Components and Contexts: Working Note 2

Peter Clark and Bruce Porter
Dept. CS, UT Austin
{pclark,porter}@cs.utexas.edu

1 Introduction

The purpose of this working note is to explore some connections between the notions of *components*, *composition*, *problem-solving*, and *contexts*, and provide food for thought. We describe the construction of concept descriptions by three methods:

1. exhaustive application of all axioms
2. selective application of axioms to answer a particular question
3. exhaustive application of axioms from particular, pre-defined axiom sets ('contexts')

2 Concept Composition

Like many others, our premise is that a concept description should be constructed from composition of components (rather than spelt out by hand). There are many different ways and levels of complexity at which a concept can be described, and the choice of which description is most suitable is task-dependent. Our hope is that by constructing a concept from components, we can easily generate these different descriptions by selecting different subsets of its (complete set of possible) components.

Our notion of 'adding a component' is simply the application of an axiom (from a large KB of axioms) to a particular instance. It's sometimes useful to use Sowa's Conceptual Graphs (CGs) to depict these axioms graphically, and view the application of an axiom as the unification/joining of the graph representing that axiom to a graph representing the concept of interest.

For example, the description of `bus` in the previous working note was constructed from the following set of axioms:

```
1. "Petrol engines use gasoline, and produce average force."
2. "A vehicle's top speed is it's engine's force / vehicle mass."
3. "A bus has an average mass."
4. "Road vehicles have petrol engines."
5. "A vehicle's fuel-tank contains the engine's fuel."
6. "Gas tanks are made of steel."
7. "Gasoline is volatile."
8. "Steel things are average strength and average weight."
9. "A gas tank is a fuel tank containing gasoline."
```

```
bus(engine => petrol_engine(force => av,
                            fuel => A: gasoline(state => volatile)),
    fuel_tank => gas_tank(contents => A,
                          material => steel,
                          strength => av,
                          weight => av),
    mass => av,
    top_speed => av).
```

(A denotes the values are co-referential).

Figure 1: Representation of bus, formed by exhaustive application of axioms.

If we exhaustively apply all these axioms (in CG terms: unify the nine graphs denoting these axioms), we obtain the concept description shown in Figure 1 (drawn as a conceptual graph in linear notation).

We can, of course, produce simpler concept descriptions by limiting the axioms which we apply. For example, just using axioms 1, 3, and 5 we get a different description:

```
bus(engine => engine(fuel => A),
    fuel_tank => container(contents => A),
    mass => av).
```

Two reasons for wanting to do this are

Efficiency: With a large KB, exhaustive application of axioms would be infeasible.

Comprehensibility: If a description is being constructed to provide/generate information for the user, then some filtering mechanism may be needed to ensure that just limited, relevant information is communicated.

The rest of this note concerns how these limits might be imposed.

3 Inference-Driven Axiom Selection

We consider the task of providing the user information 'relevant' to a particular question. A simple definition of 'relevant information' is the information required to infer the answer. We can implement this using a standard backward-chaining rule engine to select axioms ('fire rules', if you prefer) required to answer a question. As a side-effect of answering the question, a selective description of the asked-about concept is 'grown' as each needed axiom is applied. The final concept description contains just that information used to answer the question – ie. the information contained in the inference engine's working memory.
Here are two examples of query-driven selection and composition of axioms:

```
> solve(strength of fuel_tank of bus)?
axiom 8 was applied.
axiom 6 was applied.
axiom 9 was applied.
```

```
axiom 5 was applied.
axiom 1 was applied.
axiom 4 was applied.

Answer = av
Underlying object:
  --------------------------------------------------
bus(engine => petrol_engine(fuel => A: gasoline),
    fuel_tank => gas_tank(contents => A,material => steel,strength => av))
  --------------------------------------------------

> solve(top_speed of bus)?
axiom 2 was applied.
axiom 1 was applied.
axiom 4 was applied.
axiom 3 was applied.

Answer = av
Underlying object:
  --------------------------------------------------
bus(engine => petrol_engine(force => av),
    mass => av,
    top_speed => av)
  --------------------------------------------------
```

Note the two descriptions of `bus` generated in response to the different queries, shown between the horizontal bars above.

