Knowledge in Learning

Chapter 19

Concept Learning

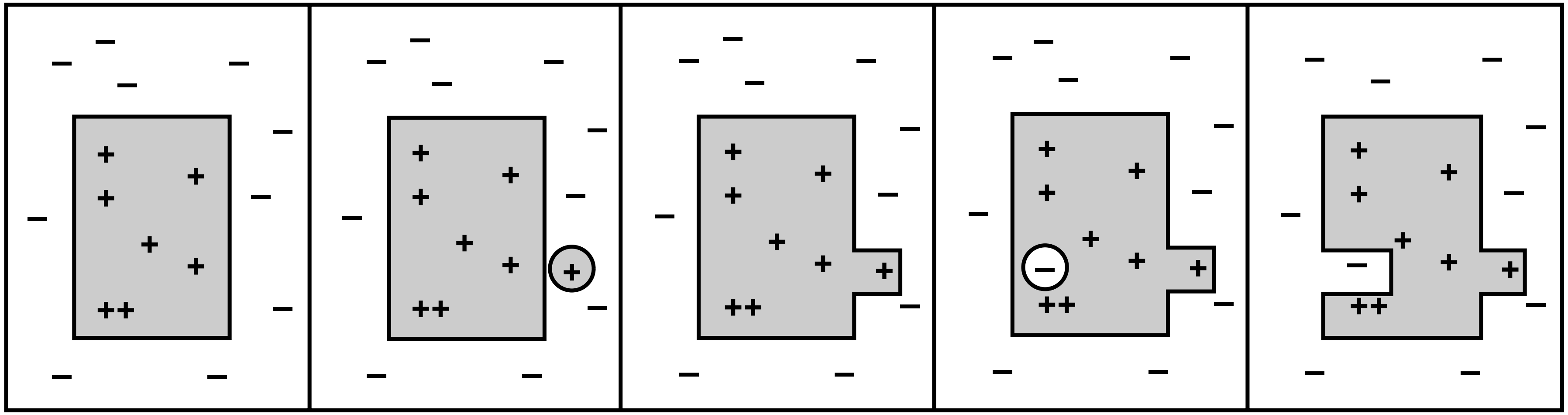
- Data Set: collection of instances = D.
- Instance: (list of attributes, class) $= d_i = (x_i, c(x_i))$
- Hypothesis: mapping $h : x_i \rightarrow c \in C$ (where C = set of classes)
- Consistent Hypothesis: $Consistent(h, D) \leftrightarrow \forall d_i \in D \; h(x_i) = c(x_i)$
- Classification = Hypothesis Elimination
  - Begin with H^* = whole hypothesis space, H.
  - For each $d_i \in D$
    - For each $h_k \in H^*$: If $h_k(x_i) \neq c(x_i)$, then $H^* \leftarrow H^* - h_i$.
  - $consistent(h_k, D) \; \forall h_k \in H^*$

H^* can be VERY LARGE
Can we work with a single h and generalize and specialize it to fit D?
Yes, but lots of search, since $\exists$ many ways to generalize and specialize!

Generalizing and Specializing a Hypothesis

- **Extension** of h = all instances that h classifies as positive.
- **Generalize** h: Changing h so as to **expand** its extension.
  - Drop a conjunct:
    red(x) $\wedge$ round(x) $\longrightarrow$ round(x).
  - Add a disjunct:
    red(x) $\wedge$ round(x) $\longrightarrow$ (red(x) $\vee$ blue(x)) $\wedge$ round(x)
- **Specialize** h: changing h so as to **contract** its extension.
  - Add a conjunct:
    red(x) $\wedge$ round(x) $\longrightarrow$ red(x) $\wedge$ striped(x) $\wedge$ round(x)
  - Drop a disjunct:
    red(x) $\vee$ blue(x) $\longrightarrow$ blue(x)

Hypothesis Refinement



(a) (b) (c) (d) (e)

a The Consistent Hypothesis (h): h agrees with all the instance classifications.

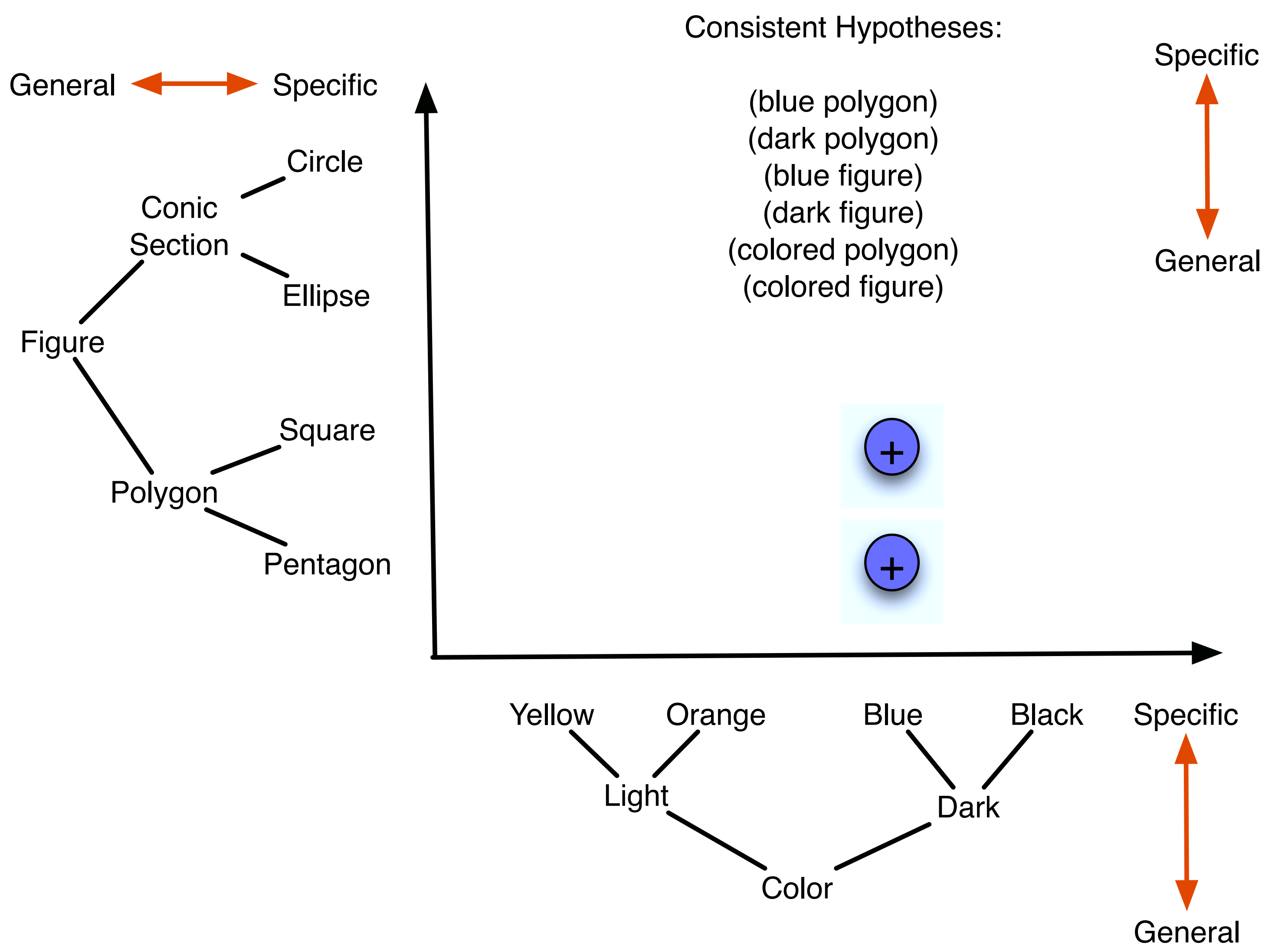
b A false negative: h(x) = -, but C(x) = +, where C(x) = correct class of instance x.

