Improving Source Selection in Analogical Reasoning
An Interactionist Approach

William A. Stubblefield
Department of Computer Science
Dartmouth College

George F. Luger
Department of Computer Science
University of New Mexico

Abstract

The success of any analogical reasoner depends upon its ability to select a relevant source. We can improve source selection by more completely integrating the process of source retrieval with analogical inference, and by using experience in solving target problems to find properties that effectively predict a source's relevance to future targets. This paper describes the design and evaluation of SCAVENGER, an analogical reasoning program that we have built to test these ideas.

1. Interactionism and source retrieval

Analogical inference is the process of reasoning about similarity: If two things are known to have certain similarities, we may infer by analogy that they are likely to have additional properties in common. Formally, an analogy is a mapping from elements of a *source*, a well understood problem solution, theory, plan, etc., to the *target*, a new problem to be solved. Analogical inference constructs this mapping, transferring knowledge from the source to the target in the process.

The success of any analogy depends upon the selection of a relevant source. Most analogical reasoners share a common structure (Hall 1989; Kedar-Cabelli 1988) that makes a number of assumptions about source selection. These assumptions include:

1. **The separation of retrieval and inference.** Many analogical reasoning programs treat source retrieval and inference as separate operations. They choose a source on the basis of known similarities to the target, prior to making any inferences.

2. **Source oriented approaches to memory organization.** To improve efficiency, analogical reasoners organize sources under a hierarchical index. Generally, they construct indices by comparing source descriptions to find properties that best distinguish them, and use these properties as indices for source retrieval (Kolodner 1993; Fisher and others 1991).

Our criticism of these assumptions derives from the *interaction theory* of metaphor (Black 1962). This theory views metaphors as complex interactions between systems of relations in the source and the target. These interactions, while primarily transferring information to the target, can also alter the semantics *of the source*. In particular, metaphors can create notions of similarity where none previously existed. Applying the interaction theory to analogical reasoning leads to the following criticisms of current source selection methods:

1. **Source retrieval and inference should be combined.** If metaphors and analogies can create similarities, how can we use similarity to select analogical sources? This circularity suggests that retrieval is itself a form of inference concerning the relevance of a source to a target. An analogical reasoning system should integrate retrieval and inference, choosing a source by evaluating the possible inferences afforded by alternative sources.

2. **Index hierarchies should be constructed through experience in solving target problems.** Most systems construct index hierarchies by comparing *sources* to find those properties that best distinguish them. Unfortunately, such algorithms can only consider a small number of properties in constructing hierarchies. The common approach to this problem is for the system designer to specify a fixed *retrieval vocabulary*. This ignores the reasoner's experience in solving target problems. An alternative approach uses information gained in solving targets to find properties that will select relevant sources.

2. An interactionist model of source retrieval

We have formalized these ideas in a computer program, called SCAVENGER, that explains empiri-

cal observations through analogies with knowledge of similar situations. We state this problem as follows:

Given:

1. A series of observations to be explained. These have a temporal order, but their causal structure is not apparent.

2. Knowledge about objects and processes in a variety of domains. These are candidate sources for explaining the observations.

Goals:

1. Select an appropriate source, and construct an analogy between it and the target observations.

2. Based on this analogy, construct a causal explanation of the observations.

3. Confirm the explanation empirically. Assume that these experiments are costly, and should be kept to a minimum.

We have represented observations to be explained as transcripts of evaluations of LISP functions. Sources are selected from a collection of known objects and methods written in the Common LISP Object System. We choose this representation in order to test our ideas using a complex language that was not of our own design. Also, this representation enables the reasoner to test its inferences by constructing and evaluating LISP function calls.

For example, consider the problem of explaining the meaning of target-class, target-function-1 and target-function-2 in the following transcript. The transcript represents the behavior of these functions as if they were evaluated by the LISP interpreter[1]:

```
> (setq x (make-instance 'target-class))
?
> (target-function-1 'a x)
?
>  (target-function-1 'b x)
?
> (target-function-2 x)
a
> (target-function-2 x)
b
```

Source objects and methods are stored along with high-level descriptions of their semantics. For example, the stack manipulation method, pop, is stored in the source base with:

1. Its argument and result types:
   pop: stack -> t

2. Its LISP definition:
   (defmethod pop ((s stack)) . . .)

3. A description of its result and side effects. In this case, the result is an element of a collection and the side-effect is to remove an item from the collection passed in as its first argument:

   result**: (accessed-element result)**

   side-effects:
   ((remove-from-collection arg-0))

To illustrate SCAVENGER's behavior, consider the example of interpreting the above transcript. Assume that sources are indexed under the index hierarchy shown in figure 1. Here, the root node has one specialization (node #2), that describes and indexes three source functions: pop, dequeue and first. These functions operate on instances of the classes stack, queue and ordered-list respectively. Note that the language used to describe functions is general enough to represent classes of similar functions.

