An Ontology Approach to Product Disassembly *

Pim Borst †
† University of Twente
Information Systems Department INF/IS
P.O. Box 217, NL-7500 AE Enschede
The Netherlands
Email: {borst,akkerman}@cs.utwente.nl

Hans Akkermans †‡
‡ Netherlands Energy Research
Foundation ECN
P.O. Box 1, NL-1755 ZG Petten
The Netherlands
Email: akkermans@ecn.nl

Abstract

In recent years, growing ecological concern has prompted for 'design for environment'. One way to achieve this is to design products that are easy to disassemble, because this improves the ability to reuse or recycle parts of a product. This paper presents a computational theory for product modeling and reasoning about product disassembly. This theory, implemented in the PROMOD system, is based on an ontology of different connection types between product components. For the task of reasoning about disassembly, the standard topological relation that expresses that two objects are connected or in contact proves to be inadequate. We therefore introduce, within a topological context, a small number of new ontological primitives concerning the rigidness of connections and the constrained degrees of freedom, which in effect are task-oriented abstractions of geometric and physical-chemical properties of products. On this basis, it is demonstrated that one can automatically generate all feasible product disassembly sequences, and in addition perform an ecological cost-benefit analysis. The latter provides a preference order over disassembly sequences, allowing to compare alternative product designs for recycling and reuse. Finally, we show how the proposed ontology for disassembly is an extension of existing ontologies dealing with physical systems, and is based on the same ontology design principles.

*This work has been carried out as part of the SUSTAIN project, with PRé Product Ecology Consultants and ECN as partners, and partially supported by the SENTER-IT Programme of the Netherlands Ministry of Economic Affairs. Helpful discussions with Mark Goedkoop (PRé) and Jan Braam (ECN) are acknowledged. We also thank Mark Goedkoop for kindly providing us with the coffee machine which we have thoroughly disassembled for the purposes of the present study.

1 Introduction

In recent years, growing ecological concern has prompted for 'design for environment' [Fiksel, 1996]. One way to achieve this is to design products that are easy to disassemble, because this improves the ability to reuse or recycle parts of a product. In analyzing these aspects, one needs to determine all feasible ways to disassemble a product. They can be jointly represented in an *AND/OR graph* [Fazio and Whitney, 1987; Sturges Jr. and Kilani, 1992], with the fully assembled product as the root, the *and-nodes* indicating a disassembly operation splitting the product into subassemblies, and the *or-nodes* representing different applicable operations that give rise to alternative ways to break up a (sub)product. In such an AND/OR graph, each subtree that has and-nodes as its leaves, and contains only one out of the alternative branches at each or-node it encounters, describes a distinct disassembly sequence.

In eco-design, an important goal is to determine the cost of disassembly as well as the environmental benefit of the subassemblies that are separated. The energy required to perform the disassembly operations might be used as a measure for the disassembly cost. All components that were separated in the disassembly process are candidates for reuse and have a positive impact on the benefits of disassembly. The degree to which the materials in a separated part of the product can be recycled is determined by the mix and the amounts of materials in that part. With such a cost-benefit analysis applied to the AND/OR graph, one can determine the disassembly sequence having the best cost-benefit ratio, as well as the impact of design decisions by comparing alternative product designs.

In this paper we will present a general ontology-based approach to two important tasks in product disassembly analysis:

1. automatically generating the AND/OR graph from a topological product model;
2. obtaining from the AND/OR graph the disassembly sequence having the best ecological cost-benefit ratio.

An ontological approach to this problem appears helpful, because it is evident in disassembly analysis that a notion of ‘connectedness’ plays a crucial foundational role [Clarke, 1981]. As we will see, the standard topological relation, that expresses that two objects are connected or in contact, as incorporated in formal topological ontologies such as in [Borst *et al.*, 1997], proves to be not adequate for this purpose. Nevertheless, we show that by extending a standard topological ontology with a small number of new ontological primitives regarding the *type* of connections, we can build product models that provide the knowledge to automatically carry out the mentioned tasks.

In Section 2 we introduce the basic concepts needed to build product models for disassembly, and illustrate them by various examples. This theory has been implemented in a knowledge system called PROMOD. Section 3 describes how it reasons with these product models for the purpose of disassembly. Section 4 surveys related work in computational (mechanical) engineering, and indicates how the proposed ontology for disassembly is an extension of existing ontologies in AI dealing with physical systems.

