Knowledge Discovery

In the age of the Internet and global collaboration, we have done very well at constructing “search engines” for people to quickly find information related to some query.

Web page publisher to web page reader, the process is **human knowledge sharing**.

Knowledge Discovery

In the machine world, we have an analog of task. For reasons of performance and economy of construction, we desire our agents to participate in **machine knowledge sharing**.

The knowledge sharing effort (**KSE**) of the early to mid nineties initiated research into the mechanisms needed to achieve knowledge sharing.

Mostly this is an economy issue. The cost of knowledge base construction is very high. We want to port to as great an extent as possible a knowledge base across both space and time. Doing so will leverage the initial cost of construction and make large-scale knowledge bases more practical to develop.

…but this is also a performance issue. Scenarios are conceivable where of some *very* large knowledge base, only some small subset of the knowledge is really used. So, in a knowledge sharing environment, the entire system can be seen as the *very* large knowledge base and some agent locally learns only what it needs to accomplish its task…thereby keeping its local knowledge base small and efficient all the while having access to more knowledge as needed.

Knowledge Discovery

Most of the work was centered on syntax, semantics, and the pragmatics of communication.

KQML (Knowledge Query and Manipulation Language) and **KIF** (Knowledge Interchange Format) are two examples of the results of the KSE.

Knowledge Discovery

Much of the work on sharing knowledge (using KQML and KIF) assumed the existence of a **common ontology** and the **relative ease of locating sources of knowledge**, both of which are practical only in **closed systems**.

A closed system in this sense is finite in its components with a relatively static structure and all components are centrally controlled and trusted.

In contrast an open system (such as the Internet) can be extremely large, highly dynamic, and makes no guarantee on the capabilities of the control or authority of other systems. Caveat emptor! Carpe Diem! (…and other appropriate Latin phrases)

Knowledge Discovery

As the Internet continues to transition from a human-centric system to a machine-centric system, we have by sharing knowledge, a unique opportunity to construct knowledge-based systems with lower cost and greater knowledge than ever.

However, on the scale of the whole Internet, we cannot assume the luxuries of a tightly controlled closed-system.

Knowledge Discovery

Specifically in an **open system**…

- Locating sources of knowledge becomes a process rather than an event
- Agents are assumed to be social, but not necessarily fully cooperative
- No common ontology is assumed to exist

An open system is scalable (there is no limit to its size – either big or small), dynamic (communications channels and components are not always available), and not centrally controlled.

Locating sources of knowledge…

This is a corollary of the scale and dynamism of an open system. An agent should not be hardcoded with models of its peers but instead incorporate mechanisms for discovering new peers and updating models as necessary.

Agents are assumed to be social…

Agents are assumed to be social, they can communicate, interact, etc, but we cannot guarantee the degree to which an agent is cooperative. Indeed it is likely (and encouraged) that entrepreneurs will exploit the open system with self-interested agents.

No common ontology is assumed to exist…

In an open system, we cannot make statements about the task domain of the system as a whole…to do so would imply that the agents are cooperative which is already explicitly stated to not be the case. As such, we cannot assume (nor enforce) a common ontology on all agents. Instead, the agents will need to negotiate and learn as they go.

Knowledge Discovery

Problem statement…

Given the environment an open system, devise a method for agents to share knowledge.

Or in other words…

Knowledge Consumers can locate and communicate with Knowledge Suppliers so that the end result is that Knowledge Consumers can integrate new knowledge on demand.

Knowledge Sources

A Knowledge Source (K-Source or K-Supplier) is composed of…

- Discrete packages of high-granularity knowledge
- Models of other K-Sources
- Ontology for description

A package of high-granularity knowledge would include knowledge at the object and collections of objects level, rather than at the predicate-logic / fact level. Cases in a CBR system are "high-granularity" whereas hand-empty or on(A,B) are definitely *not* high granularity.

The high-granularity qualification is one of practical necessity. The cost of indexing knowledge at the level of facts could be quite expensive and indeed there may even be little benefit from doing so. (We humans tend to share knowledge in high-granularity packages.)

The models of other K sources includes their aggregate index information as well as pointers to their location.

Knowledge Sources

An agent can interact with a K-Source in the following ways...

- **DESCRIBE** - returns a list of weighted terms that the K-Source believes adequately describe its knowledge in aggregate.
- **TOC** - A list of (knowledge) object IDs and their local classification
- **QUERY** - Same as TOC, but relative to some question.

DESCRIBE would return a list such as…

Astro:/luminous bodies/stars/spectral class A -> 0.5

Astro:/luminous bodies/stars/spectral class G -> 0.5

TOC would return a list such as…

0x0010 Astro:/luminous bodies/stars/spectral class A-> 1.0

0x0010 Astro:/massive bodies -> 0.3

0x001F ….

