Representing Part-Whole Relations

Leon Wong

March 19, 1997

Abstract

Part-whole relations appear throughout human knowledge. Their ubiquity suggests that they play an important role in intelligence. We conjecture that the cognitive value of partonomic knowledge lies in its ability to support translation between representations at different levels of granularity. This permits more effective problem solving by allowing reasoners to integrate the unique benefits that different representations can offer.

We describe a representation for part-whole relations that supports this perspective. Our representation captures two types of partonomic knowledge: decompositions and knowledge translations. Decompositions identify the parts that compose objects. Knowledge translations relate knowledge about objects to knowledge about their parts. Our approach organizes partonomic knowledge around a taxonomy of decomposition functions. This allows us to use inheritance to share partonomic knowledge among similar decompositions.

1 Introduction

Artificial Intelligence researchers have demonstrated how computers can achieve human-level performance on a wide variety of specialized reasoning tasks. Scaling these successes to achieve general intelligence has proven difficult, however.

Efficient problem solving and knowledge storage often require specialized representations. In order to reason about a broad range of complex phenomena, computers must be able to move fluidly between different conceptualizations of task domains, and integrate specialized knowledge and inference methods from each.

Reasoning across multiple representations requires knowledge about the relationships between their constituent concepts. Abstraction hierarchies offer one important approach for organizing related concepts. They capture the connections between terms for describing phenomena at different levels of detail, and provide semantics for translating knowledge among them.

To date, most work on abstraction hierarchies has focused on taxonomic structures: families of concepts related by generalization and specialization relations. Object-oriented and frame-based systems are two popular examples of this paradigm. This paper focuses instead on the complementary subject of partonomic abstraction hierarchies: families of concepts related by aggregation and decomposition, or part-whole, relations.

The next section of this paper discusses the cognitive role of part-whole relations. These insights motivate our strategy for representing partonomic knowledge, which we describe in the section that follows. Finally, this paper concludes with comparisons to related work and suggestions for future research.

2 Partonomic Relations in Reasoning

Part-whole relations appear throughout human knowledge: almost anything can be thought of as an aggregate of some set of parts, or as a part of some larger aggregate. The ubiquity of part-whole relations suggests that they play an important role in cognition. In the previous section, we suggested that partonomic knowledge supports patterns of reasoning that can shift between representations at different levels of granularity in order to leverage their unique computational advantages. In this section, we explore this perspective in greater detail.

We begin by showing how reasoning about objects and their parts corresponds to reasoning with representations at different levels of granularity. We then compare the relative advantages of decomposed and aggregated representations. Finally, we discuss how partonomic knowledge permits reasoning across multiple levels of granularity by allowing knowledge to be translated between decomposed and aggregated representations.

It is common to think of objects and their parts as related entities within the same knowledge representation. Even the term "part-whole relations" reinforces this perspective. This interpretation is quite natural because systems often represent knowledge about objects and their parts within the same knowledge-base.

In this paper, we view an object and its parts as different representations of the same thing. Most knowledge representations consist of two parts: a domain-independent knowledge representation language (e.g., 1st order logic), and an ontology that defines a domain-specific vocabulary for the representation language (e.g., a set of objects, functions, and relations). Mappings between aspects of a domain and concepts in an ontology define the semantics of a representation.

With this in mind, an object and its parts can be viewed as different representations because they reflect two different ways of mapping the same set of domain features to ontology concepts. Furthermore, different levels of granularity may require different ontologies to describe. Different representations may still share a common knowledge representation language and coexist within the same knowledge-base, however.

The reason this multiple representation perspective is useful is that it provides insights into the cognitive role of partonomic knowledge. Choices of representation can have profound effects on reasoning capabilities [DSS93]: they constrain both the kinds the knowledge we can capture and the types of computations we can perform efficiently. The same is also true of different aggregations and decompositions of objects.

Aggregated and decomposed representations offer complementary computational advantages. Aggregated views support more efficient reasoning about collections of objects whose combined behavior has aspects that are simpler to understand than their individual behaviors. For example, consider engineered artifacts such as automobiles. Automobiles are designed to provide drivers with simple interfaces that hide the complexities of their implementations.