2 Theory of Product Models for Disassembly

In disassembly analysis, a product is conceived of as an *assembly* consisting of *product components* with mutual *connections*. Product components are the smallest parts of a product that are relevant in a disassembly context. The most important attributes are the material(s) they are made of and the amount of these materials. With these attributes it is possible to determine to what degree the materials of a group of product components can be reused or recycled.

2.1 Component Connections

Connections specify the places of contact between product components. It is allowed that there is more than one connection between two product components. Connections have properties beyond standard topology that are important for disassembly analysis. When a component is situated between other components, the outer components are possibly in the way during disassembly. It can also be the case that the component in the middle can be removed by pulling it in another direction. But in order to do this, the product components must be connected loosely. Two properties of connections are therefore of importance: how *rigid* connections are (in the sense of physical forces) and how constrained (in a spatial or geometric sense) the *direction* of movement of components is.

Unfortunately, this implies that generally we have to deal with many, both geometric and chemical/physical concepts. This we want to avoid, practically because in the design stage of a product detailed models like 3D CAD drawings are often not yet available, and computationally because it involves strong and complex ontological commitments. However, it is possible in disassembly modeling and analysis to define task-oriented abstractions of geometric and physico-chemical connection properties that do the job. These abstractions are then

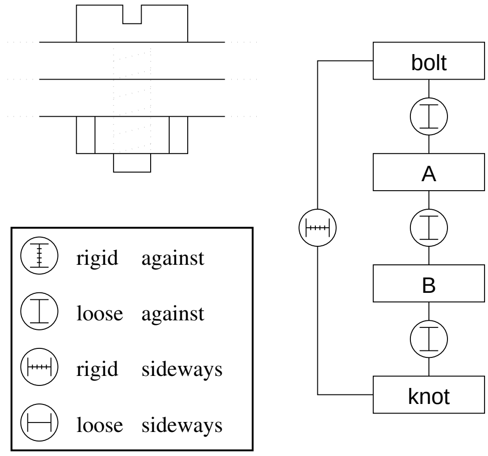

Figure 1: The four connection types and an associated component-connection model of a bolt-and-knot system.

brought into a topological ontology of component connection models by means of different connection types.

Distinguishing four types of connections is already sufficient to be able to perform practically useful disassembly analyses. These types are based on a dichotomy within two important orthogonal dimensions. The first discriminating dimension is the mentioned *rigidness* of a connection. A useful dichotomy here is whether a connection is rigid or loose, and relates to a distinction in engineering design known as force-based versus shape-based connections:

rigid: A physical or chemical *force* keeps the product components together, so this force must be overcome first to break the connection. (Example: components screwed or sticked together.)

loose: Product components are in contact with each other, but in a *purely spatial* sense without an additional binding force, so the connection disappears when the components are moved apart. (Example: a glass standing on a table.)

The second distinguishing dimension is the *direction* in which a connection restricts the movement of the connected components. The simplest possible conceptualization is to introduce the following dichotomy:

against: The connection restricts movement in the direction *perpendicular* to the surface of contact.

sideways: The connection restricts movement in a direction *within the plane* of the surface of contact.

This gives a two-by-two matrix, leading to four types of connections: *rigid-against, rigid-sideways, loose-against, loose-sideways*. Figure 1 depicts graphical representations of these types, and an example how these types are used in a product model for disassembly. When the rigid connection between the bolt and knot is broken (which requires applying a physical force), the loose-against connections in the model can be simple undone by moving the connected components away from each other.

2.2 Force Loops

In the example in Figure 1, the two plates are inbetween the two components they are connected with. In this case, the direction of all connections are the same, but generally they are not. Components can be inbetween several pairs of components in different directions. We must therefore model which connections are in the same direction, without having to specify 3D vectors. We do this by employing the topological concepts of *paths and loops* through connections to abstract and encode the geometric information. These paths can be constructed by linking the connections that are in the same direction. In the model in Figure 1 we then see one loop running through all connections.