c Generalizing h to cover x.

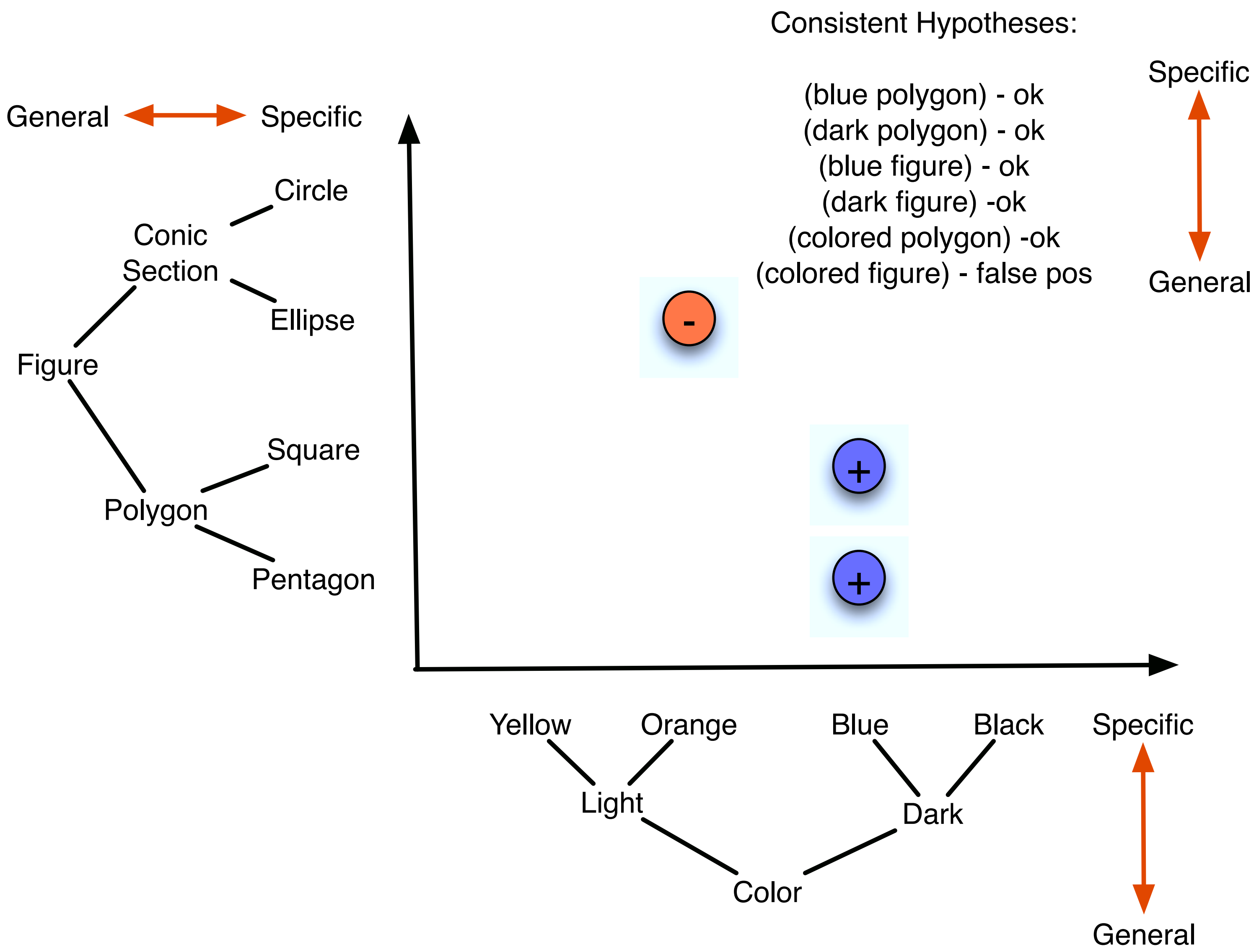
d A false positive: h(y) = +, but C(y) = -.

e Specializing h to exclude y.

