A Framework for Resolving Open-World Referential Expressions in Distributed Heterogeneous Knowledge Bases

Tom Williams and Matthias Scheutz
Human-Robot Interaction Laboratory
Tufts University, Medford, MA, USA
{williams,mscheutz}@cs.tufts.edu

Abstract

We present a domain-independent approach to reference resolution that allows a robotic or virtual agent to resolve references to entities (e.g., objects and locations) found in open worlds when the information needed to resolve such references is distributed among multiple heterogeneous knowledge bases in its architecture. An agent using this approach can combine information from multiple sources without the computational bottleneck associated with centralized knowledge bases. The proposed approach also facilitates "lazy constraint evaluation", i.e., verifying properties of the referent through different modalities only when the information is needed. After specifying the interfaces by which a reference resolution algorithm can request information from distributed knowledge bases, we present an algorithm for performing open-world reference resolution within that framework, analyze the algorithm's performance, and demonstrate its behavior on a simulated robot.

Introduction

For robotic or virtual situated agents to effectively engage in natural language interactions with humans, they must be able to identify the people, locations, and objects mentioned by their human interlocutors. This ability, known as *reference resolution* (Garrod and Sanford 1994), is necessary in order to discuss or carry out actions involving those people, locations, and objects. In a robotic or virtual component-based integrated agent architecture (e.g, DIARC, Scheutz *et al.* 2013, or ROS, Quigley *et al.* 2009), knowledge may be localized within different components instead of being centralized in a single knowledge base (KB). This paper presents an approach to solving several unique problems that arise when knowledge is distributed in this manner.

Information in an integrated architecture may be decentralized for a variety of reasons. First and foremost, there simply does not exist a single knowledge representation format that would allow a robot to efficiently deal with all representation and reasoning tasks it must perform. For example, information about entities recognized by a vision component will likely be stored in a substantially different manner than the map produced by a mapping component.

Furthermore, accumulation of knowledge into a central, homogeneous KB can create a bottleneck where computational resources become focused onto a single "stress point" rather than balanced across the architecture's components. It may make more sense, for example, to keep information about visual features such as pixels, textures, and edges localized to the vision component where they are actually processed and needed.

Finally, knowledge may be decentralized to facilitate "lazy evaluation". For example, consider a mapping component responsible for performing SLAM. On request, this component may be able to determine whether one location is "to the left down the hall" from another location, or whether two locations are within a five-minute walk of each other. However, it may not be necessary to make such decisions until explicitly requested. If knowledge is centralized in a single KB, then this information must be precomputed and asserted into the knowledge base if it cannot otherwise be inferred – a potentially unnecessary expense if the information is expensive to compute and unlikely to be requested.

The use of a distributed knowledge representation scheme, however, also presents several challenges. For example, multiple aspects of an entity may be spread across multiple KBs (e.g., visual aspects in the vision component, action-based aspects in the manipulation component, linguistic aspects in the NLP component, etc.), and each such KB may have its own form of representation (i.e., whatever form is most natural for the internal operations of that component) and its own way of evaluating queries (especially when "lazy evaluation" is employed).

Furthermore, it may be difficult to determine which KB should be queried in order to resolve a particular referential expression. For example, if an interlocutor says "the ball is in it", it may not be clear whether candidate entities to associate with "it" should be drawn from the set of objects known to the vision component or from the set of locations known to the mapping component.

The rest of this paper proceeds as follows. We first discuss previous approaches to reference resolution under different knowledge representation schemes. We then introduce a framework that allows information from domain-specific resolution techniques to be used together without the need for a centralized KB. We then present an algorithm for performing reference resolution within that framework, analyze its performance, and demonstrate its behavior on a simulated robotic agent. Finally, we discuss the results of our analysis and directions for future work.

Previous Work

Reference resolution in robotics has attracted much attention over the past decade. Previous approaches have typically fallen into one of two categories. We term the first category *domain-independent resolution*. Approaches in this category (e.g., Kruijff *et al.*; Heintz *et al.*; Lemaignan *et al.*; Daoutis *et al.* 2007; 2008; 2011; 2009) typically use a central KB in which information about disparate types of entities are stored in a homogeneous format. Techniques such as graph matching are used to resolve natural language references to entities stored in that KB. These resolution techniques may not be as effective as domain-specific techniques as they are only able to utilize information that can be encoded in the lowest common denominator representation used by the shared ontology (Gray *et al.* 1997).