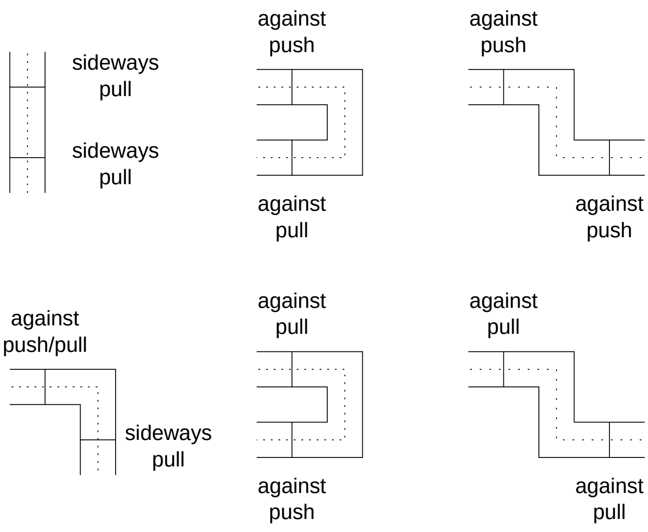

Figure 2: The possible transitions in force loops in a disassembly model.

An easy way to determine the direction of connections and the paths that go through them is to imagine what happens when the components connected by a loose connection are pulled away from each other. For example, in Figure 1, when the connection between A and B is pulled, this implies that plate A is pushed against the bolt, the bolt is pulled away from the knot, the knot is pushed against plate B and finally (although this may sound at first sight as a contradiction) plate B is pushed against plate A. The outcome of this sequence of forces depends on the geometry of the product and, as we will see later, turns out to be exactly the information required for disassembly analysis.

In a rigidly connected product, the geometric structure is such that the components connected by loose-against connections are pushed against each other. This will result in loops through connections, as is the case in the example. Therefore, we will speak of *force loops*. In cases where the product is not rigid, we will say that the loop is broken or not *intact*.

Because the force sequences in a force loop depend on geometry, only certain force transitions are possible when following a path, as shown in Figure 2. When these restrictions are obeyed, the forces on connections in an intact force loop satisfy the following rules:

- Loose-against connections are always pushed.
- Rigid-against connections are pushed or pulled.
- Sideways connections are always pulled.

When two components connected by a loose connection are pushed together, we will say that the connection is *enforced*. This can only be the case when there is a loop going through the connection that is intact. A formal ontological definition of *intact* and *enforced* reads:

$$\begin{aligned}
&\mathit{intact}(l) \leftrightarrow \neg\exists\, c\text{: } \mathit{loops\text{-}through}(l,c) \wedge \\
&\quad (\mathit{broken}(c) \vee (\mathit{loose}(c) \wedge \mathit{force}(l,c)=\mathit{pull})) \\
&\mathit{enforced}(c) \leftrightarrow \mathit{rigid}(c) \vee (\mathit{against}(c) \wedge \\
&\quad \exists\, l\text{: } \mathit{force\text{-}loop}(l) \wedge \mathit{force}(l,c)=\mathit{push} \wedge \mathit{intact}(l))\ .
\end{aligned}$$

In more complex situations, there might be more than one connection in the same direction on one side of a component, like in the model in Figure 3. This can easily be modeled by two force loops going through one connection. A situation where two connections collectively form a virtual connection is modeled in Figure 4. Determination of the loops and forces would result in the three loops that are indicated in the Figure, and the definition of *enforced* presented here must be adapted. These cases require an extension of the modelling technique that cannot be explained in this article due to lack of space.

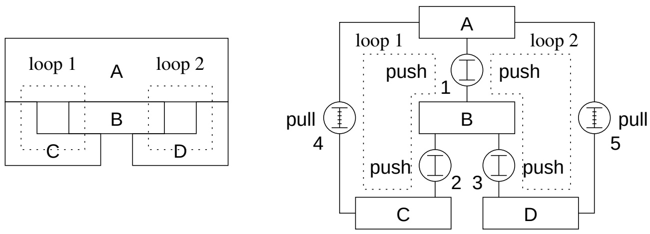

Figure 3: Each loop individually enforces the loose connection numbered 1.

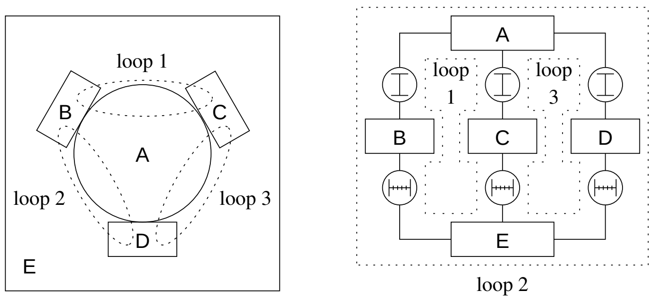

Figure 4: Two loops collectively enforce a loose connection.

