Conversion Masters in IT (MIT)
AI as Representation and Search

(Machine Learning: EDSM + Version Search Space)

Lecture 006

Sandro Spina

State Merging Algorithms

- Starting from the APTA we merge states together in order to label the unlabled states in the APTA
- By merging we introduce cycles in the APTA.
- With every merge the hypothesis becomes more general.
- We keep on merging (compatible) states until there are no more compatible merges.
- Finally we output the final hypothesis.

Evidence Driven State Merging (EDSM)

- What heuristic do we apply to this search ?
- Or rather we need to determine the order of the merges. The heuristic determines the order of how states are merged.
- *NOTE:* Different orders give different hypothesis. Why ?
- EDSM orders the merges by giving each possible merge a compatibility score.

Version Space Search

- Version Space search (Mitchell 1978, 1982) illustrates the implementation of inductive learning as search through a concept space (the set of all concept descriptions consistent with training examples).

- *The representation of learned knowledge*:
  - Eg the concept ball
  - General concept
    - Size (X, Y) $\wedge$ color (X, Z) $\wedge$ shape (X, round)
  - Less General concept
    - Size (X, small) $\wedge$ color (X, Z) $\wedge$ shape (X, round)

- *A set of operations.* Given a set of training instances, the learner must construct a generalisation that satisfies it's goals. Therefore we need operations to manipulate representations. *Eg Generalisation or specializing symbolic expressions.*

Generalisation Operators

- Size (obj1, small) $\wedge$ colour (obj1, red) $\wedge$ shape (obj1, round)
  - Replacing a single constant with a variable produces the generalizations:
- Size (obj1, X) $\wedge$ colour (obj1, red) $\wedge$ shape (obj1, round)
- Size (obj1, small) $\wedge$ colour (obj1, X) $\wedge$ shape (obj1, round)
- Size (obj1, small) $\wedge$ colour (obj1, red) $\wedge$ shape (obj1, X)
- Size (X, small) $\wedge$ colour (X, red) $\wedge$ shape (X, round)

1. Replacing constants with variables
2. Dropping conditions from a conjunction expression
3. Adding a disjunction to an expression
4. Replacing a property with it's parent in a class hierarchy. Eg primary colour is a superclass of red.

Concept Space and Heuristic search

- The concept space is the representation language, together with the operations described in previous slide, defines a space of potential concept definitions.
- Heuristic Search. The learner must search this space to find the desired concept. Learning programs must commit to a direction or order of search (inductive bias).