We term the second category *domain-dependent resolution*. Approaches in this category focus on resolving specific types of references (e.g., spatial reference resolution, Moratz and Tenbrink; Williams *et al.*; Hemachandra *et al.*; Kollar *et al.*; Zender *et al.*; Shimizu and Haas; Chen and Mooney; Matuszek *et al.*; Fasola and Matarić 2006; 2013; 2011; 2010; 2009; 2009; 2011; 2012; 2013, action resolution, Kollar *et al.*; Hewlett 2014; 2011), and use techniques that are specific to their target domain. For example, in (Fasola and Matarić 2013) "semantic fields" are used to interpret descriptions such as "near the kitchen" or "around the table", a technique which might not generalize to other resolution tasks. Also in this category are approaches like that presented in (Tellex *et al.* 2011a), which are general in principle but which must be trained to apply to a single target domain.

In recent work, we presented POWER, a hybrid approach (Williams and Scheutz 2015a; 2015b) in which a domain-independent reference resolution algorithm (of the first category of approaches) made use of a domain-dependent *consultant* that provided capabilities typically found in the second category of approaches. This is similar to mechanisms found in the knowledge representation literature (e.g., procedural attachment, Bobrow and Winograd 1977) and in some cognitive architectures (e.g., PRODIGY, Veloso *et al.* 1995).

In this work, we present a framework which extends POWER to handle multiple consultants distributed across the architecture, thus preventing a computational bottleneck. While the use of architectural components to intercede between distributed hierarchical knowledge bases during querying and assertion is not new *per se* (c.f. Gray *et al.* 1997), we believe this to be the first use of such an approach in a robotic architecture.

Framework

Assume a robotic archtiecture includes a set of n heterogeneous KBs $K = \{k_1, ..., k_n\}$. Each KB k_i is managed by a consultant c_i from $C = \{c_1, ..., c_n\}$. Each consultant performs four functions:

1. advertising the constraints it can evaluate and impose,
2. providing a set of atomic entities from its KB,
3. calculating the likelihood that a given constraint holds for a given set of atomic entities, and
4. adding, removing, or imposing constraints on its KB.

A referential expression in this framework is formulated as a set of constraints $S = \{s_1, ..., s_n\}$ where each $s \in S$ specifies a relationship (s^r, s^V) named by s^r and parameterized by $s^V = \{s^{v_1}, ..., s^{v_n}\}$. For example, "the ball in the box" might be encoded as the set $S = \{(in, \{X, Y\}), (ball, \{X\}), (box, \{Y\})\}$, where $S^V = \{X, Y\}$ contains the variables parameterizing constraints with names 'in', 'ball', and 'box', determined by $s^V_1 \cup s^V_2 \cup ... \cup s^V_n$. The goal of reference resolution is associate each variable in S^V with an entity from the architecture's KBs. The first step towards this goal is to determine which KB contains the referent for each variable.

To facilitate this process, each consultant c_i advertises the types of queries it can handle through a set of unique query templates $Q_i = \{q_{i1}, ..., q_{in}\}$, each of which specifies a relationship (q^r, q^V) named by q^r and parameterized by *kb-associated* variables $q^V = \{q^{v_1:k_1}, ..., q^{v_n:k_n}\}$. Here, each kb-associated variable $q^{v_i:k_i}$ denotes a variable v_i whose referent should be found in KB k_i. For example, the *Visual Consultant* c_O associated with KB of objects k_O might advertise the template $(in, \{X : k_O, Y : k_O\})$, and the *Spatial Consultant* c_L associated with KB of locations k_L might advertise the template $(in, \{X : k_L, Y : k_L\})$. This particular example also demonstrates how relationships that bridge knowledge bases are handled: it is assumed that relationships between pieces of knowledge stored in different KBs will be handled by exactly one of the consultants associated with those KBs. Here, information about the locations of objects is advertised to be handled by the *Spatial Consultant*. The process of associating a KB with each variable is viewed as the process of finding the optimal mapping

$t : V \rightarrow K$ from variables in S^V to KBs K, drawn from set of possible mappings T:

$$\operatorname*{argmax}_{t \in T} \prod_{s \in S} P(t|s).$$

Here, $P(t|s)$ represents the probability that mapping t correctly maps variables to KBs given that s appears in S. If a training corpus is available, $P(t|s)$ can be calculated by consulting the learned conditional distribution $P(T|s)$. Otherwise a uniform distribution may be assumed, and $P(t|s)$ can be calculated as:

$$P(t|s) = \begin{cases} 0, & \text{if } \gamma = 0. \\ 1/\gamma, & \text{otherwise.} \end{cases}$$

$$\text{where } \gamma = \sum_{c_i \in C, q \in Q_i} |matches(q, s)|.$$

Here, $|matches|$ is the number of query templates in S^Q that *match* constraint s (i.e., where $s^r = q^r$ and where s^r and q^r may be unified).

