

**INVESTIGATION OF
HYPOTHESIS-DRIVEN
CONSTRUCTIVE INDUCTION
in AQ17-HCl**

L. E. Guillen, Jr. and J. Wnek

MLI 93-13

INVESTIGATION OF HYPOTHESIS-DRIVEN CONSTRUCTIVE INDUCTION IN AQ17

ABSTRACT

Investigation of the AQ17-HCI method within the general-purpose constructive-induction system AQ17 began with experimentation involving its supporting -induction system AQ15 and a target concept from the ROBOTS domain in the EMERALD system. With the aid of the diagrammatic-visualization (DIAV) technique, AQ15 results from incremental training example sets were clearly displayed on the screen contrasting the learned concept against the target concept. AQ15 had searched successfully through the space of possible descriptions with its ultimate goal of inducing a positive output hypothesis that matched exactly the target concept.

Investigation of the constructive-induction capabilities of AQ17-HCI proceeded with an analysis of output hypotheses from the attribute-removal module AQ17(HCI/ar). The analysis disclosed that AQ17 established the attributes occurring in the positive output hypothesis as relevant based on an attribute-utility measure. It discarded all other attributes as irrelevant. As a consequence, only relevant attributes appear in the negative output hypothesis which may lead to an overlap of the hypotheses: certain combinations of attribute values occurring in the positive output hypothesis may also occur in the negative output hypothesis. As part of the analysis, I applied a method described by Wnek, which I dubbed the "altered positive output hypothesis" (APOH) method and which uses what I call "hypothesis-driven deduction" (HDD). The APOH method yielded a final positive hypothesis which matched exactly the target concept without the incremental addition of training examples.

My investigation led me to a comparison of the APOH method with Wnek's VS* (version space with the STAR) method. When the methods initially encounter a coverage overlap by positive and negative rules, each method proceeds with a successful effort to output a final hypothesis for the positive class which matches exactly the target concept-description.

KEY WORDS

Concept learning, hypothesis-driven constructive induction, diagrammatic visualization, hypothesis-driven deduction.

ACKNOWLEDGEMENTS

It gives me great pleasure to thank Janusz Wnek for his great and untiring efforts in collaborating with me on this project.

This research was done at George Mason University's Artificial Intelligence Center. Research activities of the center are supported, in part, by the Defense Advanced Research Projects Agency as administered by the Office of Naval Research under grants No. N00014-87-K-0874 and No. N00014-91-J-1854, in part, by the Office of Naval Research under grants No. N00014-88-K-0226, No. N00014-88-K-0397, and No. N00014-91-J-1351, and, in part, by the National Science Foundation under grant No. IRI-9020266.

INTRODUCTION

The goal of this project has been to investigate the hypothesis-driven constructive-induction (HCI) method and to experiment within the general-purpose constructive-induction system AQ17. Inherently, the goal included investigation of, and experimentation with, the supporting selective-induction learning program AQ15. As described later in this project report, a relation to other work became evident during experimentation with the AQ17-HCI module. A similarity was apparent between what I refer to in this paper as the "altered positive output hypothesis" (APOH) method as described by Wnek (1991) and the version-space-with-the-STAR (VS*) method (Wnek, 1992).

2 METHOD

The domain used for experimentation was the ROBOTS domain from the EMERALD system which integrates several machine-learning programs. Specifically, the concept C1 from the ROBOTS domain was the target concept used (Wnek, Sarma, Wahab, and Michalski, 1990). The concept C1 is described as follows:

Head is round and jacket is red or head is square and is holding a balloon.

The C1 concept description was the basis for the positive and negative examples used as the input training set (Wnek, 1991)

for experimentation with the AQ15 program and with the AQ17-HCI module.

The five initial positive examples or positive events for input to AQ15 and AQ17 were--

#	hs	bs	sm	ho	jc	ti
1	round	octagonal	yes	sword	yellow	no
2	square	octagonal	yes	sword	yellow	no
3	octagonal	square	no	sword	green	no
4	octagonal	round	yes	sword	blue	yes
5	octagonal	octagonal	no	balloon	green	no

The eleven initial negative examples or negative events were--

#	hs	bs	sm	ho	jc	ti
1	round	octagonal	yes	sword	yellow	no
2	square	octagonal	yes	sword	yellow	no
3	octagonal	square	no	sword	green	no
4	octagonal	round	yes	sword	blue	yes
5	octagonal	octagonal	no	balloon	green	no
6	octagonal	round	no	balloon	blue	no
7	octagonal	square	yes	flag	red	no
8	octagonal	round	no	flag	green	no
9	round	octagonal	no	flag	blue	yes
10	round	octagonal	no	flag	green	yes
11	square	round	yes	flag	yellow	yes

where attribute hs stands for head shape, bs for body shape, sm for smiling, ho for holding, jc for jacket color, and ti for tie.

Basically, the seven steps of the AQ17-HCI method by Wnek and Michalski (1991) are as follows:

1. Input the training set to induce decision rules (AQ15).
2. Analyze the output hypotheses to remove irrelevant attributes.

3. Generate a new attribute per class from a subset of highest quality rules.
4. Modify the training examples (remove irrelevant attributes and add new ones).
5. Induce new output hypotheses from the modified training set (AQ15).
6. Repeat the process until a predefined accuracy is exceeded.
7. Induce new output hypotheses from the modified training set (AQ15).

3 IMPLEMENTATION

The complete input data files containing incremental positive and negative training examples as given to AQ15 and AQ17-HCI running on a Sun SPARC station 1+ are attached as Figures A1 through A5 under Appendix A (Data) to this project report.

The data structures for AQ15 and AQ17 are prescribed by the user's guide (Hong, Mozetic, and Michalski, 1986). The system requires data to be organized as relational tables for proper input. Of two mandatory tables, the parameters table is needed to specify how the system should construct the learned rules. The other mandatory table is the variables table which declares the names of the attributes (hs, bs, sm, ho, jc, and ti) to be used for the learned rules and to describe events or training examples of the target concept. Optional domaintypes

and -names tables may be used to support the variables table. The domaintypes tables specify the domaintypes names (l2, l3, ho, and jc), the domain types (nominal, linear, structured, etc.), and number of levels for each domaintypes name (2, 3, 4, etc.) used for the attributes. The -names tables define the level numbers (0, 1, and 2 for three levels: 0 for holding sword, 1 for balloon, and 2 for flag for example) and the names for the values of the attributes (sword, balloon, and flag for attribute ho for example). The -events tables contain the training set of positive and negative examples as positive-events and negative-events for input to the system. The column headings correspond to the variables table names (attribute names hs, bs, sm, ho, jc, and ti) and the row values are those prescribed by the -names tables (red, yellow, green, and blue for example for attribute name jc). Following is the complete input file robots3.inp, with which the systems learned the target concept C1 correctly, containing the tables as described:

```

title
# text
1 "Input: robots3.inp"
2 "Output: robots3.out (AQ15) and r3.o (AQ17)"
3 "Seven positive examples and 13 negative examples:"

parameters
mode ambig trim wts echo
ic pos mini cpx tpcdvne

domaintypes
name type levels
l2 nom 2
l3 nom 3
ho nom 3
jc nom 4

```


12-names
 value name
 0 no
 1 yes

13-names
 value name
 0 round
 1 square
 2 octagonal

ho-names
 value name
 0 sword
 1 balloon
 2 flag

jc-names
 value name
 0 red
 1 yellow
 2 green
 3 blue

variables
 # type levels cost name
 1 nom 3 1.00 hs.13
 2 nom 3 1.00 bs.13
 3 nom 2 1.00 sm.12
 4 nom 3 1.00 ho.ho
 5 nom 4 1.00 jc.jc
 6 nom 2 1.00 ti.12

Pos-events

#	hs	bs	sm	ho	jc	ti
1	round	round	yes	sword	red	no
2	round	square	yes	sword	red	yes
3	round	octagonal	yes	balloon	red	yes
4	square	square	yes	balloon	red	yes
5	square	square	no	balloon	green	yes
6	round	round	yes	flag	red	yes
7	square	round	yes	balloon	blue	yes

Neg-events

#	hs	bs	sm	ho	jc	ti
1	round	octagonal	yes	sword	yellow	no
2	square	octagonal	yes	sword	yellow	no
3	octagonal	square	no	sword	green	no
4	octagonal	round	yes	sword	blue	yes
5	octagonal	octagonal	no	balloon	green	no
6	octagonal	round	no	balloon	blue	no
7	octagonal	square	yes	flag	red	no
8	octagonal	round	no	flag	green	no
9	round	octagonal	no	flag	blue	yes

10	round	octagonal	no	flag	green	yes
11	square	round	yes	flag	yellow	yes
12	square	round	yes	sword	red	yes
13	round	square	yes	balloon	green	yes

The -events tables contain the seven positive examples and 13 negative examples that the systems used successfully to learn the target concept accurately.

As I mentioned earlier, to investigate the hypothesis-driven constructive-induction (HCI) method in AQ17 (AQ17-HCI) and its supporting selective-induction learning algorithm AQ15, and to experiment with them, I used target concept C1 from the ROBOTS domain. I previously described the target concept as--

Head is round and jacket is red or

head is square and is holding a balloon.

In disjunctive normal form (DNF), each conjunction of attribute-value pairs is a disjunct. The target concept is described by the following two disjuncts:

[HS = round] & [JC = red] or

[HS = square] & [HO = balloon]

4 EXPERIMENTS AND RESULTS

I started experimentation with the following initial training set of five positive and 11 negative examples (Wnek, 1991) as I presented during discussion of the method:

Pos-events

#	hs	bs	sm	ho	jc	ti
1	round	round	yes	sword	red	no
2	round	square	yes	sword	red	yes
3	round	octagonal	yes	balloon	red	yes
4	square	square	yes	balloon	red	yes
5	square	square	no	balloon	green	yes

Neg-events

#	hs	bs	sm	ho	jc	ti
1	round	octagonal	yes	sword	yellow	no
2	square	octagonal	yes	sword	yellow	no
3	octagonal	square	no	sword	green	no
4	octagonal	round	yes	sword	blue	yes
5	octagonal	octagonal	no	balloon	green	no
6	octagonal	round	no	balloon	blue	no
7	octagonal	square	yes	flag	red	no
8	octagonal	round	no	flag	green	no
9	round	octagonal	no	flag	blue	yes
10	round	octagonal	no	flag	green	yes
11	square	round	yes	flag	yellow	yes

In DNF, this subset of the instance space is described by one disjunct for each example. Each is represented as a conjunction of attribute-value pairs (a complex). This set of examples is used by the learner to learn the target concept.

4.1 SELECTIVE-INDUCTION SYSTEM AQ15 AND DIAV

To induce a decision rule or cover (disjunction of complexes) as a selective hypothesis, I created the input file containing the initial training set and the other pertinent data needed by AQ15: description of the ROBOTS domain and parameters specifying how the rules should be formed. The file was the same as the one I introduced during the implementation discussion, but with fewer examples. Then, I gave the input file to AQ15 and redirected its output to an output file which included the initial selective hypothesis or initial learned

concept as a decision rule, a partial cover since it did not cover all of the positive examples.

To visualize the process as it evolved, I used the diagrammatic-visualization (DIAV) technique on the Macintosh IIX computer. First, I input the C1 target-concept file as depicted by Figure B1 under Appendix B (Results), which produced the target-concept diagram labeled Figure B2. Next, I output a diagram depicting the learned concept superimposed on the target concept (Wnek et al, 1990). The diagram also depicted learned-concept cells that were overgeneralized (errors of commission) and cells that were not learned (errors of omission). The error rate from this initial training set was 19.44%. The initial AQ15 output file is labeled Figure B3 and the initial DIAV diagram of the incremental learning process is attached as Figure B4, both under Appendix B (Results).

