Crosslingual Location Search

Tanuja Joshi[1], Joseph Joy[1], Tobias Kellner[1], Udayan Khurana[2], A Kumaran[1], Vibhuti Sengar[1]

[1] Microsoft Research India
196/36 2nd Main
Bangalore, 560 080, India
{v-tjoshi,josephj,t-tobik,kumarana,vibhutis}@microsoft.com

[2] Microsoft India R&D
Microsoft Campus, Gachibowli
Hyderabad, 500 032, India
udayank@microsoft.com

ABSTRACT

Address geocoding, the process of finding the map location for a structured postal address, is a relatively well-studied problem. In this paper we consider the more general problem of crosslingual location search, where the queries are not limited to postal addresses, and the language and script used in the search query is different from the one in which the underlying data is stored. To the best of our knowledge, our system is the first crosslingual location search system that is able to geocode complex addresses. We use a statistical machine transliteration system to convert location names from the script of the query to that of the stored data. However, we show that it is not sufficient to simply feed the resulting transliterations into a monolingual geocoding system, as the ambiguity inherent in the conversion drastically expands the location search space and significantly lowers the quality of results. The strength of our approach lies in its integrated, end-to-end nature: we use abstraction and fuzzy search (in the text domain) to achieve maximum coverage despite transliteration ambiguities, while applying spatial constraints (in the geographic domain) to focus only on viable interpretations of the query. Our experiments with structured and unstructured queries in a set of diverse languages and scripts (Arabic, English, Hindi and Japanese) searching for locations in different regions of the world, show full crosslingual location search accuracy at levels comparable to that of commercial monolingual systems. We achieve these levels of performance using techniques that may be applied to crosslingual searches in any language/script, and over arbitrary spatial data.

Categories and Subject Descriptors

H.3.1: Content Analysis and Indexing; H.3.3: Information Search and Retrieval

General Terms: Algorithms, Design

Keywords: Crosslingual Information Retrieval. Address Geocoding. Location Search.

1. INTRODUCTION

Address Geocoding is the well-known problem of mapping a postal address to a geocode (a geographic identifier such as a location on a map). Address geocoding features prominently in several information retrieval scenarios, such as, locating an address, finding directions, visualizing geographic distribution of a set of addresses, etc. Address geocoding is solved well by current commercial systems for certain countries [7]. A more general class of problem is *Location Search*, which deals with postal addresses, as well as informally specified textual descriptions of a location (e.g., "Corner of Grand Ave and Walnut St in Everett"). Searching for locations with unstructured information is very useful when a precise postal address is not known, but other fragments, such as, street, locality, nearby landmark, etc., are known. In this paper, we further extend the location search problem to what we call *Crosslingual Location Search:* Allowing users to search using languages and scripts different from the one in which the underlying geographic data is stored. Popular map search services, such as Google Maps™, Yahoo Maps™ and Windows Live Local™, currently handle only the English version correctly. Figure 1 is a screenshot from our prototype system that shows the successful pinpointing of the location in Snohomish, Washington, given a complex query in Japanese.

Figure 1: Crosslingual Location Search System

Table 1 gives a set of search queries written in different languages for another location.

Table 1: One sample address in multiple languages

English	Brunswick Park Gardens, 130, Barnet London
Japanese	ブルンズウィック・パーク・ガーデンズ、１３０、バーネット・ロンドン
Arabic	برنسويك بارك جاردن ، 130 ، بارنيت لندن
Hindi	ब्रुन्स्विक पार्क गार्डन्स, 130, बार्नेट लंदन

SIGIR'08, July 20–24, 2008, Singapore.

Geographic data, and particularly map data, are intrinsically tied to given regions, and hence are available predominantly in local languages. In addition, the business, resource and interoperability considerations often dictate that such data are created only for a small set of languages. Yet in today's increasingly globalized world, there is a clear need for accessing geographic information across languages. Examples range from Indian citizens who want to query in their own local languages, the land records traditionally created in English, through cross-lingual geographic indexing of documents, to visitors at the 2008 Olympics who will want to find Beijing locations using many languages other than Mandarin Chinese. Despite the clear motivation for crosslingual location searches, to the best of our knowledge, there are no academic or commercial systems that support general crosslingual location search.

A possible approach to crosslingual location search would be to create and represent all geographic entities in all languages, but this is financially and logistically unviable (for example, a country of the size of the US has several million *unique* streets, localities, landmarks, etc., and moreover, these are updated on a continual basis). Alternatively, one could use a machine translation/transliteration system to convert the query terms to the target language, and then process the results in a monolingual geocoder in the target language. However, as we show later in this paper, the linguistic ambiguities inherent in the process, increase the search space exponentially, and degrade the accuracy of results greatly. In addition, the fact that descriptions of locations and addresses are structured differently in different regions – or may be unstructured altogether – makes cross lingual location search a particularly challenging research problem.