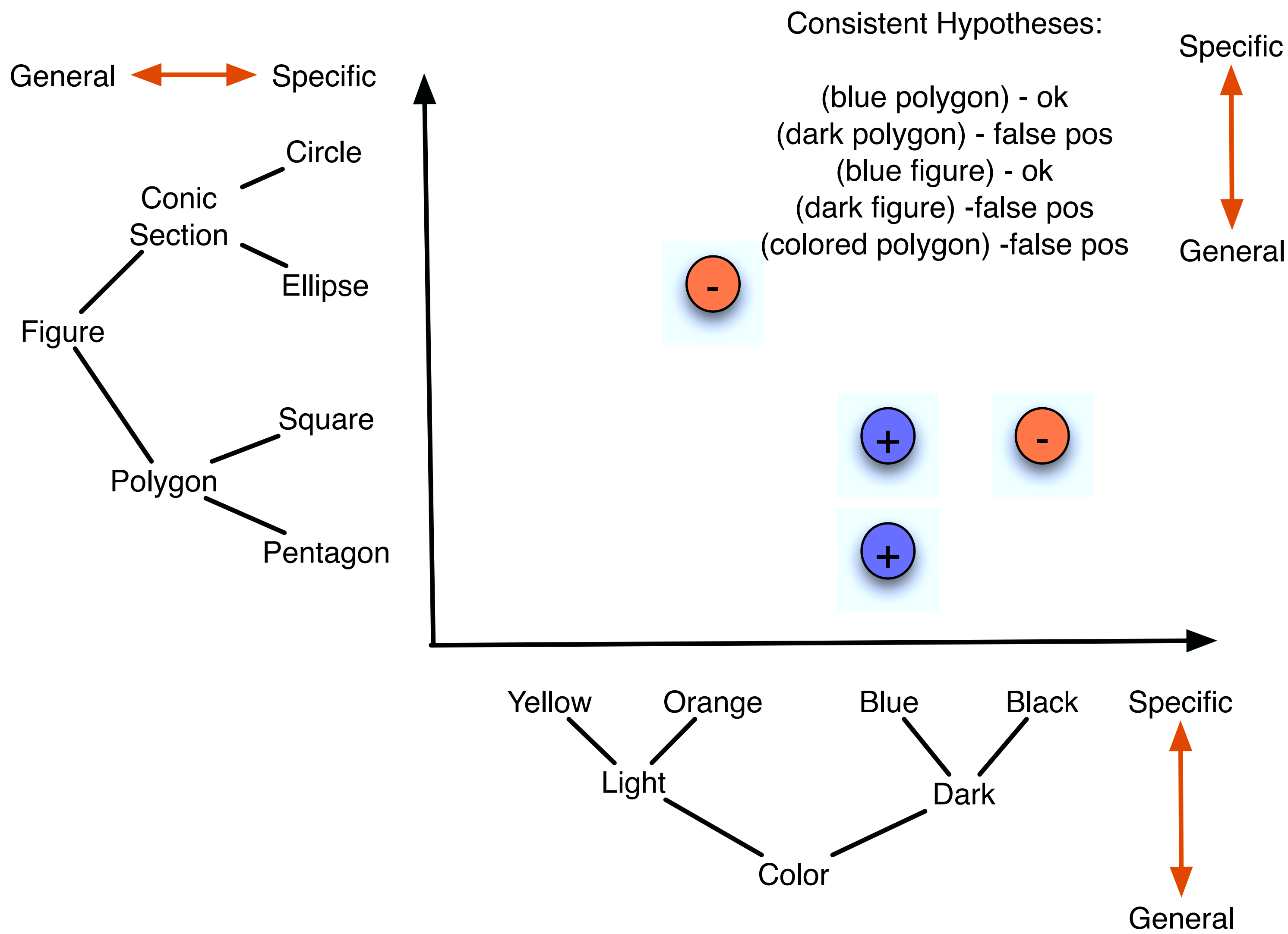
Hypothesis Filtering and Refinement



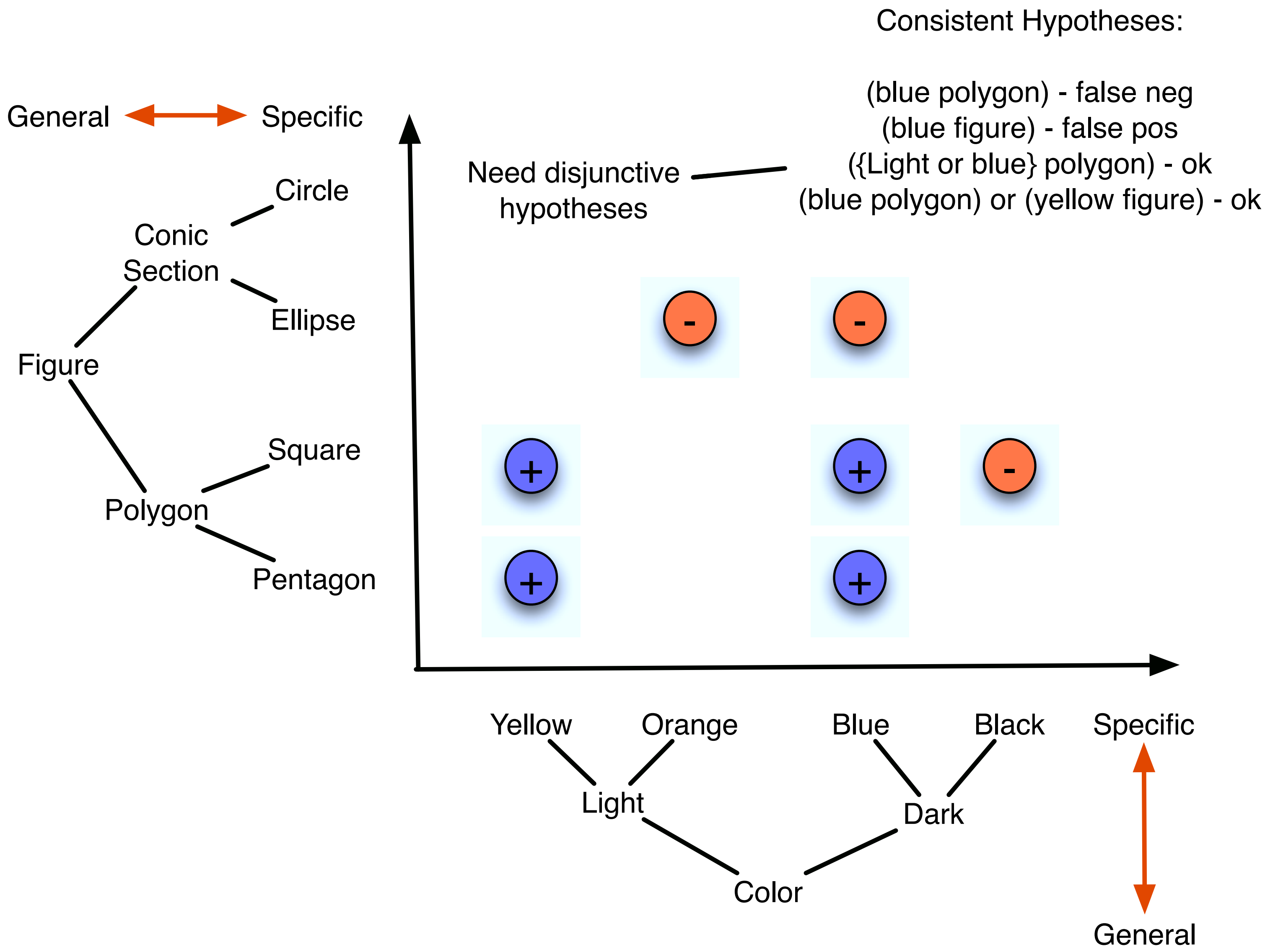
Hypothesis Filtering and Refinement (2)