Also, to improve the performance of the system, I added training data based on the overgeneralized and unlearned cells depicted by the DIAV diagram. The heuristic I used to guide the example-generation process was to add one or two positive examples to the training set over DIAV cells among those that were unlearned. To continue this process, I would add one or two negative examples to the training set over DIAV cells among those that were overgeneralized. For the subsequent steps in the selective-induction process, I used this heuristic to decide which training examples I should add.

To improve the learned concept, I added two positive examples among the cells depicted as unlearned. I ran the new input file against AQ15 with redirected output to a new output file. DIAV produced a new diagram with overgeneralized and unlearned cells. The new output file and new DIAV diagram may be found respectively as Figures B5 and B6 under Appendix B (Results). This intermediate selective hypothesis or decision rule represented an intermediate learned concept, a partial cover, with an increased error rate of 29.17%. This suggests that adding training examples does not always improve the performance accuracy of the learning system.

Next, I added a negative example to one of the cells shown as overgeneralized and repeated the AQ15 process which produced an improved learned concept depicted by DIAV with only overgeneralized cells remaining. This partial cover had an improved error rate of 8.33%. The output file and its corresponding DIAV diagram are under Appendix B (Results) as Figures B7 and B8 respectively.

Lastly, I added another negative example to one of the overgeneralized cells. With this input in file robots3.inp, AQ15 created a final hypothesis as the learned concept (Figure B9) with no overgeneralized cells nor unlearned cells in the DIAV diagram (Figure B10 under Appendix B (Results)). This final hypothesis, the learned concept, represented the final decision rule or cover satisfied by all positive examples and by no negative examples, a complete and consistent cover.

This learned concept matched the target concept with a 0% error rate. The learned and target concepts had converged.

To experiment further, I added a negative example to a cell in the correct learned-concept space. This resulted in a hypothesis that no longer matched the target-concept description, AQ15 unable to deal with "noise."

4.2 AQ17-HCI WITH INCREMENTAL EXAMPLES

During the next phase of the project, I investigated the constructive-induction capabilities of AQ17-HCI and how AQ17 learns by removing irrelevant attributes after analyzing the initial and subsequent consecutive hypotheses created by the selective-induction system AQ15. It is this sequence of the process that gives the method the name of "hypothesis-driven constructive induction" (HCI) (Wnek and Michalski, 1991). The attribute-removal module of AQ17 is known as AQ17(HCI/ar).

In analyzing the output hypotheses from AQ17(HCI/ar) in every output file, one observes that, unlike AQ15, based on an attribute-utility measure, AQ17 establishes the attributes that occur in the positive output hypothesis as relevant. AQ17 discards the other attributes as irrelevant. For that reason, only the relevant attributes, *hs*, *ho*, and *jc*, appear in the negative output hypothesis.

Another observation is that, in comparing the positive output hypothesis with the negative output hypothesis, certain combinations of attribute values cannot exist in the positive

output hypothesis since they may occur in the negative output hypothesis. If so, such combinations of attribute values in the positive output hypothesis may be modified into several forms that have removed one of the attributes and, thus, no longer occur in the negative output hypothesis. Applying the forms in as many combined ways as possible suggests new possible rules (new combinations of attribute values). Since some of the new rules cover negative examples, they overgeneralize the concept and are no longer considered. Of those that remain, the rules with disjoint covers of the whole positive example set are proposed as a new positive hypothesis.

The new positive hypothesis rules may be simplified by dropping one condition while ensuring that coverage remains consistent regarding negative examples. The simplified rules that emerge still have disjoint covers of the whole positive example set and become the final hypothesis for the positive class (Wnek, 1991). This final hypothesis matches exactly the C1 target-concept description as shown by the analysis of output file r.o under Appendix B (Results) as Figure B11.

Analysis of the next output file from AQ17 disclosed a variation from the previous positive output hypothesis: one of the rules in the positive output hypothesis is the same as one of the rules in the C1 target-concept description.

As shown by the analysis of output file r1.o attached under Appendix B (Results) as Figure B12, the process starts in the

same manner as before: certain combinations of attribute values cannot exist in the positive output hypothesis since they happen to occur in the negative output hypothesis.

After the combinations are modified into several forms that remove one of the attributes, thus precluding the occurrence of the combinations in the negative output hypothesis, new possible rules (new combinations of attribute values) are suggested by applying the forms in as many combined ways as possible. As before, those new rules that cover negative examples overgeneralize the concept and are no longer considered.

Only one of the remaining rules and the positive output hypothesis rule that already matches a rule in the C1 target-concept description together cover the whole positive example set. Thus, these two rules are proposed as a new positive hypothesis.

The simplification process, as before, ensures that coverage remains consistent regarding negative examples. After dropping possible solutions which overgeneralize the concept, consequently covering negative examples, it becomes apparent that the remaining solution may be further simplified. The new simplification process evident at this point is one in which one attribute-value pair is removed while ensuring that coverage remains consistent regarding negative examples. Of two possible solutions, the one with greater coverage is selected. This simplified rule and the positive output

hypothesis rule that already matches a rule in the C1 target-concept description now have disjoint covers of the whole positive example set. They become the final hypothesis for the positive class. As before, this final hypothesis matches exactly the C1 target-concept description.

The attached analysis of output file r2.o under Appendix B (Results) as Figure B13 discloses that the combinations of attribute values in the positive output hypothesis no longer occur in the negative output hypothesis. As in the previous positive output hypothesis, one of the rules is the same as one of the rules in the C1 target-concept description. In fact, it is exactly the same rule as before: [hs=round] [jc=red]. The other rule in the positive output hypothesis is the same as the simplified rule that resulted in the previous analysis after the first simplification dropped one condition: [hs=round,square] [ho=balloon].

As before, simplification may be obtained by dropping one attribute-value pair while ensuring that coverage remains consistent regarding negative examples. The simplification process continues exactly as in the previous case. It follows that the simplified rule thus obtained is the same as before: [hs=square] [ho=balloon].

Again, this rule and the rule that already matches a rule in the C1 target-concept description have disjoint covers of the whole positive example set. As such, they become the final

hypothesis for the positive class, which matches exactly the C1 target-concept description.

Analysis of output file r3.o as Figure B14 under Appendix B (Results) discloses that, with a training set of seven positive examples and 13 negative examples, the positive output hypothesis matches exactly the C1 target-concept description. The positive output hypothesis describes the learned concept which has converged into the target-concept description. The method used in this analysis, which I later describe as the "altered positive output hypothesis" method, is not applicable on this positive output hypothesis since there is no overlap in the hypotheses nor is simplification necessary. In fact, it is not applicable since AQ17-HCI has output a match to the target-concept description through its final iteration of the incremental addition of training examples.

In output file r4.o, found as Figure B15 under Appendix B (Results), we find that the combinations of attribute values in the positive output hypothesis do not occur in the negative output hypothesis. We also find that the positive output hypothesis rules cannot be simplified, and yet only one rule matches one of the rules in the target-concept description. The other rule in the positive output hypothesis does not match the remaining rule in the target-concept description because, if it did, the rule would cover negative example 14. Simply stated, the problem is that we purposely added negative

example 14 with attribute values within the target-concept space, thus testing AQ17's reaction to "noise." AQ17's output hypotheses seem correct for the training set given to it. However, AQ17 was unable to learn the correct concept in the presence of "noise."

4.3 COMPARISON OF THE APOH METHOD AND THE VS* METHOD

The method I have just described, which I call the "altered positive output hypothesis" (APOH) method, as illustrated by Figure 1, uses what I refer to as "hypothesis-driven deduction" (HDD). I found that there is a similarity between the APOH method and the VS* (version space with the STAR) learning method (Wnek, 1992) which is based on a three-step inductive-generalization process.

Both methods initially may encounter an overlap in coverage by positive and negative rules. As I discussed previously, the APOH method alters the positive output hypothesis by modifying some attribute-value combinations and simplifying the rules in the new proposed positive hypothesis, or by simplifying the original positive output hypothesis. The method starts with the modification of attribute-value combinations or rules in the positive output hypothesis whenever the same combinations occur in the negative output hypothesis. Such overlapping rules are depicted by the black cells in Figure 2. Modification consists of forming new combinations of attribute values in the positive output hypothesis that no longer occur in the negative output hypothesis.

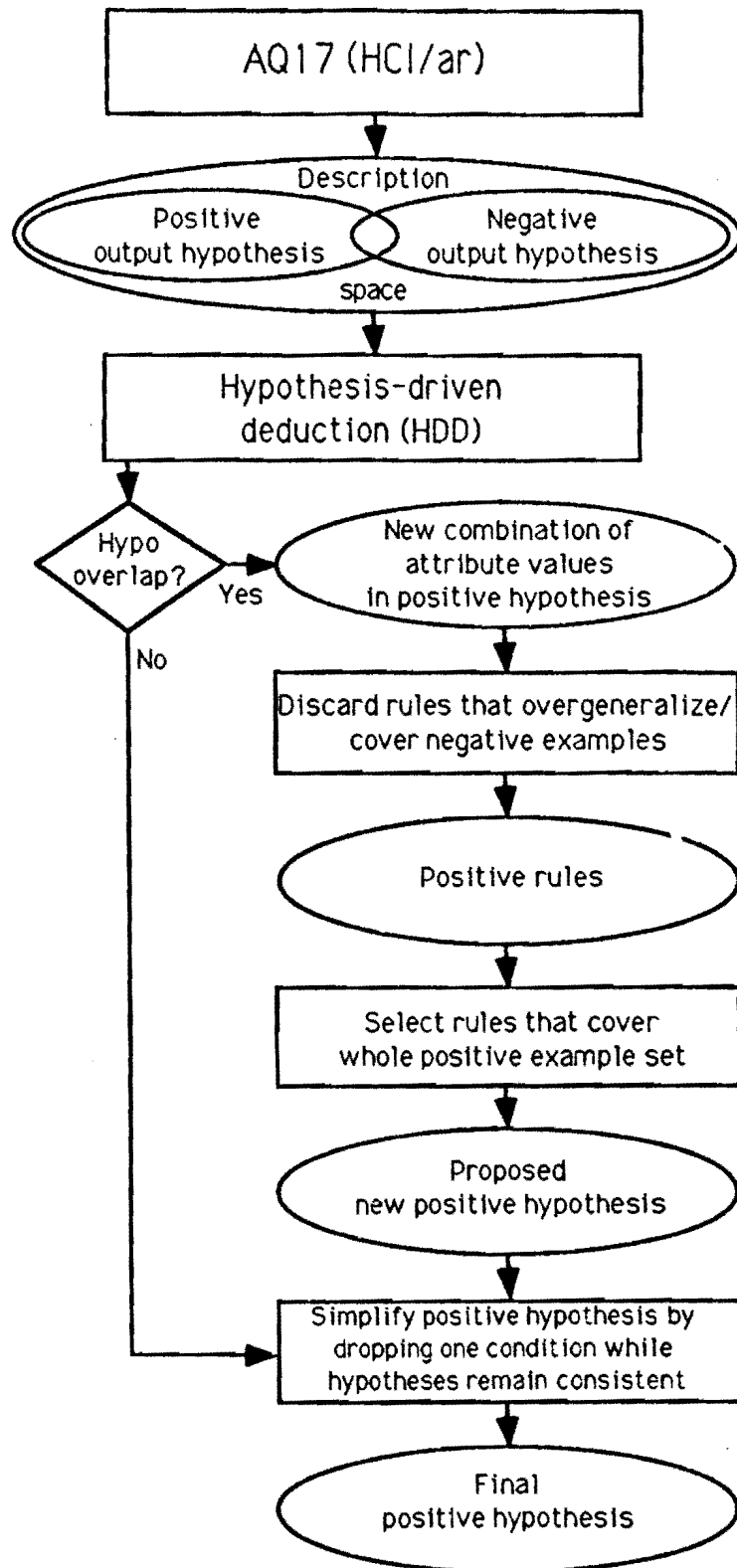
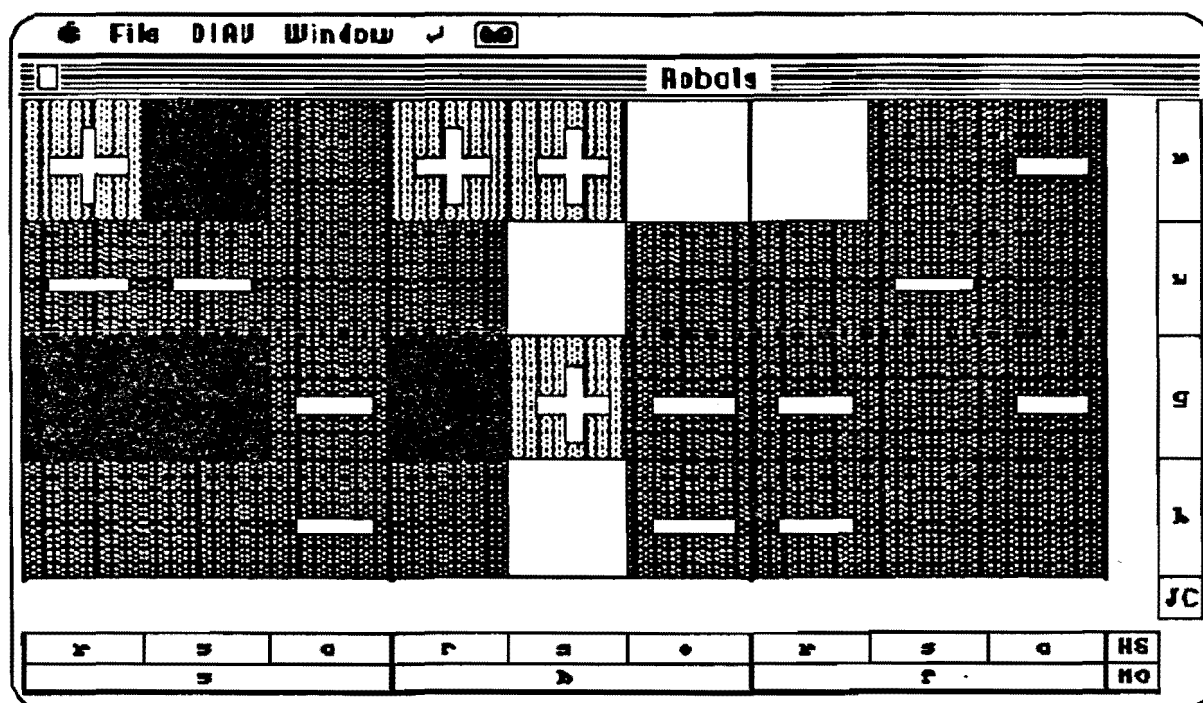