The Candidate Elimination Algorithm

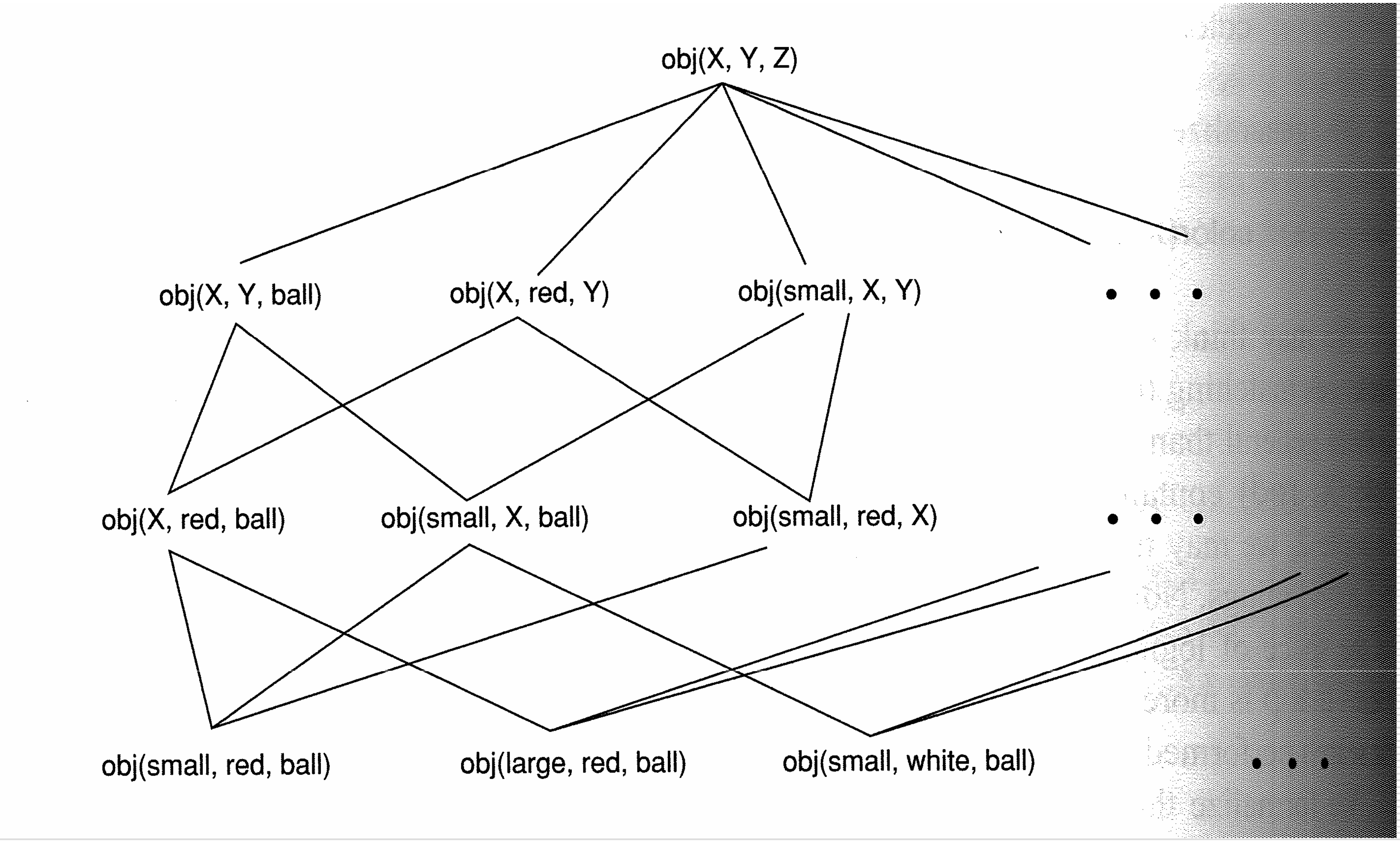
- Algorithm works by reducing the size of the version space as more examples become available.

- One can reduce the space in two ways:
  - In a specific to general direction; or
  - In a general to specific direction

- The candidate elimination algorithm combines these two approaches (bi-directional search)

- The algorithms are data driven; as in they generalize with respect to regularities found in the training data.

Eg of a concept space obj(X, Y, Z) …



Specific to general search for hypothesis set S

- Initialize S to the first +ve training instance;
- N is the set of all –ve instances seen so far;

