Service Classification in a Proto-Organic Society of Agents

Peter Weinstein and William P. Birmingham

Artificial Intelligence Laboratory
University of Michigan
1101 Beal Ave., Ann Arbor, MI 48109-2110
{peterw, wpb}@eecs.umich.edu

Abstract

Service classification helps agents select services in societies of agents that are large, dynamic, and evolving. *Services* are capabilities that agents make available to other agents. Existing systems do not use complex service descriptions, or do not handle change well. Our Service Classifier Agent (SCA) maintains a *dynamic ontology*, using description logic to organize services defined by agents at *runtime*. Agents requiring a service select the best available, utilizing knowledge derived with subsumption and other types of inference. Thus, new agents can immediately participate in the society, without requiring modification of existing agents. The SCA has demonstrated its usefulness in a society of agents organized as a computational economy.

1 Introduction

Runtime classification of services helps agents select other agents to help solve problems. *Services* are the capabilities that agents make available to other agents. To advertise, agents define their services at runtime, using concepts from a set of nested ontologies. A Service Classification Agent (SCA) classifies these definitions in a subsumption taxonomy, using description logic. To find available services that meet their needs, agents query the SCA. The taxonomy of services constitutes a *dynamic ontology*, from which concepts can be used to define additional services.

Societies of agents in the future will be large, dynamic, evolving, and terminologically heterogenous. We call these *organic societies of agents*, to highlight the decentralized way that we expect these systems to develop.

Accurate selection of services in an organic society of agents is difficult. Because the society is large, there are too many kinds of agents for any agent to know about all of them. Because it is dynamic, agent availability is in constant flux. Because it is evolving, new agents will appear, and old agents change their behavior. Finally, even if we assume a standardized syntax for domain knowledge, such as KIF [Genesereth and Fikes 92], common semantics for domain knowledge is a different story. The infinite variety of problems and contexts makes terminological heterogeneity—the use of different language to describe equivalent services—inevitable. Implications for the selection of services are obvious.

Runtime classification of services can support selection of services in organic societies of agents. Ontologies permit declarative description of services, independent of the current population of agents. The subsumption taxonomy can be used to reason about services, including selection of the best service available to meet a need. The result is that new agents can join the society and immediately participate, without requiring any modification of existing agents. This occurs when the new agent's service description best matches existing agents' requests for services. Also, the expressiveness and reasoning power of description logics may enable translation between ontologies, thus mitigating difficulties created by terminological heterogeneity. (Description logics, however, are not quite right for this task. See Weinstein and Birmingham [97a].)

Previous approaches for selecting agent services have either used expressive representations in a static way, or less generally adequate representations for societies subject to change (i.e. dynamic or evolving). For example, the PACT system [Olson et. al. 95] uses ontologies to define agent capabilities implicitly, but not to represent agent services *qua* services. Agents must know *a priori* about the other agents.

Techniques for selecting services in societies with dynamic or evolving agent populations use representations that lack either the expressiveness, solid theoretical foundations, or generic applicability of ontologies. These techniques can be characterized on a spectrum, with a tradeoff between precision and reliability, and the amount of domain knowledge required. For example, keyword-based matching is universally available, but is neither precise or dependable. Pattern matching and unification algorithms use local context around keywords to improve matching, and these can be augmented with inference rules that know about the content of expected messages [Kuokka and Harada 95]. On the knowledge-intensive end of the spectrum, context logic [Guha 92] can be used to translate rigorously between requests and service descriptions [Fikes et. al. 95]. Unfortunately, precise mapping axioms are quite difficult to develop in realistic domains, even for trained humans.

Section 2 of this paper describes runtime classification of services as currently implemented by the SCA. Section 3 describes the use of the SCA in a proto-organic society of agents that we call the Service Markets Society (SMS), developed by the University of Michigan Digital Library project (UMDL) [Birmingham et. al. 94]. In the SMS, agents negotiate to buy and sell services within a computational economy. This is an application of market-oriented programming [Wellman 93]. The SCA supports markets in the SMS in some very interesting ways. Section 4 describes our plans for future research, and Section 5 concludes.

2 Runtime classification of services

This section describes the design of an initial version of the SCA, as used in the UMDL SMS. We first describe the role of ontologies in our system. Next, we show how description logic maintains the dynamic ontology of available agent services. A more detailed discussion is available in Weinstein and Birmingham [97a].

