represented by a zero and S and E are represented by a one.

The phone book source is modeled as a set of addresses where $A = \{\alpha_1, ..., \alpha_k\}$ and $\alpha_i = \langle num_i, str_i \rangle$. Intuitively, this representation specifies that the phone book source provides all addresses (phone book entries) for the streets in Σ.

Variables: We now describe the CSP variables and their domains. These variables are used along with the predicates described above to define the constraints in the system. A solution to the CSP is an assignment of values to all of these variables. The complete set of variables is $\{s_1, .., s_m, \sharp_1, .., \sharp_m, e_{ew}, e_{ns}, a_{ns}, a_{ew}\}$ where m is the number of buildings in the image.

For each building β_i, we have one street variable s_i which takes values from Σ and one address variable $\sharp_i$ which ranges in (a subset of) the natural numbers. The variable $e_{ew} \in \{N, S\}$ indicates that even addresses lie either on the north or south side of east-west running streets. The variable $e_{ns} \in \{W, E\}$ indicates that even addresses lie on the west or east side of north-south running streets. Finally, variable $a_{ns} \in \{N, S\}$ is true if addresses get smaller as you travel in the north direction on north-south running streets and $a_{ew} \in \{W, E\}$ is true if addresses get smaller as you travel in the west direction on east-west running streets.

The variables e_{ew}, e_{ns}, a_{ns}, and a_{ew} are in the system to further constrain the problem. The information required to set these variables during problem instantiation is optional. The constraints are written in such a way that if there doesn't exist enough information in the sources to set these variables before runtime, but enough information exists in the problems' instantiation, the solver will figure out what these values should take. Otherwise, it will return solutions for all possible assignments (that satisfy the constraints) to these variables. An example of this is described in the *Global variables set* constraint below.

Constraints: There are 4 main constraints in the CSP model and they are as follows:

Constraint 1: ***Even or $\neg$Even(Odd) numbering***

$$\forall_{i,j}(s_i = j) \wedge (((north\_south(j) \wedge (side(i, j) = e_{ns})) \vee ((\neg north\_south(j) \wedge (side(i, j) = e_{ew}))) \leftrightarrow even(\sharp_i)$$

Constraint 1 is over variables $\{s_i, \sharp_i, e_{ns}, e_{ew}\}$

This constraint ensures that all assignments of address variables have the same parity (even or odd) for buildings that lie on the same street and on the same side. Thus, if two buildings are both on street *A*, which runs north-south, but one is on the east side and the other is on the west side, both buildings will not be assigned an even (or odd) address. The opposite is also true, if the buildings are on the same side of the street, they will be assigned the same parity of address (both will be odd or even). In the CSP model, this constraint is implemented as two constraints of the same type, one for each type of street in the system (north-south and east-west running). This implementation reduces the complexity of the constraint from four to three variables, as seen below:[6]

[6]The negation of each constraint is also required to implement this reduction in complexity. This negation is not shown but was incorporated into each of the constraints during implementation.

1.$\forall_{i,j}(s_i = j) \wedge north\_south(j) \wedge (side(i,j) = e_{ns}) \rightarrow even(\sharp_i)$
2. $\forall_{i,j}(s_i = j) \wedge \neg north\_south(j) \wedge (side(i,j) = e_{ew}) \rightarrow even(\sharp_i)$

Constraint 2: ***Ordering of addresses along a street***
$\forall_{i_1,i_2,j}(s_{i_1} = s_{i_2}) \wedge ((side(i_1,j) = side(i_2,j)) \wedge (north\_south(s_{i_1}) \wedge a_{ns}) \vee (\neg north\_south(s_{i_1}) \wedge a_{ew})) \leftrightarrow ((order(i_1, s_{i_1}) > order(i_2, s_{i_2})) \rightarrow (\sharp_{i_1} > \sharp_{i_2})) \wedge ((order(i_1, s_{i_1}) < order(i_2, s_{i_2})) \rightarrow (\sharp_{i_1} < \sharp_{i_2}))$
Constraint 2 is over variables $\{s_i, s_j, \sharp_i, \sharp_j, a_{ns}, a_{ew}\}$