The greater simplicity that aggregated representations offer for certain reasoning tasks may, however, prevent them from supporting distinctions necessary for other reasoning tasks. Decomposed representations offer finer-grained descriptions that allow them to express more comprehensive and detailed knowledge than aggregated views. Returning to the automobile example, diagnosing and fixing an inoperative car requires knowledge that must be associated with its constituent parts.

Decomposed views can also support more compact representations of knowledge. Consider domain theories that can describe how arbitrary combinations of objects behave. Such knowledge allows effective reasoning about a combinatorial number of different aggregates, including new aggregates outside the reasoner's experience. For example, consider electrical circuits. Knowledge about the behavior of a small set of components allows one to reason about an infinite variety of electronic devices.

Having seen some of the complementary advantages that decomposed and aggregated representations offer, it should be clear that the appropriate choice of representation can be critical in problem solving. Reasoning about complex phenomena can involve solving multiple subproblems, however, each of which may demand representations at different levels of granularity. Reasoning effectively across multiple representations requires the ability to translate knowledge between them: this is necessary for injecting subproblems into the representations to be used for reasoning, and extracting the results.

Translating between aggregated and decomposed representations requires two types of partonomic knowledge: decomposition axioms and knowledge translation axioms. Decomposition axioms correspond to traditional partonomic knowledge: they identify the parts that compose objects. This alone is insufficient for translating knowledge between decomposed and aggregated representations, however. We also need to know how knowledge about objects can be related to knowledge about their parts.

For example, knowing the set of parts that compose an automobile cannot, by itself, tell us how any knowledge we have about those parts (e.g., their masses) relates to knowledge we can express about the automobile as a whole (e.g., its mass). We need additional information, such as the conservation of mass, to connect knowledge at the decomposed level to knowledge about the aggregated level. Conservation of mass is an example of a knowledge translation axiom. In the next section we outline a representation scheme that captures both types of partonomic knowledge.

3 Representation

3.1 Overview

This section describes a representation framework for partonomic knowledge that is based on the Ontolingua knowledge representation language [Gru92]: an extension of the Knowledge Interchange Format (KIF) [GF92]. KIF is a full-featured 2nd order logic, which Ontolingua extends with frame-based capabilities. While Ontolingua's richness simplifies our representation framework, our basic approach can be generalized to other representation languages.

There are five components in our representation scheme:

- **Decompositions:** decompositions relate objects to their parts.
- **Part-Of Relation:** the `part-of` relation indicates whether an object is part of a particular aggregate. It can be useful for defining partial knowledge about decomposition relations.
- **Named Roles:** named roles allow one to refer to the parts of an object by referencing the roles they play in the aggregate. This is especially useful for referring to parts that can have no independent existence from the whole (e.g., the edge of a block).
- **Knowledge Translations:** knowledge translation axioms express the relationship between knowledge about objects and knowledge about their constituent parts. They allow translation between representations at different levels of partonomic abstraction.
- **Inter-part Relations:** inter-part relations describe the configuration of the components of an aggregate. The configuration of an object's parts often affects its aggregated behavior, and may determine whether particular aggregation abstractions are possible.

The next 5 sections describe each of these elements in greater detail.

3.2 Decompositions

We represent decompositions with unary functions that take an object and return the set of its parts. In general, there may be many useful ways to decompose objects. For example, a pizza can be considered both in terms of its slices and its constituent ingredients. We therefore allow knowledge engineers to define multiple different decomposition functions. We group decomposition functions into into classes that are arranged into a taxonomic hierarchy. At the top of this hierarchy is the `decomposer` class:

```
(define-class decomposer (?func)
  :def (and (unary-function ?func)
            (=> (member ?tuple ?func)
                (set (last ?tuple)))))
```

This class definition only constrains the number of arguments a decomposition function can have and the type of value it can return. All other information about decompositions is provided by additional axioms that describe their behavior on specific types of objects. For example, we can describe the composition of the Cleaver family in the following fashion:

```
(define-instance organization-members (decomposer))

(= (organization-members Cleavers)
   (setof Ward June Wally Beaver))
```

