Ch2. Concept Learning

S. Visa

Introduction

- Concept
  1. = boolean function that has true or false values over a specified domain of inputs (= set I of instances)
  2. = set for which instances in the set have value true

- Instance = a valid input to the concept's function

- Ex. of concepts
  - “animal” – fct. that is true for obj. that are animals and false for obj. not animals
  - “bird”

- People think of concepts in vague terms
  - e.g. hot, pleasant, appropriate
  - 1964, United States Supreme Court Associate Justice Potter Stewart said something close to the following: "I can't define pornography, but I know it when I see it“

- Concept learning
  1. an attempt to find a working definition for a concept, although it may, or may, not be possible to precisely do so
  2. the process of inferring a boolean function from a set of training pairs, E, each pair consisting of inputs and the corresponding boolean output

     $(x\ ,\ c(x))$ where $c(x) = 0$ or 1 (no or yes)

Example - EnjoySports

- Training examples – see table

Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

- Each x is an attribute vector

e.g. x=(sunny,warm,normal,strong,warm,same)
➔c(x)=1

- Note:
  - Sky has 3 possible values (Sunny, Rainy, Overcast)
  - all other attributes have 2

Hypothesis

- H = hypothesis space for concepts = power set of n instances = 2^n

  Ex. 4 instance in instance space I: hot, high, cold, low

  → H has 2^4 = 16 possible different concepts: (hot,high); (cold,high); (hot,low); (cold,low); (hot,cold,low); (hot, high, cold, low); … (high); (cold)…

- h = boolean function, possible candidate to be the concept c
  ➔ any set of instances for the concept
  ➔ any h makes concept prediction for all instances
  ➔ instances in the hypothesis set are true and all other false
  ➔ defines a concept (it might not be my concept c!)

- c = concept = target hypothesis

- Goal of concept learning
  1. find h = c on the entire domain
  2. (less preferably) find h = c on training data and approximates well on rest
  3. (even less preferably, but most commonly) find h = c on high percentage of training data and approximates well on rest

- Generalization
  - a hypothesis that approximates well c in the entire domain
  - NOT an h that learns perfectly the training data! → overfitting

Hypothesis as conjunctions of attributes

- Represent hypothesis h as a tuple
  - ?→ any value for this attribute satisfies h (don't care)
  - 0→ no possible value for this attr. satisfies h
  - Specific value

- Ex.
  - h = (0,0,0,0,0,0) → most specific h
    - there are no days to EnjoySport
    - classify **all instances as "no"**
  - h = (0,Cold,0,0,0,0)
    - same as above (because " 0 and whatever" is 0)
    - if any attr. has 0➔ no choice of input can satisfy h
  - h = (?,?,?,?,?,?) → most general h
    - EnjoySport on all days
    - classify **all instances as "yes"**
  - h = (?,Cold,High,?,?,?)
    - EnjoySport on Cold days with High humidity, no matter what the status of the sky, wind, water, or forecast
    - Cold and high

On the number of hypotheses

- 3*2*2*2*2*2=96 possible inputs (instance space I)
  ➔2^{96} possible concepts (~10^{28}) – HUGE search space!!

- ➔using conjunction repres. (and) → 5*4*4*4*4*4 = 5,120 hypotheses (I added 0 and ?)

- We usually search for c (or an approx of c) in a smaller subset of H:

  1+4*3*3*3*3 = 973 hypothesis

  when using "and"-ing (1 is from null h in which tuples have at least one 0)

- instance vs. hypothesis – can an instance be a h?
  - Yes, but (most likely) it will recognize only that specific instance

Prototypical concept learning problem

- Given:
  - Instances I
    - possible days described by attributes Sky, AirTemp, …
  - Target fct. c
    - EnjoySport: I → {0,1}
  - Hypothesis H
    - conjunction of attributes
    - e.g. (?, Cold, High, ?,?,?)
  - Training examples D
    - positive and negative examples of c
    - e.g. $(x_1, c(x_1)), (x_2, c(x_2)), \ldots, (x_n, c(x_n))$
- Determine:
  - h s.t. h(x) = c(x) for all x in domain D

Ex. - Concept learning problem

1. Task T
   – find h consistent with training data, c(i) = h(i) for every i in D
2. Performance P
   – no. of instances of D for which c(i) = h(i)
3. Experience E
   – training examples from D

- Inductive Learning Assumption
  – "small" hypothesis that do well on the training examples will do well on unobserved examples!