This constraint assures that all assignments of address numbers adhere to the ordering of the buildings on a given street. For example, if we are looking at buildings on the north side of a east-west running street, the address numbers assigned to the buildings will be consistent with their ordering on that street. Therefore, if there exist three buildings, b_1, b_2, and b_3 and order(b_1) $<$ order(b_2) $<$ order(b_3), then address(b_1) $<$ address(b_2) $<$ address(b_3). This is of course subject to the assignment of a_{ew} or a_{ns}. This is analogous to walking down a street in one direction and expecting all of the numbers to either get bigger or smaller. As with the even numbering constraint, this constraint is implemented as two constraints of the same type, one for each direction of street in the system. In doing this, we reduce the complexity of the constraint, making it over five rather then six variables, as seen below:[7]

1.$\forall_{i_1,i_2,j}(s_{i_1} = s_{i_2}) \wedge (side(i_1,j) = side(i_2,j)) \wedge ((north\_south(s_{i_1}) \wedge a_{ns}) \rightarrow ((order(i_1, s_{i_1}) > order(i_2, s_{i_2})) \rightarrow (\sharp_{i_1} > \sharp_{i_2})) \wedge ((order(i_1, s_{i_1}) < order(i_2, s_{i_2})) \rightarrow (\sharp_{i_1} < \sharp_{i_2}))$
2. $\forall_{i_1,i_2,j}(s_{i_1} = s_{i_2}) \wedge (side(i_1,j) = side(i_2,j)) \wedge (\neg north\_south(s_{i_1}) \wedge a_{ew})) \rightarrow ((order(i_1, s_{i_1}) > order(i_2, s_{i_2})) \rightarrow (\sharp_{i_1} > \sharp_{i_2})) \wedge ((order(i_1, s_{i_1}) < order(i_2, s_{i_2})) \rightarrow (\sharp_{i_1} < \sharp_{i_2}))$

Constraint 3: ***Phone book numbers***
$\forall_\alpha \exists_i (s_i = \alpha_{str}) \wedge (\sharp_i = \alpha_{num})$
where $i \in \{i' | on\_street(i', \alpha_{str})\}$
This constraint checks to make sure that all of the values present in the phone book have been assigned to a building on the image. Since all of the values in the phone book correspond to buildings that must exist, the system assures that these addresses are assigned to a building. As mentioned earlier, this type of reasoning would have led to the conclusion that the building thought to be the FDSP headquarters (which turned out to be the Chinese Embassy) could not have possibly been correct because there existed one phone book entry for Tresnjin Cvet and only one building that could have be on that street.

Constraint 4: ***Global variables set***
This is a constraint that checks to see if the variables e_{ew}, e_{ns}, a_{ns}, a_{ew} have been set correctly, meaning if any of these variables were instantiated to a particular value, then the solution returned must have the corresponding variable set to the instantiated value. The system also allows the problem to be instantiated without knowing all of these variable values. However, setting one or more of these values will further constrain the problem and lead to fewer solutions. But, if any of these variable values are not set, the CSP provides solutions based on the possible assignments for each unset variable. This constraint allows the system to prune the number of possible solutions in the presence of additional information from online sources.

For example, if a given street has two houses on the west side and five on the east side, and the phone book has three even entries and one odd entry for that street, we know that e_{ns} must equal {E} (indicating that even numbers lie on the east side of north-south running streets) because the *Phone book numbers* constraint tells us that all phone book entries must be assigned to a building. Therefore, the only valid assignment in this case would be to assign the houses on the east side even numbers because there aren't enough houses on the west side to assign all of the even phone book entries to them. Therefore, all solutions produced by the CSP for this example will have e_{ns} = {E}.

All of the constraints are *hard* constraints. We are currently working on including *soft* constraints which would further constrain the problem in the presence of additional information. This is discussed further in the Future Work section.