2.3 Disassembly Operations

Disassembly operations performed on a product induce changes in its disassembly model. According to the above conceptualization of connection types, two kinds of modification operators are distinguished: *loosening* operations that change a connection from being rigid to loose, and *breaking* operations that delete loose connections. The first require applying a force (energy) to undo the rigidness, while the latter refer to changing the spatial location by simply moving the components apart.

Disassembly operations like cutting, sawing and unscrewing change the types of the connections involved from rigid to loose. In the bolt-knot example, loosening the bolt is modeled by replacing the rigid-sideways connection by a loose-sideways connection. Loosening operations can cause force loops to break. As a result, some loose connections will no longer be enforced and certain parts of the product (called subassemblies) can be easily separated from the rest of the product, by breaking operations. Again, the bolt-knot example in Figure 1 illustrates this. When the connection between the bolt and the knot has been loosened, the bolt (or the knot) can be removed from the product. This actually breaks (deletes) the connections between the bolt and the knot and the bolt and the plate.

2.4 Subassemblies

Loosening of rigid connections may cause that groups of components can be removed from the product. If this is the case, there may be no enforced connections between the group and the rest of the product. Furthermore we demand that the group does not contain smaller groups of components that can be separated. In other words, internal connections in a group have to be enforced. In the ontology for disassembly, groups of components having these properties are called *subassemblies* (formally defined in Sec. 4 as an extension of a general systems ontology).

Each subassembly is a candidate for removal, but a direction has to be found in which the subassembly can be moved away from the rest of the product. The geometric information captured in the force loops can be reused to find the desired direction.

When a subassembly has a loose external connection that is *not* linked to an external connection by a force loop, it means that no component blocks the subassembly in the direction of the connection. Because the other external connections of the subassembly have other directions and are not enforced, it implies that the subassembly can be removed.

When an external connection has *one* force loop that links it to another external connection of the subassembly, it may be that this latter connection leads to a component that blocks the subassembly. The geometric information in the force loop can be used to see whether this component actually blocks or not. This can be seen in Figure 5.

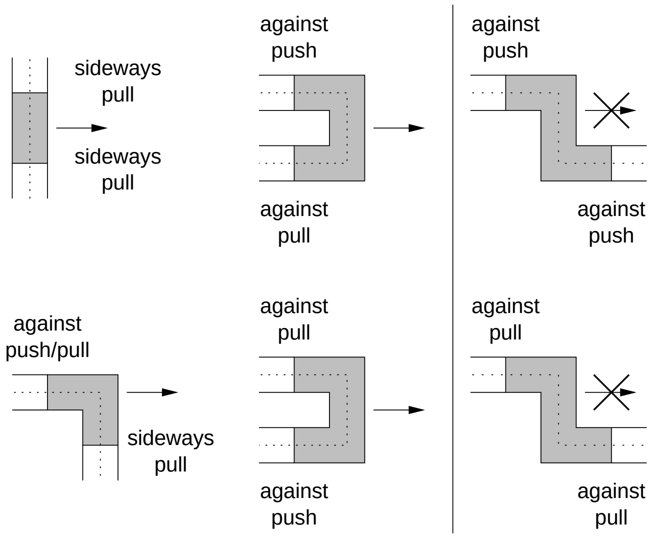

Figure 5: Situations where a subassembly can be removed (left) or is blocked (right).

It may also be the case that an external connection is linked to *more than one* other external connection by force loops. Each external component of these connections may then be blocking the subassembly. Therefore, the subassembly can only be removed when *all* force loops involved match a situation depicted left of the vertical line in Figure 5.