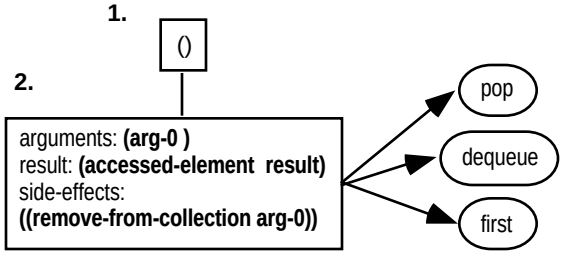

Figure 1

The algorithm proceeds as follows:

1. Preprocess the target transcript. This step extracts available information on the types of target function arguments. It will use this information to constrain analogical matches. SCAVENGER also assigns a default, "vanilla" semantics to each target. For example, both target functions would be as-

[1] The "?" indicates an unspecified (or unknown) value; this adds an element of noise to the observations to be explained.

sumed to have no side effects, and return a new object as their result:

result: **(new-object result)**
side-effects: **()**

2. Search the hierarchy for all indices whose function descriptions match a subset of the target functions. In our example, target-function-2 matches the description in index 2. The root index, which has no function description, will match any target. On matching an index, transfer its function descriptions to the matching target functions. This is an analogical inference; it is also non-monotonic in that it may later be retracted. SCAVENGER handles non-monotonicity by maintaining multiple *interpretations* of the behavior of target functions. In this example, a search of the index hierarchy produces two interpretations: the match with the root will leave all functions with their default semantics (call this interpretation-1). The match with node #2 interprets target-function-2 as returning an item from a collection, and changing the collection as a side-effect (interpretation-2).

3. Rank these different interpretations. SCAVENGER uses a number of heuristics to do so. The most important favors interpretations resulting from matches with nodes deep in the index hierarchy. Another heuristic constructs a graph[2] of the target transcript using both default function descriptions and those inferred from the index match. It analyzes this graph for properties such as connectedness, simplicity, etc. By transferring information from the source to the target and using it to evaluate candidate sources, a proposed analogy can actually create new notions of similarity.

4. Select the best ranked of the unevaluated interpretations, and create a set of partial analogical mappings between target functions and sources stored under the index. Doing this to interpretation-2 results in three partial analogies (figure 2). Note that the match with the root would yield one partial analogy in which no targets are mapped.[3]

5. Complete each partial analogy. Each analogy can have multiple completions, although type constraints acquired by the initial match limits possibilities. For example, the mapping of target-function-2 onto pop will also cause target-class to be mapped onto the source class: stack. Consequently target-function-1 can only map onto the function, push.

Partial analogies

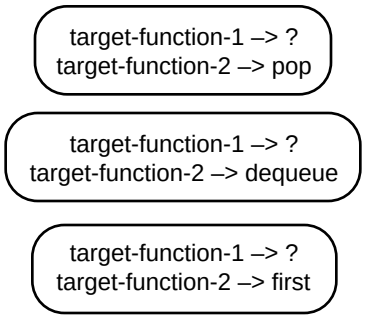

Figure 2

6. Evaluate the analogies produced in step 5 by using the sources to execute the target transcript. In this example, the only analogy that will execute the target correctly maps target-class onto stack, target-function-1 onto push and target-function-2 onto pop. If none of the analogies produced by the current interpretation behave correctly, go to step 4; otherwise, return this solution[4] and continue to step 7. The graph of the transcript constructed using the function descriptions acquired through the analogy is the desired explanation of the target.

7. Update the index hierarchy by trying to specialize the index node that produced the successful analogy. SCAVENGER does this by searching for a source function in the successful analogy that was not indexed under the node being specialized, and whose description differs from those of the failed sources. SCAVENGER uses a variation of the information-theoretic evaluator used in the ID3 induction algorithm (Quinlan 1986) to rank alternative specializations. For example, if we assume that push and enqueue have identical descriptions, but the function for inserting objects into an ordered-list has a different description, SCAVENGER would produce the specialization of figure 3. In using success in solving target problems to determine the relevance of different source methods, SCAVENGER implements a form of analogical transfer from the target back to the source.