Solving the CSP Model

Our framework uses CPlan (van Beek & Chen 1999), a constraint satisfaction planner. We chose CPlan for our system because it uses a CSP model, which is a purely declarative representation of domain knowledge and is thus independent of any algorithm. A solution in our framework consists of an assignment of a street and address for each building in a satellite image. However, a solution may contain multiple assignments per building. This is possible because the CSP solver can return multiple solutions for a given problem instance. For the final solution in our framework, we union all of the solutions returned to provide one final set of possible assignments. Therefore, if one building had the assignment A_1 in some solution and A_2 in some other, then the final solution for the problem will contain assignments A_1 and A_2.

While our goal is to assign streets and addresses to buildings with 100% accuracy, there are cases where this is not possible. Still, our results show that in fact there are only a few cases where a building is assigned multiple addresses. Such a situation usually occurs when our system does not have enough information to determine one value for the "optional" variables e_{ew}, e_{ns}, a_{ns}, and a_{ew}. Furthermore, the current solution provides assignments which are of equal probability. We are exploring the possibility of incorporating probabilities into the solution, i.e. a building B would be assigned street address sa_1 with a probability of 0.7 and sa_2 with a probability of 0.3. This is further discussed in the Future Work section.

[7] The negation of each constraint is also required to implement this reduction in complexity.

Experimental Evaluation

To evaluate our approach, we divided the experiments into two sets. The first set, which we call synthetic problems, consists of a sample area that we generated ourselves. We manually came up with the layout of the buildings in our "image," the streets, and the phone book entries . The experiments in this set were divided into four scenarios described in the Synthetic Problems section. These scenarios are used to show the flexibility and reasoning power of our system over varying degrees of available information. The second experiment was run on a real-world scenario. We tested on a neighborhood in El Segundo, CA, which was used as a test location for the work done by (Bakshi, Knoblock, & Thakkar 2004). The purpose of this experiment is to show that our technique applies to real-world situations and could be used in a real geospatial context. For these experiments, we used data from the phone book,[8] vector data provided by NGA,[9] and satellite imagery from Terraservice.[10]

The results for both experiments are presented as measures of precision and recall. Recall corresponds to the percentage of buildings correctly identified by the solver over the total number of buildings in the image. Therefore, if any of the assignments to a given building contain the correct assignment, we consider this building to be correctly identified. We take all the correct assignments made, and divide by the total number of assignments to calculate precision. Since buildings may have multiple assignments, not all of them will be correct. Therefore, precision measures what percentage of the assignments returned were actually correct. For example, if we have two buildings in an image, two assignments to one building, three to the other, and a correct assignment is made to both, the recall would be 100% and the precision would be 40%.

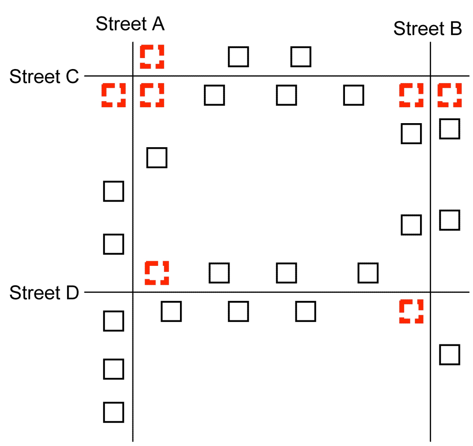

Figure 2: Synthetic Problem layout

Synthetic Problems

To illustrate the functionality of our system, we used a synthetic layout we created, as shown in Figure 2. In this street configuration, there were 29 houses and 4 streets. The most interesting buildings in this case are the ones marked with dotted lines. These are corner lots that could belong to one of two streets. Furthermore, the "phone book" in this case contained the following entries: Street A = {2,3,4,5,6,7,8,9,11,13}, Street B = {1,2,3,4,5,6,7,8}, Street C = {1,2,3,4,5}, and Street D = {1,2,3,4,5,6}.