Figure 1. Altered positive output hypothesis (APOH) method.



 Positive output hypothesis

 Negative output hypothesis

 No hypothesis

 Both hypotheses (overlap)

 Positive example

 Negative example

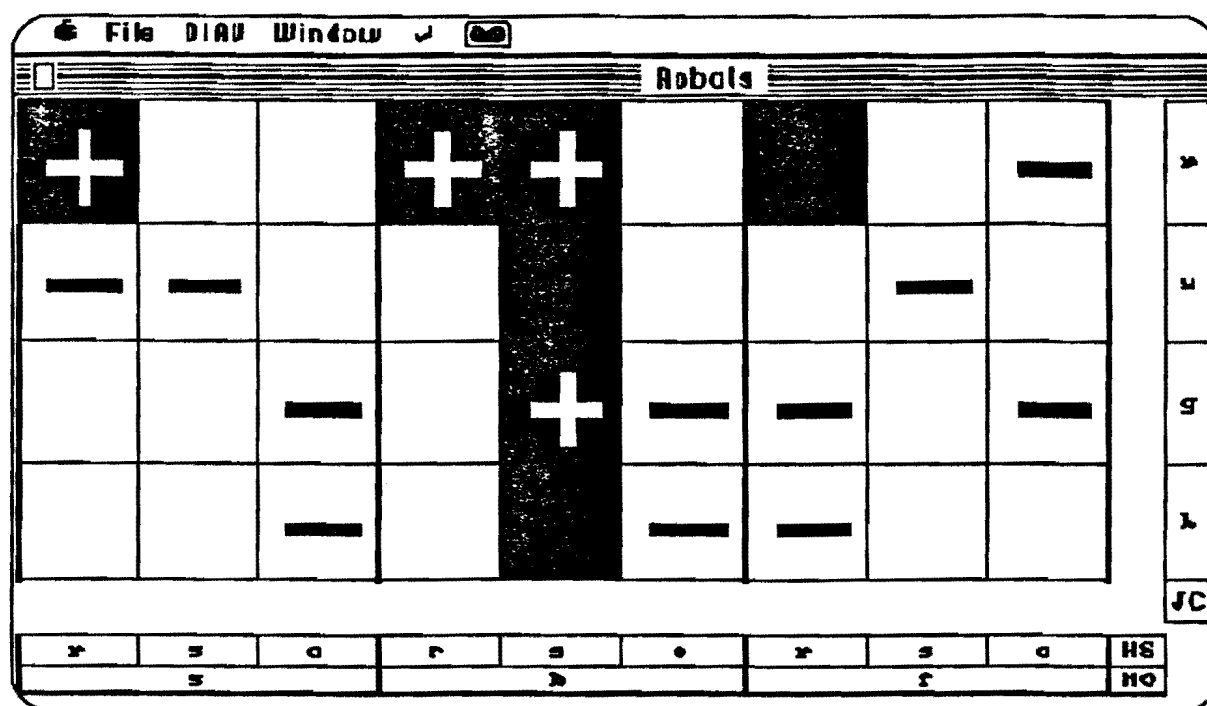
Figure 2. Positive and negative concepts overlap.

As was the case with Wnek's version space concept definitions (Wnek, 1992), the white cells (instances) are not covered by either hypothesis and the black cells (instances) are covered by the positive and by the negative output hypotheses, representing an overlap. Attribute-value combinations that occur in the negative output hypothesis should not occur also in the positive output hypothesis.

Also, as in the VS* method (Wnek, 1992), the positive concept or positive output hypothesis covers the cells depicted in light gray, whereas the negative concept or negative output hypothesis covers the cells depicted in dark gray. Positive and negative examples occur respectively among those cells, thus appropriately covered by their respective hypothesis.

Whether the output hypotheses overlap or not, the APOH method also simplifies the positive hypothesis, successfully reducing it to a learned concept that matches exactly the C1 target concept as depicted by Figure 3.

In the VS* method, the final inductive learning step yielded rules that correctly matched the target-concept description of the CIRCLE (Wnek, 1992). The APOH method altered the positive output hypothesis as previously described, and the final "hypothesis-driven deduction" step yielded a final hypothesis for the positive class which matched exactly the C1 target-concept description. As was with the VS* algorithm, not all white instances in the APOH algorithm were generalized to the final concept description.



Positive example



Negative example



C1 target concept and final hypothesis for positive class

Figure 3. Final positive hypothesis by APOH method matches C1 target concept.

The VS* method uses three inductive generalization learning steps. The first one builds a version space with 'the most specific classification rules for the "positive" concept, and the most general classification rules for the "negative" concept,' as presented by Wnek (1992). As happened with the APOH method, the white cells in the concept diagram defining the version space are not covered by either rule. Both positive and negative rules cover the black instances, representing an overlap. During the second step, AQ17's hypothesis-driven constructive induction (HCI/v) transforms the version space and creates new training examples. The final step yields a concept description that matches the target concept of the CIRCLE (Wnek, 1992).

Sharing some similarities, two different methods obtained an equivalent result. The final hypothesis for the positive class matched exactly the target-concept description: rules describing the CIRCLE in the case of the VS* learning method or rules describing the C1 concept in the case of the APOH method.

5 CONCLUSION

Experimentation with hypothesis-driven constructive induction in AQ17 began with a training set of positive and negative examples from the ROBOTS domain. Each input file included relational tables containing data about the object attributes such as attribute names, domain types and levels, and attribute values; and parameters telling the learning system

how to form the learned rules. Additionally, each file included event tables that held the positive and negative events or instances of the concept to be learned. As incremental training example sets were input to AQ15 as part of sequential input files, the learning system normally improved its learned concept as it came closer and closer to matching the target-concept description exactly. In the final iteration, the selective-induction system AQ15 had learned the target concept correctly. The sequence of output hypotheses could be appreciated better with the aid of the diagrammatic-visualization (DIAV) technique on the Macintosh IIX computer. DIAV diagrams provided a graphical comparison between the learned concept and the target concept at each stage. The errors of commission were depicted by cells that were overgeneralized, the system having overlearned, whereas the errors of omission were depicted by cells within the target concept that had not been learned.

Analysis of output hypotheses from the attribute-removal module AQ17(HCI/ar) with the same set of incremental input files indicated that the attributes appearing in the positive output hypothesis were designated by AQ17 as relevant based on an attribute-utility measure. The other attributes, now irrelevant, were no longer used. For this reason, only the relevant attributes existed in the negative output hypothesis as well. In the initial hypotheses, this condition led to an overlap of output hypotheses. That is, some combinations of attribute values in the positive output hypothesis may also

occur in the negative output hypothesis. To eliminate the inconsistency in such a learned concept, I used a method described by Wnek (1991). I have referred to the method as the "altered positive output hypothesis" (APOH) method since concept consistency was obtained by forming new combinations of attribute values in the positive hypothesis which led to new rules from which a new proposed hypothesis emerged. A simplification process followed which produced a positive hypothesis that matched the target-concept description exactly. I have called the deductive inference used in the APOH method "hypothesis-driven deduction" (HDD). Using HDD, the APOH method led to a correctly learned concept without incremental input of training sets.

A comparison of the APOH method as described by Wnek (1991) with the VS* method proposed by Wnek (1992) disclosed similarities that included a final positive output hypothesis in each case which was complete and consistent, covering all positive examples and no negative ones. As did the APOH method, the VS* method initially encountered an inconsistent learned concept, an overlap in output hypotheses. VS* countered by building a version space with the most specific rules for the positive hypothesis and the most general rules for the negative hypothesis. Then it applied AQ17-HCI/v constructive transformations to transform the version space and to create new training examples. The VS* method progressed through its third step of its inductive-generalization process to yield a learned concept which

REFERENCES

- (1) Hong, J., Mozetic, I., and Michalski, R. S., "AQ15: Incremental Learning of Attribute-Based Descriptions from Examples. The Method and User's Guide," Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, 1986.
- (2) Wnek, J., Sarma, J., Wahab, A., and Michalski, R. S., "Comparing Learning Paradigms via Diagrammatic Visualization: A Case Study in Single Concept Learning Using Symbolic, Neural Net, and Genetic Algorithm Methods," Center for Artificial Intelligence, George Mason University, Fairfax, VA, 1990.
- (3) Wnek, J., Robots/C1/clra.doc File, Artificial Intelligence Center, George Mason University, Fairfax, VA, September 20, 1991.
- (4) Wnek, J. and Michalski, R. S., "Hypothesis-driven Constructive Induction in AQ17: A Method and Experiments," in Proceedings of the IJCAI-91 Workshop on Evaluating and Changing Representation in Machine Learning, K. Morik, F. Bergadano, and W. Buntine (Eds.), pp. 13-22, Sydney, Australia, 1991.
- (5) Wnek, J., "Transformation of Version Space with Constructive Induction: The VS* Algorithm," Reports of Machine Learning and Inference Laboratory, MLI 92-02, Artificial Intelligence Center, George Mason University, Fairfax, VA, 1992.