Organizing decomposition functions into a taxonomic classification hierarchy offers a number of benefits. One advantage is that it provides users with a intuitive and familiar structure for organizing large numbers of decomposition functions. More importantly, however, taxonomies allow us to use inheritance to share knowledge among similar decompositions. For example, making the `pizza-slices` and decomposer a member of the `physical-partitioner` class allows it to inherit knowledge about mass conservation:

```
(define-class physical-partitioner (?decomp)
  :def (and (decomposer ?decomp)
            ;; mass conservation axiom
            (=> (physical-object ?x)
                (= (mass ?x)
                   ;; sum of part masses
                   (reduce-set + (value ?decomp ?x) mass 0)))))

(define-class food-decomposer (?decomp)
  :def (physical-partitioner ?decomp))

(define-object pizza-slices (food-decomposer))
```

This lets us conclude that the mass of a pizza is equal to the sum of the masses of its slices. The `reduce-set` function that appears in the definition of the `physical-decomposer` class is similar to the lisp `reduce` function, and has the following KIF definition:

```
(define-function reduce-set (?function ?set ?key ?initial-value)
  :result-variable ?reduction
  :def
  (and (function ?function)
       ;; commutative and associate ensure result indep of order
       ;; of processing elements - this is required to make
       ;; reduce-set a function
       (commutative ?function)
       (associative ?function)
       (set ?set)
       (function ?key)
       (=> (= ?set (setof @items))
           (= ?reduction
              (reduce-list ?function
                           (listof @items)
                           ?key
                           ?initial-value))))

(define-function reduce-list (?function ?list ?key ?initial-value)
  :result-variable ?reduction
  :def
  (and (function ?function)
       (list ?list)
       (function ?key)
       (if (null ?list)
           (= ?reduction ?initial-value)
           (= ?reduction
              (value ?function
                     (key (first ?list))
                     (reduce ?function
     (rest ?list)
     ?key
     ?initial-value))))))
```

As the amount of knowledge about decompositions grows, the benefits of taxonomic inheritance increase because new decompositions may be able to inherit significant portions of their semantics from previously defined classes of decomposers. Sharing knowledge in this fashion results in more compact knowledge-bases, and simplifies knowledge engineering and maintenance.

In addition to allowing us to organize decomposition functions taxonomically, Ontolingua's support for 2nd order logic offers two other representational benefits. First, it lets axioms quantify over decomposition functions. This is useful in a variety of situations. For instance, we can use universal quantification to express knowledge about collections of decompositions. Existential quantification can be useful for reasoning in contexts where the appropriate decomposition is unknown. For example, a home buyer might want to know whether a house contains any asbestos. We could express this as a query about whether there is some physical decomposition of the house that contains a part that made of asbestos:

```
(exists (?func ?part)
        (and (physical-decomposer ?func)
             (member ?part (value ?func house))
             (asbestos ?part)))
```

Another advantage of using 2nd order logic for describing decompositions is that it allows us to ascribe properties to decomposers. We can then easily associate knowledge with all decomposers sharing particular combinations of properties. For example, if a decomposition is disjoint, we can conclude that no two parts of an object under that decomposition share common subparts:

```
(=> (and (decomposer ?d1)
         (disjoint ?d1)
         (part-of ?parta ?whole ?d1)
         (part-of ?partb ?whole ?d1))
    (not (exists (?subpart ?d2)
                 (and (decomposer ?d2)
      (part-of ?subpart ?parta ?d2)
      (part-of ?subpart ?partb ?d2))))))
```

The disjointness property can be useful for identifying the decompositions that obey certain conservation laws. For example, the members of the `physical-partitioner` class must be disjoint decomposers in order to support mass conservation.

While we have relied heavily on the expressiveness of 2nd order logic throughout this section, note that it is possible to use a representation strategy similar to the one we have outlined in 1st order logics by reifying decomposition functions.