The four trials run using this layout included: (1) Providing the system with all information (all phone book entries, even/odd, and ascending/descending information), (2) Not providing any information about which side of a street contains even/odd numbers, (3) Randomly taking out 5 samples from the phone book entries for Street A, and (4) Same as trial (3) but also withholding the even/odd information. The results of the four trials using this layout are shown in Table 1.

Trial Type	Precision	Recall
All information available	100%	100%
All info except even/odd	100%	100%
Missing phone book entries	85.3%	96.6%
Missing entries and no even/odd	58.6%	96.6%

Table 1: Synthetic Problem Results

When the system is provided with all information, it identifies all buildings correctly, as seen by the 100% levels of precision and recall. If the even/odd information is withheld, the system still reaches 100% precision and recall. This is because there is enough information in the phone book for the system to figure out which side the even and odd addresses must be on. In the third trial with missing entries from the phone book, the recall level stays high and precision drops. This is because there are multiple possible assignments for some of the buildings, leading to a drop in precision. Finally, in the last trial, which provides the system with the least amount of information, precision drops dramatically. However, recall stays very high. This is caused by the fact that there are many possible assignments that satisfy the constraints of the system. However, the problem is still constrained enough where one of the solutions contains the right assignment for almost all of the buildings.

In fact only one building did not have the correct assignment in its respective set of assignments. This was caused by the fact that the combination of deleted phone book entries for Street A led to one of the addresses on Street A never getting the correct assignment. This is caused by an implementation decision we made. Since each variable's domain needs to be finite, when deciding on the variable domain size during problem instantiation, we set the domain for address variables as the range $1 - n$ where n is the largest address seen. Therefore, it is possible that if a building has an address outside of the range of the domain, the correct assignment can not be made. This was the case in our experiments of scenarios (3) and (4) and explains the drop in recall.

Real-world Scenarios

To show the validity of our approach in real world scenarios, we ran two trials on one of the blocks in El Segundo used in

[8]http://www.whitepages.com/
[9]http://www.nga.mil
[10]http://terraservice.net

(Bakshi, Knoblock, & Thakkar 2004). This area consisted of a block with 34 houses and four cross streets, as seen in Figure 3. The results can be seen in Table 2. The lower levels of precision and recall can be explained by the fact that the phone book was incomplete with respect to this area. Therefore, our system had difficulties determining the correct location of the corner lots.

Source Used	Precision	Recall
Phone book source	54.7%	94.1%
Property tax source	100%	100%

Table 2: Real World Problem Results

Furthermore, the two buildings that were not labeled correctly did not have an entry in the phone book. Even though one of their labels contained the correct street, the address number was incorrect because the system did not know about such an address. It should be noted that even/odd information was not available for this block, yet our approach was able to figure this out.

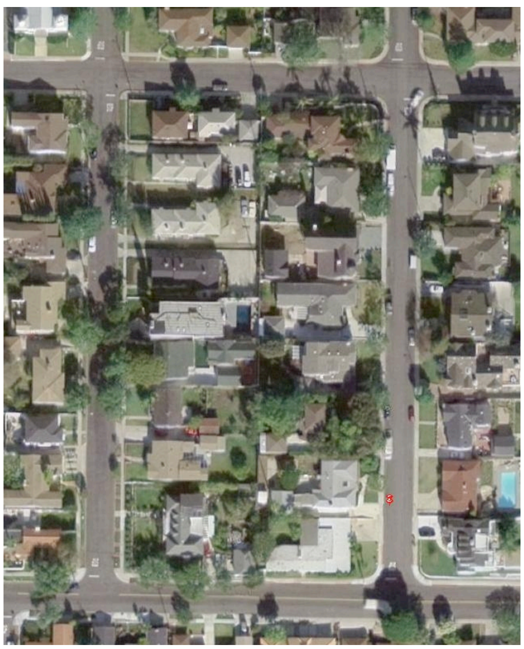

Figure 3: El Segundo Region

However, the problem of incomplete data can be addressed by introducing more sources into the system. Therefore, we ran another trial replacing the phone book data with the property tax data source used in (Bakshi, Knoblock, & Thakkar 2004). This source provided the system with a complete set of house addresses for this area. Table 2 shows that if the system has enough information, it can produce 100% levels of precision and recall. These results validate our theory that the more complete a source, the better the results.