APPENDIX A: DATA

Figure A1. Input file robots.inp for AQ15 and AQ17-HCI.
(continued)

Pos-events

#	hs	bs	sm	ho	jc	ti
1	round	round	yes	sword	red	no
2	round	square	yes	sword	red	yes
3	round	octagonal	yes	balloon	red	yes
4	square	square	yes	balloon	red	yes
5	square	square	no	balloon	green	yes

Neg-events

#	hs	bs	sm	ho	jc	ti
1	round	octagonal	yes	sword	yellow	no
2	square	octagonal	yes	sword	yellow	no
3	octagonal	square	no	sword	green	no
4	octagonal	round	yes	sword	blue	yes
5	octagonal	octagonal	no	balloon	green	no
6	octagonal	round	no	balloon	blue	no
7	octagonal	square	yes	flag	red	no
8	octagonal	round	no	flag	green	no
9	round	octagonal	no	flag	blue	yes
10	round	octagonal	no	flag	green	yes
11	square	round	yes	flag	yellow	yes

APPENDIX A: DATA

Figure A2. Input file robots1.inp for AQ15 and AQ17-HCI.

```

title
# text
1 "Input: robots1.inp"
2 "Output: robots1.out (AQ15) and r1.o (AQ17)"
3 "Seven positive examples and 11 negative examples:"

parameters
mode ambig trim wts echo
ic pos mini cpx tpcdvne

domaintypes
name type levels
12 nom 2
13 nom 3
ho nom 3
jc nom 4

12-names
value name
0 no
1 yes

13-names
value name
0 round
1 square
2 octagonal

ho-names
value name
0 sword
1 balloon
2 flag

jc-names
value name
0 red
1 yellow
2 green
3 blue

variables
# type levels cost name
1 nom 3 1.00 hs.13
2 nom 3 1.00 bs.13
3 nom 2 1.00 sm.12
4 nom 3 1.00 ho.ho
5 nom 4 1.00 jc.jc
6 nom 2 1.00 ti.12

```

APPENDIX A: DATA

Figure A2. Input file robots1.inp for AQ15 and AQ17-HCI.
(continued)

Pos-events

#	hs	bs	sm	ho	jc	ti
1	round	round	yes	sword	red	no
2	round	square	yes	sword	red	yes
3	round	octagonal	yes	balloon	red	yes
4	square	square	yes	balloon	red	yes
5	square	square	no	balloon	green	yes
6	round	round	yes	flag	red	yes
7	square	round	yes	balloon	blue	yes

Neg-events

#	hs	bs	sm	ho	jc	ti
1	round	octagonal	yes	sword	yellow	no
2	square	octagonal	yes	sword	yellow	no
3	octagonal	square	no	sword	green	no
4	octagonal	round	yes	sword	blue	yes
5	octagonal	octagonal	no	balloon	green	no
6	octagonal	round	no	balloon	blue	no
7	octagonal	square	yes	flag	red	no
8	octagonal	round	no	flag	green	no
9	round	octagonal	no	flag	blue	yes
10	round	octagonal	no	flag	green	yes
11	square	round	yes	flag	yellow	yes

APPENDIX A: DATA

Figure A3. Input file robots2.inp for AQ15 and AQ17-HCI.

```

title
# text
1 "Input: robots2.inp"
2 "Output: robots2.out (AQ15) and r2.o (AQ17)"
3 "Seven positive examples and 12 negative examples:"

parameters
mode ambig trim wts echo
ic pos mini cpx tpcdvne

domaintypes
name type levels
12 nom 2
13 nom 3
ho nom 3
jc nom 4

12-names
value name
0 no
1 yes

13-names
value name
0 round
1 square
2 octagonal

ho-names
value name
0 sword
1 balloon
2 flag

jc-names
value name
0 red
1 yellow
2 green
3 blue

variables
# type levels cost name
1 nom 3 1.00 hs.13
2 nom 3 1.00 bs.13
3 nom 2 1.00 sm.12
4 nom 3 1.00 ho.ho
5 nom 4 1.00 jc.jc
6 nom 2 1.00 ti.12

```

APPENDIX A: DATA

Figure A3. Input file robots2.inp for AQ15 and AQ17-HCI.
(continued)

Pos-events

#	hs	bs	sm	ho	jc	ti
1	round	round	yes	sword	red	no
2	round	square	yes	sword	red	yes
3	round	octagonal	yes	balloon	red	yes
4	square	square	yes	balloon	red	yes
5	square	square	no	balloon	green	yes
6	round	round	yes	flag	red	yes
7	square	round	yes	balloon	blue	yes

Neg-events

#	hs	bs	sm	ho	jc	ti
1	round	octagonal	yes	sword	yellow	no
2	square	octagonal	yes	sword	yellow	no
3	octagonal	square	no	sword	green	no
4	octagonal	round	yes	sword	blue	yes
5	octagonal	octagonal	no	balloon	green	no
6	octagonal	round	no	balloon	blue	no
7	octagonal	square	yes	flag	red	no
8	octagonal	round	no	flag	green	no
9	round	octagonal	no	flag	blue	yes
10	round	octagonal	no	flag	green	yes
11	square	round	yes	flag	yellow	yes
12	square	round	yes	sword	red	yes

APPENDIX A: DATA

Figure A4. Input file robots3.inp for AQ15 and AQ17-HCI.

```

title
# text
1 "Input: robots3.inp"
2 "Output: robots3.out (AQ15) and r3.o (AQ17)"
3 "Seven positive examples and 13 negative examples:"

parameters
mode ambig trim wts echo
ic pos mini cpx tpcdvne

domaintypes
name type levels
12 nom 2
13 nom 3
ho nom 3
jc nom 4

12-names
value name
0 no
1 yes

13-names
value name
0 round
1 square
2 octagonal

ho-names
value name
0 sword
1 balloon
2 flag

jc-names
value name
0 red
1 yellow
2 green
3 blue

variables
# type levels cost name
1 nom 3 1.00 hs.13
2 nom 3 1.00 bs.13
3 nom 2 1.00 sm.12
4 nom 3 1.00 ho.ho
5 nom 4 1.00 jc.jc
6 nom 2 1.00 ti.12

```

APPENDIX A: DATA

Figure A4. Input file robots3.inp for AQ15 and AQ17-HCI.
(continued)

Pos-events

#	hs	bs	sm	ho	jc	ti
1	round	round	yes	sword	red	no
2	round	square	yes	sword	red	yes
3	round	octagonal	yes	balloon	red	yes
4	square	square	yes	balloon	red	yes
5	square	square	no	balloon	green	yes
6	round	round	yes	flag	red	yes
7	square	round	yes	balloon	blue	yes

Neg-events

#	hs	bs	sm	ho	jc	ti
1	round	octagonal	yes	sword	yellow	no
2	square	octagonal	yes	sword	yellow	no
3	octagonal	square	no	sword	green	no
4	octagonal	round	yes	sword	blue	yes
5	octagonal	octagonal	no	balloon	green	no
6	octagonal	round	no	balloon	blue	no
7	octagonal	square	yes	flag	red	no
8	octagonal	round	no	flag	green	no
9	round	octagonal	no	flag	blue	yes
10	round	octagonal	no	flag	green	yes
11	square	round	yes	flag	yellow	yes
12	square	round	yes	sword	red	yes
13	round	square	yes	balloon	green	yes

APPENDIX A: DATA

Figure A5. Input file robots4.inp for AQ15 and AQ17-HCI.

```

title
# text
1 "Input: robots4.inp"
2 "Output: robots4.out (AQ15) and r4.o (AQ17)"
3 "Seven positive examples and 14 negative examples:"

parameters
mode ambig trim wts echo
ic pos mini cpx tpcdvne

domaintypes
name type levels
12 nom 2
13 nom 3
ho nom 3
jc nom 4

12-names
value name
0 no
1 yes

13-names
value name
0 round
1 square
2 octagonal

ho-names
value name
0 sword
1 balloon
2 flag

jc-names
value name
0 red
1 yellow
2 green
3 blue

variables
# type levels cost name
1 nom 3 1.00 hs.13
2 nom 3 1.00 bs.13
3 nom 2 1.00 sm.12
4 nom 3 1.00 ho.ho
5 nom 4 1.00 jc.jc
6 nom 2 1.00 ti.12

```

APPENDIX A: DATA

Figure A5. Input file robots4.inp for AQ15 and AQ17-HCI.
(continued)

Pos-events

#	hs	bs	sm	ho	jc	ti
1	round	round	yes	sword	red	no
2	round	square	yes	sword	red	yes
3	round	octagonal	yes	balloon	red	yes
4	square	square	yes	balloon	red	yes
5	square	square	no	balloon	green	yes
6	round	round	yes	flag	red	yes
7	square	round	yes	balloon	blue	yes

Neg-events

#	hs	bs	sm	ho	jc	ti
1	round	octagonal	yes	sword	yellow	no
2	square	octagonal	yes	sword	yellow	no
3	octagonal	square	no	sword	green	no
4	octagonal	round	yes	sword	blue	yes
5	octagonal	octagonal	no	balloon	green	no
6	octagonal	round	no	balloon	blue	no
7	octagonal	square	yes	flag	red	no
8	octagonal	round	no	flag	green	no
9	round	octagonal	no	flag	blue	yes
10	round	octagonal	no	flag	green	yes
11	square	round	yes	flag	yellow	yes
12	square	round	yes	sword	red	yes
13	round	square	yes	balloon	green	yes
14	round	square	yes	flag	red	no

APPENDIX B: RESULTS

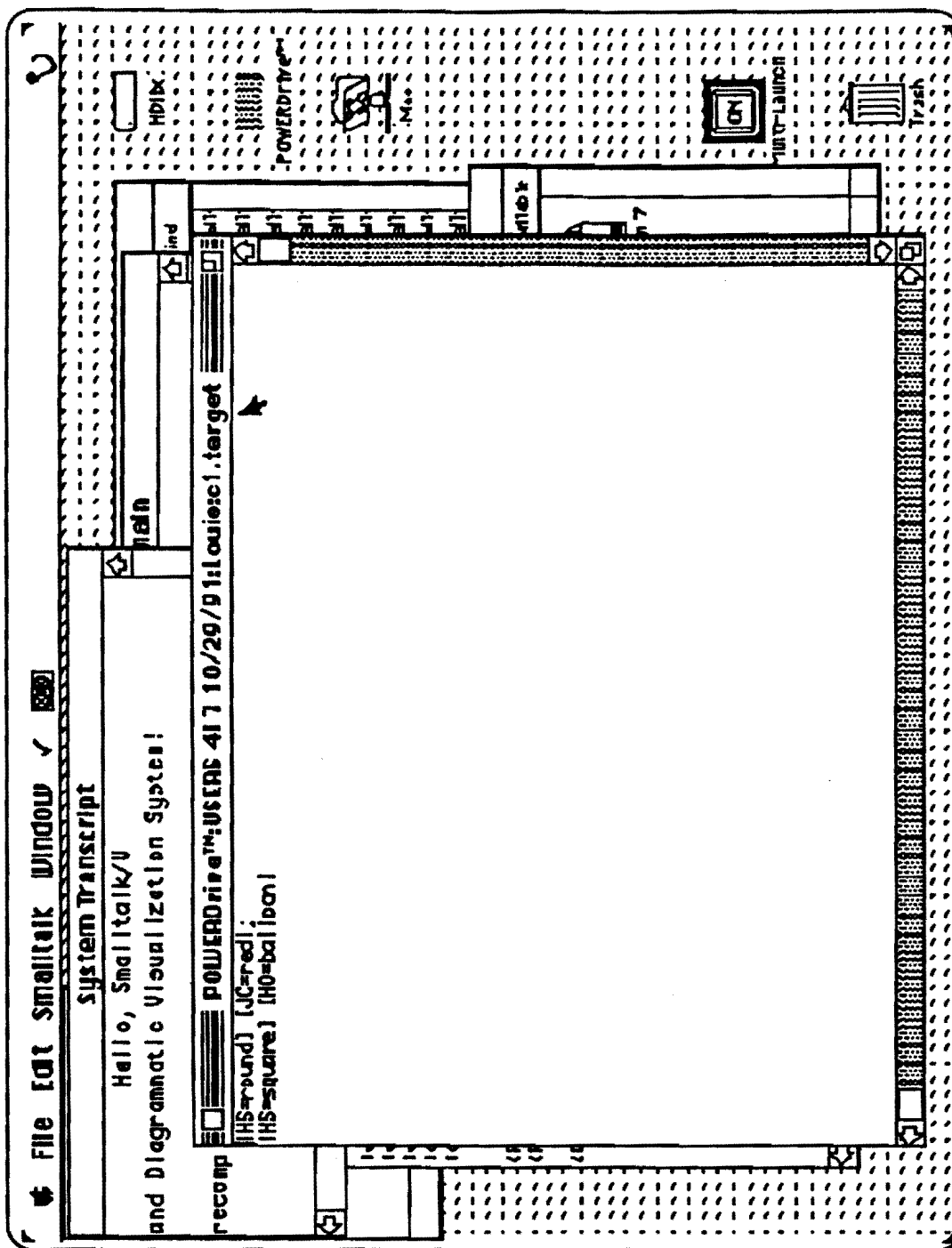
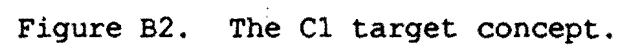