In this paper, we develop an underlying set of techniques to tackle the crosslingual location search problem. These techniques use a combination of transliteration, translation and abstraction to map text query fragments to spatial regions in the underlying map data, and spatial constraints among the elements of map data to narrow down the search space. We present our implementation as an end-to-end system, which can successfully handle crosslingual queries of the kind listed in Table 1, accurately computing the geographic position to within 1 km in most cases. Our current implementation supports queries in 4 genetically distinct languages with different scripts (namely, Arabic, English, Hindi and Japanese), over geographic data (in English) from different regions (USA, UK, and India) of the world. As no comparable systems exist currently, we evaluate our system by comparing its performance to two baselines: first, commercial monolingual location search systems and, second, a sequential approach of feeding transliteration results into these commercial monolingual geocoding systems. We show experimentally that our crosslingual system compares well with the state-of-the-art monolingual geocoders, and is significantly more robust and accurate than sequential transliteration followed by monolingual geocoding.

The core strength and the novelty of our approach lies in its generality, and the way it completely integrates address translation/transliteration and spatial constraint processing. The key contributions of this paper are:

1. A precise formulation of the crosslingual location search problem as an end-to-end problem.
2. The use of spatial constraints to manage ambiguity introduced by transliteration and translation.
3. A comparative evaluation of crosslingual location search for a variety of languages over geographic data covering different regions.

Our approach has been implemented as a fully functional end-to-end system, as described in [8].

2. BACKGROUND

Address geocoding is a specific instance of location search, concerned with finding the geographic coordinates given a postal address. Address geocoding is central to services provided by companies such as ESRI™, as well as online location based services such as Google Maps™. Although the problems of address parsing and geocoding are widely studied [7,14,18], we find very few end-to-end address geocoding techniques discussed in the literature; [3] describes one such system, which is a geocoding system with an address parser based on HMMs and a rule-based matching engine, applied to the Australian National Address File system.

In earlier work [17], we describe a system that supports *monolingual* location search queries. A location search query contains a subset of terms that, taken together, identify the spatial scope (typically, a location on a map) of the query. Location search queries can include *structured* (postal) addresses, as well as informal, or *unstructured* descriptions of locations. In this paper, we present a precise formulation of the problem, and focus on computing the spatial scope of the query as accurately and as robustly as possible when the query is in a script that is *different* from that of the names of the underlying geographic entities in the geocoding system.

Over the last decade, the CLEF [4] forum had been active in promoting crosslingual IR research, and one of its subtasks, namely GeoCLEF [6], specifically focuses on geographic information retrieval scenarios. While the systems showcased are either text based systems or monolingual geocoders working on multiple languages, the system we present here is truly a crosslingual location search system.

Location search is challenging even with the assumption that the query and the underlying spatial data share the same language. Figure 2 contains the key reasons for this.

- Geometric coordinates need to be computed
- Address formats differ widely across countries[15]
- Unstructured queries can have endless variations in format
- Terms in the query are not always delimited, so boundaries between terms can be ambiguous
- Misspelled, irrelevant (not matching anything) or conflicting (e.g., a legitimate but wrong postal code) terms are present
- Non-unique (e.g., many roads called "Main St"), or similar names (e.g., spelling variations of "Auckland") are present

Figure 2: Same-language location search challenges

Nevertheless, there are several commercial location service providers, such as Google™ Maps, Windows™ Live Maps, and Yahoo™ Maps, which provide good results for several countries and are somewhat tolerant to misspellings and address variations. *Crosslingual* location search adds a significant level of difficulty to the monolingual location search problem, as the additional issues listed in Figure 3 need to be overcome.

- Proper nouns need to be transliterated, a process that is inherently ambiguous.
- Common nouns (such as "Hospital" or "Road") may be transliterated *or* translated.
- Orderings of terms may be changed because of language and local addressing conventions.

Figure 3: Crosslingual sources of ambiguity

The bulk (not all) of words in geographic entity names are proper nouns, and in general cannot be translated readily like other parts of speech (such as, common nouns, adjectives, verbs, etc.) that are a part of standard bilingual dictionaries. Where source and target language share the same script, the problem is not readily apparent as the proper nouns are generally identical (for example, "Berlin" is correct in German, English, and Italian, though some variations in orthography and pronunciation exist, such as, "Antwerp" is "Antwerpen" in Dutch, "London" is "Londres" in Spanish, etc.). For different scripts, a name in the source language needs to be *transliterated,* that is, converted into a string in the target script which preserves the original pronunciation of the source word, yet conforming to the pronunciation rules of the target language.

Since the mapping between pronunciation and spelling is not fully deterministic in both languages, transliteration is an inherently ambiguous process. For example, the transliterations of "بلمر" (representing "Palmer") could be "Palmer", "Bilmar" etc. [1], as Arabic does not distinguish between "b" and "p" and short vowels may not be represented in Arabic orthography. Similarly, the English transliterations of Hindi syllable "की" could be highly ambiguous, as many English phonetic constructs, such as, "key", "kee", "ki" and "kea" are possible. As a consequence, for each source word, several transliteration candidates have to be considered to achieve a reasonable confidence that the correct transliteration is included. This leads to a large number of possible interpretations of a multiword query, as illustrated in Table 2, which shows some machine transliteration results for the Arabic address fragment "بلمر رود ريدبريدج لندن", which *should* be transliterated as "Palmer Road Redbridge London".

