In AAAI Fall Symposium on Computational Approaches to Representation Change during Learning and Development 2007.

Handling Granularity Differences in Knowledge Integration

Doo Soon Kim and Bruce Porter
The University of Texas at Austin
Computer Science Department
{onue5, porter}@cs.utexas.edu

Abstract

Knowledge integration is a process of combining two different knowledge representations together. This task is important especially in learning where new information is combined with prior knowledge or in understanding where a coherent knowledge representation should be generated out of several knowledge fragments. A challenging problem in KI is handling granularity differences, i.e. combining together two knowledge representations with granularity differences. This paper presents an algorithm to find such correspondences between two representations with a granularity difference and to combine the two representations together based on the correspondences. The algorithm uses coarsening operators which generate coarse-grained representations from a representation. At the end, we introduce a large scale project in which the algorithm will be used.

Knowledge Integration (KI) is a process of combining two pieces of knowledge[1] together. KI is an important step in human activities such as learning, reading comprehension, reasoning and others. For example, in learning, one must combine the new knowledge with the old, and in reading comprehension, one must relate together individual sentences in a text which often differ in level of detail. Just as in human activities, KI is also important in many AI tasks such as knowledge acquisition, story understanding, and multidocument summarization. All these tasks commonly require the combination of several pieces of knowledge to generate a coherent knowledge representation.

A challenging problem in KI is integrating together two knowledge representations with granularity differences. Simply putting the representations together without considering the granularity differences could result in an incorrect representation. Ideally, the two representations should be aligned together based on identification of which parts in the fine-grained correspond to which parts in the coarse-grained. This paper presents an algorithm to find the correspondences between two representations with a granularity difference and to combine the representations together based on the correspondences. The algorithm uses coarsening operators which generate coarse-grained representations from a representation.

[1] In this paper, by knowledge, we mean factual knowledge which can be represented by logic. We will use conceptual graphs or logical predicates as our notation.

Example

This section compares a good and a bad KI on the two texts below about *blood circulation*. These texts used in the learning-by-reading project (Barker *et al.* 2007) were excerpted from a biology textbook. Fig. 1a and fig. 1b show the encoding of the long and the short text respectively.[2] The Component Library (Barker, Porter, & Clark 2001) was used as the ontology for the encoding.

Text1:

Hearts pump blood through the body. Blood carries oxygen to organs throughout the body. Blood leaves the heart, then goes to the lungs where it is oxygenated. The oxygen given to the blood by the lungs is then burned by organs throughout the body. Eventually the blood returns to the heart, depleted of oxygen. It is then pumped by the heart back to the lungs.

Text2:

Heart pumps blood to circulate around the body.

Fig. 2 shows a bad combination by a simple graph join (Sowa 1984) on the two representations in fig. 1. In this combination, Move3 and Move1 are unified together. Notice that Move3 is from the coarse-grained representation (fig. 1a), and Move1 is from the fine-grained representation(fig. 1b). Consequently, Move13, resulting from the unification of Move1 and Move3, comes to have two destinations (Lung1 and Heart12), such that its English interpretation would be *"Blood moves from the Heart to the Heart and the Lung."* In contrast, fig. 3 illustrates a good combination in which Move1 is identified as a superevent of Obtain1, Move1, Carry1, Move2, and thus subevent relations are estabilished between Move1 and Obtain1, Move1, Carry1, Move2. The English interpretation for this combination would be *Blood moves from Heart to Heart. This*

[2] Due to the limitation of space, some information in the long text is not encoded.