In order to determine the most likely mapping of entities to variables, we must first obtain a set of candidates for each variable, drawn from the appropriate KB. This is performed by choosing the consultant c_v associated with each variable

v, and requesting a list of candidate entities from that consultant by calling $getCandidates(c_v)$. Each possible combination of variable-entity bindings is called a *hypothesis* h. The set of these hypotheses is called H. Then, the process of reference resolution can be modeled as:

$$\underset{h\in H}{\operatorname{argmax}} \prod_{s\in S} P(s|t,h).$$

Here, $P(s|t,h)$ represents the probability that constraint s is an accurate description of the state of the world, given variable-entity mapping h and variable-KB mapping t (as described above). This value is calculated by the component that advertises the relationship matching s with variable-KB mapping t. As it will likely be prohibitively expensive to examine every hypothesis h, we present an algorithm to efficiently search through hypothesis space H.

Algorithms

The distributed POWER algorithm (i.e., DIST-POWER, Algorithm 1) takes four parameters: a query S, a set of consultants C, a mapping T from the set of variables S^V to the set of KBs K managed by C, and a priority queue of initial hypotheses H. C is assumed to be sorted according to some ordering, such as by $|s_i^V|$, so that constraints with only one variable (e.g., $(room, X)$) will be examined before constraints containing multiple variables (e.g., (in, X, Y)), limiting the size of the search space considered. T is assumed to be sorted according to, e.g., the prepositional attachment of the variables contained in T, as described in (Williams and Scheutz 2015b). Each hypothesis h in H contains (1) a set of unapplied constraints h^S, (2) a list of candidate bindings h^B, and (3) $h^P = p(h^B|S,T)$, which is used as that hypothesis' priority.

Algorithm 1 DIST-POWER (S, C, T, H)

```
 1: $S$: list of relationship constraints
 2: $C$: set of consultants
 3: $T$: optimal mapping from $S^V \to K$
 4: $H$: set of initial hypotheses
 5: if $H = \emptyset$ then
 6:     $\alpha = S[0]^V[0]$
 7:     $c_\alpha = find\_consultant(C, T, S[0])$
 8:     for all $\phi \in getCandidates(c_\alpha)$ do
 9:         $push(H, \{\{\alpha \to \phi\}, S, 1.0\})$
10:     end for
11: end if
12: $A$ = resolve$(S, C, T, H, \emptyset)$
13: if $A \neq \emptyset$ and $(A[0])^V \neq S^V$ then
14:     $A = posit(A[0], C, S)$
15: end if
16: return $A$
```

If H is initially empty, DIST-POWER initializes it with a set of hypotheses $\{h_0, ...h_n\}$ where $h_i^S = S$, h_i^B maps the first variable found in $S[0]$ to the i^{th} candidate returned by $getCandidates(c_\alpha)$ (where consultant c_α is determined by $find\_consultant$, i.e., the process described previously), and $h^P = 1.0$ (Algorithm 1, lines 5-11).

Algorithm 2 resolve(S, C, T, H, A)

```
 1: while $H \neq \emptyset$ do
 2:     $h = pop(H)$
 3:     $s = h^S[0]$
 4:     if $(\exists v \in s^V \mid v \notin h^B)$ then
 5:         for all $\phi \in getCandidates(c_v)$ do
 6:             $enqueue(H, h^B \cup (b \to c), h^S, h^P)$
 7:         end for
 8:     else
 9:         $c_h = find\_consultant(C, T, \{s\})$
10:         $h^P = h^P * apply(c_h, s, h^B)$
11:         $h^S = h^S \setminus s$
12:         if $(h^P > \tau)$ then
13:             if $(h^S = \emptyset)$ then
14:                 $A = A \cup h$
15:             else
16:                 $H = H \cup h$
17:             end if
18:         end if
19:     end if
20: end while
21: if $A = \emptyset$ and $T = \emptyset$ then
22:     return resolve$(prune(S, T[0]), C, tail(T), H, A)$
23: else
24:     return $A$
25: end if
```

Resolution is then performed using resolve(S, C, T, H, A) (Algorithm 2), which performs a best-first search over the set of possible assignments from values provided by consultants in C to variables in S. If a solution of sufficient probability cannot be found (line 21), $resolve$ tries again with a restricted set of variables (line 22), recursing until it either finds a sufficiently probable solution or runs out of variables to restrict. This process extends the POWER algorithm (Williams and Scheutz 2015b) in order to choose the *best* consultant for resolution from a set of distributed consultants, instead of only handling a single consultant as POWER did. We thus refer the reader to (Williams and Scheutz 2015b) for the details of the POWER algorithm itself. The POWER algorithm is similar to the algorithm presented in (Tellex *et al.* 2011b), in which a beam search is performed through an initial domain of salient objects in order to identify the most probable satisfaction of an induced probabilistic graphical model. We chose best-first search instead of beam search as a large number of relatively equally likely candidates may exist at each step. When resolving a reference to some "room", for example, it would be imprudent to discard places that did not fall in the top few most "room-like" candidates since there may be hundreds of places that satisfy this constraint to a high degree. We instead rely on a lower probability threshold τ to keep the search space tractable.