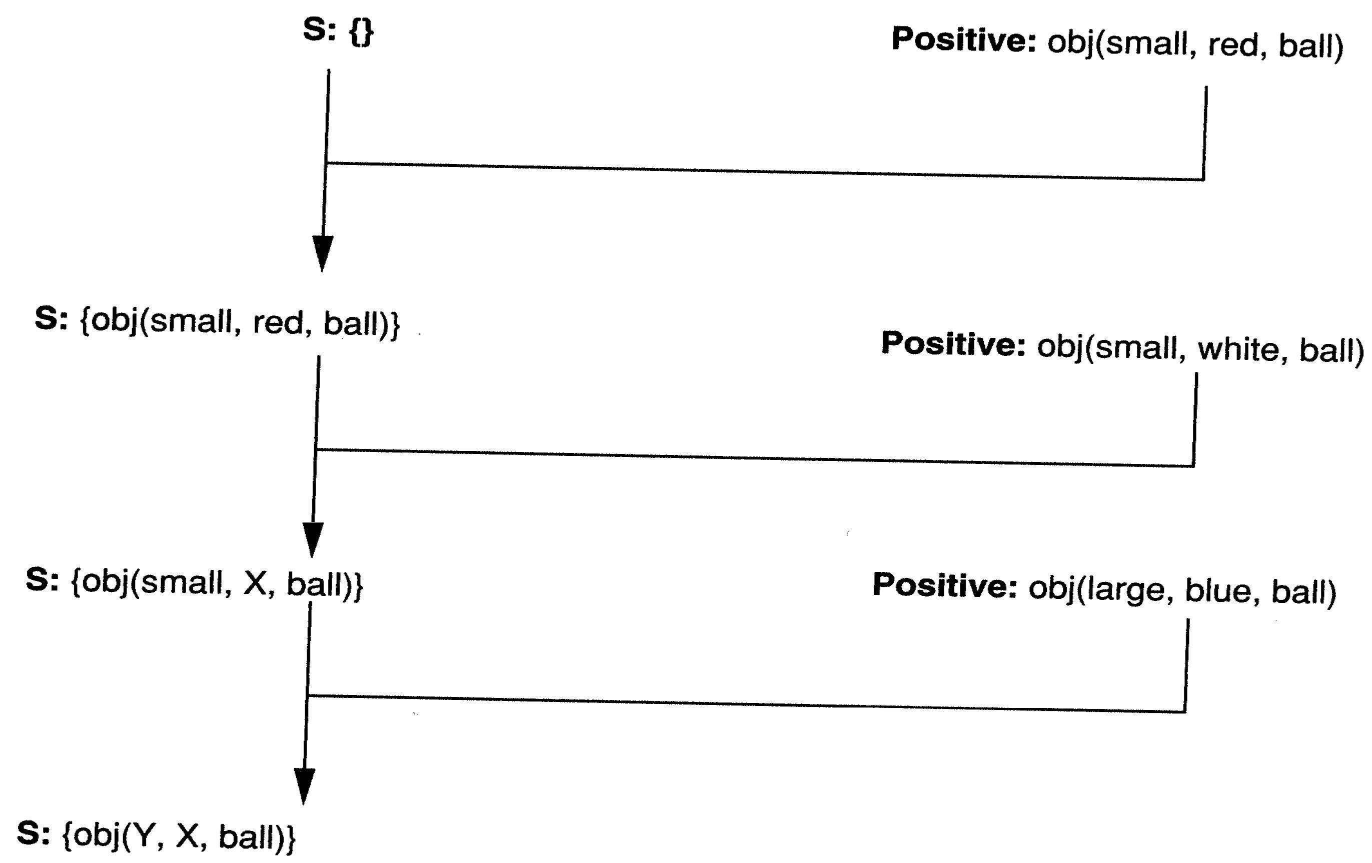
- For each +ve instance p
  - Begin
  - For every s in S, if s does not match p, replace s with its most specific generalization that matches p;
  - Delete from S all hypothesis more general than some other hypothesis in S
  - Delete from S all hypothesis that match a previously observed –ve instance in N;
  - End;
- For each –ve instance n
  - Begin
    - Delete all members of S that match n;
    - Add n to N to check future hypothesis for overgeneralization;
  - End;