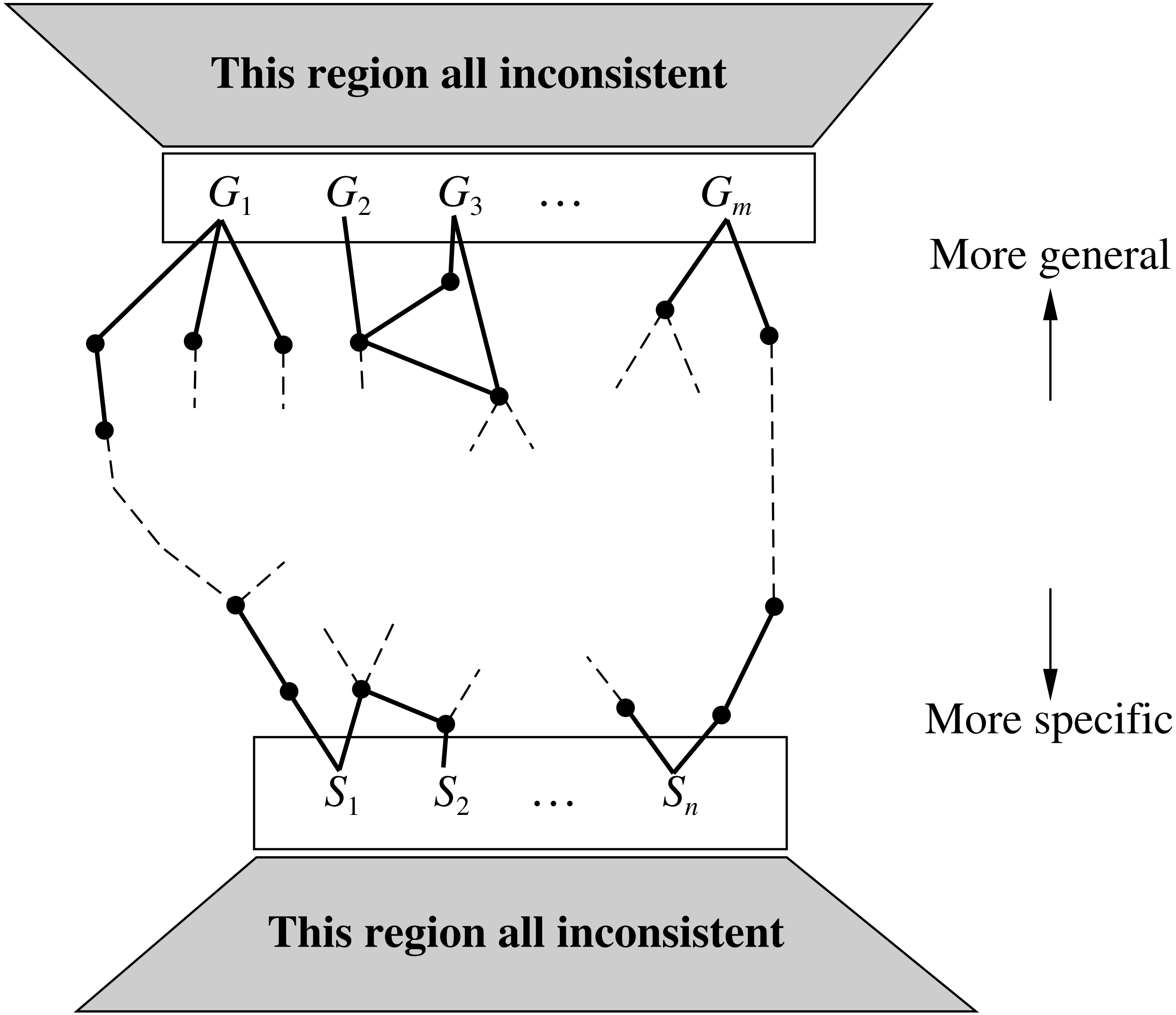
Hypothesis Filtering and Refinement (3)



Hypothesis Filtering and Refinement (4)



Version Space



Beauty of the Version Space

- The version space represents the entire space of consistent hypotheses.
- But only **implicitly** via the boundaries of that space:
  - S - the set of most specific hypotheses, all of which cover every positive example and no negative examples, but as **few** of the other instances as possible.
  - G - the set of most general hypotheses, all of which cover every positive example and no negative examples, but as **many** of the other instances as possible.
- As examples are presented, the version space contracts by:
  - Generalizing the hypotheses in S to cover new positive examples.
  - Specializing the hypotheses in G to avoid covering new negative examples.
- When all pos and neg examples have been seen, the current version space represents all possible hypotheses that are consistent with each example.