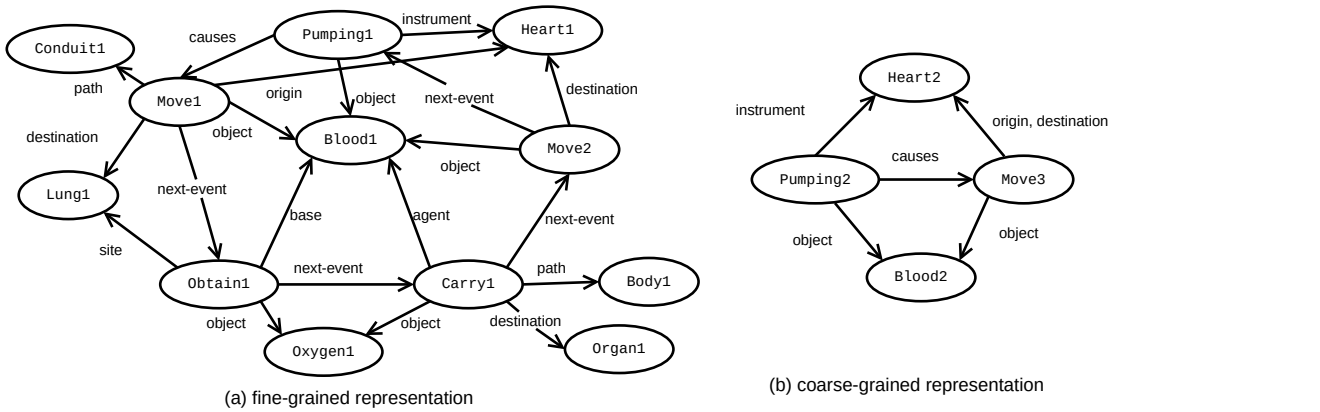


Figure 1: Encodings on the two texts on blood circulation: The fine-grained is shown in the left and the coarse-grained in the right.

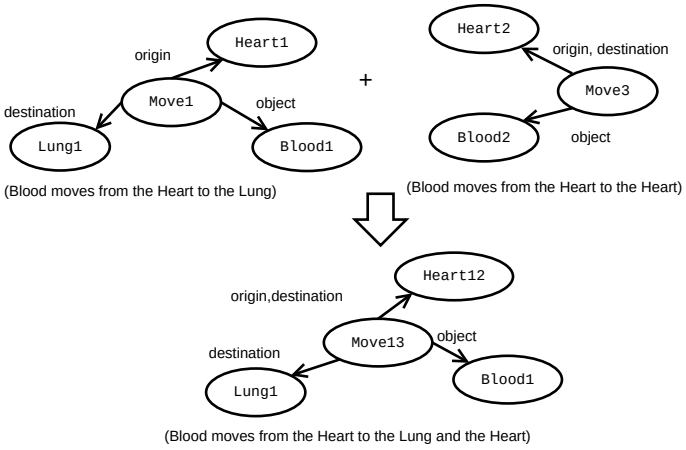


Figure 2: Bad combination of the two representations in fig. 1 by a simple graph join (Sowa 1984). English glosses are given below each representation. Because of the space limitation, the relevant parts of the represetations from fig. 1 are shown. In this combination, Move1 and Move3 are unified, and consequently the resulting node, Move13, has two destinations, Lung1 and Heart12. (<name><number1><number2> labels a node created by the unification of <name><number1> and <name><number2>. For example, Move13 is a node created by the unification of Move1 and Move3.)

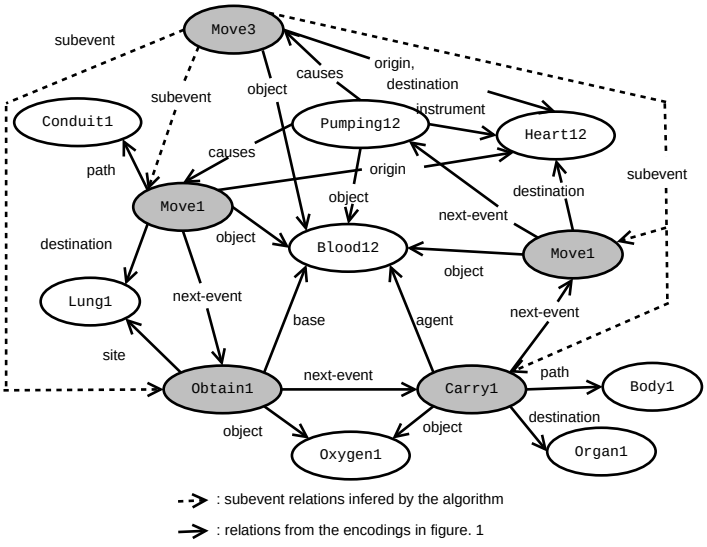


Figure 3: Good combination of the two representations in fig. 1 using our algorithm. Move3 is identified as a superevent of Move1, Obtain1, Carry1, and Move2

movement has 4 subevents: Blood moving to the lung, Blood obtainining oxygen, Blood carrying oxygen, and Blood moving to the heart.