3.3 Part-Of Relation

The `part-of` relation indicates whether an object is among the components of a particular aggregate, with respect to a specific decomposition function. It has the following definition:

```
(define-relation part-of (?part ?whole ?decomposer)
  :def (and (decomposer ?decomposer)
            (member ?part (value ?decomposer ?whole))))
```

While the definition is almost trivial, the concept is common and intuitive enough to be worth representing. Often we can only identify some subset of the parts that compose an object, but can still perform useful reasoning with this information. The `part-of` relation provides a simple way to express such knowledge.

In the following example we see that is not always necessary to know the full decomposition of an object in order to deduce useful properties about that object:

```
(=> (and (dish ?d)
         (part-of ham ?d food-ingredients))
    (not (vegetarian ?d)))
```

3.4 Named Roles

It is often useful to be able to refer to the parts of an object by the roles they play in the aggregate. For example, we may wish to talk about the President of the United States without knowing who he is. Named roles are especially useful for referring to parts whose existence is dependent on the existence whole (e.g., the edge of a block): in such cases, we are unlikely to have named constants defined for those objects.

We can refer to the objects that fill particular part roles in an aggregate by defining functions on aggregates that return those objects. This lets us generalize our decomposition axioms by allowing us to use role selector functions to identify objects' parts. For example, we might describe the decomposition of typical nuclear families in the following fashion:

```
(= (organization-members ?family)
   (union (setof (father ?family)
                 (mother ?family))
          (children ?family)))
```

Notice that there is no need to distinguish role selector functions in any special fashion. Their relevant semantics can be derived from the way they appear decomposition and knowledge translation axioms.

Most role selector functions return a single part and have the aggregated object as their only argument. Also, in many classes of structured composites, every part has a well-defined role. This regularity allows us to simplify the decomposition axioms for such cases by using the `decomposition-roles` relation:

```
(define-relation decomposition-roles (?class ?decomp ?roles)
  :def
  (and (class ?class)
       (decomposer ?decomp)
       (set ?roles)
       (=> (instance-of ?object ?class)
```

```
(= (value ?decomp ?object)
   (reduce union
                 ?roles
                 (lambda (?role)
                    (setof (value ?role ?object)))
                 (setof ))))))
```

This relation holds if instances of `?class` decompose into the set of objects that the specified role selectors in `?roles` return, under the decomposition function `?decomp`. For example, we can describe the composition of most single-battery flashlights in the following fashion:

```
(decomposition-roles single-battery-flashlight
                     device-parts
                     ;; set of flashlight part selector functions
                     (setof flashlight-battery
                            flashlight-bulb
                            flashlight-lens
                            flashlight-reflector
                            flashlight-switch
                            flashlight-case))
```

3.5 Knowledge Translations

Knowledge translation axioms play a role in partonomic hierarchies that is analogous to the role of inheritance semantics in taxonomic hierarchies: they provide a means for relating knowledge expressed at different levels of abstraction. This allows the results of reasoning under different conceptualizations to be combined to solve complex problems. We saw one example of a knowledge translation in our axiom for mass conservation: it described how the mass of an aggregate is related to the mass of its parts under physical partitions.

There are no special mechanisms that must be employed to define knowledge translation axioms: any axiom that relates knowledge about aggregates to knowledge about their parts can be used for reasoning across different levels of granularity. Furthermore, it seems that translation axioms span a sufficiently rich spectrum of knowledge that they require the full expressiveness of a general knowledge representation language.

To see this, observe that knowledge translations often require in-depth domain knowledge: the behavior of aggregates of physical objects can require arbitrary knowledge about the physical sciences, and the behavior of aggregates of people can require arbitrary knowledge about the social sciences. Representations for knowledge translation axioms can therefore demand as much expressive power as the representations used for domain theories at both the aggregate and decomposed levels of granularity.