A subassembly is called a *free* subassembly when it can be removed. To see whether a subassembly is free, an external connection has to be found that matches one of the three cases described above. This can be formalized as follows:

$$
\begin{aligned}
&\textit{free(a)} \leftrightarrow \textit{subassembly(a)} \wedge \\
&\quad \exists\ \textit{c1: in-boundary(c1,a)} \wedge \neg\textit{broken(c1)} \wedge \\
&\quad\quad \forall\ \textit{l,c2: loops-through(l,c1)} \wedge \textit{loops-through(l,c2)} \wedge \\
&\quad\quad\quad \textit{in-boundary(c2,sa)} \wedge \textit{c1} \neq \textit{c2} \rightarrow \\
&\quad\quad\quad\quad \textit{broken(c2)} \vee \textit{sideways(c1)} \vee \textit{sideways(c2)} \vee \\
&\quad\quad\quad\quad (\textit{against(c1)} \wedge \textit{against(c2)} \wedge \textit{force(l,c1)} \neq \textit{force(l,c2)})
\end{aligned}
$$

Note that this definition only takes into account blocking subassemblies that are in direct contact with the subassembly. For situations where components that are not in direct contact prevent subassemblies to be removed (or connections to be loosened), such as closed covers or lids, the *state* concept has been introduced. This makes it possible to ensure for instance that the lid has been opened or that the cover has been removed before subassemblies are removed from the product.

3 The PROMOD System for Disassembly Analysis

It is now easy to give an algorithm that automatically generates the AND/OR tree of a product. Given a product model, at any point in the disassembly process, disassembly operations effectuate one of two changes in the model:

- connections can be loosened;

- free subassemblies can be removed.

In the algorithm, the disassembly operations are not considered themselves, but indirectly through the modifications they cause in the model. Then, a suitable algorithm that generates the disassembly tree (in a depth-first way) is:

```
algorithm disassemble m
  do cost-benefit analysis
  foreach connection c in m do
    if c can be loosened then
      m' = m
      loosen c in m'
      disassemble m'
    endif
  end
  foreach subassembly a in m do
    if a is free then
      m' = m
      remove a from the rest in m'
      disassemble m'
    endif
  end
end
```

The algorithm recursively invokes itself on a model `m'` that is a copy of the original model `m` with a modification operation applied to it. In this way, the original model is available to branch off the other alternatives for disassembly of model `m`.

PROMOD is a prototype KBS that implements the product models for disassembly as well as the AND/OR tree generation algorithm described above. In addition, it contains a simple form of ecological cost-benefit analysis, as follows. It uses a list of product components that have to be removed from the product for recycling or reuse. Each loosening or removal operation applied to the model accounts for some given ecological cost (e.g. 5 units for loosening a connection, 1 unit for removing a subassembly). The ecological benefits depend on the degree to which the components that were marked as components to be recycled or reused have been separated from the product. For simplicity, the cost-benefit analysis does not calculate an ecological benefit, but instead a penalty (e.g. 15 units) for each unwanted component that is still attached to one of the components that should be removed. The total evaluation score of a situation in the disassembly process is then defined as $1/(cost + penalty)$.

We will now discuss an example indicating the usability and reasoning power of the PROMOD system implementing the theory for product disassembly of this paper. Figure 6 shows a model of a coffee machine that has been analyzed by the system. Connection `c8` can only be loosened when the base has been separated from the case (i.e. when both components are not part of a single subassembly). The component `block` has been marked as the component to be removed from the product.

Figure 7 shows part of the results of the disassembly analysis by PROMOD. For each sequence of disassembly operations applied to the coffee machine it gives information on the subassemblies and the state of the subassemblies, on the

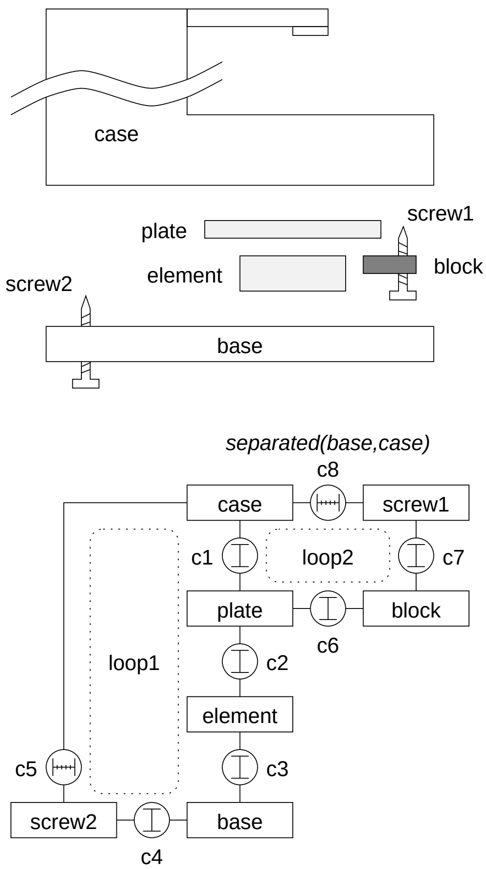

Figure 6: Disassembly model of a coffee machine.