The current SCA is implemented using the UMDL agent class [Durfee, Kiskis, and Birmingham 96], and the Loom description-logic system [MacGregor 91]. An interactive demonstration of the SCA is available on the web at http://www.umich.edu/~peterw/Ontology/sca.html.

2.1 Role of ontologies

The most fundamental reason to use ontologies is that we need language to encode declarative descriptions of complex agent services. Declarative descriptions are required to establish a space of services independent of the implementation of the current set of agents.

Representing complexity in digital libraries, and any other “real-world” domain, demands expressiveness that is substantially more powerful than that of relational databases. Services have many descriptive dimensions (attributes), may be partially described at any level of granularity (with any combination of dimensions), and may be viewed from many perspectives (accessed by different sequences of attribute values). For example, an agent might seek a service to recommend a collection of articles on volcanoes, for high-school audiences, to be contracted for via auction. Alternatively, the agent might seek an auction that sells a service to recommend collections. (In Figure 1, follow links either from “recommend-dlcollection” at the top, or “auction” at the bottom of the figure). In an ontology, both perspectives can be represented simultaneously, and retrieval supports queries on any level of granularity. Declarative formalisms with less expressiveness than ontologies, such as relational databases, force a commitment to a particular ordering of dimensions and a particular perspective.

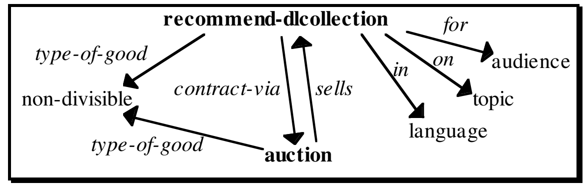

Figure 1: Multi-dimensional, multi-perspective complexity

Because ontologies are costly to develop, many researchers are working on ways to define ontologies so that they may be reused in different problem solving contexts [Gruber 93]. To promote reuse, the UMDL ontology is divided into nested modules (each of which is itself called an ontology) [Weinstein and Alloway 97]. The most general includes library content and services that we consider to be part of a "generic" digital library. To avoid reinventing common concepts such as “duration” or “periodical”, we embed the UMDL ontologies in the Frame and other ontologies in the library maintained by the Knowledge Systems Lab at Stanford [Farquhar 96]. The second module adds concepts specific to the UMDL implementation, such as auctions. The third module describes agent services. We call this last ontology "dynamic" because agents define new service concepts at runtime. In contrast, "static" ontologies are either fixed, or are changed slowly over time by committees of persons.

To facilitate development of ontology definitions, current versions are maintained in stylized natural language [Uschold and King 95]. Currently, we manually translate from English to Loom. Eventually we hope to automate, as fully as possible, translations from natural language to KIF, and from KIF to Loom and other representation systems [Gruber 93].

2.2 Description logic

We use description logic to maintain the ontology of agent services. Description-logic systems use subsumption to automatically place new concepts in taxonomies, permitting classification at runtime. The concept taxonomy can be used in sophisticated ways to reason about services and their relationships to other services.

To classify a service, an agent submits a service description to the SCA. The service description can use terms from any of the three UMDL ontologies. The agent also includes a preferred label in the classification request. There are three cases of classification, as follows:

- If the service description is classified as a new concept, the preferred label is returned as the recommended service label.
- If the service description is equivalent to a service that has already been classified, then the previously existing label will be returned.
- If the concept is new but the label has already been used for another service description, the SCA automatically generates a new label.

If an agent provides multiple services, it classifies each one separately. It is also possible for several different kinds of agents to provide the same service—the SCA ensures that they all use the same label.

The example in Figure 2 is a description for a service to recommend a collection on the topic “science”, for “middle school” audiences. The first line indicates inheritance from the “recommend-dlcollection” concept, and the second and third lines further restrict the values of its slots. An initial colon (:) identifies a Loom keyword. Other symbols are from the ontologies. Service concepts are always treated as "defined" (not primitive) to support recognition based on their descriptions.

```
(:and recommend-dlcollection
      (:filled-by recommend-dlcollection.has.audience
            middle-school)
      (:filled-by recommend-dlcollection.has.topic science)))
```

Figure 2: Example service description for classificaion

In KL-ONE description-logic systems, there is a deeply rooted bifurcation between concepts, which denote sets of objects, and instances, which denote individual objects. For example, classification operates on concepts, but retrieval is over instances. This causes problems for the SCA, because Loom refuses to classify concept descriptions using the subsumption relations of role fillers that are instances. Thus, a service description exactly like that in Figure 2, except with “biology” substituted for “science”, would not be subsumed by the Figure 2 description, because the :filled-by keyword indicates an instance role filler rather than a concept value restriction. See Weinstein and Birmingham [97a] for various ways to work around this problem, and Woods [91] for a description logic with a graceful integration of concepts and instances.