General-to-specific ordering of hypotheses

- h_2 is more general than or equal to h_1 if h_1 is a subset of $h_2 \rightarrow h_2 \geq_g h_1$
  - h_2 classifies as yes all instances of h_1, plus possible some more
  - ex.

    h1= (sunny,?,?,strong,?,?)

    h2= (sunny,?,?,?,?,?)

    (Q: if both do equally well on training, which one you prefer?)

- h_1 is more specific than or equal to $h_2 \rightarrow h_1 \leq_s h_2$

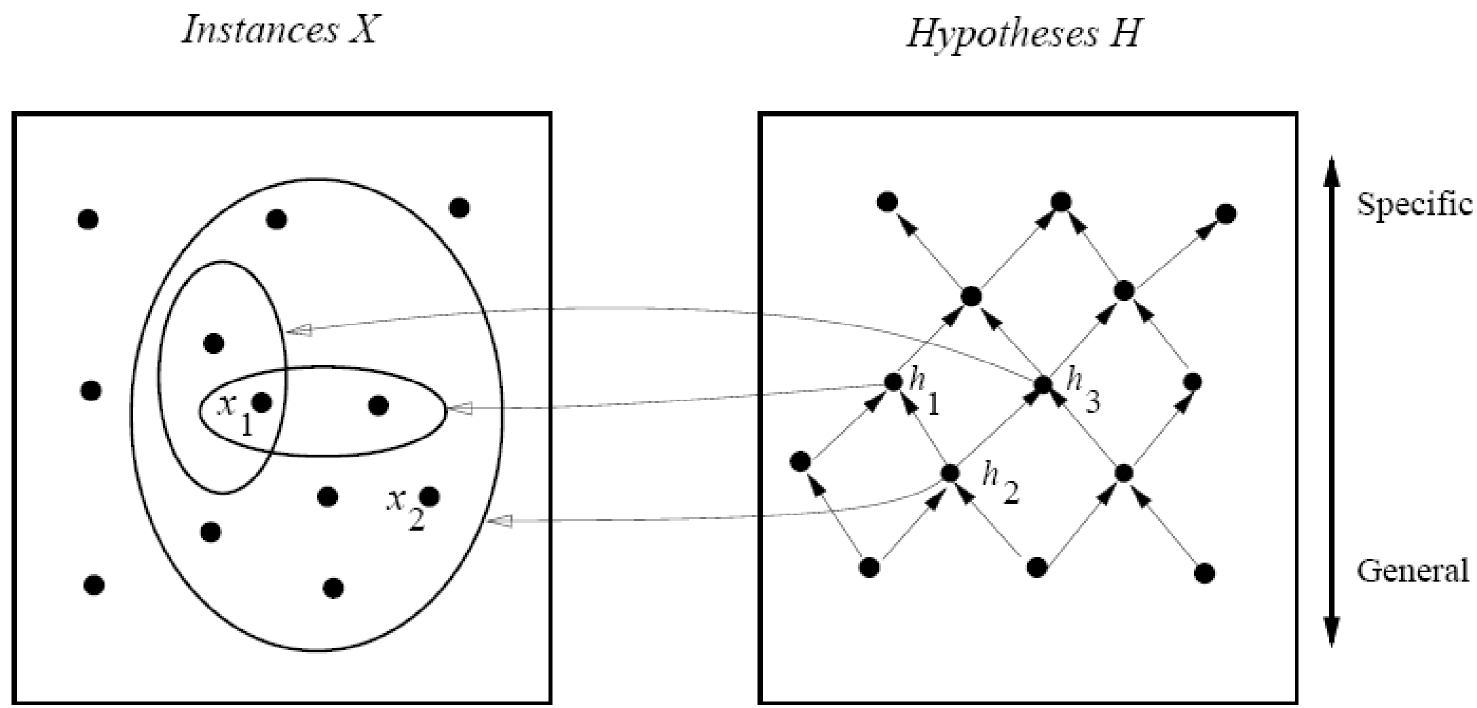
  ➔ h_1 eliminates some of the instances that h_2 said where true