groups of components that are separated, and on the ecological rating of the situation. The rating consists of three numbers: the ecological cost of disassembly, the total component penalty and the sum of these numbers.

```
Situations considered for 'Coffee Machine'

sequence: ((break c5) (remove screw2) (remove base)
           (break c8) (remove screw1) (remove block))
asys:     ((free case) (blocked plate) (separate screw1)
           (free element) (separate base) (separate block)
           (separate screw2))
clumps:   ((case plate element) (screw1) (base) (block)
           (screw2))
rating:      14    0   14
sequence: ((break c5) (remove screw2) (remove base)
           (break c8) (remove screw1))
asys:     ((free case) (blocked plate) (separate screw1)
           (free element) (separate base) (free block)
           (separate screw2))
clumps:   ((case plate element block) (screw1) (base)
           (screw2))
rating:      13   75   88

[many sequences removed]

sequence: ((break c5))
asys:     ((free case plate screw1 block) (blocked element)
           (blocked base) (free screw2))
clumps:   ((case plate screw1 element base block screw2))
rating:       5  150  155
sequence: NIL
asys:     ((separate case plate screw1 element base block
            screw2))
clumps:   ((case plate screw1 element base block screw2))
rating:       0  150  150
```

Figure 7: Disassembly analysis result from the PROMOD system for the coffee machine.

4 Related Work and Discussion

Computational Mechanical Engineering. The problem to obtain the AND/OR graph of a product has also been considered in computational (mechanical) engineering, and several approaches are reported in the literature.

In the *reversed fishbone* approach [Ishii and Lee, 1996], the designer has to come up with the preferred way to disassemble the product him/herself. This is graphically specified in a reversed fishbone diagram which can be considered as a subgraph of the AND/OR graph. The disadvantage of this approach is that when the design of the product changes, possibly large parts of the diagram have to be changed. This makes this method less suited for comparing alternative designs.

A method to obtain the AND/OR graph automatically is to generate it from a geometric model of the product. This is done in the *degrees of freedom* (Dof) approach [Khosla and Mattikali, 1989] where a 3D CAD drawing of the product is utilized. After a design modification the AND/OR graph can automatically be recomputed, but the drawback is that it needs extensive input information, which is not always available in the early stages of product design.

The third approach uses a topological *liaison diagram* [Fazio and Whitney, 1987; Sturges Jr. and Kilani, 1992], nodes denoting components and edges physical connections. The AND/OR graph is generated from this diagram plus additional relations specifying a partial ordering over the breaking of connections. Our theory is related to this approach. Its novel and distinguishing aspect is that by a clever choice of connection types, it can capture geometric information otherwise only available in 3D geometric models.

Ontological Engineering. In AI, much work has been done on geometric and spatial reasoning, e.g. [Joskowicz and Sacks, 1991; Cohn *et al.*, 1995]. Regarding other aspects of physical systems, an extensive collection of ontologies is now available, see [Top and Akkermans, 1994; Borst *et al.*, 1997; Gruber and Olsen, 1994] and the KSE/OntoLingua library of ontologies.

The PHYSSYS ontology [Borst *et al.*, 1997], for example, formalizes aspects of physical systems, including physical processes and related engineering mathematics, the latter by incorporating the ENGMATH ontology [Gruber and Olsen, 1994]. In order to construct large ontologies in a modular fashion, general abstract ontologies have been defined separately, such as mereology, general topology and systems theory. To define specific physical system aspects, these are imported and extended or specialized. A similar approach to enhance modularity and genericity in ontology design has been proposed in the context of a medical ontology library by [Heijst *et al.*, 1997].