Table 2: Transliteration ambiguity example

بلمر	رود	ريدبريدج	لندن
blemer	rod	redbridge	landon
plemer	rud	redpridge	lendon
belmer	rood	ridbridge	lindon
balmer	rhod	ridpridgere	landen
blimer	road	dpredge	lenden
…	…	…	…

Another source of ambiguity is that queries can contain common nouns, which can both be *translated* and transliterated. For example, depending on context, either the transliteration "al-matar" or the translation "airport" may be preferable for the word "المطار" in an Arabic language query. Consider the impact of having to consider many alternatives to *each* word in the query, especially in the absence of tokens such as commas to delimit one potential term from another. If k transliterations/translations need to be considered for each word in an n word query, which can be partitioned into subsequences in 2^{n-1} ways, then the total number of possible combinations of non-overlapping subsequences to be considered is $k^n 2^{n-1}$, which, even for the short (four word) Arabic query above, with k=5, amounts to 5000 combinations.

The crosslingual challenges summarized in Figure 3 compound with the monolingual challenges listed in Figure 2. In fact, we claim in this paper that the ambiguity in spellings, names and orderings inherent in crosslingual matching make simple composition of translation/transliteration with monolingual location search an *unviable* option. Our experimental results in Section 5 confirm this.

3. APPROACH AND ALGORITHMS

A key aspect of our approach is harnessing the power of *spatial constraints* to resolve the ambiguities discussed above. In the above example there were many possible interpretations for the English candidate transliterations of the Arabic location query. However, of these, only sets of entities that *spatially overlap* are plausible candidates. Thus the fact that 3 particular entities named "Palmer Road", "Redbridge", and "London" (in the U.K.) all spatially overlap, gives a powerful reason to pick this interpretation from all possible interpretations of the query, as illustrated in Figure 4. Moreover, the *region of overlap* (the gray region in the figure) of these three entities defines the geometric scope of the result.

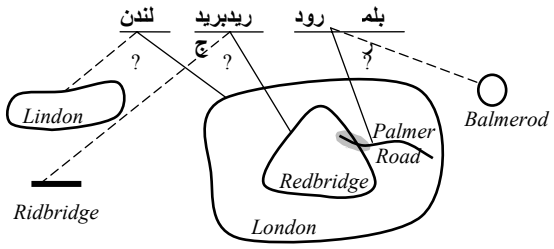

Figure 4: Illustrating use of spatial constraints

To systematically exploit such spatial constraints, we reformulate the crosslingual location search problem as an end-to-end problem, perhaps for the first time. Our definition is intentionally more general than location search over map data – it covers monolingual and crosslingual text based search over arbitrary spatial data.

3.1 Problem Definition

The following definition of the crosslingual and monolingual location search problem is a more precise and general form of the definition in [17], which considers only monolingual location queries. A *query* is a text string about which our only assumption is that it can be parsed into a list of *tokens* (typically, words, numbers, etc.), represented by $Q = (q_1, q_2, q_3, \ldots, q_n)$. Collectively, the terms in query Q are assumed to identify some spatial region. A query *subsequence* is a list of contiguous tokens of Q. We represent the subsequence $(q_i, q_{i+1}, q_{i+2}, \ldots, q_j)$ as q_{i-j} $(i \leq j)$. For example, given the sample query "ブリジュポート ウェイ と パシフィク ハイウェイ、 レクウ", q_{2-3} represents subsequence "ウェイ と."

Some subsequences (such as "Palmer Road", perhaps after translation or transliteration) are expected to refer to named entities in a *spatial repository* consisting of geometric objects.

The presence of these subsequences is the *defining* aspect of location search queries. We model the underlying spatial repository as a set of *entities* that have *geometry* (points, lines, polygons, etc., embedded in a universal metric space) and one or more textual *names*. The spatial repository typically includes entities whose geometry represents the spatial boundaries of landmarks, roads, rivers, city boundaries, etc. Names typically identify an entity (e.g. "Redbridge"), but they can also represent classifications of the entity (e.g., "park"). Entities can have multiple names (e.g., a street can have multiple names), and the same name can apply to multiple entities (e.g., many streets can share the name "Main Street"). An entity, e, is thus represented as a pair, $e = (g, \{n_1, n_2, \ldots, n_n\})$, where g represents its geometry, and the n_i's are the names associated with the entity.

An *Interpretation*, I, of a query Q, with respect to spatial repository S, is a set of one-to-one mappings from non-overlapping subsequences of Q, to distinct spatial entities in S. An interpretation thus has the form $\{(q_{i1\text{-}j1},e_1), (q_{i2\text{-}j2},e_2), \ldots, (q_{im\text{-}jm},e_m)\}$. There are a very large number of possible Interpretations, but only a small number are consistent, and, informally, the goal of location search is to find the most consistent interpretations.

Consistent Interpretations have two key properties:

1. There must be *textual affinity* between each subsequence and a name of its corresponding entity in the Interpretation. Informally, the subsequences should "sound like", or more generally, be "related to", the names of the entities that they are mapped to. This needs to hold true even if the query is in a different script from that of the names of the entity. For example, the subsequences "Bridge Port Way", and also "ブリジュポート ウェイ" (roughly "*BuRiJyuPorTo Wai*" in Japanese Katakana script) both have textual affinity to an entity named "Bridgeport Way". Textual affinity can include semantic or ontological similarities, as between "Main Street" and "Main Road."