Once $resolve$ returns set of candidate solutions A to DIST-POWER (line 24), that set is examined. If A is nonempty, and if the best solution in A does not contain candidate bindings for all variables found in S, then new representations are posited for the entities associated with the missing variables, as described in (Williams and Scheutz 2015b) (Algorithm 1, lines 13-14). These new representations are added to the appropriate KBs by the appropriate consultants, and assigned new identifiers which are used to update A before it is returned.

If A contains exactly one hypothesis, that hypothesis represents the entity likely described by the utterance. If A is empty, no known entity matched the description, and the robot may need to ask for clarification. If A contains more than one hypothesis, the description matched multiple entities, and the robot may need to ask for clarification.

Proof-of-Concept Demonstration

In this section we present a proof of concept demonstration of our proposed algorithm and framework. The purpose of this demonstration is two-fold: First, we will demonstrate that the proposed algorithm and framework behave as intended, that is, that they allow resolution to be performed when the requisite information is distributed across various databases, and that they allow resolution to be performed without knowledge (on the part of the algorithm itself) as to (1) the format of the knowledge stored in each KB, and (2) the techniques necessary for extracting the relevant knowledge from each KB. Second, we will demonstrate that the algorithm and framework have been fully integrated into a robotic architecture in order to perform tasks natural to human-robot interaction scenarios.

The proposed algorithm was integrated into a Resolver component of *ADE* (Scheutz 2006) (the implementation middleware of the Distributed, Integrated, Affect, Reflection Cognition (*DIARC*) architecture, Scheutz *et al.* 2013), which uses a distributed heterogeneous knowledge representation scheme: the architecture has a *Belief, Goal, and Dialog* management component which tracks general information and the beliefs of other agents, but information about visual targets, for example, is localized in the *Vision* component, and information about spatial entities is localized in the *Spatial Expert* component. To implement the proposed framework, a set of "consultants" were implemented to interface with KBs of known objects, locations, and people. Each consultant performed four functions:

1. Each advertised the types of queries it handled by exposing a list of formulae such as $in(W - objects, Y - locations)$. This formula, for example, states that the consultant which advertises it is able to assess the degree to which some entity from the `objects` knowledge base is believed to be in an entity from the `locations` knowledge base.
2. Each provided a method which returned a set of numeric identifiers of the atomic entities in its associated KB.
3. Each provided a method which, given formula p (e.g., $in(X - objects, Y - locations)$) and mapping m from variable names to numeric identifiers, (e.g., from X and Y to 22 and 25) would return the probability that relationship p held under the variable bindings specified in m. In this example, the appropriate consultant would return the degree to which it believed object 22 to be in location 25.
4. Each provided a method which, given a set of formulae with some unbound variables, would posit new representations to associate with those unbound variables, store the knowledge of their properties represented by those formulae, and return new variable bindings accounting for the newly posited entities.

The Resolver provided a DIST-POWER method which, given a set of formulae S, calculated optimal mapping T and executed the DIST-POWER (S,C,T,H) algorithm.

As a proof of concept demonstration, we examined a robot's behavior in interpreting the utterance "Jim would like the ball that is in the room across from the kitchen" (assumed to be uttered by an agent named "Bob"). This utterance is represented as:
$Stmt(Bob, self, and(wouldlike(Jim, X), ball(X), in(X, Y), room(Y), acrossfrom(Y, Z), kitchen(Z)))$:
A statement from "Bob" to the robot (i.e., "self"), where the head of the and list (i.e., $\{wouldlike(Jim, X)\}$) represents the literal semantics of the sentence, and the tail of the and list represents the properties which must be passed to the Resolver for resolution.

We will now describe the behavior of the Resolver R as it follows the DIST-POWER algorithm, detailing the state of R's hypothesis queue at several points throughout the trace of the algorithm. In order to provide an easily describable example, we limited the number of entities in the initial populations of each KB to three or four entities. The robot's knowledge base of locations contained a hallway and several rooms, including a kitchen, and a room across from it which only contained, to the robot's knowledge, a table. The robot's knowledge base of objects contained the table and several boxes and balls. We will use o as shorthand for `objects` and l as shorthand for `locations`.