---

2 A variation of a data-flow graph.

3 If all other indices fail to produce a plausible analogy, SCAVENGER will eventually fail back to the root. This will lead to an exhaustive search of the source base under step 5, guaranteeing that the algorithm will find a plausible analogy, should one exist.

4 In general, there may be multiple plausible analogies for a given target. SCAVENGER selects the best of these solutions heuristically.

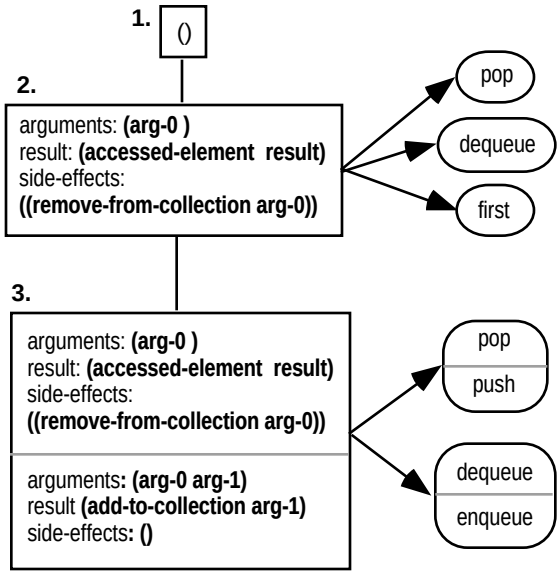

Figure 3

3. Evaluation

A central question in evaluating SCAVENGER is whether there is enough similarity between target problems to create generally useful indices. In repeated trials on 17 training instances, using a source base of 12 classes and 68 methods, the learning algorithm demonstrated a 65% speedup (figure 4). The learning algorithm also generalizes well across different target instances. Testing it on a different set of problems from those used to train it showed a 52% speed up. Note that this ability to generalize, is sensitive to the language used for describing sources.

In addition, the learner discovers interesting combinations of source functions. In the example of this paper, it found that combinations of insertion and accessor functions for collections were interesting.

We have tested the algorithm on a variety of domains, including the LISP transcript interpretation problem described above, the problem of finding bugs in failed plans, and the explanation of observations in naive physics, and these results hold across domains.

4. Conclusion

Our work has investigated the ramifications of the interaction theory of metaphor for analogical source retrieval. In particular, it has shown the viability of transferring knowledge of the relevance of source properties from the target back to the source, using this to update the organization of source memory. This idea that criteria for determining the relevance of analogical sources arises out of their interaction with target problems is one of the most interesting and potentially useful implications of the interaction theory of metaphor.

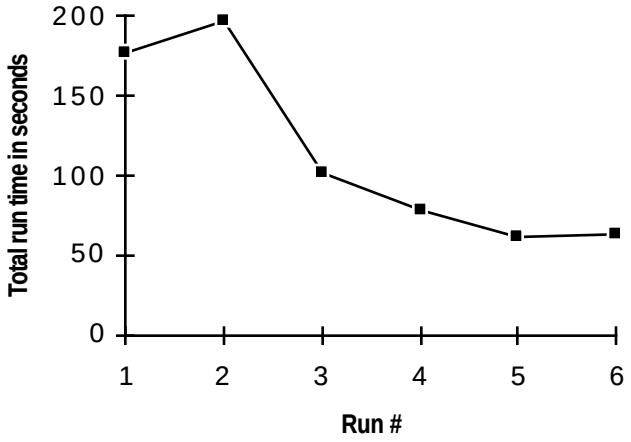

Figure 4

References

Black, Max. 1962. *Models and Metaphors.* Ithaca, NY: Cornell University Press.

Fisher, Douglas H., Michael Pazzani J., and Pat Langley, ed. 1991. *Concept Formation:Knowledge and Experience in Unsupervised Learning*. San Mateo, Cal.: Morgan Kaufmann.

Hall, R.P. 1989. Computational Approaches to Analogical Reasoning: A Comparative Analysis. *Artificial Intelligence* 39 (1): 39-120.

Helman, David H., ed. 1988. *Analogical Reasoning: Perspectives from Artificial Intelligence, Cognitive Science and Philosophy*. Dordrecht: Kluwer Academic.

Kedar-Cabelli, S. 1988. Analogy - From a Unified Perspective. In David H. Helman (1988).

Kolodner, Janet. 1993. *Case-Based Reasoning.* San Mateo, Cal.: Morgan Kaufmann.

Quinlan, J. R. 1986. Induction of Decision Trees. *Machine Learning* 1 (1): 81-106.