Agents can use a variety of strategies to learn about classified services. Often a single query suffices, but a series of queries can also be used to systematically explore the dynamic ontology. Loom's query expression language has full first-order expressiveness. Variables may be chained to traverse ontology relationships, either to restrict matches to services with specified role fillers, or to reveal fillers for services that are otherwise selected. If the query is successful, the SCA returns a list of sets of bindings, where each set includes a value for each variable in the query expression. The example in Figure 3 is a service description in a query that asks for a service to recommend a collection, where the service is suitable for an audience that is a kind of school. Symbols starting with a question mark (?) are variables.

```
(:and (recommend-dlcollection ?service)
      (recommend-dlcollection.has.audience ?service ?audience)
      (school ?audience))
```

Figure 3: Example service description for a query

Loom's query language can be extended with custom predicate functions that accept or reject every combination of bindings considered by the query. The SCA also provides several "wrapper" functions that are invoked once for the entire query. Whereas predicate functions execute within Loom queries, wrapper functions typically include a Loom query. The example in Figure 4 shows an example of the most-specific-subsuming wrapper function, which ranks all subsuming services in order by the weighted proximity of their role fillers to the ideal. This wrapper is used to search upwards in the concept taxonomy through increasingly general services for the first that is available. The first phrase is a Loom query that identifies candidate services. The next line characterizes the ideal solution. The :priorities list weights the importance of each query dimension. The :attenuation-factors quantify the judgment of proximity, compensating for the density of development of the static ontology in which the role fillers are defined.

```
(:wrapper #'most-specific-subsumer
   '(:and   (recommend-dlcollection ?service)
            (recommend-dlcollection.has.audience
               ?service ?audience)
            (school ?audience)
             (recommend-dlcollection.has.topic ?service ?topic)
            (topic ?science))
   '((middle-school audience) (biology ?topic))
   :priorities (2 1)
   :attenuation-factors (0.6 0.8))
```

Figure 4: Wrapper for *most specific subsumer*

3 Service classification to support markets

This section discusses the role of the SCA in the UMDL’s SMS (Service Markets Society). The SMS is a computational economy. Agents buy and sell services within markets. Markets are implemented with auctions, which provide a bidding protocol whereby buyers and sellers negotiate over price. The SCA supports markets in the SMS in several interesting ways.

The market-oriented programming paradigm benefits from extensive work in economics. Under certain conditions, agents making distributed economic decisions, each to maximize their own utility, will converge on behavior that achieves pareto-optimal resource allocation for the society as a whole [Wellman 93]. As a selling agent’s computational load increases, it costs more to produce marginal services. So the agent asks for a higher price, receives less business, and effectively sheds load to other producers.[1]

The SMS is a simulation, and does not deliver digital-library services to end users. We do not currently identify the quantity of a service. For example, a purchase might be for an hour's worth of query planning, or a bundle of 100 queries.

The SMS includes six kinds of agents. Three of these use the SCA. Several dozen instances of these agents may be active simultaneously. At any time, new instances can be generated, or active ones closed. All SMS agents use the

[1] Another digital library organized as a computational economy has been developed at Zuno, a subsidiary of Mitsubishi (see http://www.dlib.com/). In this system, Profile Selector agents use Q-learning to select Search agents that have performed well at receiving documents that match the current query profile [Ferguson and Karakoulas 96].

same SCA, and thus subscribe to the same ontologies: the SMS is not terminologically heterogenous.

Services are sold by Query Planning Agents (QPAs). These agents are instantiations of the Task-Planning Agent, which is a generalized, goal pursuing procedural reasoner [Vidal and Durfee 95]. Each QPA is configured upon creation to provide query planning service specialized for one of 49 (hypothetical) possible services. This space of services is the cross product of seven values for each of two attributes. The possible values are arranged in small subsumption hierarchies. The attribute “audience”, for example, can be “any-audience”, “school”, “professional”, “middle-school”, “high-school”, or “government” or “business” as kinds of professional.