The same design principles hold for the construction of an ontology for disassembly. General topology defines generic properties regarding connectedness. The nature of these connections is left open. For disassembly, it is straightforward to import this general ontology, and extend it by writing axioms introducing the proposed four types of connections. In the same vein, *subassemblies* as discussed in Sec. 2.4 can be formally defined using concepts from the abstract ontology of systems theory that is part of PHYSSYS:

$$\begin{aligned}
&\textit{subassembly(a)} \leftrightarrow \textit{subsystem-of(a,model)} \wedge \\
&\quad (\textit{in-boundary(c,a)} \rightarrow \neg \textit{enforced(c)}) \wedge \\
&\quad (\forall\ \textit{o1,o2: part-of(o1,a)} \wedge \textit{part-of(o2,sa)} \rightarrow \\
&\qquad \textit{connected(o1,o2)}) \wedge \\
&\quad \neg \exists\ \textit{o1,o2,c: part-of(o1,sa)} \wedge \textit{part-of(o2,sa)} \wedge \\
&\qquad\qquad \textit{connects(c,o1,o2)} \wedge \neg \textit{enforced(c)}.
\end{aligned}$$

Various abstract ontologies are reused here, including mereology (*part-of*), topology (*connects*) and systems theory (*subsystem-of, in-boundary*). Also, the formal definition given in Sec. 2.4 of a *free* subassembly reuses general systems theory. Thus, in the formal definition of the disassembly viewpoint, existing abstract ontologies theory can be reused, and modularly extended to enable reasoning about subassemblies and whether or not they are free for removal in disassembly.

5 Conclusion

In this paper, a new theory and modelling technique for disassembly of products has been presented. It is based upon an ontology that introduces unconventional but simple geometric and physical task-oriented abstractions in a topological context, by introducing connection types. Thus, advantages of several existing techniques for disassembly analysis are combined.

We have indicated how the ontology for disassembly can be neatly embedded in a broader ontology for physical systems, corroborating generic ontology construction principles proposed in the AI literature.

The proposed theory of product models has been implemented in a working prototype KBS. We have shown how the system is able to automatically generate the full disassembly graph and to carry out an ecological cost-benefit analysis.

References

[Borst *et al.*, 1997] W.N. Borst, J.M. Akkermans, and J.L. Top. Engineering ontologies. *International Journal of Human-Computer Studies*, 46:365–406, 1997.

[Clarke, 1981] B. L. Clarke. A calculus of individuals based on 'connection'. *Notre Dame Journal of Formal Logic*, 22(3):204–218, 1981.

[Cohn *et al.*, 1995] A.G. Cohn, D.A. Randell, and Z. Cui. Taxonomies of logically defined qualitative spatial relations. *International Journal of Human-Computer Studies*, 43:831–846, 1995.

[Fazio and Whitney, 1987] T.L. De Fazio and D.E. Whitney. Simplified generation of all mechanical assembly sequences. *IEEE Journal of Robotics and Automation*, RA-3(6):640–658, 1987.

[Fiksel, 1996] J. Fiksel. *Design For Environment: Creating Eco-Efficient Products and Processes*. McGraw-Hill, Inc., New York, 1996.

[Gruber and Olsen, 1994] T.R. Gruber and G.R. Olsen. An ontology for engineering mathematics. In J. Doyle, P. Torasso, and E. Sandewall, editors, *Proceedings Fourth International Conference on Principles of Knowledge Representation and Reasoning*, pages 258–269, San Mateo, CA, 1994. Morgan Kaufmann.

[Heijst *et al.*, 1997] G. Van Heijst, A.Th. Schreiber, and B.J. Wielinga. Using explicit ontologies in KBS development. *International Journal of Human-Computer Studies*, 46:183–292, 1997.

[Ishii and Lee, 1996] K. Ishii and B.H. Lee. Reverse fishbone diagram: A tool in aid of design for product retirement. In *ASME Design for Manufacturability Conference*, Irvine, California, August 1996. 96-DETC/DFM-1272, ASME DTC/CIE Proceedings CD, ISBN 0-7918-1232-4.

[Joskowicz and Sacks, 1991] Leo Joskowicz and Elisha Sacks. Computational kinematics. *Artificial Intelligence*, 51:381–416, 1991.

[Khosla and Mattikali, 1989] Pradeep K. Khosla and Raju Mattikali. Determining the assembly sequence from a 3-D model. *Journal of Mechanical Working Technology*, 20:153–162, 1989.

[Sturges Jr. and Kilani, 1992] R.H. Sturges Jr. and M.I. Kilani. Towards an integrated design for an assembly evaluation and reasoning system. *Computer-Aided Design*, 24(2):67–79, February 1992.

[Top and Akkermans, 1994] J.L. Top and J.M. Akkermans. Tasks and ontologies in engineering modelling. *International Journal of Human-Computer Studies*, 41(4):585–617, December 1994.