R first calculates optimal mapping T, and returns $\{X : o, Y : l, Z : l\}$, determining that the first constraint to be examined will be $ball(X)$. R thus instantiates its hypothesis queue by requesting a set of candidate entities for X from the consultant associated with KB o, which produces $\{o_1, o_2, o_3, o_4\}$. R then requests from o the probability of each of $\{ball(o_1), ball(o_2), ball(o_3), ball(o_4), \}$ being true, and receives back, respectively, $0.82, 0.92, 0.0, 0.0$. Since $0.0 < 0.1$ (the chosen value of τ), the hypotheses with mappings $X : o_3$ and $X : o_4$ are thrown out, and the other two hypotheses are returned to H, resulting in hypothesis queue:

Binding	Unconsidered Constraints	P
$\{X : o_2\}$	$\{room(Y), kitchen(Z), in(X, Y), acrossfrom(Y, Z)\}$	.92
$\{X : o_1\}$	$\{room(Y), kitchen(Z), in(X, Y), acrossfrom(Y, Z)\}$	.82

The next constraint to be considered is $room(Y - l)$. Since $\{X : o_2\}$ does not contain a candidate identifier for Y, R requests the initial domain of Y from l, receives $\{l_1, l_2, l_5, l_6\}$, and replaces the first hypothesis with a set of four hypotheses which each have a different binding for Y but share the original P value and set of unconsidered constraints. $P(in(o_2, l_i))$ is then assessed for each of these four hypotheses, resulting in, respectively, $0.82, 0.92, 0.0, 0.6$. The third hypothesis is thrown out and the others are returned to H with updated probabilities, producing:

Binding	Unconsidered Constraints	P
$\{X : o_2, Y : l_2\}$	$\{kitchen(Z), in(X, Y), acrossfrom(Y, Z)\}$	.846
$\{X : o_1\}$	$\{room(Y), kitchen(Z), in(X, Y), acrossfrom(Y, Z)\}$	.820
$\{X : o_2, Y : l_1\}$	$\{kitchen(Z), in(X, Y), acrossfrom(Y, Z)\}$	.754
$\{X : o_2, Y : l_6\}$	$\{kitchen(Z), in(X, Y), acrossfrom(Y, Z)\}$	.736

As the hypothesis with binding $\{X : o_2, Y : l_2\}$ is then the most likely hypothesis and the next constraint to consider is $kitchen(Z)$, Z is expanded with candidate locations, each checked for the $kitchen(Z)$ property. As only location 2 is known to be a kitchen, the first hypothesis is replaced with a single new hypothesis, with probability 0.762. This causes the hypothesis with binding $\{X : o_1\}$ to become the most probable hypothesis, resulting in the above process being repeated for that hypothesis, producing:

Binding	Unconsidered Constraints	P
$\{X : o_2, Y : l_2, Z : l_2\}$	$\{in(X, Y), acrossfrom(Y, Z)\}$	.762
$\{X : o_2, Y : l_1\}$	$\{kitchen(Z), in(X, Y), acrossfrom(Y, Z)\}$	.754
$\{X : o_1, Y : l_2\}$	$\{kitchen(Z), in(X, Y), acrossfrom(Y, Z)\}$	.754
$\{X : o_2, Y : l_6\}$	$\{kitchen(Z), in(X, Y), acrossfrom(Y, Z)\}$	.736
$\{X : o_1, Y : l_1\}$	$\{kitchen(Z), in(X, Y), acrossfrom(Y, Z)\}$	.672
$\{X : o_1, Y : l_6\}$	$\{kitchen(Z), in(X, Y), acrossfrom(Y, Z)\}$	.656

When the next best hypothesis is examined, it will be eliminated, as o_2 is not known to be located in l_2. Indeed, as no ball is known to exist in a room across from a kitchen, all hypotheses are systematically eliminated. Once this has finished, DIST-POWER removes the head of T and tries the entire above process again, with $T = \{Y : l, Z : l\}$ and $S = \{room(Y), acrossfrom(Y, Z), kitchen(Z))\}$. The elimination of X from these sets suggests that X refers to an entity which is not yet known to the robot. This time, the initial hypothesis queue is, after considering the first formula in S (i.e., $room(Y)$):

Binding	Unconsidered Constraints	P
$\{Y : l_2\}$	$\{kitchen(Z), acrossfrom(Y, Z)\}$	.92
$\{Y : l_1\}$	$\{kitchen(Z), acrossfrom(Y, Z)\}$	.82
$\{Y : l_6\}$	$\{kitchen(Z), acrossfrom(Y, Z)\}$	.8