Services are purchased by User Interface Agent (UIAs). These agents seek to buy one of the same 49 services potentially provided by QPAs. Since this service may not be provided by any active QPA, UIAs identify the best available service using either of two search strategies: increasing generality (most-specific-subsuming), or increasing specificity (most-general-subsumed). The search criteria and strategy can be changed at any time. UIAs seek to buy services periodically, at a rate subject to interactive modification.

Typically, one Auction agent is active for each service provided by a QPA. At each clearing of the auction, the QPA offering the lowest price is selected to sell the service at that price. Since all surplus goes to the buyer, UIAs bid the highest price they are willing to pay. There is no iteration in the bidding; sellers and buyers are not aware of current prices until they are contacted to conduct a transaction.

Auction Manager Agents (AMAs) define markets, and maintain an appropriate population of auctions to service each market. In the SMS, a single AMA spawns an auction for each service provided by a QPA. A QPA asks the AMA for an auction to sell its services; if no auction is active selling that service, the AMA spawns a new auction. The AMA uses the SCA to classify the auction service description, and then checks the registry to see if an auction providing that service is active.

The Registry Agent uses a relational database to hold the network addresses of each active agent and its metadata, such as the service label provided by the SCA.

Figure 5 illustrates the SCA’s interactions with the other agents. Agents are in ovals, ontologies in rectangles, and the arrows connecting agents represent messages, paraphrased in natural language. To communicate with the SCA, agents use terminology from the nested ontologies. The thin line around the SCA is jagged to show that the set of available terms is dynamic; both QPAs and the AMA add concepts to the agent services ontology. Messages use font styles that correspond to the ontology labels (plain, italics, and upper-case), to highlight the source of their terminology.

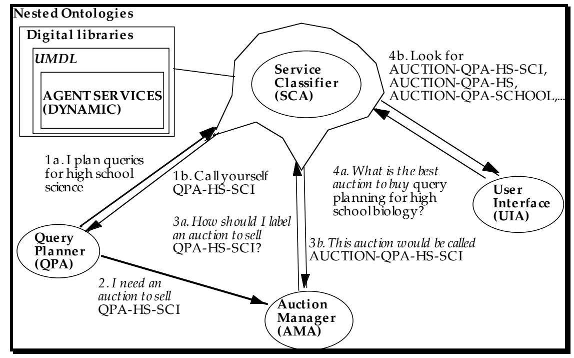

Figure 5: Agent interaction with the SCA in the SMS

Figure 5 illustrates the ability of the SCA to construct new terminology from existing terms at runtime. Concepts in the SCA’s dynamic ontology hide knowledge, just as words chunk meaning in natural language. Auctions sell some service, but they don't need to know anything about that service. The AMA asks the SCA for a service label for an auction, using the QPA’s service label, but it does not know anything about that service. The SCA classifies the auction service using characteristics inferred from the QPA’s service label. Thus, the SCA can respond to the UIA’s request for an auction, which is phrased in terms from the static ontologies, rather than the label which the AMA used to define the auction service. This appropriate hiding of knowledge reduces overall system complexity, and increases reusability and maintainability.

In future organic societies, auction managers may make extensive use of the SCA’s knowledge. First, ontologies are the natural place to represent the type of auction that is appropriate for marketing a particular service. Second, auction managers may need to define markets for sets of services, rather than services with exactly the same classification. Efficient resource allocation depends on maintaining suitable numbers and ratios of seller and buyer agents. These quantities can be controlled by adjusting the generality or specificity of the service to be auctioned.

The most dramatic contribution of the SCA to the SMS derives from its declarative description of services, and its ability to rank available services given a target and search strategy. When a new QPA provides a service that better meets a UIA’s needs, the UIA switches to buying services from the new QPA, without requiring any intervention to modify the UIAs behavior. This occurs because the UIA periodically repeats its search for the best available service. Furthermore, if the new service subsequently becomes unavailable, the UIA reverts to its previous supplier.

4 Discussion

We call the SMS a “proto-organic” society of agents because it is designed to invite third party development of new agents. The SCA permits new agents to participate immediately on entry. Auctions quantify the value of their contribution, facilitating remuneration. Due to the lack of entry barriers, new development can be decentralized. To enable the growth of organic societies of agents, however, will require further development of SCAs (or other agents with equivalent functionality). In this section we briefly discuss three issues: terminological heterogeneity, knowledge sharing, and scaling up.