Figure B1. Target-concept file for DIAV



APPENDIX B: RESULTS

Figure B3. Output file robots.out from AQ15.

```

title
# text
1 "Input: robots.inp"
2 "Output: robots.out (AQ15) and r.o (AQ17)"
3 "Five positive examples and 11 negative examples:"

parameters
run mode  ambig  trim  maxstar  echo  criteria  operata
1  ic      pos    mini   10      tpcdvne  default  off

default-criteria
# criterion  tolerance
1  maxnew    0.00
2  minsel    0.00

domaintypes
type  levels  cost  name
nom   2       1.00  l2
nom   3       1.00  l3
nom   3       1.00  ho
nom   4       1.00  jc

variables
# type  levels  cost  name
1  nom    3     1.00  hs.l3
2  nom    3     1.00  bs.l3
3  nom    2     1.00  sm.l2
4  nom    3     1.00  ho.ho
5  nom    4     1.00  jc.jc
6  nom    2     1.00  ti.l2

l2-names
value  name
0      no
1      yes

l3-names
value  name
0      round
1      square
2      octagonal

```

APPENDIX B: RESULTS

Figure B3. Output file robots.out from AQ15.
(continued)

ho-names

value	name
0	sword
1	balloon
2	flag

jc-names

value	name
0	red
1	yellow
2	green
3	blue

Pos-events

#	hs	bs	sm	ho	jc	ti
1	round	round	yes	sword	red	no
2	round	square	yes	sword	red	yes
3	round	octagonal	yes	balloon	red	yes
4	square	square	yes	balloon	red	yes
5	square	square	no	balloon	green	yes

Neg-events

#	hs	bs	sm	ho	jc	ti
1	round	octagonal	yes	sword	yellow	no
2	square	octagonal	yes	sword	yellow	no
3	octagonal	square	no	sword	green	no
4	octagonal	round	yes	sword	blue	yes
5	octagonal	octagonal	no	balloon	green	no
6	octagonal	round	no	balloon	blue	no
7	octagonal	square	yes	flag	red	no
8	octagonal	round	no	flag	green	no
9	round	octagonal	no	flag	blue	yes
10	round	octagonal	no	flag	green	yes
11	square	round	yes	flag	yellow	yes

APPENDIX B: RESULTS

Figure B3. Output file robots.out from AQ15.
(continued)

Pos-outhypo

cpx

1 [hs=round,square] [ho=sword,balloon] [jc=red,green] (total:5, unique:5)

Neg-outhypo

cpx

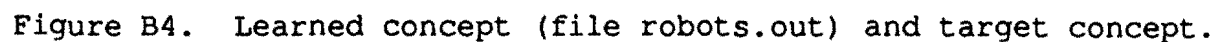
1 [hs=round,octagonal] [jc=yellow,green,blue] (total:9, unique:5)

2 [hs=octagonal] (total:6, unique:2)

This run used (milliseconds of CPU time):

System time: 0

User time : 116



APPENDIX B: RESULTS

Figure B5. Output file robots1.out from AQ15.

```

title
# text
1 "Input: robots1.inp"
2 "Output: robots1.out (AQ15) and r1.o (AQ17)"
3 "Seven positive examples and 11 negative examples: "

parameters
run mode ambig trim maxstar echo criteria operata
1 ic pos mini 10 tpcdvne default off

default-criteria
# criterion tolerance
1 maxnew 0.00
2 minsel 0.00

domaintypes
type levels cost name
nom 2 1.00 l2
nom 3 1.00 l3
nom 3 1.00 ho
nom 4 1.00 jc

variables
# type levels cost name
1 nom 3 1.00 hs.l3
2 nom 3 1.00 bs.l3
3 nom 2 1.00 sm.l2
4 nom 3 1.00 ho.ho
5 nom 4 1.00 jc.jc
6 nom 2 1.00 ti.l2

l2-names
value name
0 no
1 yes

l3-names
value name
0 round
1 square
2 octagonal

```

APPENDIX B: RESULTS

Figure B5. Output file robots1.out from AQ15.
(continued)

ho-names

value	name
0	sword
1	balloon
2	flag

jc-names

value	name
0	red
1	yellow
2	green
3	blue

Pos-events

#	hs	bs	sm	ho	jc	ti
1	round	round	yes	sword	red	no
2	round	square	yes	sword	red	yes
3	round	octagonal	yes	balloon	red	yes
4	square	square	yes	balloon	red	yes
5	square	square	no	balloon	green	yes
6	round	round	yes	flag	red	yes
7	square	round	yes	balloon	blue	yes

Neg-events

#	hs	bs	sm	ho	jc	ti
1	round	octagonal	yes	sword	yellow	no
2	square	octagonal	yes	sword	yellow	no
3	octagonal	square	no	sword	green	no
4	octagonal	round	yes	sword	blue	yes
5	octagonal	octagonal	no	balloon	green	no
6	octagonal	round	no	balloon	blue	no
7	octagonal	square	yes	flag	red	no
8	octagonal	round	no	flag	green	no
9	round	octagonal	no	flag	blue	yes
10	round	octagonal	no	flag	green	yes
11	square	round	yes	flag	yellow	yes

APPENDIX B: RESULTS

Figure B5. Output file robots1.out from AQ15.
(continued)

Pos-outhypo

```
# cpx
1 [hs=round,square] [ho=sword,balloon] [jc=red,green,blue] (total:6, unique:3)
2 [jc=red] [ti=yes] (total:4, unique:1)
```

Neg-outhypo

```
# cpx
1 [ho=sword,flag] [jc=yellow,green,blue] (total:8, unique:5)
2 [hs=octagonal] (total:6, unique:3)
```

This run used (milliseconds of CPU time):

```
System time:      0
User time :      217
```

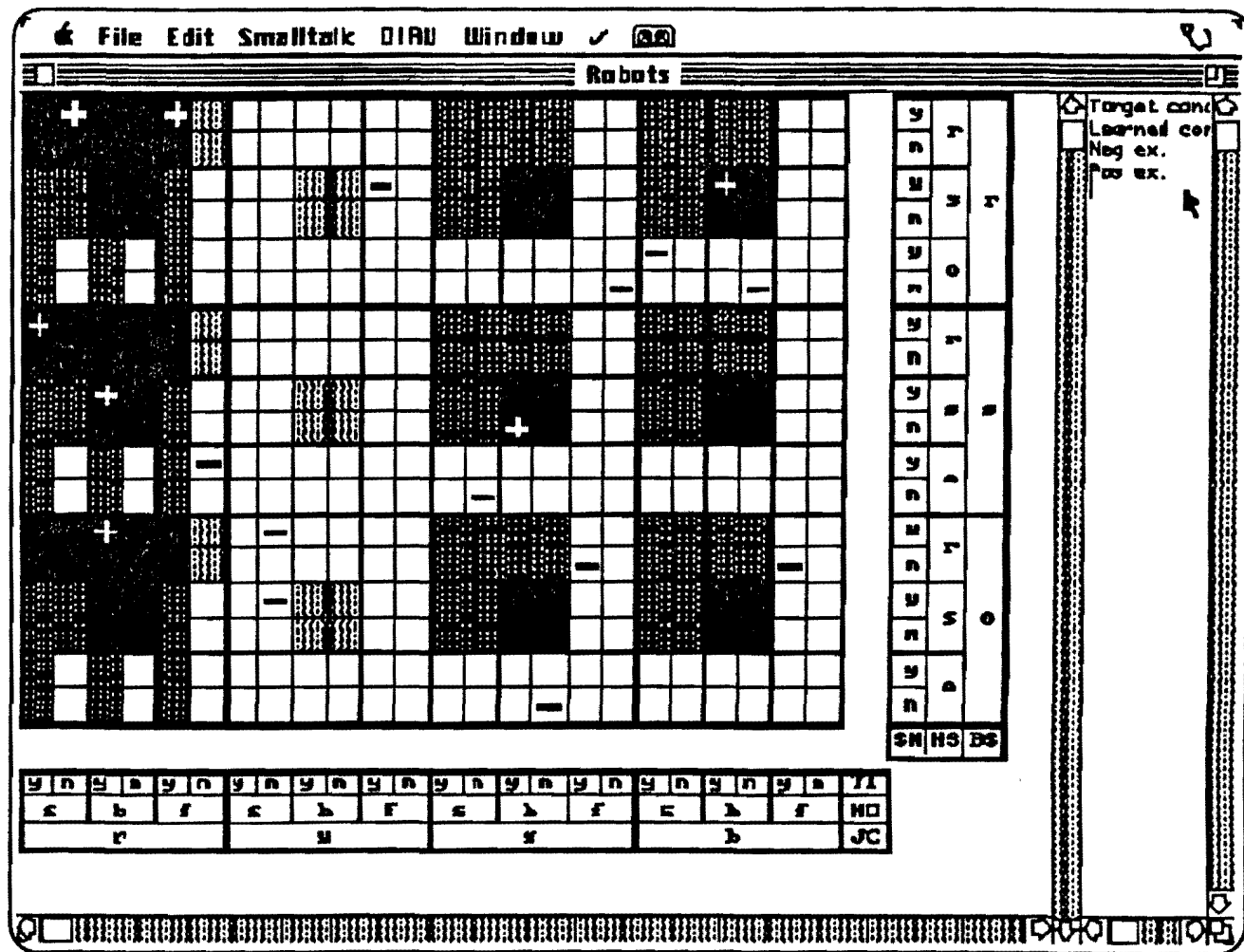


Figure B6. Learned concept (file robots1.out) and target concept.