- maximally specific hypothesis for examples in D
  - a hypothesis, **s**, which contains only positive instances of D and such that any more specific hypothesis is missing a positive instance of D that is in s

- maximally general hypothesis
  - an hypothesis, **g**, which contains only positive instances of D, but such that any more general hypothesis includes a negative instance from D

- Obs.: there can be more than one s and g for a given training set D

Instance, hypothesis, and more-general-than



x_1 = <*Sunny, Warm, High, Strong, Cool, Same*>
x_2 = <*Sunny, Warm, High, Light, Warm, Same*>

h_1 = <*Sunny, ?, ?, Strong, ?, ?*>
h_2 = <*Sunny, ?, ?, ?, ?, ?*>
h_3 = <*Sunny, ?, ?, ?, Cool, ?*>

Questions: $h_2 \geq h_1$? $h_3 \geq h_2$?

Find-S: finding a maximally specific hypothesis

1. Initialize h to the most specific hypothesis in H
2. For each positive training instance x
   - For each attribute constraint a_i in h

     If the constraint a_i in h is satisfied by x

     Then do nothing

     Else replace a_i in h by the next more general constraint that is satisfied by x
3. Output hypothesis h

Find-S - example

- Start with most specific h=(0,0,0,0,0,0)→all predictions are "no"
- Look at pos. ex. one by one adjusting h s.t. it classifies correctly the current pos. ex.
- Never look at neg. ex.
- Apply Find-S to EnjoySports training set

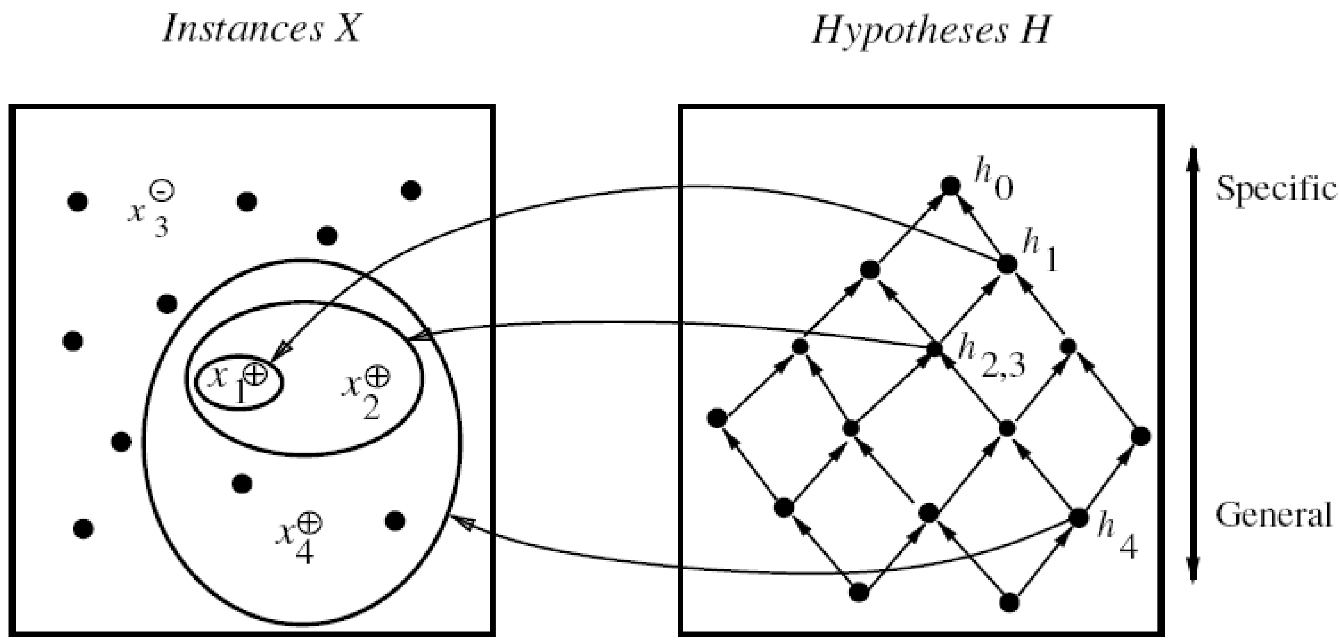
Next pos. ex.	New hypothesis
	h_0=(0,0,0,0,0,0)
x_1=(s,w,n,s,w,s)	h_1=(s,w,n,s,w,s)
x_2=(s,w,h,s,w,s)	
x_4=(s,w,h,s,c,c)	