Despite the lack of constraints on the form of knowledge translations, many translation axioms share common properties. The following list provides some examples of the dimensions along which knowledge translations can vary:

- **Emergent vs. Aggregated Properties:**

  In some cases, an aggregate-level property is also meaningful at the decomposed level and there exists a well-defined relationship between the two. For example, mass is a property of physical objects that is meaningful from astronomic to subatomic scales. Conservation laws dictate that masses sum in all exhaustive disjoint partitions of physical objects throughout this range. In other cases, a property that is meaningful at the aggregated level may arise from different properties at the decomposed level. For example, the temperature of a gas arises from the kinetic energies of its constituent molecules. Temperature is not a meaningful concept at the level of individual particles. The relationship between molecular kinetic energy and gas temperature is still well-defined, however.

- **Configuration-Dependence:**

  An aggregate's behavior may depend on the relationships that exist between its parts. For example, the amount of volume that a jar and its lid displaces depends critically on whether the lid is screwed onto the jar. In other cases, such as mass conservation, knowledge translations can be stated without regard to inter-part relations. These translation axioms tend to be much easier to represent and reason about.

- **Combination Semantics:**

  The properties of objects may combine in different ways to produce the properties of their aggregates. Many properties, for instance mass and electrical charge, are additive. A variety of other combination semantics are common, however. For example, a food item is vegetarian if all its ingredients are vegetarian. The combination function in this case is logical `and`. A chain is only as strong as its weakest link. Chain strength aggregates with the `minimum` function.

Note that we do not necessarily have to have explicit knowledge translation axioms for every property that we wish to relate across different levels of granularity. In many situations, once we have knowledge translations for some primitive set of attributes, we can deduce the relationships between derived properties across different levels of granularity. For example, consider the task of relating per-capita incomes across different levels of population aggregation. It is certainly possible to define a knowledge translation axiom that computes the appropriate weighted average. In cases where we have knowledge translation axioms for population counts and total income, however, this is unnecessary. We can simply define per capita income in terms of total income and population, and then rely on the translations for those properties to carry knowledge across aggregation and decomposition abstractions.

3.6 Inter-part Relations

The relationships that exist between the parts of an object are often critical for determining the behavior of the aggregate. For example, two electrical circuits may use the same components, but arrange them in different topologies to produce radically different behaviors.

As we stated earlier, knowledge translations can involve in-depth domain-specific knowledge. Arbitrary relations may therefore be necessary for expressing translation axioms. For this reason, our representation places no constraints on the inter-part relations that can be stated. It is also not necessary to specially distinguish inter-part relations in any way: the knowledge translation axioms that reference these relations provide all the necessary semantics.

This need for expressive freedom was the motivation behind having decomposition functions return sets of objects' parts: no structured collection can adequately support the full range of inter-part relations one might require. By separating knowledge about part composition from inter-part relations, we allow the latter to remain unconstrained by the former.

4 Related Work

In [RN95], Russell and Norvig describe an approach frequently taken in modeling part-whole relations. Their approach differs from ours in two important respects. The first difference from our framework is that they assume only a single canonical decomposition exists for each object. Because of this, they do not make explicit distinctions between different types of decompositions.

This approach can be problematic when applied to large-scale knowledge-bases. We have seen that objects often have several useful decompositions. Also, when representations fail to distinguish between different types of decompositions it forces the knowledge engineer to associate knowledge translation axioms with the specific objects or classes of objects being composed or decomposed. This can lead to repetition of translation knowledge, which may undermine their maintenance and reuse.

The second significant difference between [RN95] and our approach is that our framework has no explicit treatment of `part-of` transitivity: the notion that if B is a part of A and C is a part of B, then C is a part of A. Transitivity is an issue that many researchers attempt to address, but its lack of emphasis in our framework is intentional.

There are three reasons why we do not explicitly address transitivity. First, the concept is trivially accounted for in a representation framework where `part-of` relations are stated with respect to specific decomposition functions.

It is always possible to define a new decomposition that combines the effects of successive decompositions. The real question is whether the resulting decomposition is at all valuable. This same question can be asked of any part-whole representation scheme with `part-of` transitivity, however.

This brings us to our second reason for omitting special treatment for transitivity: transitivity by itself seems to have little cognitive value. The main reason transitivity, and part-whole relations in general, appears useful is to support knowledge translation. For example, if we know that a piston is part of an engine, and that the engine is part of a car, transitivity and mass conservation allow us to conclude that the mass of the piston contributes to the mass of the car. In our framework, however, transitivity is unnecessary because we can simply combine translation axioms for mass across both the decomposition of the car and the decomposition of the engine. We never need to consider the decomposition that would result from the composition of the two individual decomposition operations.