To support terminological heterogeneity, an SCA needs to be able to answer requests for services that use terminology that is not in its ontologies. We formulate the problem as illustrated in Figure 6. The request is channeled through another SCA, so we can assume knowledge of the ontologies involved, and the availability of inference capabilities on both sides of the transaction. We assume that the SCAs share ontologies with "deep" conceptualization of concepts and relations that is not semantically restrictive in practice. We also assume a normalizable syntax for requests, by expecting all concepts for services to inherit from a concept for a generic service.

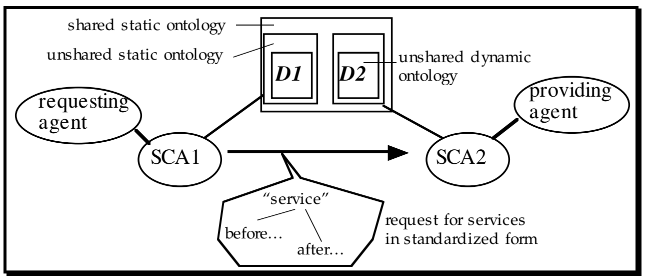

Figure 6: An outsider request for services

Our plan for building and evaluating matches between requests for services and services known to the SCA is described in Weinstein and Birmingham [97b]. An SCA will identify lists of similarities and differences between a request and candidate services. Agents sensitive to problem context will use these lists to reason about candidates' suitability. Feedback from agents using recommended services will be used to improve subsequent translations.

To enable ontology reuse and "knowledge-sharing", the SCA should accept descriptions in a knowledge-interchange language, such as KIF, not a syntax required for a particular representation system, such as Loom. This transition will present some obstacles. It will be necessary to execute a translation tool (such as Ontolingua) at runtime. It will also be necessary to either remove, or work around, the gap in expressiveness between the interchange language and the representation language.

Finally, in large societies of agents it is impossible for any agent, even an SCA, to know about all of the agents. To support society-wide searches for the best available service, ontologies of ontologies will prove useful, with meta-service classifier agents to help select service-classification service. Note that the structure and behavior of meta-service classifiers will be no different than that of regular service classifiers, except for the content of their ontologies.

5 Summary

We have shown that runtime classification of services can support accurate selection of agent services in societies of agents that are dynamic, and evolving. Ontologies are sufficiently expressive to permit representation of complex services, with multiple dimensions, at any level of granularity, and from multiple perspectives. Service descriptions do not reference particular agents, but refer to classes of services. A subsumption taxonomy organizes the service descriptions, and can be used to rank available services by how well they fit given search criteria. Therefore, an agent can define a new service and immediately have customers, if the new service description satisfies the search criteria of agents requesting services better than existing service descriptions. This capability is demonstrated in the UMDL SMS system. Support for service selection despite terminologically heterogenous service descriptions, however, is a goal for future research.

We have also demonstrated that service classification can be used to define markets, at runtime, in societies of agents implemented as a computational economy. In these societies markets may be defined for each classified service. If markets are defined for sets of services, knowledge maintained by service classifiers will be used even more intensively.

Together, runtime classification of services and dynamic resource allocation using markets provide potent incentives for attracting the development of new agents by third parties. New agents can immediately participate in such a society, without requiring the modification of existing agents. Also, the value of services provided by the new agents is automatically calculated, based on supply and demand, and the participation of these agents in the completion of system tasks. Thus, ways to remunerate third party developers can be devised that avoid the need for predicting the contributions of new agents. We therefore call the UMDL SMS a "proto-organic" society of agents: because its infrastructure is designed to catalyze the development of truly large, dynamic, evolving, and terminologically heterogenous societies of agents.

Acknowledgments

Tracy Mullen, Anisoara Nica, Sunju Park, and Jose Vidal each contributed substantially to the SMS. Dan Kiskis provided vital technical support, and Edmund Durfee was our steady guide and critic.

This work was supported by the NSF/ARPA/NASA Digital Library Initiative under grant CERA IRI-9411287.

References

[Birmingham et. al. 94] Birmingham, W.P., K.M. Drabenstott, C.O. Frost, A.J. Warner, K. Willis. 1994. The University of Michigan Digital Library: This is Not Your Father's Library. Digital Library '94 Proceedings, June 1994, 53-60.