2. There must be *spatial coherence* amongst all entities in the Interpretation. Informally, this means that all the entities in the Interpretation must roughly overlap in space. We define spatial coherence as a Boolean property of a set of entities, parameterized by a boundary-width parameter β. A set of entities is spatially coherent with boundary-width β if there is a region in space, where all the entities, first expanded outwards by β, overlap. The region of overlap is called the *geometric scope* of the Interpretation. In Figure 5, the entities e_1 and e_2 (but not e_3) are spatially coherent, with geometric scope indicated by the shaded area.

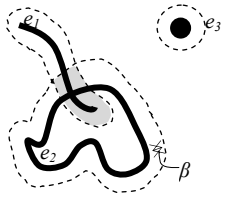

Figure 5: Illustrating spatial coherence

In terms of these definitions, we now define the crosslingual location search problem as follows:

> *Find a set of* ***Interpretations*** *that maximizes* ***textual affinity*** *while preserving* ***spatial coherence****, and return the* ***geometric scopes*** *of these interpretations.*

The definition can be made more precise by introducing a textual affinity function, say α, that computes a textual similarity score for an interpretation, and cast the problem as: jointly optimize for maximum α and minimum β. However, the definition of α, and of the joint optimization function are highly domain specific and not easy to pin down. In this paper we take the pragmatic approach of using fixed values of β for each entity type, and crafting an α that improves end-to-end performance of the system with respect the specific domain of cross lingual location search over geographic (map) data.

3.2 Computing Interpretations

Our algorithm that computes a ranked list of Interpretations first constructs an ordered list of "match candidates" (MCs). Each MC, ($q_{i\text{-}j}$, *name*), maps query subsequence $q_{i\text{-}j}$ to a name *name* in the spatial repository, S. Recall that many entities in S may share the same name. Let *Entities*(S, *name*) be the set of entities that share the name *name*. Figure 6 contains our algorithm, XL-QUERY. Figure 7 illustrates our system that implements the algorithm. The algorithm computes the list of match candidates using a combination of transliteration and translation, abstraction (defined later), and approximate text lookup. The algorithm then incrementally computes Interpretations, preserving spatial coherence, using a depth first algorithm.

XL-QUERY uses several pre-built indexes and access functions:

1. One or more language-specific dictionaries, with access function $LSD(l,w)$, that returns for a word w in language l, its translations (if any) in the language of the underlying data.
2. A second Fuzzy Text Index, $FI_A(S)$, used for looking up abstractions (defined below) of the names in S.

The algorithm uses several pluggable functions:

1. A Machine Transliteration (MT) function, $T(l, w)$, that performs machine transliteration of word w in language l to the target language, as elaborated in the next section.
2. An Abstraction function, $A(w)$, that transforms a word w into a form that captures the "phonetic essence" of w along the lines of SOUNDEX [16]. The same abstraction function must be used both for building $FI_A(S)$ and during query processing.
3. A ranking function $R(I)$ that assigns a real number score to Interpretation I. This function constructs a composite score for all the MCs that make up the Interpretation. This function is used to create the final ranking of the Interpretations found during query processing.

The **FindInterpretations** function implements the TEXSPACE algorithm from [17]. The algorithm's input includes a geometric region or scope, called *Focus*. The scope is successively narrowed as the working interpretation grows. The **Filter** function needs to compute the boolean expression "$Entities(name_y)$ are spatially coherent with *Focus*." This may seem computationally expensive, since this is asking the question: does the scope represented by *Focus* spatially overlap any of the entities with name $name_y$ (after first growing their respective boundaries by β as explained in the previous section). However, as explained in [8], by using the linear quadtrees [5] to represent the spatial extents of the collection of spatial entities, as well as *Focus*, we can efficiently compute spatial coherence. In fact, we have found that the *overall* computation time taken up by function **FindInterpretations** is typically less than 15% of the overall execution time.

XL-QUERY(*Q*, *S, F*)

// Return a ranked list of Interpretations of query *Q* given spatial
// repository *S* and initial focus *F*. See text for explanation.

1. **for each** token q_i in query *Q*
   a. Detect language *l* from q_i
   b. $x_i \leftarrow T(l, q_i)$, a list of transliterated candidate words
   c. $y_i \leftarrow LSD(l, q_i)$, a list of translated words (if any)
   d. $z_i \leftarrow A(x_i)$, a set of phonetic abstractions of transliterations
2. **for each** $n(n+1)/2$ possible subsequences q_{i-j} of *Q*
   a. $L_{TL} \leftarrow$ a list of top ranked *k* candidate subsequences derived from the x_i's and y_i's, based on transliteration probabilities
   b. $L_A \leftarrow$ a list of top ranked *k* candidate abstracted sequences derived from the z_i's
   c. $MCL_{TL} \leftarrow$ set of MCs resulting from looking up L_{TL} in precomputed approximate index $FI(S)$
   d. $MCL_A \leftarrow$ set of MC's resulting from looking up L_A in precomputed approximate abstracted index $FI_A(S)$
   e. $MCList \leftarrow MCL_{TL} \cup MCL_A$
   f. FoundInterpretations ← **FindInterpretations**(*MCList*, *F*, {})
   g. **return** *FoundInterpretations*, ordered by composite Interpretation ranking function *R*

FindInterpretations(*MCList*, *Focus, WorkingInterpretation*)

// Recursively grow partially constructed Interpretation
// *WorkingInterpretation*, and narrowing *Focus,* by trying
// successive elements of *MCList* in turn.