Difficulties in handling granularity differences in KI

We confine ourselves in this section to the difficulties in handling granularity differences, although there are other difficult problems in KI (Murray 1995).

- **finding correspondences between the fine-grained and the coarse-grained** The parts in the fine-grained that are detail descriptions of parts in the coarse-grained should be identified. Just putting them together without this consideration could produce an incorrect representation as in fig. 2.
- **revealing unspecified relations** The unspecified relations between the fine-grained and the coarse-grained should be explicitly represented in the combined representation. For example, given *"The heart pumps blood around the body"* and *"The right ventricle contracts blood to the lung,"* the combined representation has to represent explicitly that the RIGHT VENTRICLE is a part of the HEART, the CONTRACT action is a subevent of the PUMPING action, the LUNG is a part of the BODY. The more related they are, the more useful the combined knowledge would be.
- **resolving corefered entities** Entities corefered by different representations should be identified. For example, in fig. 1, BLOOD, HEART, and PUMPING are entities corefered by the two representations.

Problem description and Algorithm

This section presents a KI algorithm to handle granularity differences. The algorithm takes in two representations which represent the same topic: a fine-grained representation and a coarse-grained representation. The algorithm coarsens the granularity of the fine-grained representation repeatedly until there is an enough graph match (Sowa 1984) between the coarsened fine-grained representation and the coarse-grained representation (Algorithm1).

Algorithm 1 Finding correspondences between two representations with granularity differences

Input : K_{fine}, a fine-grained representation
K_{coarse}, a coarse-grained representation
Output : combined knowledge representation
Algorithm
while K_{fine} is not matched with K_{coarse} well enough (Sowa 1984) **do**
$K_{fine} \leftarrow$ transform K_{fine} into a more coarse-grained represenation using the coarsening operators.
Keep the information of how K_{fine} gets transformed.
end while
Combine K_{fine} and K_{coarse} together based on the transformation information recorded during the coarsening.

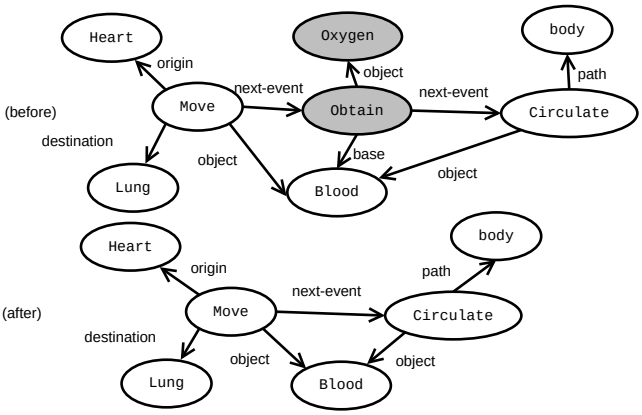


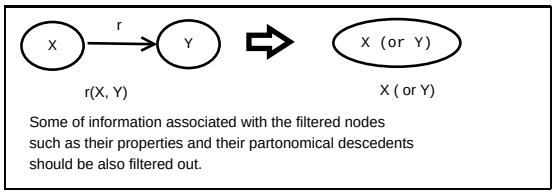
Figure 4: Example of filtering: *To pick up oxygen* is filtered out.