APPENDIX B: RESULTS

Figure B7. Output file robots2.out from AQ15.

```

title
# text
1 "Input: robots2.inp"
2 "Output: robots2.out (AQ15) and r2.o (AQ17)"
3 "Seven positive examples and 12 negative examples:"

parameters
run mode  ambig  trim  maxstar  echo  criteria  operata
1  ic      pos    mini   10      tpodvne  default  off

default-criteria
# criterion  tolerance
1  maxnew    0.00
2  minsel    0.00

domaintypes
type  levels  cost  name
nom   2       1.00  12
nom   3       1.00  13
nom   3       1.00  ho
nom   4       1.00  jc

variables
#  type  levels  cost  name
1  nom   3       1.00  hs.13
2  nom   3       1.00  bs.13
3  nom   2       1.00  sm.12
4  nom   3       1.00  ho.ho
5  nom   4       1.00  jc.jc
6  nom   2       1.00  ti.12

12-names
value  name
0      no
1      yes

13-names
value  name
0      round
1      square
2      octagonal

```

APPENDIX B: RESULTS

Figure B7. Output file robots2.out from AQ15.
(continued)

ho-names

value	name
0	sword
1	balloon
2	flag

jc-names

value	name
0	red
1	yellow
2	green
3	blue

Pos-events

#	hs	bs	sm	ho	jc	ti
1	round	round	yes	sword	red	no
2	round	square	yes	sword	red	yes
3	round	octagonal	yes	balloon	red	yes
4	square	square	yes	balloon	red	yes
5	square	square	no	balloon	green	yes
6	round	round	yes	flag	red	yes
7	square	round	yes	balloon	blue	yes

Neg-events

#	hs	bs	sm	ho	jc	ti
1	round	octagonal	yes	sword	yellow	no
2	square	octagonal	yes	sword	yellow	no
3	octagonal	square	no	sword	green	no
4	octagonal	round	yes	sword	blue	yes
5	octagonal	octagonal	no	balloon	green	no
6	octagonal	round	no	balloon	blue	no
7	octagonal	square	yes	flag	red	no
8	octagonal	round	no	flag	green	no
9	round	octagonal	no	flag	blue	yes
10	round	octagonal	no	flag	green	yes
11	square	round	yes	flag	yellow	yes
12	square	round	yes	sword	red	yes

APPENDIX B: RESULTS

Figure B7. Output file robots2.out from AQ15.
(continued)

Pos-outhypo

```
# cpx
1 [hs=round] [jc=red] (total:4, unique:3)
2 [hs=round,square] [ho=balloon] (total:4, unique:3)
```

Neg-outhypo

```
# cpx
1 [ho=sword,flag] [jc=yellow,green,blue] (total:8, unique:3)
2 [hs=square,octagonal] [ho=sword,flag] (total:7, unique:1)
3 [hs=octagonal] (total:6, unique:2)
```

This run used (milliseconds of CPU time):

```
System time:      0
User time :      233
```

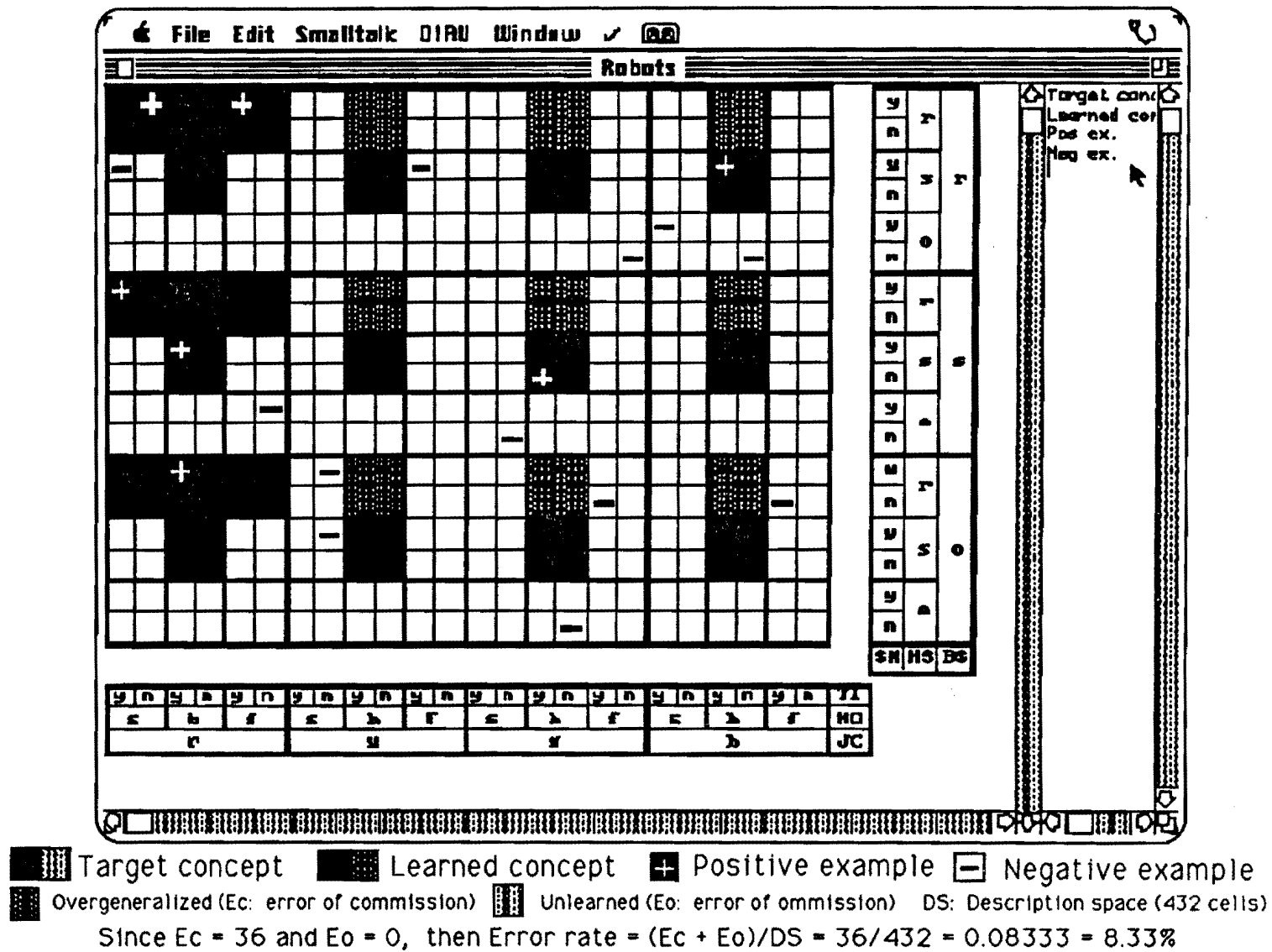


Figure B8. Learned concept (file robots2.out) and target concept.

APPENDIX B: RESULTS

Figure B9. Output file robots3.out from AQ15.

```

title
# text
1 "Input: robots3.inp"
2 "Output: robots3.out (AQ15) and r3.o (AQ17)"
3 "Seven positive examples and 13 negative examples:"

parameters
run mode  ambig  trim  maxstar  echo  criteria  operata
1  ic    pos    mini   10    tpcdvne  default  off

default-criteria
# criterion  tolerance
1  maxnew    0.00
2  minsel    0.00

domaintypes
type  levels  cost  name
nom    2    1.00  l2
nom    3    1.00  l3
nom    3    1.00  ho
nom    4    1.00  jc

variables
# type  levels  cost  name
1  nom    3    1.00  hs.l3
2  nom    3    1.00  bs.l3
3  nom    2    1.00  sm.l2
4  nom    3    1.00  ho.ho
5  nom    4    1.00  jc.jc
6  nom    2    1.00  ti.l2

12-names
value  name
0      no
1      yes

13-names
value  name
0      round
1      square
2      octagonal

```

APPENDIX B: RESULTS

Figure B9. Output file robots3.out from AQ15.
(continued)

ho-names

value	name
0	sword
1	balloon
2	flag

jc-names

value	name
0	red
1	yellow
2	green
3	blue

Pos-events

#	hs	bs	sm	ho	jc	ti
1	round	round	yes	sword	red	no
2	round	square	yes	sword	red	yes
3	round	octagonal	yes	balloon	red	yes
4	square	square	yes	balloon	red	yes
5	square	square	no	balloon	green	yes
6	round	round	yes	flag	red	yes
7	square	round	yes	balloon	blue	yes

Neg-events

#	hs	bs	sm	ho	jc	ti
1	round	octagonal	yes	sword	yellow	no
2	square	octagonal	yes	sword	yellow	no
3	octagonal	square	no	sword	green	no
4	octagonal	round	yes	sword	blue	yes
5	octagonal	octagonal	no	balloon	green	no
6	octagonal	round	no	balloon	blue	no
7	octagonal	square	yes	flag	red	no
8	octagonal	round	no	flag	green	no
9	round	octagonal	no	flag	blue	yes
10	round	octagonal	no	flag	green	yes
11	square	round	yes	flag	yellow	yes
12	square	round	yes	sword	red	yes
13	round	square	yes	balloon	green	yes

APPENDIX B: RESULTS

Figure B9. Output file robots3.out from AQ15.
(continued)

Pos-outhypo

```
# cpx
1 [hs=round] [jc=red] (total:4, unique:4)
2 [hs=square] [ho=balloon] (total:3, unique:3)
```

Neg-outhypo

```
# cpx
1 [hs=round,octagonal] [jc=yellow,green,blue] (total:9, unique:6)
2 [hs=square,octagonal] [ho=sword,flag] (total:7, unique:4)
```

This run used (milliseconds of CPU time):

```
System time:      0
User time :      233
```

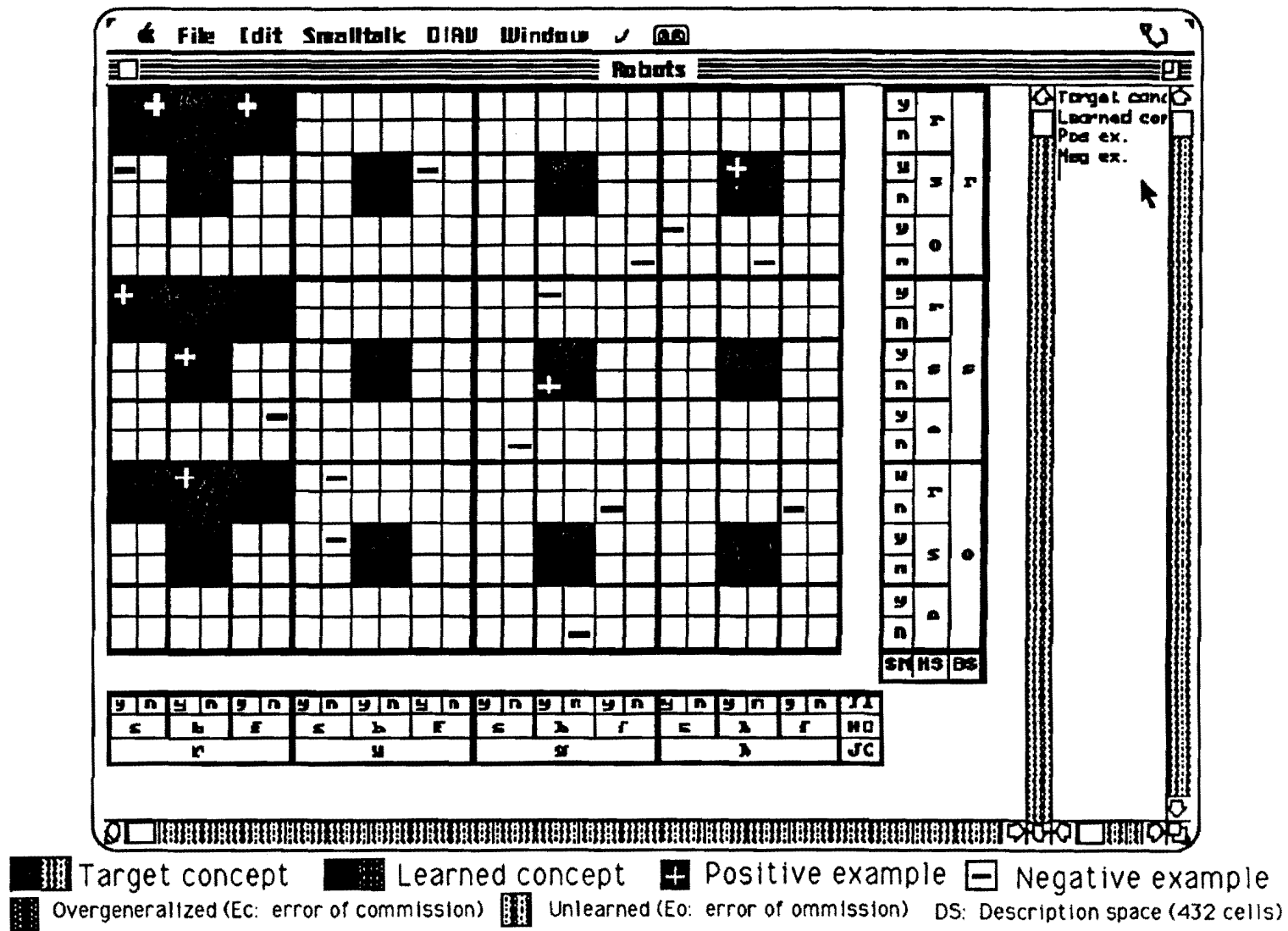


Figure B10. Learned concept (file robots3.out) and target concept.