3. **if** *MCList* is empty
   then return {*WorkingInterpretation*}
4. *NextMC* ← head of *MCList*
5. *NewWorkingInterpretation* ← append *NextMC* to end of *WorkingInterpretation*
6. *newFocus* ← geometric intersection of *Focus* and entities in *NextMC*
7. *CompatibleMCs* ← **Filter**(*NextMC, MCList, newFocus*)
8. // Perform recursive depth-first-search
   return
   FindInterpretation (*CompatibleMCs*, *NewFocus, NewWorkingInterpretation*))
   ∪ **FindInterpretation** (*MCList - NextMC*, *Focus, WorkingInterpretation*)

Filter (*NextMC, MCList, Focus*)

// Return the MCs in *MCList* that are textually non-overlapping
// and spatially coherent with *NextMC*.

9. $(q_{i-j}, name) \leftarrow NextMC$
10. *filteredList* ← {}
11. **for each** $(q_{k-l}, name_y)$ in *MCList*
    a. **if** (q_{k-l} does not textually overlap q_{i-j})
       ∧ ($Entities(name_y)$ are spatially coherent with *Focus*) // See text for explanation.
       then add $(q_{k-l}, name_y)$ to *filteredList*
12. **return** *filteredList*

Figure 6: Computing Interpretations

3.3 Machine Transliteration

The transliteration process, $T(l, w)$ can be implemented either by using hand-crafted language specific transliteration rules, or by a system which uses machine learning to build a statistical transliteration model from training data. Unlike rule-based systems, statistical transliteration systems have the advantage that they scale well for many languages pairs, as they follow generic, language-independent approaches and need only appropriate training data to be adapted to new language pairs.

Such statistical machine transliteration systems typically calculate the transliteration probability $P(t|s)$ (the probability of a target language name *t* being the transliteration of a source language name *s*), by segmenting *s* and *t* into n Transliteration Units (TUs) that typically represent zero or more characters, and expressing the transliteration probability as:

$$P(t \mid s) \approx \prod_i P(t_i \mid s_i)$$

where, *s* is represented as the sequence $(s_1 \ldots s_n)$ and, *t*, by $(t_1 \ldots t_n)$.

Early papers [10] used a generative approach that explicitly models the transliteration of a source language TU string *s* to a target language string *t* (i.e. $P(t|s)$) as a series of conversions from the grapheme space (spelling) of the source language to the phoneme space (pronunciation), and then to the grapheme space of the target language. However, subsequent research publications report [1,13] that a direct mapping from source to target TUs without an intermediate phonetic representation often leads to better results. Such *grapheme-to-grapheme* transliteration approaches usually follow a two-step training process: First, an alignment algorithm is used on the name pairs in the training data to align each TU of a source language word with a TU in the corresponding target language word, to build a probability model based on the evidence. Then, a classifier is trained to predict the probability of a target language TU, given a source language TU and its context (e.g. the preceding and following TUs).

A range of different statistical machine transliteration systems were compared in [13]. They find that, in scenarios where pronunciation information for the source word cannot be looked up from an external resource (e.g. pronunciation dictionary), grapheme-to-grapheme transliteration using a maximum entropy classifier (and a context window of *n* preceding and following source TUs) outperforms all other approaches [9]. As we show later in Section 4, the maximum entropy classifier based transliteration used in our crosslingual location search system, performs at a level comparable to the best of the breed [11]. Given that there are no available pronunciation dictionary resources for worldwide geographical entities written in diverse scripts, this is an appropriate approach to machine transliteration for crosslingual location search.

4. IMPLEMENTATION

We have implemented an end-to-end location search system that takes a location search query in one of the supported languages as input and directly produces Interpretations. Distinguishing features of our implementation are that the geocoding is deeply integrated within the system and that it uses no language- or region-specific rules. Instead, our system employs the machine-learning based transliteration and the general text/space processing algorithms presented in the previous section. This has allowed us to easily build a system to support new languages and diverse geographic regions. Our system currently supports source queries in Arabic, English, Hindi and Japanese, for underlying map data that is in English. We

have indexed detailed map data to build a system that supports cross-lingual queries over several large cities in three countries.