4 Context-Driven Axiom Selection

4.1 Contexts

Using an inference engine to identify relevant axioms is one method of custom-building a concept description. However, this is potentially intractible in a large KB, and also unintuitively computes relevance as the result, rather than a precursor, of problem-solving.

An additional method for selecting relevant axioms is to group axioms into *contexts*, representing different aspects or 'viewpoints' of a concept, and then only work with (all) axioms from context(s) potentially relevant to a solution. In this way, a reasoning system works only (and hence more efficiently) with a subset of the knowledge potentially available to it. An axiom may belong to several contexts. For example (the below grouping of axioms is not ideal, but serves to illustrate the idea):

```
CONTEXT 1: ENGINE-RELATED AXIOMS
 1. "Petrol engines use gasoline, and produce average force."
 2. "A vehicle's top speed is it's engine's force / vehicle mass."
 4. "Road vehicles have petrol engines."
 5. "A vehicle's fuel-tank contains the engine's fuel."

CONTEXT 2: STRUCTURE-RELATED AXIOMS
 3. "A bus has an average mass."
 6. "Gas tanks are made of steel."
 8. "Steel things are average strength and average weight."

CONTEXT 3: FUEL-RELATED AXIOMS
```

```
1. "Petrol engines use gasoline, and produce average force."
5. "A vehicle's fuel-tank contains the engine's fuel."
6. "Gas tanks are made of steel."
7. "Gasoline is volatile."
9. "A gas tank is a fuel tank containing gasoline."
```

So, for example, the contexts required to answer the following questions are:

Query	**Contexts Required**	**(Axioms Needed)**
`strength of fuel_tank of bus?`	`Engine, Structure, Fuel`	(1,4,5,6,8,9)
`top_speed of bus?`	`Engine, Structure`	(1,2,3,4)
`state of fuel of engine of bus?`	`Engine, Fuel`	(1,4,7)
`material of fuel_tank of bus?`	`Engine, Fuel`	(1,4,5,6,9)

In an ideal world, it should be possible to identify which contexts are potentially relevant to solving a problem *without* having to actually solve it.

4.2 Contexts in CYC

Guha's notion of contexts[1] was motivated by an additional issue, namely that different Knowledge Enterers used terminology in inconsistent ways. Contexts isolated different sets of axioms (eg. from different knowledge enterers), and lifting rules defined how to transform an axiom from one context so it could be added to another. Most importantly, a run-time *problem-solving context* (PCS) is constructed by collecting axioms from different relevant contexts.

The illustration of contexts in Section 4 can be viewed as a special case of Guha's contexts, where different contexts are already compatible (ie. terminology is used consistently) and hence all axioms are directly liftable into other contexts without modification (ie. there is one trivial lifting rule saying all axioms are liftable[2]). The combined context formed to answer a specific query (eg. combined contexts `Engine, Structure` to answer `top_speed of bus?`) corresponds to Guha's problem-solving context.

5 Closing Remarks

This brief note hopefully has highlighted the following points:

1. It's desirable to construct concept descriptions from components, each encapsulating different aspects of a concept.

2. Axioms, components, if...then... rules, conceptual graphs, and frames can essentially be thought of as synonyms.

3. Via answering a question, an inference engine produces a simplified concept description containing just information relevant to the answer.

4. Contexts can be used as a means of limiting information available to an inference engine, to help achieve tractibility.

[1]See Blair, P., Guha, R. V., and Pratt, W. (1992). *Microtheories: An ontological engineer's guide* Tech Rept CYC-050-92, MCC, Austin, TX

[2]In fact Guha uses a more general version of this lifting rule called the Default Coreference Axiom (DCA), which states that all axioms are liftable unless we know some specific reason for not doing so.