Coarsening operators

The coarsening operators convert a fine-grained representation into a more coarse-grained one. (Stell & Worboys 1999) and (Ramalingam & Hornsby 2001) identified a similar type of coarsening operators to handle spatial granularity differences.
We defined three types of coarsening operators: filtering, generalization, and abstraction. Each type is represented as a rule. The antecedent of a rule represents a pattern that is coarsened, and the consequent represents the result of coarsening the antecedent. By matching an antecedent of a rule with a part of a knowledge representation and replacing the part with the consequent, the representation can be transformed into a more coarse-grained one.
Each rule will be illustrated with a diagram in which nodes denote instances, and edges denote relations among the instances. The corresponding logical notation is provided.

1. Filtering

Some of the information in a representation can be filtered out in a coarse granularity.



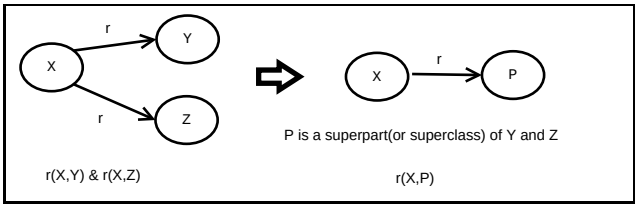
Example
(before) The blood flows from the heart to the lung *to pick up oxygen*[3] and then circulates around the body.
(after) The blood flows from the heart to the lung and then circulates around the body.

[3]Throughout the examples, *italic words* represent the coarsened parts

Fig. 4 shows the example in a conceptual graph notation.
To prevent any part of a representation from being filtered out, it is necessary to determine which parts are allowed to be left out. For example, *"The blood picks up oxygen and then circulates around the body"*, resulting from filtering out *"The blood flows from the heart to the lung"* from the above example, is not a good description of blood circulation. For now, we leave this problem as our future research problem. We speculate that coherence plays an important role. In the example, FLOW is coherent with CIRCULATE in that they together represent the continuation-of-movement viewpoint, whereas PICK-UP and CIRCULATE do not constitue any coherent viewpoint.

2. Generalization

A few specific descriptions can be generalized into a more general description.



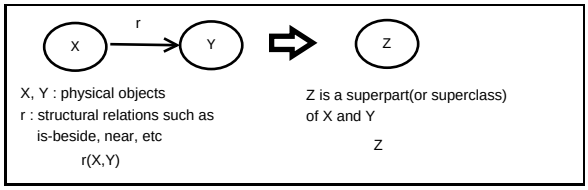
Example
(before) The *right ventricle* is below the *right atrium* in a heart. The *left ventricle* is below the *left atrium* in a heart.
(after) A ventricle is below the atrium in the heart.

Only common relations from Y and Z are carried over to P.

3. Abstraction

A detail description can be abstracted out. This abstraction looks slightly different in different domains.

structural abstraction Structural domains describe physical objects and relations among them.



Example
(before) A blood conduit is connected to an atrium. *An atrium is above a ventricle.* A ventricle is connected to another blood conduit.
(after) A blood conduit is connected to a heart, which is connected to another conduit.

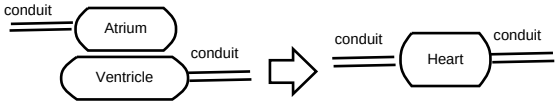
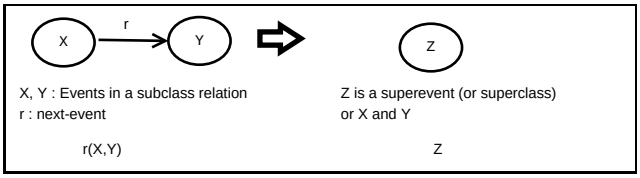


Figure 5: A picture to illustrate the example. Atrium and ventricle are abstracted into heart.

All relations associated with X and Y can be carried over to Z.

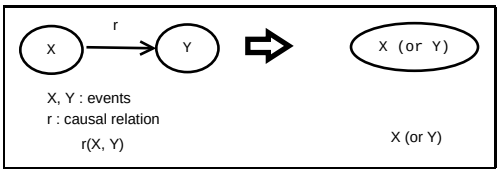
functional abstraction Functional domains describe events and temporal relations among them.



Example
(before) Jack moves from LA to Chicago, and then moves from Chicago to NY.
(after) Jack moves from LA to NY.