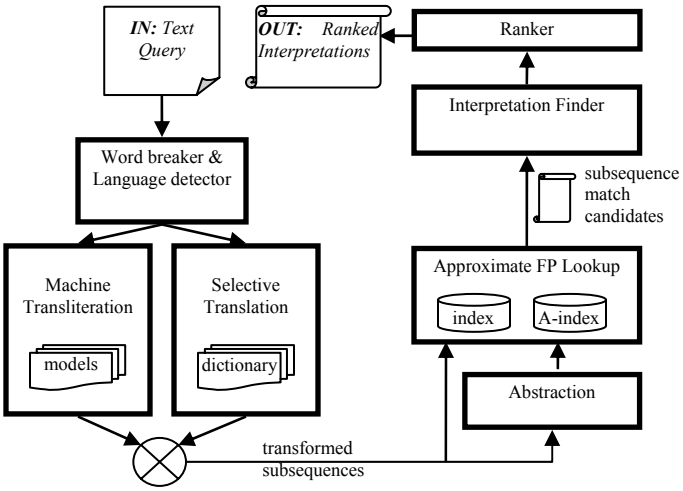

Figure 7: System diagram

Figure 7 illustrates our system, which implements the XL-QUERY algorithm described in Figure 6. We now explain how a query flows through the various components of our system. The query is first passed into a **Word Breaker and Language Detector**, which detects the language based on Unicode code pages (this could be extended to other language detection algorithms), and breaks the character stream into words using whitespace and punctuation marks as guidelines. The individual words are then passed into our **Machine Transliteration** (MT) and **Selective Translation** (ST) components. MT implements a Maximum Entropy classifier, details of which are provided later in this section. The ST component contains language-specific dictionaries that contain a relatively small set of common words which often are translated instead of transliterated (translations for "road", "garden", "lane" etc.). While our current implementation translates only common nouns as above, the ST component may be enhanced with known mappings between the query language entities and target language entities (for example, "Vienna" is "Wien" in German, and "Becs" in Hungarian). Such mapping may be obtained from multilingual Gazetteers, and when implemented, may improve the quality of crosslingual location search significantly. The MT component suggests multiple transliterations for each input word, and the ST may add additional words. The set of word alternatives are then combined into a set of most promising transformed subsequences of the query, by taking the cross-product of the word-level candidates. This set of transformed subsequences is ranked by probabilistic estimates provided by the MT system and a small number (currently 4) of top-ranked candidate subsequences are then passed on to the next stage. This number was determined based on experiments on the efficacy of considering various numbers of transliteration permutation alternatives. The multiple transformed subsequences are then passed through our **abstraction** component, a more discriminating form of the SOUNDEX technique, which reduces textual variations inherent in transliteration. The transformed subsequences, along with their abstracted versions, are then passed on to our **Approximate Footprint Lookup** (AFL) component, which looks up the *spatial footprint* (a linear quad-tree [5] representation) of entity names that approximately match each transformed subsequence. Our approximate text matching system is based on [2]. The resultant spatial footprints are passed into our **Interpretation Finder**, which implements the FindInterpretations depth-first-search described in the previous section, to come up with a ranked list of candidate Interpretations. Finally, found interpretations are ranked by a pluggable **Ranker**, which presents the more relevant results first. The Ranker is not specific to crosslingual search and its role is described in [17] and in more detail in [8].

Our MT system uses Viterbi training alignment and a Maximum Entropy classifier to generate target language transliteration candidates for a source language word. We use our own alignment algorithm to align each single character of a training source word with zero or more characters in the target language transliteration. Because it takes the strictly monotonic nature of alignments in transliteration into consideration, it is better suited for the task than general machine translation alignment tools. Based on the resulting alignments, we train a maximum entropy classifier (adapted from [12]) to estimate the probability that source language character, in the context of the 3 preceding and the 3 following source characters, should be transliterated to a given target language string. After training (on 15000 name-pairs for each source-target language combination) is finished, a beam search decoder is used to obtain the *n* most likely transliteration candidates for a (previously unseen) source language word.

Figure 8 illustrates transliteration performance for transliteration from three different languages into English, obtained on a test set of 2,700 words. It plots the probability that a transliteration, which either exactly matches the reference transliteration ("exact") or has an edit distance of at most one character ("$\Delta \leq 1$ char"), is found among the highest ranked *n* ("Top *n*") transliteration candidates. The graph shows that there is a high probability (70% for Arabic, 86% for Hindi and 88% for Japanese) of having at least one very close match (within one char edit distance) amongst the top four transliteration candidates, and the quality improvements are asymptotic beyond top four candidates.

Our location search system indexes around 240,000 entities (street names, locality names, postal codes etc.) belonging to three big cities, one each from UK, USA and India, on a 3.2GHz (Pentium 4) system with 2GB RAM. The generation of 15 transliteration candidates for a source word takes approximately 30 ms (without any optimization for speed) while fuzzy text lookup performs within 10 ms per subsequence.

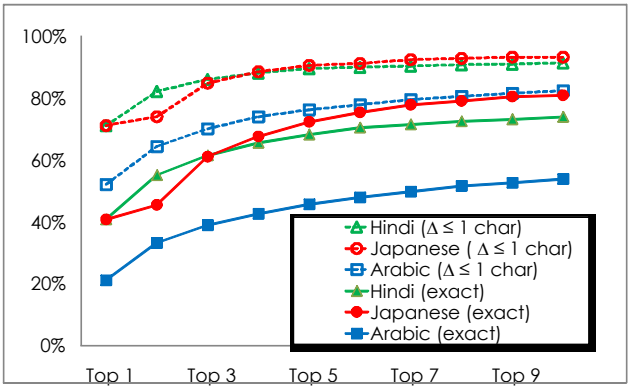

Figure 8: Transliteration performance

The approximate text lookup for a *full query* can be expensive since a single location search can yield several tens of subsequences with word breaking, transliteration/translation and

subsequence regeneration. We use caching to reduce the overall cost of Approximate Footprint Lookup.