After going through the same resolution process, the final hypothesis queue will be:

Binding	Unconsidered Constraints	P
$\{Y : l_1, Z : l_2\}$	$\{\}$	.702

DIST-POWER then instructs the `objects` consultant to create a new representation for X, the new identifier for which is then used to update the hypothesis queue:

Binding	Unconsidered Constraints	P
$\{X : o_5, Y : l_1, Z : l_2\}$	$\{\}$	.702

DIST-POWER then instructs both the `objects` and `locations` consultants to maintain consistency with S under the bindings of the remaining hypothesis h. This results in the `objects` consultant asserting into its KB that o_5 is a ball, and the `locations` consultant asserting into its KB that l_1 contains o_5.

R then uses h^B to convert $wouldlike(Jim, X)$ into $wouldlike(Jim, o_5)$. The utterance $Stmt(Bob, self, wouldlike(Jim, o_5))$ is then returned to the Dialog module of DIARC's Belief, Goal and Dialog Management component.

While resolution confidence could be used to determine whether to ask for clarification, we currently pass the utterance directly to a pragmatic reasoning component, which uses a set of pragmatic rules to produce a set of candidate underlying intentions (Williams *et al.* 2015a). One such rule in this set is:

$$Stmt(S, L, wouldlike(C, O)) \xrightarrow{[0.95,0.95]} goal(L, bring(L, O, C)),$$

indicating the robot is 95% sure[1] that when S tells L that C would like O, their intention is for L to have a goal to bring O to C. The robot thus determines that Bob wants it to bring object o_5 (which is in room l_1) to Jim. The robot responds "Okay" and drives to l_1 to retrieve object o_5.

Quantitative Analysis

In this section we analyze the performance of DIST-POWER compared to our previous, non-distributed, POWER algorithm. This analysis is not presented as an evaluation *per se*, but rather to demonstrate that the DIST-POWER algorithm, in addition to providing new capabilities and opportunities for easier integration, provides improved efficiency: even *without* the use of heuristics and domain-dependent tricks. This analysis is thus presented as a baseline which may be improved upon using such heuristics. Future work should include an extrinsic, task-based evaluation.

For this analysis we generated forty KBs: five each of sizes *n* = 20, 40, . . . , 160 where *n* indicates the number of entities in each KB. In each KB, half of the entries were locations in a random floor plan (i.e., rooms, halls, intersections and floors) with various properties with randomly assigned likelihoods; the rest were objects (i.e., balls, boxes and desks), each randomly assigned properties and room of location. Baseline performance was assessed by measuring the average time taken by POWER to evaluate the query associated with "the box in the room" for each set of five KBs.

We then generated forty additional pairs of KBs: five pairs each of sizes *(n_1, n_2)* = (10,10), (20,20), (30,30), . . . , (80,80) such that the first KB dealt with all location-based knowledge and the second KB dealt with object-related knowledge. Performance of DIST-POWER was established by measuring the average time taken to evaluate the query associated with "the box in the room" for each set of five KB *pairs*.

Figure 1 shows the results of this experiment: along the horizontal axis are the sum sizes of KBs used in each

[1] As indicated by the Dempster-Shafer theoretic *confidence interval* $[0.95, 0.95]$. For more on our use of this uncertainty representation framework, we direct the reader to our previous work (Williams *et al.* 2015a).

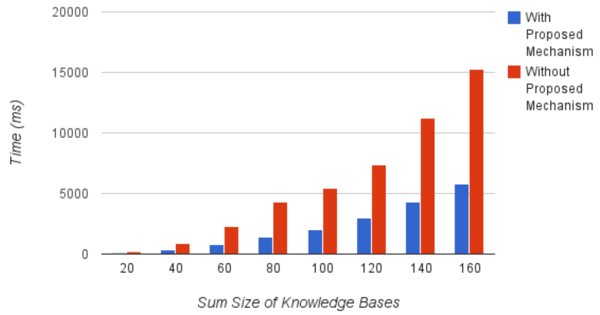

Figure 1: Performance Differences

test case (e.g., "40" refers to the KB containing 40 entities used when analyzing performance *without* the proposed mechanism, and the two KBs containing 20 entities each used when analyzing performance *with* the proposed mechanism.) Along the vertical axis is the average time taken, for each set of KBs of each size, to perform the simple query described. From these results one may observe the performance improvement effected through use of the proposed algorithm: up to 3x speedup among the examined cases.