Candidate Elimination Algorithm

Init G to max-general hypos
Init S to max-specific hypos
$\forall d_i \in D$ do:

- If $C(d_i) = +$ then:
  - $\forall g \in G \ni$ inconsistent(g,d): $G \leftarrow G - g$
  - $\forall s \in S \ni$ inconsistent(s,d):
    - $S \leftarrow S - s$
    - Add all **minimal generalizations** s_{mg} of s to S, where:
      - consistent(s_{mg},d_i), and
      - $\exists g \in G \ni$ more-general(g,s_{mg})
    - $\forall s_1, s_2 \in S \ni$ more-general(s_1, s_2) $S \leftarrow S - s_1$

Candidate Elimination Algorithm (2)

- If $C(d_i) = -$ then:
  - $\forall s \in S \ni$ inconsistent(s,d): $S \leftarrow S - s$
  - $\forall g \in G \ni$ inconsistent(g,d):
    - $G \leftarrow G - g$
    - Add all **minimal specializations** g_{ms} of g to G, where:
      - · consistent(g_{ms},d_i), and
      - · $\exists s \in S \ni$ more-general(g_{ms},s)
    - $\forall g_1, g_2 \in G \ni$ more-general(g_1, g_2) $G \leftarrow G - g_2$

The target concept is precisely learned when G = S.
Before this convergence of G and S, the system may give ambiguous classifications of some test cases: G may include it, while S may exclude it.
E.g. (blue ellipse) in the upcoming example.

Candidate Elimination Algorithm (3)

In general:

- S set summarizes (in most specific form) ALL pos examples seen so far.
  - $\forall h(\exists s \in S \ni$ more-general(s,h)$) \rightarrow$ h fails to cover at least one pos eg., d+
  - Thus, d+ is a false negative of h.
- G set summarizes (in most general form) ALL neg examples seen so far.
  - $\forall h(\exists g \in G \ni$ more-general(h,g)$) \rightarrow$ h includes at least one neg eg., d-
  - Thus, d- is a false positive of h.

Candidate Elimination Example

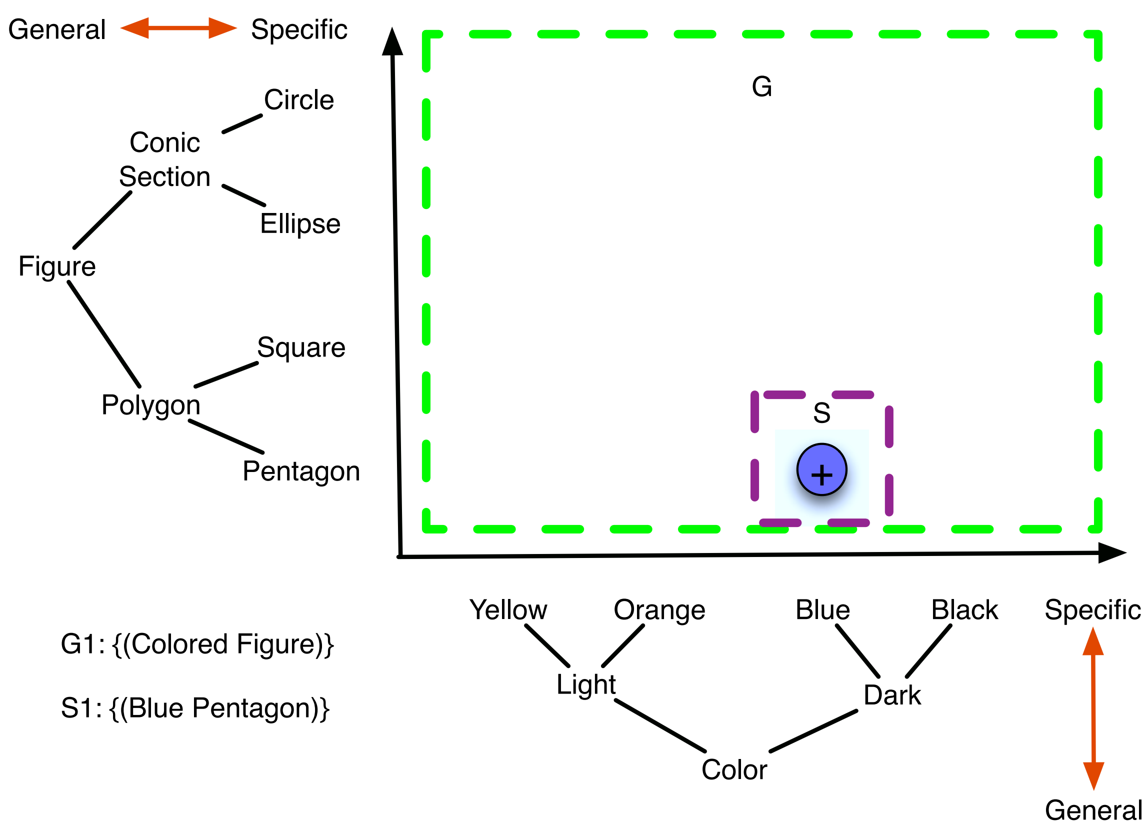
Assume the following list of training examples:

1. (blue pentagon) - positive
2. (blue square) - positive
3. (orange ellipse) - negative
4. (black square) - negative

Use Candidate Elimination to filter the hypothesis space.