[Durfee et. al. 96] Durfee, Edmund H., Daniel L. Kiskis, William P. Birmingham. 1996. The Agent Architecture of the University of Michigan Digital Library. To appear in a Special Issue of the *IEE British Computer Society Journal of Software Engineering on Developing Agent Based Systems*.

[Farquhar et. al. 96] Farquhar, A., R. Fikes, J. Rice. 1996. The Ontolingua Server: a Tool for Collaborative Ontology Construction. KAW96. Also available as KSL-TR-96-26.

[Ferguson and Karakoulas 96] Ferguson, Innes A., Grigoris J. Karakoulas. 1996. Multiagent learning and adaptation in an information filtering market. Proceedings of the AAAI Spring Symposium on Adaptation, Coevolution and Learning in Multiagent Systems. ftp://ai.iit.nrc.ca/pub/ksl-papers/NRC-39186.ps.Z.

[Fikes et. al. 95] Fikes, Richard, Robert Engelmore, Adam Farquhar, Wanda Pratt. 1995. Network-Based Information Brokers. Proceedings of the Spring Symposium on Information Gathering from Distributed, Heterogenous Environments.
http://www.isi.edu/sims/knoblock/sss95/fikes.ps.

[Genesereth and Fikes 92] Genesereth, Michael R., Richard E. Fikes. 1992. Knowledge Interchange Format Version 3.0 Reference Manual, Report Logic-92-1, Computer Science Dept., Stanford University.

[Gruber 93] Gruber, Thomas R. 1993. A Translation Approach to Portable Ontology Specifications. http://www-ksl.stanford.edu/yearindex.html.

[Guha 92] Guha, Ramanathan V. 1992. *Contexts: A formalization and some applications*. Ph.D. dissertation, Stanford University.

[Kuokka and Harada 95] Kuokka, Daniel, Larry Harada. 1995. Supporting Information Retrieval via Matchmaking. Proceedings of the Spring Symposium on Information Gathering from Distributed, Heterogenous Environments. http://www.isi.edu/sims/knoblock/sss95/kuokka.ps.

[MacGregor 91] MacGregor, Robert M. 1991. The Evolving Technology of Classification-based Knowledge Representation Systems. pp. 385-400 in Sowa, John. F. (ed.), *Principles of Semantic Networks*, Morgan Kaufmann Publishers, San Mateo, California.

[Olsen et. al. 95] Olsen, Gregory R., Mark R. Cutkosky, Jay M. Tenenbaum, Thomas R. Gruber. 1995. Collaborative Engineering Based on Knowledge Sharing Agreements. *Concurrent Engineering* V3 N2 p. 145 - 159.

[Uschold and King 95] Uschold, Mike, Martin King. 1995. Towards a Methodology for Building Ontologies. Artificial Intelligence Applications Institute technical report AIAI-TR-183, July 1995.

[Vidal and Durfee 95] Vidal, Jose M., Edmund H. Durfee. 1995. Task Planning Agents in the UMDL. Proceedings of the 1995 Intelligent Information Agents Workshop. http://ai.eecs.umich.edu/people/jmvidal/papers/tpa/ tpa.html.

[Weinstein and Alloway 97] Weinstein, Peter, Gene Alloway. 1997. Seed Ontologies: growing digital libraries as distributred, intelligent systems. Proceedings of the Second ACM International Conference on Digital Libraries.

[Weinstein and Birmingham 97a] Weinstein, Peter, William P. Birmingham. 1997. Runtime Classification of Agent Services. Proceedings of the AAAI-97 Spring Symposium on Ontological Engineering.

[Weinstein and Birmingham 97b] Weinstein, Peter, William P. Birmingham. 1997. Matching Requests for Agent Services with Differentiated Vocabulary. Student abstract, Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI-97).

[Wellman 93] Wellman, Michael. 1993. A Market-Oriented Programming Environment and its Application to Distributed Multicommodity Flow Problems. *Journal of Artificial Intelligence Research* 1 (1993) pp. 1-23.

[Woods 91] Woods, W. A. 1991. Understanding Subsumption and Taxonomy: A Framework for Progress. pp. 45-94 in Sowa, John. F. (ed.), *Principles of Semantic Networks*, Morgan Kaufmann Publishers, San Mateo, California.