Discussion

One will notice that both algorithms show performance exponential in the number of stored entities, due to the use of best-first search over, e.g., beam search. However, the complexity of both algorithms when used in the real world would likely be substantially reduced, for several reasons. First, the consultants used by DIST-POWER did not use any heuristics when returning the set of initial candidates to consider. While these would certainly be employed in practice, but using them here would have conflated the performance of the algorithm with the performance of those heuristics, which is beyond the scope of this paper.

Second, complexity would be significantly reduced by tracking the entities in, e.g., the robot's short term memory, and checking against those entities before querying the robot's knowledge bases. In fact, we are currently working to integrate DIST-POWER into a larger resolution framework inspired by Gundel et al.'s *Givenness Hierarchy* (Gundel *et al.* 1993), which will both substantially reduce complexity and allow a robot to resolve references occurring in a wider variety of linguistic forms (Williams *et al.* 2015b).

We also note that in order to have a consistent evaluation, the POWER and DIST-POWER algorithms were provided with information represented in the same way. However, one of the primary advantages of the DIST-POWER algorithm is that information need not be represented in a single format; the information stored in the `locations` knowledge base could just as easily have been represented in a topological map rather than as a database of formulae. In fact, this was the case for our proof-of-concept demonstration.

Finally, we would like to discuss how the experiments demonstrate the architectural commitments of *DIARC* facilitated by DIST-POWER. First, *DIARC* does not prescribe any single knowledge representation. This is facilitated by distributing information amongst KBs of *heterogeneous* representation. Second, *DIARC* uses formulae for *intercomponent communication* whenever possible. This is facilitated by accepting queries represented as sets of formulae. Finally, *DIARC* components should perform processing asynchronously, with components possibly spread across multiple computers. This is facilitated by allowing information and processing to remain localized in separate components, rather than using a single centralized KB. However, DIST-POWER is not incremental or parallelized, aspects which would yield tighter adherence to this architectural commitment, suggesting directions for future work.

Future Work

We have already presented several directions for future work, including parallelization, incrementalization, and emplacement within a larger resolution framework, in order to both increase efficiency and bring our approach closer in line with psycholinguistic reference resolution theories. In this section we present two additional directions for future work.

First, the probability of the best candidate referent, and the number of possible candidate referents, should be used to initiate resolution clarification requests. Doing this appropriately will require the algorithm to be able to distinguish *uncertainty* from *ignorance*; ideally, the algorithm would be able to distinguish between a consultant responding that it *does not know* whether a certain object has a certain property, and that consultant simply returning a fairly low probability that an object has a certain property. This could be effected, e.g., through a Dempster-Shafer theoretic approach, similar to that seen in (Williams *et al.* 2015a).

Second, we will investigate the performance of DIST-POWER when different heuristics are used by its components, and under different constraint-ordering strategies. For example, it may be more efficient to consider rarer constraints first so as to quickly prune the search space. On the other hand, it may be more efficient to instead sort constraints by cost, so that expensive constraints are only examined after establishing that less expensive constraints hold.

Conclusion

In this paper we introduced a framework for performing open-world reference resolution in an integrated architecture with knowledge distributed among heterogeneous KBs. We then presented the DIST-POWER algorithm for efficiently searching the space of candidate referential hypotheses, along with an objective analysis of algorithm performance and a proof-of-concept demonstration of behavior on a simulated robot, showing how the algorithm helps address the challenges of performing reference resolution with a distributed, heterogeneous knowledge representation scheme.

Acknowledgments

This work was funded in part by grant #N00014-14-1-0149 from the US Office of Naval Research.

References

Daniel Bobrow and Terry Winograd. An overview of KRL, a knowledge representation language. *Cognitive science*, 1(1):3–46, 1977.

David Chen and Raymond Mooney. Learning to interpret natural language navigation instructions from observations. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence*, 2011.

Marios Daoutis, Silvia Coradeshi, and Amy Loutfi. Grounding commonsense knowledge in intelligent systems. *Journal of Ambient Intelligence and Smart Environments*, 2009.

Juan Fasola and Maja J Matarić. Using semantic fields to model dynamic spatial relations in a robot architecture for natural language instruction of service robots. In *IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems*, 2013.

Simon Garrod and Anthony Sanford. Resolving sentences in a discourse context: How discourse representation affects language understanding. In M. Gernsbacher, editor, *Handbook of Psycholinguistics*. Academic Press, 1994.

Peter Gray, Alun Preece, NJ Fiddian, and et al. Gray, WA. Kraft: Knowledge fusion from distributed databases and knowledge bases. In *DEXA Workshop*, 1997.

Jeanette Gundel, Nancy Hedberg, and Ron Zacharski. Cognitive status and the form of referring expressions in discourse. *Language*, pages 274–307, 1993.