- Init: G = {(Colored, Figure) }
- Init: S = {(nil, nil)}

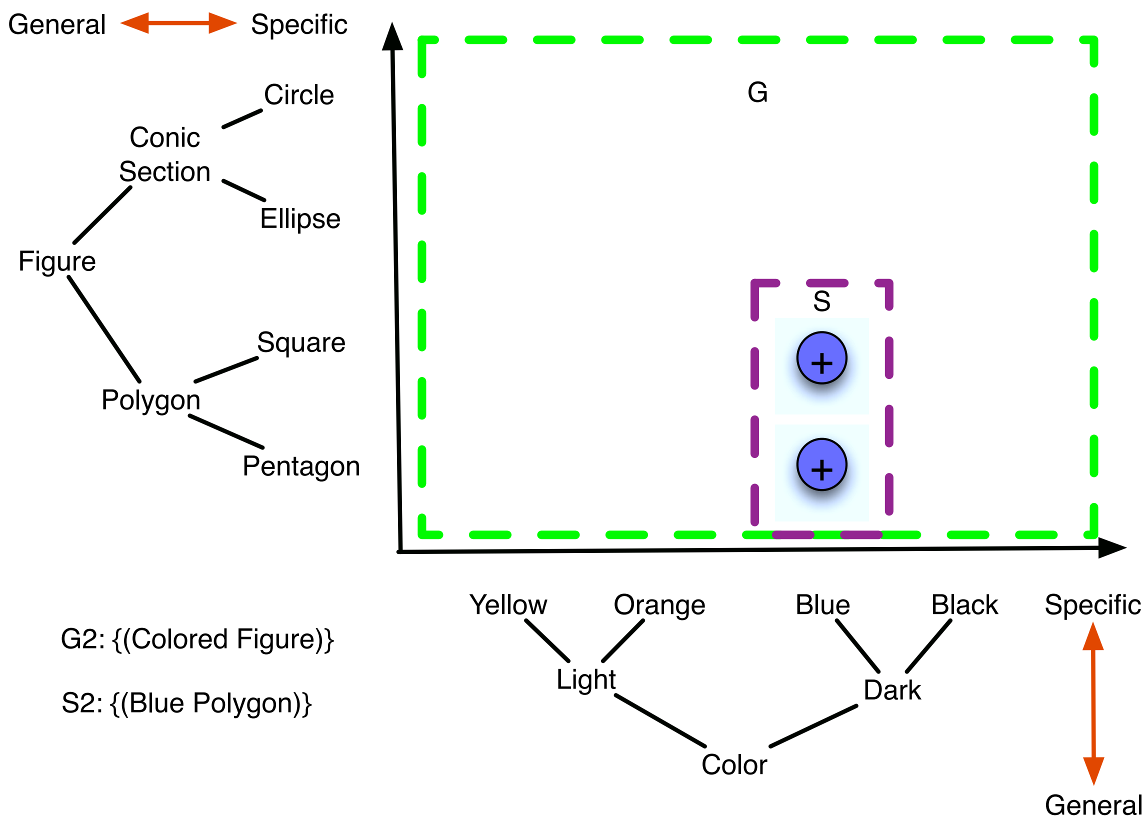
Candidate Elimination Example (2)



On seeing d_1 = (blue pentagon)(+)

- G is unchanged, since G's only member is consistent with d_1.
- S's only member is inconsistent with d_1, so it is removed and minimally generalized to cover d_1.

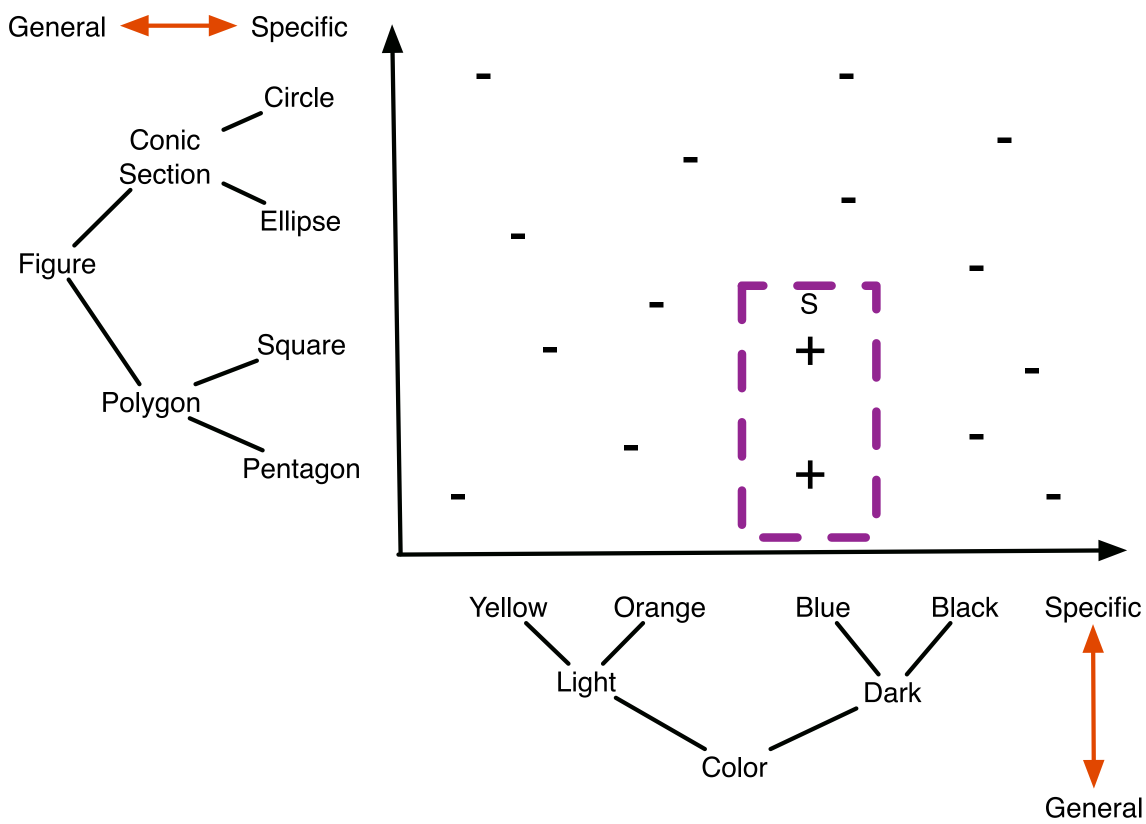
Candidate Elimination Example (3)



On seeing $d_2 =$ (blue square)$(+)$

- G is unchanged, since G's only member is consistent with d_2.
- S's only member is inconsistent with d_2, so it is removed and minimally generalized to cover d_2.