Notice that this works even in cases where the decompositions applied at each level are radically different. The main point here is that knowledge translation axioms make very explicit the kinds of reasoning often associated with transitivity, and eliminate the need to make transitivity properties explicit.

Our final reason for avoiding the subject of part-whole transitivity is that it does not seem to hold consistently when people reason about part-of relations. This observation has been made by several researchers [WCH87], and many unsuccessful attempts to resolve the issue have been made.

Despite the problems we have mentioned, approaches of the kind described by Russell and Norvig can be quite useful and are common in practice. Quite often, practical systems deal with sufficiently focused tasks and narrow domains that problems with implicit decompositions and transitivity do not arise.

Researchers have, however, attempted to devise richer theories of part-whole relations by introducing taxonomies of part-whole relations as we have. [WCH87] represents one instance of this approach. In [WCH87], Winston, Chaffin and Herrmann attempt to explain intransitivity phenomena in linguistics by introducing 6 types of part-whole relations: component-integral, member-collection, portion-mass, stuff-object, feature-activity, and place-area. They explain intransitivity by claiming that transitivity only holds among part-of relations of the same type.

For example, if Simpson's arm is a part of Simpson, and Simpson is a member of the Philosophy department, most people are uncomfortable with the conclusion that Simpson's arm is a part of the Philosophy department. [WCH87] suggests that the failure of transitivity is due the mixture of component-integral and member-collection part-of relations.

There are several problems with [WCH87]. One difficulty is that the theory still fails to provide a satisfactory explanation for intransitivity. If we state that Simpson is the chairman of the philosophy department, our previous example seems to exhibit no more transitivity despite our substitution of a second component-integral relation for our original member-collection relation. Other researchers have had similar difficulty adequately characterizing the intransitivity of part-whole relations.

Even if we ignore these problems, the theories in [WCH87] do not seem to provide a sufficient basis for developing representations for partonomic knowledge. They provide little justification for why transitivity may be useful to capture, and give few insights into what other aspects of part-whole relations require representation.

Our conjecture is that the intuitiveness of a part-of relation actually depends on the amount of translation knowledge we have about it. Simpson's arm may be a part of the Philosophy department in some bizarre decomposition that partitions according to organization member body parts, but people find this relation counter-intuitive because it has no cognitive value: we have little knowledge about such decompositions and even less cause to reason about them.

While [WCH87] may exhibit several weaknesses, their taxonomy of decomposition types could still provide a useful set of base classes for our representation scheme. They capture intuitively satisfying distinctions between types of part-whole relations and merit further study.

In [GP96], Gerstl and Pribbenow present an alternative hierarchy of decomposition operations that may also deserve some attention. They claim that their classification of part-whole relations has greater generality than the linguistically-motivated taxonomy in [WCH87]. Their examples focus on the descriptive capabilities of their classification scheme, rather than its potential utility in reasoning, however. In general, it seems that extensive experience with both schemes will be necessary in order to evaluate them properly.

Note that both [GP96] and [WCH87] imply that a small taxonomy of part-whole relations can provide an adequate set of distinctions. The representation framework we have outlined will likely induce a much richer hierarchy of concepts, stemming from our desire to associate highly domain and context-dependent knowledge translations with different decomposition schemes.

In [AFGP96], Artale, Franconi, Guarino, and Pazzi move beyond transitivity and argue for the necessity of a variety of additional features in representations of part-whole knowledge. They also evaluate several representation schemes against these criteria. The features they advocate include: reification of aggregates, differentiation of part-of relations from other attributes of objects, transitivity, the ability to refer to parts by their role names, the ability to express "vertical" relationships between parts and wholes, and the ability to express "horizontal" constraints among the parts of a whole.