A QUERY returns a list similar to the TOC, but the list is relative to some question…such as…

“Astro://Planets/Sol/Mars”

Knowledge Sinks

A Knowledge Sink (K-Sink or K-Consumer) is composed of…

- A knowledge base
- Models of known K-Sources
- Ontology for description

Note that if the knowledge base is organized into a collection of high-granularity knowledge objects, then then the K sink needs only a socially extroverted quality to be functionally equivalent to a K source…this is intentional. ;)

Subproblems: Locating

Locating K-Sources can be accomplished in a several ways…

- Explicit introduction
- Broadcast
- Referral
- Via facilitator

Explicit introduction is the simplest and most obvious. Here the K sink (agent) is "given" a list of K sources. On boot-strap the K sink then initiates communication with the K sources on the list in order to construct models of their capabilities.

Broadcast works only on the local subnet…it is akin to shouting out an introduction in a room and hoping that someone interesting is close enough to hear you.

K sources that index other K sources can at their option respond to referral queries. It is important to note that this is optional as there may be reasons why a K source does *not* want to refer a K sink to its own sources. For instance, some search company may gain revenue from answering questions from clients. The company's real asset is in the other K sources that it has indexed. If they were to give these sources away, they would be losing their assets.

If a facilitator is known (such as the specialized K source that indexes only other K sources), the facilitator can be queried for referrals. By definition the facilitator is "cooperative", in other particular, it will answer referral queries…but they might come at a price! ☺

Subproblems: Representing K Sources

- For some K-Source, KS_0 the aggregate knowledge of KS_0 is the union of its local knowledge and the knowledge of its peers.
- A K-Source represents itself using its own local ontology
- A K-Source describes itself as a weighted vector of the most specialized terms in its ontology

So a K source describes itself as a weighted vector of terms from its own description ontology. The sum of the vector is one…so choose carefully! A higher weight implies that more knowledge is contained in that area.

The weighted vector could be interpreted as the proportion of knowledge objects that have that classification.

Note also that ontologies as used for description are inherently hierarchical. So…for some non-leaf node, the weight on the node is the sum of the weights of its children. This property is useful for other K sources and K sinks that lack the specialized richness of some ontology. The upper layers, being more abstract and general are more likely to be included in other ontologies.

For example.

KS0 is an astronomical K source. So its local ontology is very descriptive of celestial bodies.

Astronto:/luminous bodies/stars/spectral class A -> 0.5

Astronto:/luminous bodies/stars/spectral class G -> 0.5

Where as some arbitrary other K agent may have

Common:/stars

And may resolve KS0 to

Subproblems: Ontologies

Ontologies used for classification and description are essentially thesauri and contain the following relationships:

- **IS-A** – generalization / specialization
- **BT** / **NT** - broader / narrower term
- **INSTANCE-OF** – global objects
- **SYN** – same meaning
- **DEF** – a description as a sentence of other terms

So an ontology used here for description is inherently hierarchical.

The IS-A relationships, as edges on a digraph, can carry membership weights in the range of (0,1]

BT / NT allows for specifying terms that are outside of a true taxonomy -- where IS-A is not appropriate.

DEF is a "sentence" of other terms. In this case a sentence is essentially a weighted vector of terms such that the term does not include itself in its description.

Example should follow.

IS-A allows the ontology to express OO-style taxonomies.

INSTANCE-OF allows the ontology to describe implicitly global *objects* rather than classes. For instance, an ontology describing the ecosystem of the Earth may have "Earth" as an INSTANCE-OF "Planet" Likely, most ontologies will only describe classes of objects rather than objects.

Subproblems: Ontologies

- In an open system, ultimately there is no "common" ontology other than the natural language that is used to create it.
- An ontology that has all its DEF statements defined in some natural language is said to be **NL-grounded**
- An ontology that has some or all of its DEF statements defined in the terms of another ontology is said to be **lifted**

Examples…

DEF(Astro:/Earth,"the origin of H. sapiens")

DEF(Astro:/luminous bodies/stars,"massive bodies of primarily hydrogen and trace heavy elements that radiate from the process of fusion induced by hyper-gravitation")

DEF(Astro:/luminous bodies/stars, <physics:/fusion +.7, physics:/gravity +.3>)

DEF(physics:/fusion, "the process of transforming two or more atomic nuclei of low mass into one atomic nuclei of higher mass while liberating energy")

DEF(physics:/gravity, "the force between two massive bodies")

So…phys is an NL-grounded ontology, while Astro is a lifted ontology.

Subproblems: Ontologies

- Ontologies in open systems need to be distinguished by **namespaces**.
- A namespace is some network-unique identifier that prefixes the name of the ontology.
- Within a namespace, a term has only one meaning. Thus namespaces alleviate **polysemy**

For instance:

DEF(Astro:/Earth, “the planet on which H. sapiens originated”)

DEF(Ag:/Earth,”soil for planting”)

Related Work: UMDL

- University of Michigan Digital Library
- An open system
- Agents are social, but not necessarily cooperative
- Ontology-based
- Ontologies can be seeded and grown dynamically.
- Many classes of agents

Related Work: InfoSleuth

- Dynamic federated information system
- MAS with specialized agents (UI, ontology, etc)
- Uses word vectors for ontological resolution

Related Work: DOGGIE

- Distributed Ontology Gathering Group Integration Environment
- Uses **distributed collective memory** for comparing and learning foreign ontologies
- Two different classifications can be applied to some common set of objects. The classification schemes are then commensurated from the results of the co-description.