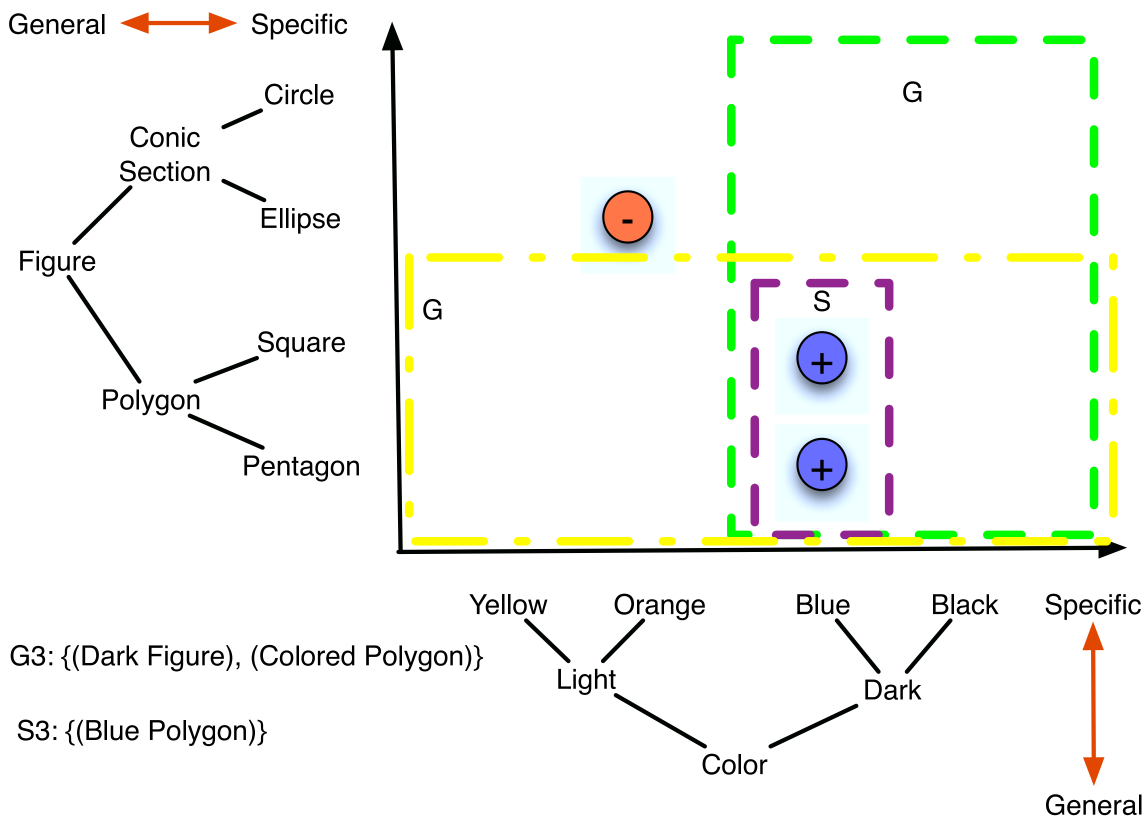
Semantics of a Hypothesis



The hypotheses in S and G have the same semantics:

- Everything that satisfies their description is a positive example.
- Everything else is a negative example.

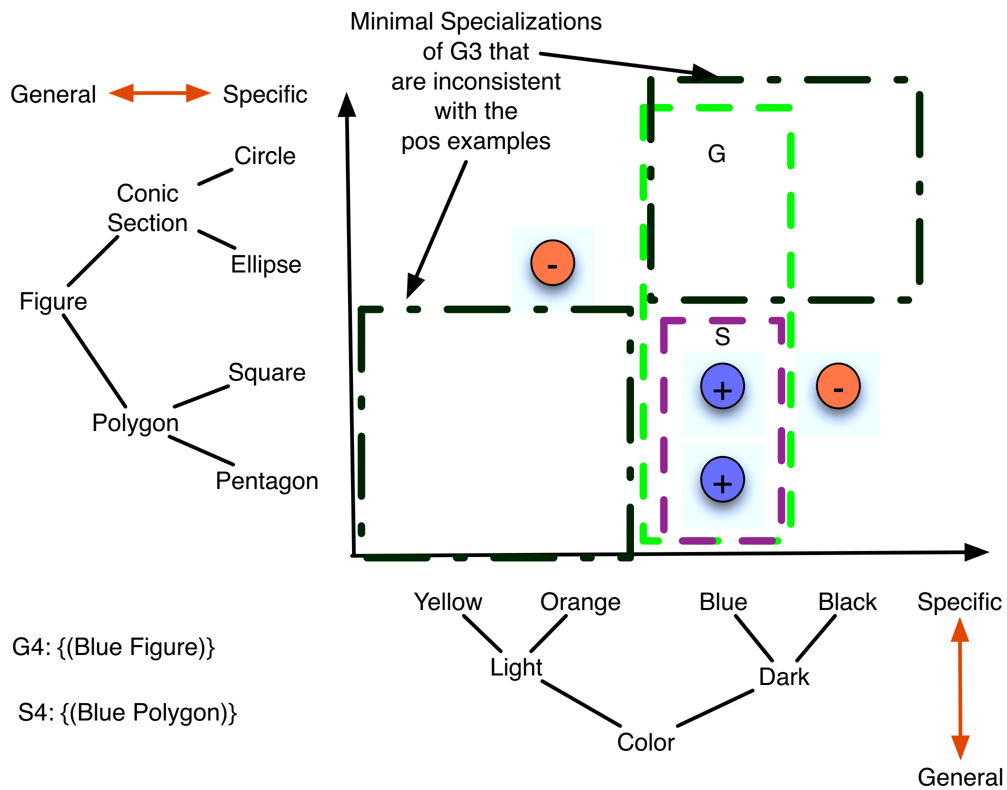
Candidate Elimination Example (4)



On seeing $d_3 =$ (orange ellipse)(-)

- G's only member is inconsistent with d_3, so it is removed and minimally specialized to avoid d_3.
- S's only member is consistent with d_3, so no change.