Related Work

Constraint satisfaction problems (CSP) (Marriott & Stuckey 2003; Van Hentenryck 1989) have been an active research topic. There has been a lot of work done on building solvers, optimizing, formalizing, etc. for CSPs. Our work focuses on applying CSPs to a new domain in a novel way.

The work done by (Bakshi, Knoblock, & Thakkar 2004) presents methods to accurately geocode addresses using publicly available data sources. The authors present two different approaches that can be used to improve traditional geocoders. The end result of this work is accurate latitude and longitude coordinates for buildings in a given area. This work also uses online sources to improve the accuracy of building labels. The authors' goal is to precisely identify the location of buildings in a satellite image, which is different from our goal of providing a set of labels to buildings in an image. Furthermore, this work assumes that the sources used to identify all buildings in an image are complete (contain all of the buildings for a given area). This is a valid assumption to make when considering property tax websites, however such sources are not available for most areas of the world. Therefore, this approach may not be universally applicable.

Littman (Littman, Keim, & Shazeer 2002) presents a CSP approach to solving crossword puzzles. This work implements a probabilistic CSP approach to filling out a crossword puzzle using the PROVERB system. This is similar to our work in that it also leverages the power of constraint programming to solve an assignment problem (assigning letters to squares). However, the crossword assignment problem is very different from the building assignment problem we are trying to solve. One key difference is the types of constraints needed in each problem. We believe that this work serves as an excellent starting point for our future direction in assigning probabilities to building labels, something we describe in the next section.

Finally, there has been work done in identifying buildings in satellite imagery and merging geospatial databases using computer vision approaches, as seen in (Agouris & Stefanidis 1996; Agouris *et al.* 2000; Doucette *et al.* 1999). While some of the goals in this work are similar (identifying objects in images), the work is more focused on the actual detection of buildings in the images. This varies from our goal of labeling and reasoning over specific buildings in images. As mentioned earlier, we assume that we have a tool available which will identify buildings in images. Therefore, this work could fit in well with our system as part of a "preprocessing" step.

Discussion and Future Work

In this article, we presented a constraint satisfaction approach to performing geospatial reasoning. This approach focuses on leveraging the power of constraint programming and the availability of public information sources to allow for accurate labeling of buildings in satellite imagery. Our approach is general enough to be used in a geospatial context, for example as a tool in automatic gazetteer creation. Our results show that our framework allows for reasoning over missing data (even/odd street information), leading to accurate labeling of buildings in the absence of complete data. Also, in the presence of complete data, our approach can label buildings with 100% accuracy.

We are focusing our future work on improving the accuracy and informativeness of the solutions provided by the system. This can be done in two ways. First, by incorporating the notion of *soft* constraints (Dechter 1989), we can return smaller solution sets. If we model certain sources as soft constraints, we can view them as being "optional". If one of these sources is available, its information can be used to further constrain the problem. However, if this source is unavailable, a solution will still be returned.

We are studying modeling this problem using probabilistic or stochastic CSPs. This approach returns assignments with associated probabilities. This eliminates binary (yes/no) assignments of addresses to buildings, and introduces the likelihood of an assignment being correct. The main challenge with this approach is how to model this domain using probabilities. Namely, how to systematically determine with what probabilities a building is on street *a* and on street *b*. We are evaluating the effectiveness of using distance from a street as a metric in this case.