Our framework for representing partonomic knowledge satisfies many of these criteria. We reify both parts and wholes, permit arbitrary vertical and horizontal relationships, and allow objects to be referenced by their roles in aggregates. We have already discussed the kind of support we provide for transitivity, and [AFGP96] goes no further than [WCH87] in specifying desiderata for theories of transitivity. Lastly, we differentiate parts from other attributes of wholes to the extent that decomposition axioms explicitly identify them. More explicit distinctions seem undesirable since they would serve little purpose, and might involve unnecessarily restrictive ontological commitments.

The survey portion of [AFGP96] focuses on description logics and object data modeling paradigms. It is hardly surprising that none of the methods surveyed satisfied all the evaluation criteria, given that the representation frameworks were not designed with those criteria in mind. Furthermore, most of the methods surveyed were constructed for specialized task domains that may not have required generic partonomic knowledge representations. A common problem we observed in most of the methods surveyed, however, was lack of expressive power. We have already suggested that arbitrarily complex knowledge may need to be associated with part-whole relations, and that representations require the full expressiveness of rich knowledge representation languages. It is hardly surprising that the object data models and description logics surveyed showed significant limitations in this regard.

5 Future Work

Our current work on partonomic knowledge representation is in its earliest stages, and requires extensive development on a wide variety of fronts.

On the applied side, we need to accumulate experience with representing and reasoning about partonomic knowledge. Test cases from several different domains should provide intuitions about the generality and usefulness of the framework we have established. In addition, they may reveal the need for modifications and enhancements to our representation scheme.

After the basic representation framework becomes stable, the next task will be to accumulate a useful library of partonomic knowledge. In particular, a rich taxonomy of decomposition operations with their associated knowledge translation axioms could serve as a valuable collection of reusable knowledge.

The taxonomic organization of decomposition schemes can facilitate the incremental development of such a library. Initially, users can create decompositions and knowledge translations for their task-specific needs. When significant similarities appear between different decompositions, they can be grouped under common superclasses that encompass their commonality. Incremental development is important because of the vast numbers of decompositions that could potentially be useful.

Significant theoretical work is also necessary for furthering this research. This paper made several unsupported conjectures regarding the cognitive value of partonomic knowledge. The rationale for our design will benefit from a formal characterization of the computational advantages of reasoning with representations that span multiple levels of granularity. Such an analysis may reveal the need for new inference techniques. With the ability to reason across multiple different conceptualizations of a task, search spaces expand and new techniques for controlling inference may become crucial.

With a formal understanding of the computational benefits that partonomic knowledge supports, it would be interesting to develop theories for how useful partonomic concept hierarchies can be learned. Such work could provide useful insights to knowledge engineers building large-scale systems, and might advance our progress toward intelligent agents that possess human-level learning capabilities.

6 Acknowledgments

This work was supported under a Stanford University School of Engineering Fellowship, and ??? some ARPA grant ???

References

[AFGP96] Alessandro Artale, Enrico Franconi, Nicola Guarino, and Luca Pazzi. Part-whole relations in object-centered systems: an overview. *Data and Knowledge Engineering*, 1996.

[DSS93] Randall Davis, Howard Shrobe, and Peter Szolovits. What is a knowledge representation? *AI Magazine*, 14(1):17–33, 1993.

[GF92] Michael R. Genesereth and Richard E. Fikes. Knowledge interchange format version 3.0 reference manual. Technical Report Logic-92-1, Stanford University Logic Group, January 1992.

[GP96] Peter Gerstl and Simone Pribbenow. A conceptual theory of part-whole relations and its applications. *Data and Knowledge Engineering*, 20:305–322, 1996.

[Gru92] Thomas R. Gruber. Ontolingua: A mechanism to support portable ontologies. Technical Report KSL-91-66, Stanford University Knowledge Systems Lab, November 1992.

[RN95] Stuart J. Russell and Peter Norvig. *Artifical Intelligence: A Modern Approach.* Prentice Hall, Englewood Cliffs, NJ 07632, 1995.

[WCH87] Morton E. Winston, Roger Chaffin, and Douglas Herrmann. A taxonomy of part-whole relations. *Cognitive Science*, 11(4):417–444, 1987.