APPENDIX B: RESULTS

Figure B11. Altered positive output hypothesis (APOH) method on AQ17-HCI output file r.o.

Pos-events

#	hs	bs	sm	ho	jc	ti
1	round	round	yes	sword	red	no
2	round	square	yes	sword	red	yes
3	round	octagonal	yes	balloon	red	yes
4	square	square	yes	balloon	red	yes
5	square	square	no	balloon	green	yes

Neg-events

#	hs	bs	sm	ho	jc	ti
1	round	octagonal	yes	sword	yellow	no
2	square	octagonal	yes	sword	yellow	no
3	octagonal	square	no	sword	green	no
4	octagonal	round	yes	sword	blue	yes
5	octagonal	octagonal	no	balloon	green	no
6	octagonal	round	no	balloon	blue	no
7	octagonal	square	yes	flag	red	no
8	octagonal	round	no	flag	green	no
9	round	octagonal	no	flag	blue	yes
10	round	octagonal	no	flag	green	yes
11	square	round	yes	flag	yellow	yes

Output hypotheses from AQ17:

Pos-outhypo

#	cpx
1	[hs=round,square] [ho=sword,balloon] [jc=red,green] (total:5, unique:5)

Neg-outhypo

#	cpx
1	[hs=round,octagonal] [jc=yellow,green,blue] (total:8, unique:5)
2	[hs=square,octagonal] [ho=sword,flag] (total:6, unique:3)

AQ17 has established the three attributes that occur in Pos-outhypo as relevant: hs, ho, and jc. For that reason, only hs, ho, and jc appear in Neg-outhypo.

Comparing Pos-outhypo and Neg-outhypo, the following four combinations of attribute values cannot exist in Pos-outhypo since they occur in Neg-outhypo:

```
[hs=round] [ho=sword] [jc=green]
[hs=round] [ho=balloon] [jc=green]
[hs=square] [ho=sword] [jc=red]
[hs=square] [ho=sword] [jc=green]
```

APPENDIX B: RESULTS

Figure B11. Altered positive output hypothesis (APOH) method on AQ17-HCI output file r.o.
(continued)

They may be put in the following forms:

1. If keep[hs=round] [ho=sword] then remove[jc=green]
2. If keep[hs=round] [jc=green] then remove[ho=sword]
3. If keep[ho=sword] [jc=green] then remove[hs=round]
4. If keep[hs=round] [ho=balloon] then remove[jc=green]
5. If keep[hs=round] [jc=green] then remove[ho=balloon]
6. If keep[ho=balloon] [jc=green] then remove[hs=round]
7. If keep[hs=square] [ho=sword] then remove[jc=red]
8. If keep[hs=square] [jc=red] then remove[ho=sword]
9. If keep[ho=sword] [jc=red] then remove[hs=square]
10. If keep[hs=square] [ho=sword] then remove[jc=green]
11. If keep[hs=square] [jc=green] then remove[ho=sword]
12. If keep[ho=sword] [jc=green] then remove[hs=square]

We can apply the rules above in six ways to obtain new rules:

- | | | | | | |
|----------------|-------------------|--------------------|----------------|----------------|----------|
| 1. (1,4,7,10): | [hs=round,square] | [ho=sword,balloon] | | Pos{1,2,3,4,5} | Neg{1,2} |
| 2. (2,5,8,11): | [hs=round,square] | | [jc=red,green] | Pos{1,2,3,4,5} | Neg{10} |
| 3. (3,6,9,12): | | [ho=sword,balloon] | [jc=red,green] | Pos{1,2,3,4,5} | Neg{3,5} |
| 4. (4,8): | [hs=round,square] | [ho=balloon] | [jc=red] | Pos{3,4} | Neg{} |
| 5. (1,4,9): | [hs=round] | [ho=sword,balloon] | [jc=red] | Pos{1,2,3} | Neg{} |
| 6. (6,8,11): | [hs=square] | [ho=balloon] | [jc=red,green] | Pos{4,5} | Neg{} |

Since rules 1, 2, and 3 cover negative examples, they overgeneralize the concept and are no longer considered.

Rules 5 and 6 have disjoint covers of the whole positive example set. Therefore, they are proposed as a new Pos-hypo:

- | | | | | | |
|-----------|-------------|--------------------|----------------|------------|-------|
| 1st rule: | [hs=round] | [ho=sword,balloon] | [jc=red] | Pos{1,2,3} | Neg{} |
| 2nd rule: | [hs=square] | [ho=balloon] | [jc=red,green] | Pos{4,5} | Neg{} |

We simplify the rules by dropping one condition while ensuring that coverage remains consistent regarding negative examples as follows:

1st rule:

- | | | | | | |
|----|------------|--------------------|----------|--------------|--------|
| a) | [hs=round] | [ho=sword,balloon] | | Pos{1,2,3} | Neg{1} |
| b) | [hs=round] | | [jc=red] | Pos{1,2,3} | Neg{} |
| c) | | [ho=sword,balloon] | [jc=red] | Pos{1,2,3,4} | Neg{} |

Removing [jc=red] makes the rule cover a negative example. Thus, we drop a) as a solution. Although c) has greater coverage than b), b) has adequate coverage and is simpler than c). Therefore, we pick b) as the solution.

APPENDIX B: RESULTS

Figure B11. Altered positive output hypothesis (APOH) method on AQ17-HCI output file r.o.
(continued)

So, the first rule in the new Pos-hypo is--

[hs=round] [jc=red] Pos{1,2,3} Neg{}

2nd rule:

- | | | | | | |
|----|-------------|--------------|----------------|------------|--------|
| a) | [hs=square] | [ho=balloon] | | Pos{4,5} | Neg{} |
| b) | [hs=square] | | [jc=red,green] | Pos{4,5} | Neg{} |
| c) | | [ho=balloon] | [jc=red,green] | Pos{3,4,5} | Neg{5} |

Removing [hs=square] makes the rule cover a negative example. Thus, we drop c) as a solution. The coverage by a) and b) is the same and a) is simpler than b). We pick a) as the solution.

So, the second rule in the new Pos-hypo is--

[hs=square] [ho=balloon] Pos{4,5} Neg{}

It is apparent that the two simplified rules in the new Pos-hypo still have disjoint covers of the whole positive example set.

THE FINAL HYPOTHESIS FOR THE POSITIVE CLASS IS--

[hs=round] [jc=red]
[hs=square] [ho=balloon]

THIS MATCHES EXACTLY THE C1 TARGET-CONCEPT DESCRIPTION!

APPENDIX B: RESULTS

Figure B12. Altered positive output hypothesis (APOH) method on AQ17-HCI output file r1.o.

Pos-events

#	hs	bs	sm	ho	jc	ti
1	round	round	yes	sword	red	no
2	round	square	yes	sword	red	yes
3	round	octagonal	yes	balloon	red	yes
4	square	square	yes	balloon	red	yes
5	square	square	no	balloon	green	yes
6	round	round	yes	flag	red	yes
7	square	round	yes	balloon	blue	yes

Neg-events

#	hs	bs	sm	ho	jc	ti
1	round	octagonal	yes	sword	yellow	no
2	square	octagonal	yes	sword	yellow	no
3	octagonal	square	no	sword	green	no
4	octagonal	round	yes	sword	blue	yes
5	octagonal	octagonal	no	balloon	green	no
6	octagonal	round	no	balloon	blue	no
7	octagonal	square	yes	flag	red	no
8	octagonal	round	no	flag	green	no
9	round	octagonal	no	flag	blue	yes
10	round	octagonal	no	flag	green	yes
11	square	round	yes	flag	yellow	yes

Output hypotheses from AQ17:

Pos-outhypo

#	cpx
1	[hs=round,square] [ho=sword,balloon] [jc=red,green,blue] (total:6, unique:3)
2	[hs=round] [jc=red] (total:4, unique:1)

Neg-outhypo

#	cpx
1	[ho=sword,flag] [jc=yellow,green,blue] (total:8, unique:5)
2	[hs=octagonal] (total:6, unique:3)

AQ17 has established the three attributes that occur in Pos-outhypo as relevant: hs, ho, and jc. For that reason, only hs, ho, and jc appear in Neg-outhypo.

Comparing Pos-outhypo and Neg-outhypo, the following four combinations of attribute values cannot exist in Pos-outhypo since they occur in Neg-outhypo:

```
[hs=round] [ho=sword] [jc=green]
[hs=round] [ho=sword] [jc=blue]
[hs=square] [ho=sword] [jc=green]
[hs=square] [ho=sword] [jc=blue]
```

APPENDIX B: RESULTS

Figure B12. Altered positive output hypothesis (APOH) method on AQ17-HCI output file r1.o.
(continued)

They may be put in the following forms:

1. If keep[hs=round] [ho=sword] then remove[jc=green]
2. If keep[hs=round] [jc=green] then remove[ho=sword]
3. If keep[ho=sword] [jc=green] then remove[hs=round]
4. If keep[hs=round] [ho=sword] then remove[jc=blue]
5. If keep[hs=round] [jc=blue] then remove[ho=sword]
6. If keep[ho=sword] [jc=blue] then remove[hs=round]
7. If keep[hs=square] [ho=sword] then remove[jc=green]
8. If keep[hs=square] [jc=green] then remove[ho=sword]
9. If keep[ho=sword] [jc=green] then remove[hs=square]
10. If keep[hs=square] [ho=sword] then remove[jc=blue]
11. If keep[hs=square] [jc=blue] then remove[ho=sword]
12. If keep[ho=sword] [jc=blue] then remove[hs=square]

We can apply the rules above in three ways to obtain new rules:

1. (1,4,7,10): [hs=round,square] [ho=sword,balloon] [jc=red] Pos{1,2,3,4} Neg{}
2. (2,5,8,11): [hs=round,square] [ho=balloon] [jc=red,green,blue] Pos{3,4,5,7} Neg{}
3. (3,6,9,12): [ho=sword,balloon] [jc=red,green,blue] Pos{1,2,3,4,5,7} Neg{3,4,5,6}

Since rule 3 covers negative examples, it overgeneralizes the concept and is no longer considered.

Only rule 2 above and Pos-outhypo rule 2 together cover the whole positive example set. Therefore, they are proposed as a new Pos-hypo:

- 1st rule: [hs=round,square] [ho=balloon] [jc=red,green,blue] Pos{3,4,5,7} Neg{}
- 2nd rule: [hs=round] [jc=red] Pos{1,2,3,6} Neg{}

We simplify the rules by dropping one condition while ensuring that coverage remains consistent regarding negative examples as follows:

1st rule:

- a) [hs=round,square] [ho=balloon] Pos{3,4,5,7} Neg{}
- b) [hs=round,square] [jc=red,green,blue] Pos{1,2,3,4,5,6,7} Neg{9,10}
- c) [ho=balloon] [jc=red,green,blue] Pos{3,4,5,7} Neg{5,6}

Removing [ho=balloon] makes the rule cover negative examples. Thus, we drop b) as a solution. Removing [hs=round,square] makes the rule cover negative examples. Thus, we drop c) as a solution. Therefore, we pick a) as the solution.