General to specific search for hypothesis set S

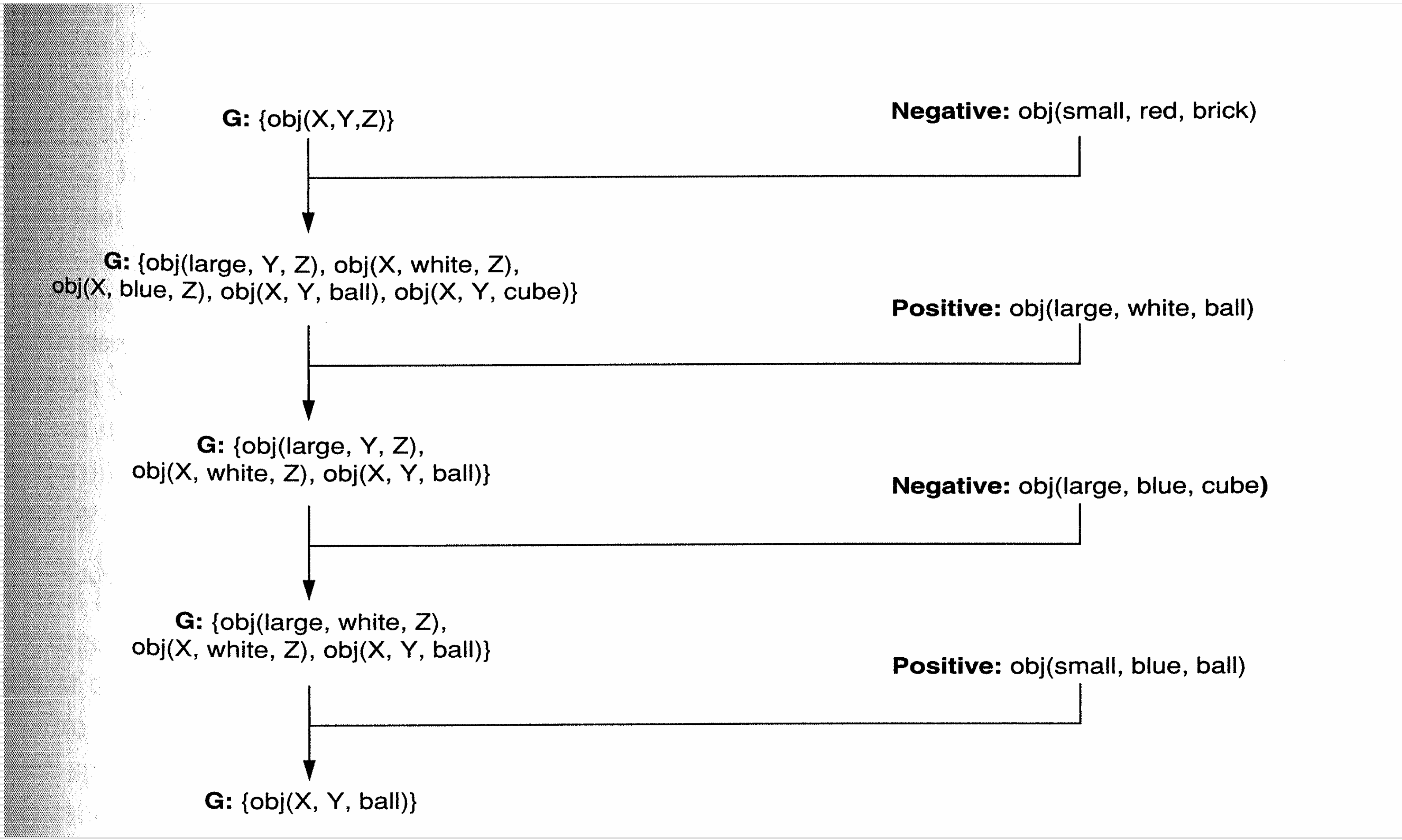
- Initialize G to contain the most general concept in the space;
- P contains all +ve examples seen so far;

- For each -ve instance n
  - Begin
  - For every g in G that matches n, replace g with its most general specialization that do not match n;
  - Delete from G all hypothesis more specific than some other hypothesis in G;
  - Delete from G all hypothesis that fail to match some +ve examples in P;
  - End;
- For each +ve instance p
  - Begin
    - Delete from G all hypothesis that fail to match p;
    - Add p to P;
  - End;

Eg of using the specific → generic algo



Eg of using the generic → specific algo



Combining the two together

- Keep both sets S and G.
- S contains the most specific description;
- G contains the most generic;
- If G=S and both are singletons, then the algorithm has found a single concept that is consistent with all the data and the algorithm halts;
- If G and S become empty, then there is no concept that covers all +ve instances and none of the −ve instances.

Eg Combining the two together