Furthermore, the issue of scalability needs to be addressed. We are currently researching the effect problem size has on efficiency and satisfiability. We have begun dealing with efficiency by prioritizing variables. Such an approach forces the solver to make assignments to variables which are involved in the most constraints. However, more tests need to be done to determine the satisfiability of large problems. We are exploring ways to reduce the parity of the constraints by either subdividing them furthermore or by exploiting the properties captured by a given constraint using two or more lower parity constraints.

Finally, we envision a system with the capability to "plug in" region-specific information such as numbering schemes, red/black numbering in Italy, etc. as constraints or in another manner that makes sense. The assumptions made for this paper allowed us to test our problem solving approach and determine its viability. Even though these assumptions may not be universally applicable, we believe a "plug in" capability would enable us to apply our approach to most regions of the world.

Acknowledgments

We would like to thank Mark Carman for his help in coding for and using CPLAN, and the reviewers for their helpful comments on the paper.

This research is based upon work supported in part by the Air Force Office of Scientific Research under grant number FA9550-04-1-0105, in part by the National Science Foundation under Award No. IIS-0324955, and in part by the Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory, Air Force Materiel Command, USAF, under agreement number F30602-00-1-0504.

The U.S. Government is authorized to reproduce and distribute reports for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of any of the above organizations or any person connected with them.

References

Agouris, P., and Stefanidis, A. 1996. Integration of photogrammetric and geographic databases. *International Archives of Photogrammetry and Remote Sensing, ISPRS XVIIIth Congress* 31:24–29.

Agouris, P.; Beard, K.; Mountrakis, G.; and Stefanidis, A. 2000. Capturing and modeling geographic object change: A spatio-temporal gazeteer framework. *Photogrammetric Engineering and Remote Sensing* 66(10):1224–1250.

Bakshi, R.; Knoblock, C. A.; and Thakkar, S. 2004. Exploiting online sources to accurately geocode addresses. In *GIS '04: Proceedings of the 12th annual ACM international workshop on Geographic information systems*, 194–203. ACM Press.

Chen, C.-C.; Thakkar, S.; Knoblock, C. A.; and Shahabi, C. 2003. Automatically annotating and integrating spatial datasets. In *Proceedings of the Eighth International Symposium on Spatial and Temporal Databases (SSTD 2003)*, volume 2750 of *Lecture Notes in Computer Science*.

Chen, C.-C.; Shahabi, C.; and Knoblock, C. A. 2004. Utilizing road network data for automatic identification of road intersections from high resolution color orthoimagery. In *Proceedings of the 2nd Workshop on Spatio-Temporal Database Management - STDBM'04*.

Dechter, R. 1989. *Constraint Processing*. The MIT Press.

Doucette, P.; Agouris, P.; Musavi, M.; and Stefanidis, A. 1999. Automated extraction of linear features from aerial imagery using kohonen learning and gis data. *Lecture Notes in Computer Science* 1737:20–33.

Hill, L. L. 2002. Content standards for digital gazetteers. In *Proceedings of the Networked Knowledge Organization Systems/Services Workshop on Digital Gazetteers: Integration into Distributed Digital Library Services at the ACM/IEEE Joint Conference on Digital Libraries*.

Lin, C., and Nevatia, R. 1998. Building detection and description from a single intensity image. *Computer Vision and Image Understanding Journal* 72(2):101–121.

Littman, M. L.; Keim, G. A.; and Shazeer, N. 2002. A probabilistic approach to solving crossword puzzles. *Artificial Intelligence* 134(1-2):23–55.

Marriott, K., and Stuckey, P. J. 2003. *Programming with Constraints*. Morgan Kaufmann.

Pickering, T. 1999. Oral Presentation by Under Secretary of State Thomas Pickering on June 17, 1999 to the Chinese Government Regarding the Accidental Bombing of the PRC Embassy in Belgrade.

van Beek, P., and Chen, X. 1999. CPlan: A constraint programming approach to planning. In *The Sixteenth National Conference on Artificial Intelligence (AAAI-99)*, 585–590.

Van Hentenryck, P. 1989. *Constraint Satisfaction in Logic Programming*. The MIT Press.