Some relations associated with X and Y should be prohibited from being carried over to Z if they describe a state which results from the event X but are overriden by the event Y. In the above example, Chicago is an intermediate location which is nullified later by the next move.

causal abstraction Causal domains describe events and causal relations among them.



Example
(before) John mixed oxygen and hydrogen together. As a result, water was produced.
(after)
1. Mixing together oxygen and hydrogen by John produced water.
2. Water is produced out of oxygen and hydrogen by John.

The difference from the above two cases is that either X or Y is chosen in the consequent. Fig. 6 shows two ways of abstracting the **before** text in the example.
If the cause (X in the above diagram) is retained in the consequent, the information associated with Y should not be carried over except the information describing effects by Y. In the example, (result water) is carried over(the dotted line in fig. 6b). If the effect (Y in the diagram) is retained in

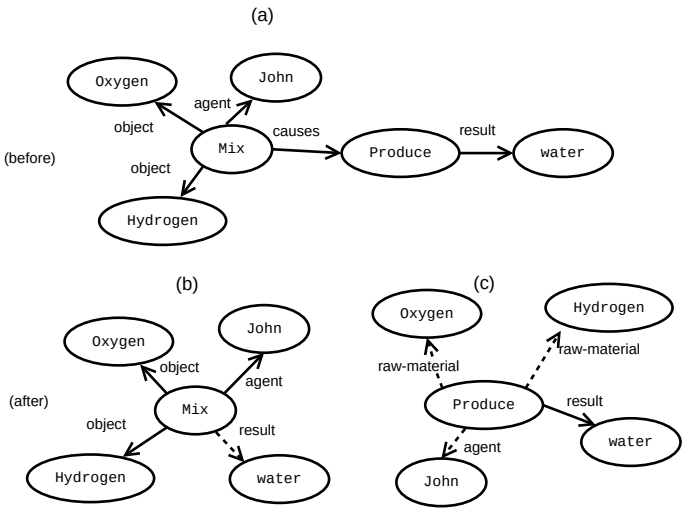
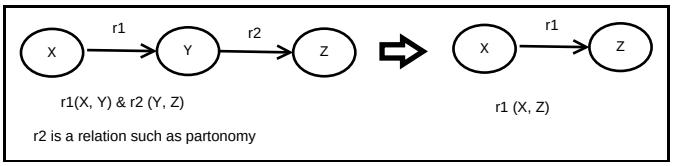


Figure 6: Two ways of abstracting the **before** text in the example.

the consequent, only information describing the cause of X such as an agent of X or a causal event of X should be carried over. In the example, (agent John), (raw-material Oxygen) and (raw-material Hydrogen) are carried over (the dotted lines in fig. 6c)

4. Referring at different level of granularity

A same thing can be referred at different level of granularity.



Example
(before) John moves *to Austin*. *Austin is a part of Texas.*
(after) John moves to Texas.

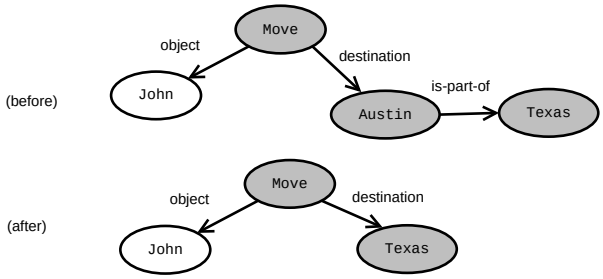


Figure 7: The destinations of MOVE are different in the level of detail. Two relations, DESTINATION and IS-PART-OF, gets abstracted into DESTINATION

In the above example, the two knowledge representations have different destinations (Austin and Texas respectively), though both express the same meaning. These destinations are different in the level of granularity. (Yeh, Porter, & Barker 2003) identified a similar type of rules called "transfers-thru" in their library of transformation rules.

Handling granularity difference as state space search

The operators in the previous section can coarsen a fine-grained representation in many different ways. To handle multiple possible outputs, we reformulate our problem as a state space search. We will use G_{fine} and G_{coarse} to denote a fine-grained representation and a coarse-grained representation to be matched, respectively.