Candidate Elimination Example (5)



On seeing d_4 = (black square)(-)

- Both of G's members are inconsistent with d_4, so remove and specialize both. But only one of the specializations is more general than a member of S (i.e. covers the pos egs.).
- S's only member is consistent with d_4, so no change.

Pros and Cons of Candidate Elimination

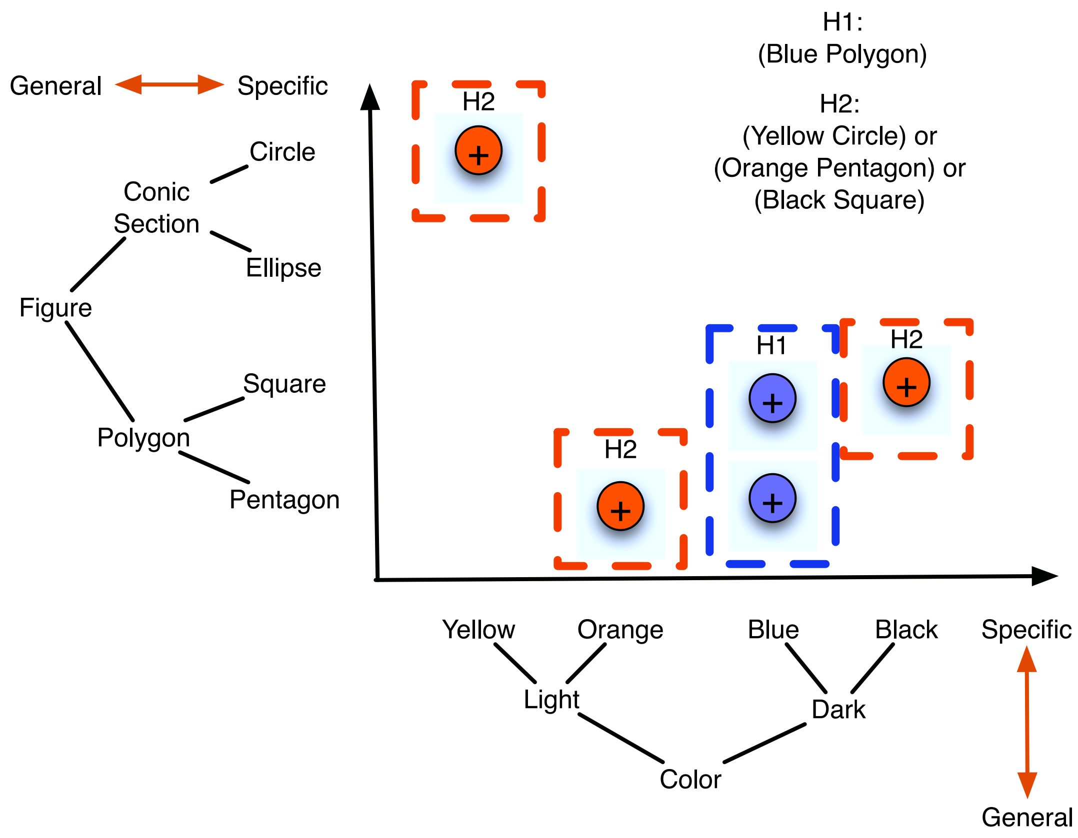
Pros:

- One-shot learning
- Independent of ordering of instances
- Elegant model for hypothesis-space filtering

Cons:

- Cannot handle noisy data (i.e. pos examples that are really negative).
- Difficulties with disjunctive concepts (e.g. (red polygon) or (dark circle))
- Totally dependent upon the attribute hierarchy.

Inductive Learning Bias



- 4 x 4 = 16 instances $\longrightarrow 2^{16} = 65536$ hypotheses.
- But only 7 x 7 = 49 conjunctive hypos are expressible in the rep.
- The rep **strongly biases** what the system can learn.

Expressibility - Generalizability Tradeoff

- Assume that unlimited disjunctions are allowed in the hypotheses.
- Consider a simple training set: $x_1(+), x_2(+), x_3(-), x_4(-)$
- After seeing these examples, the candidate-elimination algorithm would have:
  - G $= \{(\neg x_3 \wedge \neg x_4)\}$
  - S $= \{(x_1 \vee x_2)\}$
  - since these are the most general and most specific (respectively) hypotheses that:
    * are expressible in the representation language
    * contain all pos examples and exclude all neg examples.
- But now, any new example, x_5, will be ambiguous, since G will consider it positive, and S will consider it negative.
- **Only** the previously-seen examples can be unambiguously classified.
- To learn target concept, system must see **every** pos example of it!
- Cannot generalize beyond what it sees → memorization, not learning!

Inductive Leaps

- As shown above, a representation in which where EVERY possible combination of instances is a legal hypothesis:
  - has no inductive bias, but
  - has no ability to generalize beyond what it sees.
  - So it has no ability to classify previously-unseen examples.
- The inductive bias in a language enables **inductive leaps** beyond the immediate evidence.
  - In generalizing an $s \in S$, the new s will often include more pos egs than seen so far.
  - In specializing a $g \in G$, the new g will often exclude more neg egs than seen so far.
- In both cases, the system **takes a chance**: it makes an inference that is not purely deductive!
- So induction, like abduction, = non-deductive (possibly faulty) reasoning.
- Rep, via its bias, determines types of risk the learning system takes.