So, the first rule in the new Pos-hypo is—

[hs=round,square] [ho=balloon] Pos{3,4,5,7} Neg{}

APPENDIX B: RESULTS

Figure B12. Altered positive output hypothesis (APOH) method on AQ17-HCI output file rl.o.
(continued)

We can further simplify the first rule by dropping one attribute-value pair while ensuring that coverage remains consistent regarding negative examples as follows:

- a) [hs=round] [ho=balloon] Pos{3} Neg{}
- b) [hs=square] [ho=balloon] Pos{4,5,7} Neg{}

Since b) has greater coverage than a), we pick b) as the first rule.

So, the first rule in the new Pos-hypo is now—

[hs=square] [ho=balloon] Pos{4,5,7} Neg{}

2nd rule:

- a) [hs=round] Pos{1,2,3,6} Neg{1,9,10}
- b) [jc=red] Pos{1,2,3,4,6} Neg{7}

Removing any condition makes the rule cover negative examples, thus inconsistent regarding negative examples. Therefore, the second rule cannot be simplified.

So, the second rule in the new Pos-hypo remains as before—

[hs=round] [jc=red] Pos{1,2,3,6} Neg{}

It is apparent that the simplified first rule and the second rule in the new Pos-hypo have disjoint covers of the whole positive example set.

THE FINAL HYPOTHESIS FOR THE POSITIVE CLASS IS—

[hs=square] [ho=balloon]
[hs=round] [jc=red]

THIS MATCHES EXACTLY THE C1 TARGET-CONCEPT DESCRIPTION!

APPENDIX B: RESULTS

Figure B13. Altered positive output hypothesis (APOH) method on AQ17-HCI output file r2.o.

Pos-events

#	hs	bs	sm	ho	jc	ti
1	round	round	yes	sword	red	no
2	round	square	yes	sword	red	yes
3	round	octagonal	yes	balloon	red	yes
4	square	square	yes	balloon	red	yes
5	square	square	no	balloon	green	yes
6	round	round	yes	flag	red	yes
7	square	round	yes	balloon	blue	yes

Neg-events

#	hs	bs	sm	ho	jc	ti
1	round	octagonal	yes	sword	yellow	no
2	square	octagonal	yes	sword	yellow	no
3	octagonal	square	no	sword	green	no
4	octagonal	round	yes	sword	blue	yes
5	octagonal	octagonal	no	balloon	green	no
6	octagonal	round	no	balloon	blue	no
7	octagonal	square	yes	flag	red	no
8	octagonal	round	no	flag	green	no
9	round	octagonal	no	flag	blue	yes
10	round	octagonal	no	flag	green	yes
11	square	round	yes	flag	yellow	yes
12	square	round	yes	sword	red	yes

Output hypotheses from AQ17:

Pos-outhypo

#	cpx
1	[hs=round] [jc=red] (total:4, unique:3)
2	[hs=round,square] [ho=balloon] (total:4, unique:3)

Neg-outhypo

#	cpx
1	[ho=sword,flag] [jc=yellow,green,blue] (total:8, unique:3)
2	[hs=octagonal] [hs=round,square] [ho=sword,flag] (total:7, unique:1)
3	[hs=octagonal] (total:6, unique:2)

AQ17 has established the three attributes that occur in Pos-outhypo as relevant: hs, ho, and jc. For that reason, only hs, ho, and jc appear in Neg-outhypo.

Comparing Pos-outhypo and Neg-outhypo discloses that the combinations of attribute values in Pos-outhypo do not occur in Neg-outhypo.

APPENDIX B: RESULTS

Figure B13. Altered positive output hypothesis (APOH) method on AQ17-HCI output file r2.o.
(continued)

We try to simplify the Pos-outhypo rules by dropping one condition while ensuring that coverage remains consistent regarding negative examples as follows:

1st rule:

- a) [hs=round] Pos{1,2,3,6} Neg{1,9,10}
- b) [jc=red] Pos{1,2,3,4,6} Neg{7,12}

Removing either condition makes the rule cover negative examples, thus inconsistent regarding negative examples. Therefore, the first rule cannot be simplified.

So, the first rule in the new Pos-hypo remains as before—

[hs=round] [jc=red] Pos{1,2,3,6} Neg{}

2nd rule:

- a) [hs=round,square] Pos{1,2,3,4,5,6,7} Neg{1,2,9,10,11,12}
- b) [ho=balloon] Pos{3,4,5,7} Neg{5,6}

Removing either condition makes the rule cover negative examples, thus inconsistent regarding negative examples. Therefore, the second rule cannot be simplified by removing a condition.

However, we try simplifying the second rule by dropping one attribute-value pair while ensuring that coverage remains consistent regarding negative examples as follows:

- a) [hs=round] [ho=balloon] Pos{3} Neg{}
- b) [hs=square] [ho=balloon] Pos{4,5,7} Neg{}

Since b) has greater coverage than a), we pick b) as the solution.

So, the second rule in the new Pos-hypo is—

[hs=square] [ho=balloon] Pos{4,5,7} Neg{}

It is apparent that this simplified second rule and the first rule have disjoint covers of the whole positive example set.

THE FINAL HYPOTHESIS FOR THE POSITIVE CLASS IS—

[hs=round] [jc=red]
[hs=square] [ho=balloon]

THIS MATCHES EXACTLY THE C1 TARGET-CONCEPT DESCRIPTION!

APPENDIX B: RESULTS

Figure B14. Altered positive output hypothesis (APOH) method on AQ17-HCI output file r3.o.

Pos-events

#	hs	bs	sm	ho	jc	ti
1	round	round	yes	sword	red	no
2	round	square	yes	sword	red	yes
3	round	octagonal	yes	balloon	red	yes
4	square	square	yes	balloon	red	yes
5	square	square	no	balloon	green	yes
6	round	round	yes	flag	red	yes
7	square	round	yes	balloon	blue	yes

Neg-events

#	hs	bs	sm	ho	jc	ti
1	round	octagonal	yes	sword	yellow	no
2	square	octagonal	yes	sword	yellow	no
3	octagonal	square	no	sword	green	no
4	octagonal	round	yes	sword	blue	yes
5	octagonal	octagonal	no	balloon	green	no
6	octagonal	round	no	balloon	blue	no
7	octagonal	square	yes	flag	red	no
8	octagonal	round	no	flag	green	no
9	round	octagonal	no	flag	blue	yes
10	round	octagonal	no	flag	green	yes
11	square	round	yes	flag	yellow	yes
12	square	round	yes	sword	red	yes
13	round	square	yes	balloon	green	yes

Output hypotheses from AQ17:

Pos-outhypo

#	cpx
1	[hs=round] [jc=red] (total:4, unique:4)
2	[hs=square] [ho=balloon] (total:3, unique:3)

Neg-outhypo

#	cpx
1	[hs=round,octagonal] [jc=yellow,green,blue] (total:9, unique:6)
2	[hs=square,octagonal] [ho=sword,flag] (total:7, unique:4)

THE OUTPUT HYPOTHESIS FOR THE POSITIVE CLASS MATCHES EXACTLY THE C1 TARGET-CONCEPT DESCRIPTION!

APPENDIX B: RESULTS

Figure B15. Altered positive output hypothesis (APOH) method on AQ17-HCI output file r4.o.

Pos-events

#	hs	bs	sm	ho	jc	ti
1	round	round	yes	sword	red	no
2	round	square	yes	sword	red	yes
3	round	octagonal	yes	balloon	red	yes
4	square	square	yes	balloon	red	yes
5	square	square	no	balloon	green	yes
6	round	round	yes	flag	red	yes
7	square	round	yes	balloon	blue	yes

Neg-events

#	hs	bs	sm	ho	jc	ti
1	round	octagonal	yes	sword	yellow	no
2	square	octagonal	yes	sword	yellow	no
3	octagonal	square	no	sword	green	no
4	octagonal	round	yes	sword	blue	yes
5	octagonal	octagonal	no	balloon	green	no
6	octagonal	round	no	balloon	blue	no
7	octagonal	square	yes	flag	red	no
8	octagonal	round	no	flag	green	no
9	round	octagonal	no	flag	blue	yes
10	round	octagonal	no	flag	green	yes
11	square	round	yes	flag	yellow	yes
12	square	round	yes	sword	red	yes
13	round	square	yes	balloon	green	yes
14	round	square	yes	flag	red	no

Output hypotheses from AQ17:

Pos-outhypo

#	cpx
1	[hs=round] [ho=sword, balloon] [jc=red] (total:3, unique:3)
2	[hs=round] [ho=balloon] (total:3, unique:3)

Neg-outhypo

#	cpx
1	[hs=round, octagonal] [jc=yellow, green, blue] (total:9, unique:6)
2	[hs=square, octagonal] [ho=sword, flag] (total:7, unique:4)

AQ17 has established the three attributes that occur in Pos-outhypo as relevant: hs, ho, and jc. For that reason, only hs, ho, and jc appear in Neg-outhypo.

Comparing Pos-outhypo and Neg-outhypo discloses that the combinations of attribute values in Pos-outhypo do not occur in Neg-outhypo.

APPENDIX B: RESULTS

Figure B15. Altered positive output hypothesis (APOH) method on AQ17-HCI output file r4.o.
(continued)

We try to simplify the Pos-outhypo rules by dropping one condition while ensuring that coverage remains consistent regarding negative examples as follows:

1st rule:

- a) [hs=round] [ho=sword,balloon] Pos{1,2,3} Neg{1,13}
- b) [hs=round] [jc=red] Pos{1,2,3,6} Neg{14}
- c) [ho=sword,balloon] [jc=red] Pos{1,2,3,4} Neg{12}

Removing any condition makes the rule cover negative examples, thus inconsistent regarding negative examples. Therefore, the first rule cannot be simplified.

2nd rule:

- a) [hs=square] Pos{4,5,7} Neg{2,11,12}
- b) [ho=balloon] Pos{3,4,5,7} Neg{5,6,13}

Removing either condition makes the rule cover negative examples, thus inconsistent regarding negative examples. Therefore, the second rule cannot be simplified.

As a result, the Pos-outhypo cannot be simplified and has the following coverage:

```
[hs=round] [ho=sword,balloon] [jc=red] Pos{1,2,3} Neg{}
[hs=square] [ho=balloon] Pos{4,5,7} Neg{}
```

It is apparent that the Pos-outhypo fails to cover positive example 6. The reason is a valid one.

AQ17 is using a more restricted attribute language limited to those attributes deemed to be relevant: hs, ho, and jc. As such, the attributes have the same values in negative example 14 as in positive example 6. This is precisely why the Neg-outhypo fails to cover negative example 14 with the following coverage:

```
[hs=round,octagonal] [jc=yellow,green,blue] Pos{} Neg{1,3,4,5,6,8,9,10,13}
[hs=square,octagonal] [ho=sword,flag] Pos{} Neg{2,7,11,12}
```

Taking another look at choice b) in our attempt to simplify the first rule above sheds further light. That choice as the first rule and the original second rule in a new Pos-hypo would have matched exactly the CI target-concept description. However, choice b) had the following coverage:

- b) [hs=round] [jc=red] Pos{1,2,3,6} Neg{14}

Even though it added cover for positive example 6, it also covered negative example 14 and could not be used as a simplification of the first rule.

APPENDIX B: RESULTS

Figure B15. Altered positive output hypothesis (APOH) method on AQ17-HCI output file r4.o.
(continued)

Simply stated, the problem is that we purposely added negative example 14 with attribute values within the target concept space, thus testing AQ17's reaction to "noise."

AQ17's output hypotheses seem correct for the training set given to it. However, it failed to learn the correct concept in the presence of "noise."

This is a likely candidate problem for future research!