- **Initial state** A pair of G_{fine} and G_{coarse}
- **State** A pair of the coarsened G_{fine} and G_{coarse}
- **Operators** Coarsening operators
- **Goal** Succeed if G_{fine} and G_{coarse} in the pair are matched to each other well enough with a conceptual graph match (Sowa 1984).

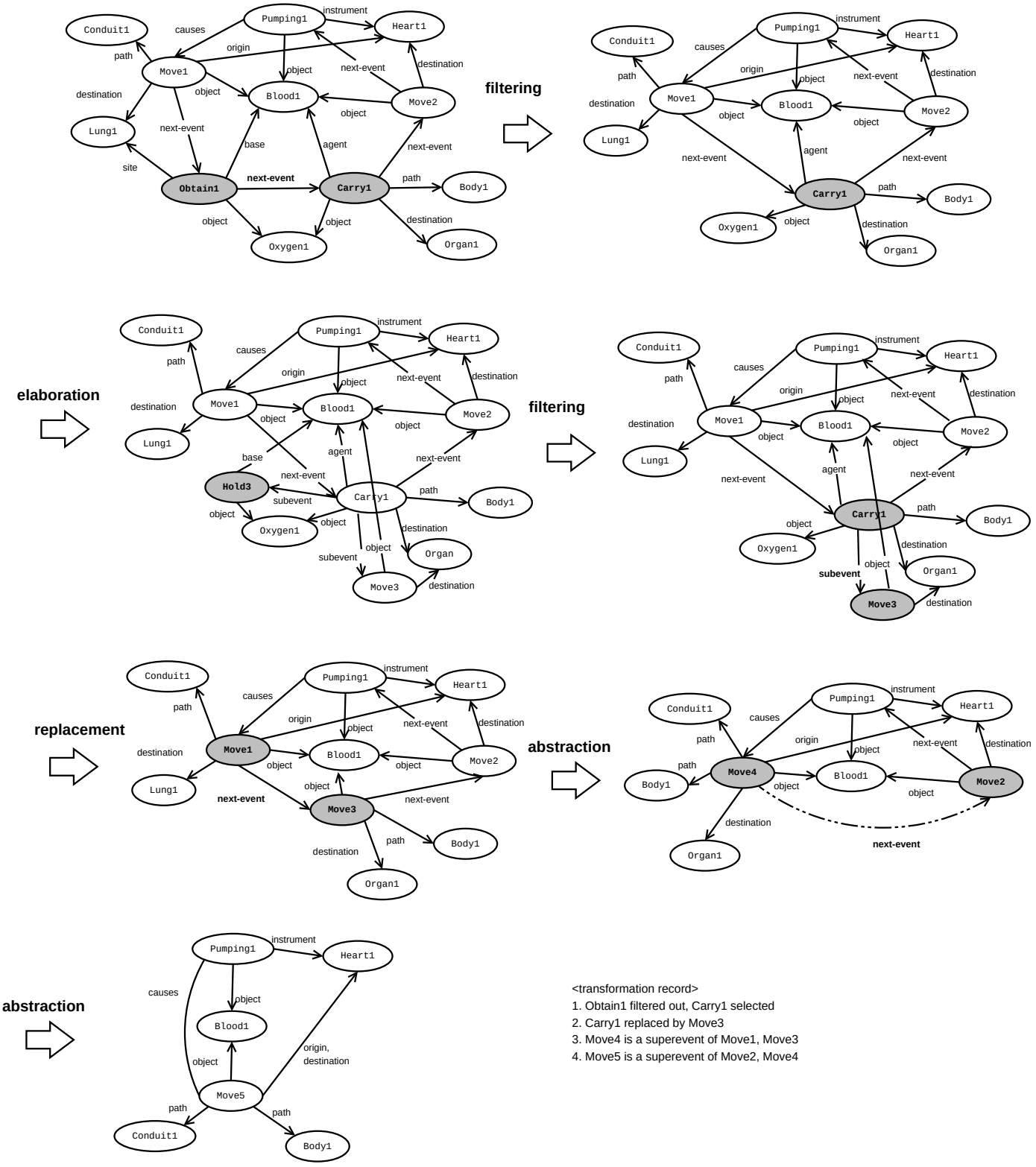
As in other search-based methods, it would suffer from the state explosion problem. One heuristic of preventing the state explosion is to use the coarse-grained input as a guidance for the search. For example, the coarsening operators may be restricted to apply only to unmatched parts between the fine-grained and the coarse-grained.