5. RESULTS

In this section, we first establish a baseline using our transliteration module and commercial monolingual location search systems, since no other comparable crosslingual location search system exists. Based on our experiments, we find that our system enables broad crosslingual support for a wide variety of location search queries, with results that compare well with the best *monolingual* location search providers. Furthermore, we establish that straightforward approach of transliteration followed by monolingual geocoding, performs poorly across the board.

5.1 Experimental Data

A set of 275 random English address queries, both structured and unstructured, covering geographic regions from the United States and India were collected from users and user location search query logs. The Arabic, Hindi and Japanese versions of these test queries were created manually by an external agency to remove any bias. Thus, we have 275 test queries in each of these languages, making up a total of 825 crosslingual queries. A sample query, along with its transliterations in all three languages, is given in Table 1. For each English query, the gold standard geographic location (Latitude, Longitude) was obtained by majority consensus among multiple commercial location search engines, namely, Google Maps™, Windows Live Local™, and Yahoo Maps™, or by manually locating it on a map.

5.2 Baseline Definition

As there are currently no commercial or academic crosslingual location search systems available, we construct a baseline, using our transliteration system and the commercial location search engines (referred to as, *T + CS*) listed above, as follows: we first transliterate each of the test queries (in Arabic, Hindi and Japanese) to English using our transliteration engine, and then send the four highest ranked transliteration candidates to the three commercial location search engines. We also experimented with larger number of transliteration candidates, but found moving beyond 4 transliterations did not improve the baseline result quality significantly. This is also corroborated by the graph in Figure 8, which shows only asymptotic improvement in transliteration quality, beyond the top-4 candidates, especially for the fuzzy matching. For example, consider the Japanese query "835, セントラル・アベニューエン, ケント, ワシングトン" (a manually created Japanese version of "835, Central Ave N, Kent"). For this Japanese query, the machine transliteration system would typically create candidates such as "835 sentral avenu en kento", etc. It should be noted here that our geocoder contains data for specific regions in India, US and UK, and hence we were restricted to selecting the test queries only for these areas. To ensure parity in comparing our results with commercial services that are set up for the whole world, we make one important modification. We append the names of the region and country, in error-free English, to each query that were sent to the commercial engines to narrow down the scope of the search.

We evaluated the three commercial location search engines, and here we are presenting as the baseline, the performance of *the best of the three commercial services*, when supplied with the four highest ranked transliterations from our transliteration system.

5.3 Experimental Results

Of the 275 test queries, 155 were *structured* addresses, conforming to the local conventions for postal addresses, and the rest were *unstructured* addresses informally describing a geographic location.

Table 3: Experimental Results

	Queries geocoded within 1 km (in %)							
Language	STRUCTURED				UNSTRUCTURED			
	USA		India		USA		India	
	T+CS	Our	T+ CS	Our	T+ CS	Our	T+ CS	Our
ENG	98	85	18	94	22	93	1	92
ARA	34	73	8	79	4	75	0	83
HIN	53	76	12	85	6	85	0	92
JAP	8	78	9	91	3	88	0	86

Table 3 shows our experimental results for both structured and unstructured test addresses from US and India. The numbers denote the percentage of queries for which the geocoding result was within 1 km distance of the gold standard location.

In the left half of Table 3, we show the results for geocoding 155 structured or full postal addresses, in English, Arabic, Hindi and Japanese. It should be noted that in the monolingual case, the unmodified English test queries were provided to the baseline system (along with the region and country information), as well as to our implementation. We observe that our system's performance is marginally poorer than the baseline on monolingual structured data for the USA, but it is significantly better on monolingual Indian queries. For crosslingual structured queries, our system considerably outperforms the baseline combination of transliteration and commercial location search engine in all scenarios.

In the right half of Table 3, we present results for experiments with 120 random unstructured queries. For these, we synthesized unstructured location descriptions by selecting combinations of overlapping features from the underlying repository from each city such as intersecting roads, localities, etc. (described in more detail in [17]). We followed the same process of transliterating and appending the region and country information as in structured queries. The results show that for unstructured queries our system very significantly outperforms the baseline system in the monolingual case, and even more so for in the crosslingual case.

Our claim that integrating transliteration results in an end-to-end system (rather than combining transliterations and monolingual geocoding) will lead to crosslingual performance that is close to monolingual levels is clearly supported by our experimental results illustrated in Figure 9. The figure shows performance (queries geocoded within 1 km, on structured data for US locations) for crosslingual queries, relative to the performance on the original monolingual (English) queries. It illustrates that, even if queries are written in a completely different language and script, our system can still geocode (within 1 km) between 85% (for Arabic) and 90% (Hindi, Japanese) of the queries it can geocode in English. In comparison, the baseline system supplied with the highest ranked 4 transliterations of the foreign language queries, achieves only 8% (Japanese), 35% (Arabic) and 54%

(Hindi) of the monolingual baseline performance. We have deliberately chosen structured queries for US locations as a basis for demonstrating the relative crosslingual performance, because this is the scenario where the commercial baseline has strongest performance on monolingual queries.