Fredrik Heintz, Jonas Kvarnström, and Patrick Doherty. Knowledge processing middleware. In *Simulation, Modeling, and Programming for Autonomous Robots*, pages 147–158. Springer, 2008.

Sachithra Hemachandra, Thomas Kollar, Nicholas Roy, and Seth Teller. Following and interpreting narrated guided tours. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2011.

Daniel Hewlett. *A framework for recognizing and executing verb phrases*. PhD thesis, University of Arizona, 2011.

Thomas Kollar, Stefanie Tellex, Deb Roy, and Nicholas Roy. Toward understanding natural language directions. In *Proceeding of the 5th ACM/IEEE International Conference on Human-Robot Interaction*, pages 259–266, 2010.

Thomas Kollar, Stefanie Tellex, Deb Roy, and Nicholas Roy. Grounding verbs of motion in natural language commands to robots. In *Experimental Robotics*. Springer, 2014.

Geert-Jan M Kruijff, Pierre Lison, Trevor Benjamin, Henrik Jacobsson, and Nick Hawes. Incremental, multi-level processing for comprehending situated dialogue in human-robot interaction. In *Symp. on Language and Robots*, 2007.

Séverin Lemaignan, Raquel Ros, Rachid Alami, and Michael Beetz. What are you talking about? grounding dialogue in a perspective-aware robotic architecture. In *RO-MAN*, pages 107–112, 2011.

Cynthia Matuszek, Evan Herbst, Luke Zettlemoyer, and Dieter Fox. Learning to parse natural language commands to a robot control system. In *Proc. of the 13th Int'l Symposium on Experimental Robotics (ISER)*, 2012.

Reinhard Moratz and Thora Tenbrink. Spatial reference in linguistic human-robot interaction. *Spatial Cognition and Computation*, pages 63 – 106, 2006.

Morgan Quigley, Josh Faust, Tully Foote, and Jeremy Leibs. Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, 2009.

Matthias Scheutz, Gordon Briggs, Rehj Cantrell, Evan Krause, Tom Williams, and Richard Veale. Novel mechanisms for natural human-robot interactions in the diarc architecture. In *Proceedings of AAAI Workshop on Intelligent Robotic Systems*, 2013.

Matthias Scheutz. ADE: Steps toward a distributed development and runtime environment for complex robotic agent architectures. *Applied Artificial Intelligence*, 2006.

Nobuyuki Shimizu and Andrew Haas. Learning to follow navigational route instructions. In *Proceedings of the 21st Int'l Jont Conf. on Artifical intelligence*, 2009.

Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R Walter, Ashis Gopal Banerjee, Seth Teller, and Nicholas Roy. Approaching the symbol grounding problem with probabilistic graphical models. *AI Mag.*, 2011.

Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R Walter, Ashis Gopal Banerjee, Seth Teller, and Nicholas Roy. Understanding natural language commands for robotic navigation and mobile manipulation. In *Proc. of the 25th AAAI Conf. on Artificial Intelligence*, 2011.

Manuela Veloso, Jaime Carbonell, Alicia Perez, Daniel Borrajo, Eugene Fink, and Jim Blythe. Integrating planning and learning: The prodigy architecture. *Journal of Experimental & Theoretical Artificial Intelligence*, 7(1):81–120, 1995.

Tom Williams and Matthias Scheutz. A domain-independent model of open-world reference resolution. In *Proc. of the 37th annual meeting of the Cognitive Science Society*, 2015.

Tom Williams and Matthias Scheutz. POWER: A domain-independent algorithm for probabilistic, open-world entity resolution. In *IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems*, 2015.

Tom Williams, Rehj Cantrell, Gordon Briggs, Paul Schermerhorn, and Matthias Scheutz. Grounding natural language references to unvisited and hypothetical locations. In *Proc. of the 27th AAAI Conf. on Artificial Intelligence*, 2013.

Tom Williams, Gordon Briggs, Bradley Oosterveld, and Matthias Scheutz. Going beyond command-based instructions: Extending robotic natural language interaction capabilities. In *Proceedings of 29th AAAI Conference on Artificial Intelligence*, 2015.

Tom Williams, Stephanie Schreitter, Saurav Acharya, and Matthias Scheutz. Towards situated open world reference resolution. In *AAAI Fall Symposium on AI for HRI*, 2015.

Hendrik Zender, Geert-Jan Kruijff, and Ivana Kruijff-Korbayová. Situated resolution and generation of spatial referring expressions for robotic assistants. In *Proceedings of the 21st Int'l Joint Conf. on Artifical Intelligence*, 2009.