Working Example

Fig. 8 illustrates a sequence of transformations of the fine-grained representation in fig. 1a. This sequence would be a solution path in a state space search. The figure shows that the transformed representation finally comes to have the same granularity as the coarse-grained. Based on the transformation records kept during the transformations, our algorithm is able to estabilish the subevent relations between the events in the two representations. Fig. 3 shows the KI result on the two representation in fig. 1 by our algorityhm based on the transformation records.

Application to Learning-by-Reading

Learning-by-Reading is a large scale project in which the goal is developing a system to build a knowledge base by reading a text. In the system, the role of the KI module is to integrate together logical forms generated by the NL system. Please see (Barker *et al.* 2007) for details. Failure of handling granularity differences in logical forms during KI was one of the brittlenesses in last year's system. For example, the topic sentence of a paragraph is typically coarse-grained and the subsequent ones are more detailed. We plan to overcome this brittleness by incorporating the algorithm presented in this paper. The coarsening operators surfaced with the Component Library will be used. Then, how much the readibility is increased by our algorithm will be evaluated by measuring the correctness of the representation integrated out of logical forms.



<transformation record>
1. Obtain1 filtered out, Carry1 selected
2. Carry1 replaced by Move3
3. Move4 is a superevent of Move1, Move3
4. Move5 is a superevent of Move2, Move4

Figure 8: The fine-grained representation gets coarsened repeatedly by the coarsening operators until being aligned with the coarse-grained input well enough by the conceptual graph match (Sowa 1984). The gray nodes are the ones on which the operators are applied. From the transformation records, which keep the information on how the fine-grained has been converted, we can infer that Move5 is a superevent of Obtain1, Move1, Move2 and Carry1. The above figure contains two operations which are not decribed in this paper: **elaboration** and **replacement**. **Elaboration** is an operation where more knowledge is added to elaborate a concept. In the above example, CARRY is elaborated as having HOLD and MOVE as its subevent. **Replacement** is an operation to replace a concept with its partonomical child. In the example, CARRY is replaced with MOVE, which is a subevent of the CARRY

Acknowledgments

This research was funded by DARPA under contract NBCHD030010.

References

Barker, K.; Agashe, B.; Chaw, S. Y.; Fan, J.; Glass, M.; Hobbs, J.; Hovy, E.; Israel, D.; Kim, D. S.; Mulkar, R.; Patwardhan, S.; Porter, B.; Tecuci, D.; and Yeh, P. 2007. Learning by reading: A prototype system, performance baseline and lessons learned. In *Proceedings of Twenty-Second National Conference on Artificial Intelligence*.

Barker, K.; Porter, B.; and Clark, P. 2001. A library of generic concepts for composing knowledge bases. In *Proceedings of the international conference on Knowledge capture*, 14–21. ACM Press.

Murray, K. S. 1995. *Learning as Knowledge Integration*. Ph.D. Dissertation, The University of Texas at Austin, Department of Computer Sciences.

Ramalingam, C., and Hornsby, K. 2001. Generalizing graphs using amalgamation and selection. In *Proceedings of the First COSIT Workshop on Spatial Vagueness, Uncertainty, and Granularity*.

Sowa, J. 1984. *Conceptual Structures.* Addison-Wesley.

Stell, and Worboys. 1999. Generalizing graphs using amalgamation and selection. In *SSD: Advances in Spatial Databases*. LNCS, Springer-Verlag.

Yeh, P. Z.; Porter, B.; and Barker, K. 2003. Using transformations to improve semantic matching. In *Proceedings of the international conference on Knowledge capture*, 180–189.