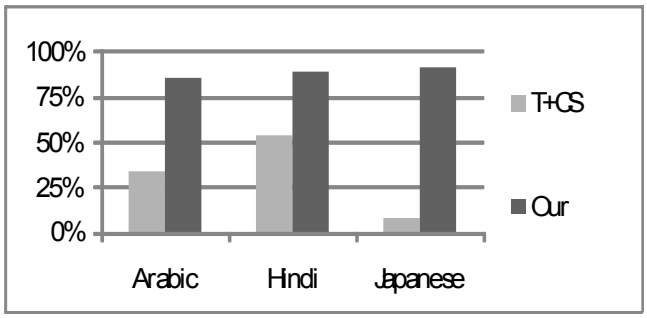

Figure 9: Relative crosslingual performance

Our system performs dramatically better than the baseline of transliteration followed by monolingual geocoding. Even for the USA, where the baseline outperforms us on structured monolingual data, we achieve between 75% and 88% on crosslingual unstructured data, whereas the baseline is between 3% and 6%.

6. CONCLUSIONS AND FUTURE PLANS

In this paper we addressed, possibly for the first time, the problem of processing complex *crosslingual location search* queries, where the underlying map data are in a different language than that of the query. We demonstrate empirically that the simple composition of machine transliteration followed by monolingual geocoding performs poorly, as the ambiguities in the transliteration/translation process compound to unacceptable levels for multi-word queries. We then reformulate the crosslingual location search as an *end-to-end* problem: finding the geometric scope of a query by simultaneously maximizing the crosslingual *textual affinity* and *spatial coherence* (i.e., innovatively using *spatial constraints* to effectively resolve translation ambiguities). We provide results for queries in Arabic, English, Hindi and Japanese, over detailed English map data covering multiple cities. Experiments confirm full crosslingual location search accuracy at levels comparable to that of commercial monolingual systems.

We are pursuing several promising extensions to this work: Currently, we are working on extending our algorithms to include "what" terms, to support local search queries such as "hospitals near Bellevue downtown" (and their crosslingual versions). In addition, our current prototype system [8] that is deployed in 3 demographics supporting 4 languages, is being extended as a distributed system that can handle two orders of magnitude larger data than that used for results in this paper. In addition, we are also exploring improving coverage of entity *translation* between languages, by mining comparable corpora for equivalent entities.

Our approach is generic, and may be used for crosslingual searches in any language/script, over arbitrary spatial data. We hope that this facilitates its adoption in spatial domains beyond geographic location search, such as crosslingual queries over scientific spatial data.

7. ACKNOWLEDGMENTS

We would like to thank the Microsoft Windows Live Local team for providing us Gigabytes of detailed spatial map vector data for several countries. We thank Kalika Bali for several sessions of fruitful discussions on linguistics/phonetics and specific language related issues. In addition, we thank Osama Shabaneh and Pushkar Chitnis for their significant help in validating Arabic and Japanese queries and search results.

REFERENCES

[1] Al-Onaizan, Y. and Knight K. Machine transliteration of names in Arabic text. In *Proc. of ACL Workshop on Computational Approaches to Semitic Languages*, 2002.

[2] Chaudhary, S., Ganjam, K., Ganti, V., and Motwani, R. Robust and efficient fuzzy match for online data cleaning. In *Proc. SIGMOD* 2003.

[3] Christen, P., Churches, T. and Willmore, A. A probabilistic geocoding system based on a national address file. In *Proc. 3rd Australasian Data Mining Conf.*, 2004.

[4] CLEF Forum. http://www.clef-campaign.org/.

[5] Gargantini, I. An effective way to represent quadtrees. In *Comm. of the ACM*, 1982.

[6] GeoCLEF. http://ir.shef.ac.uk/geoclef/.

[7] Goldberg, D. W., Wilson, J. P., and Knoblock, C. A. From text to geographic coordindates: The current state of geocoding. In *J. Urban and Regional Information Systems Assoc.*, 2006.

[8] Joshi, T., Joy, J., and Sengar, V. Robust Location Search. Technical Report MSR-TR-2008-41, Microsoft Research, 2008.

[9] Goto, I., Kato, N., Uratani, N. and Ehara, T. Transliteration considering context information based on the Maximum entropy method. In *Proc. MT Summit IX,* 2004.

[10] Knight, K. and Graehl, J. Machine transliteration. *Computational Linguistics, 24(4)*, 1998.

[11] Kumaran, A. and Kellner, T., A generic framework for machine transliteration, In *Proc. SIGIR*, 2007.

[12] Lin, Dekang. MaxEnt Classifier. 2003. http://www.cs.ualberta.ca/~lindek/maxent.tgz.

[13] Oh,J., Choi, K. & Isahara, H. A comparison of different machine transliteration models. *Artificial Intelligence Research,* 2006.

[14] Pouliquen, B., Steinberger, R., Ignat, C., and D. E. Groeve, T. Geographical information recognition and visualization in texts written in various languages. In *Proc. 19th Annual ACM Sym. on Applied Computing,* 2004.

[15] Rhind, G.R. *Global Sourcebook of Address Data Management A Guide to Address Formats and Data in 193 Countries***.** Gower Publishing Ltd, 2005.

[16] Russell, R. Soundex. US Patent 1,261,167, 1918.

[17] Sengar, V., Joshi, T., Joy, J., Prakash, S., and Toyama, K. Robust location search from text queries. In *Proc. ACM GIS,* 2007.

[18] Viola, P. and Narasimhan, M. Learning to extract information from semi-structured text using a discriminative context free grammar. In *Proc. SIGIR*, 2005.