Find-S - example

- Start with most specific h=(0,0,0,0,0,0)→all predictions are "no"
- Look at pos. ex. one by one adjusting h s.t. it classifies correctly the current pos. example
- Never looks at neg. ex.
- Apply Find-S to EnjoySports training set

Next pos. ex.	New hypothesis
	h_0=(0,0,0,0,0,0)
x_1=(s,w,n,s,w,s)	h_1=(s,w,n,s,w,s)
x_2=(s,w,h,s,w,s)	h_2=(s,w,?,s,w,s)
x_4=(s,w,h,s,c,c)	h_3=(s,w,?,s,?,?)

Find-S - example



h_0 = <∅, ∅, ∅, ∅, ∅, ∅>

x_1 = <*Sunny Warm Normal Strong Warm Same*>, +

h_1 = <*Sunny Warm Normal Strong Warm San*

x_2 = <*Sunny Warm High Strong Warm Same*>, +

h_2 = <*Sunny Warm ? Strong Warm Same*>

x_3 = <*Rainy Cold High Strong Warm Change*>, -

h_3 = <*Sunny Warm ? Strong Warm Same*>

x_4 = <*Sunny Warm High Strong Cool Change*>, +

h_4 = <*Sunny Warm ? Strong ? ?* >

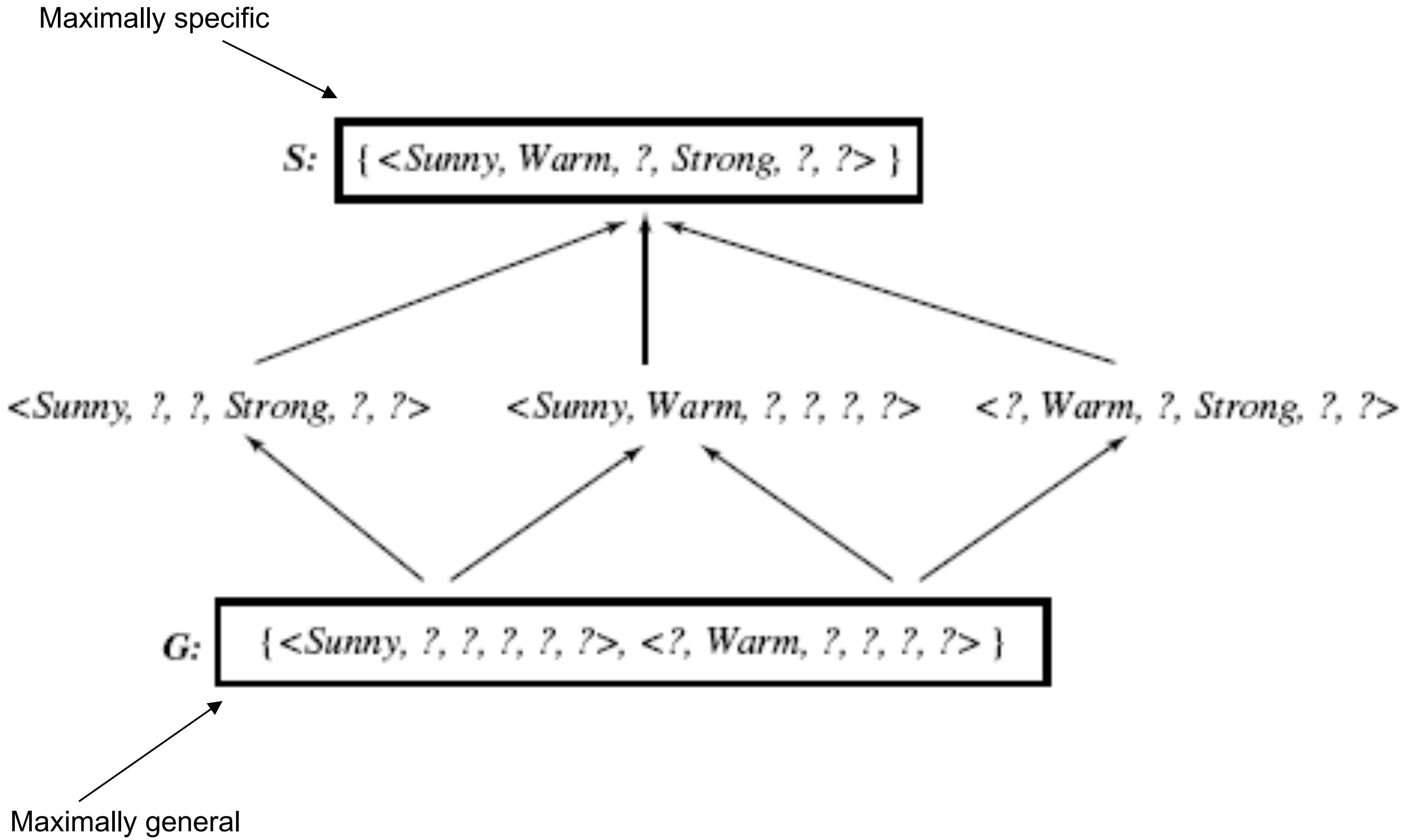
Obs. and limitations of Find-S

- Covers all (only) pos training examples
- Yield the most specific conjunction hypothesis that matches training data but maybe a more general conjunction hypothesis learns better c (better generaliz.)
- Can’t tell whether has learned c – maybe by looking at counterex. I could have learned c better
- Learns poorly more complicated data

Version space (VS)

- **Version space** = subset of all h from H consistent with all tr. data in D
- h is **consistent** with tr. ex. D of target concept ⇔ h(x) = c(x) for all (x, c(x)) in D
- We describe/repres. VS by the **general** (G) and **specific** (S) boundaries
- G of VS = set of its maximally general members
- S of VS = set of its maximally specific members
- Th.2.1(p.32) Version space representation th.: every member of VS lies between G and S

Ex. of version space



Note that S and G are hypotheses but also sets of maximally specific and maximally general hypotheses!

Candidate elimination alg.(p.33)

1. Start with
   - S_0 most specific h (all attr. 0)
   - G_0 most general h (all attr. ?)
2. Repeatedly adjust S and G by removing any h that is inconsistent with the tr. ex.
   - when h - inconsistent with pos. ex. p ➔ h is too specific
     - if $h \in G$, eliminate h
     - if $h \in S$, moderate h to include p
   - when h - inconsistent with neg. ex. n ➔ h is too general
     - if $h \in S$, eliminate h
     - if $h \in G$, moderate h to exclude n

OBS. Convergence of Candidate elimination alg. (p.37)

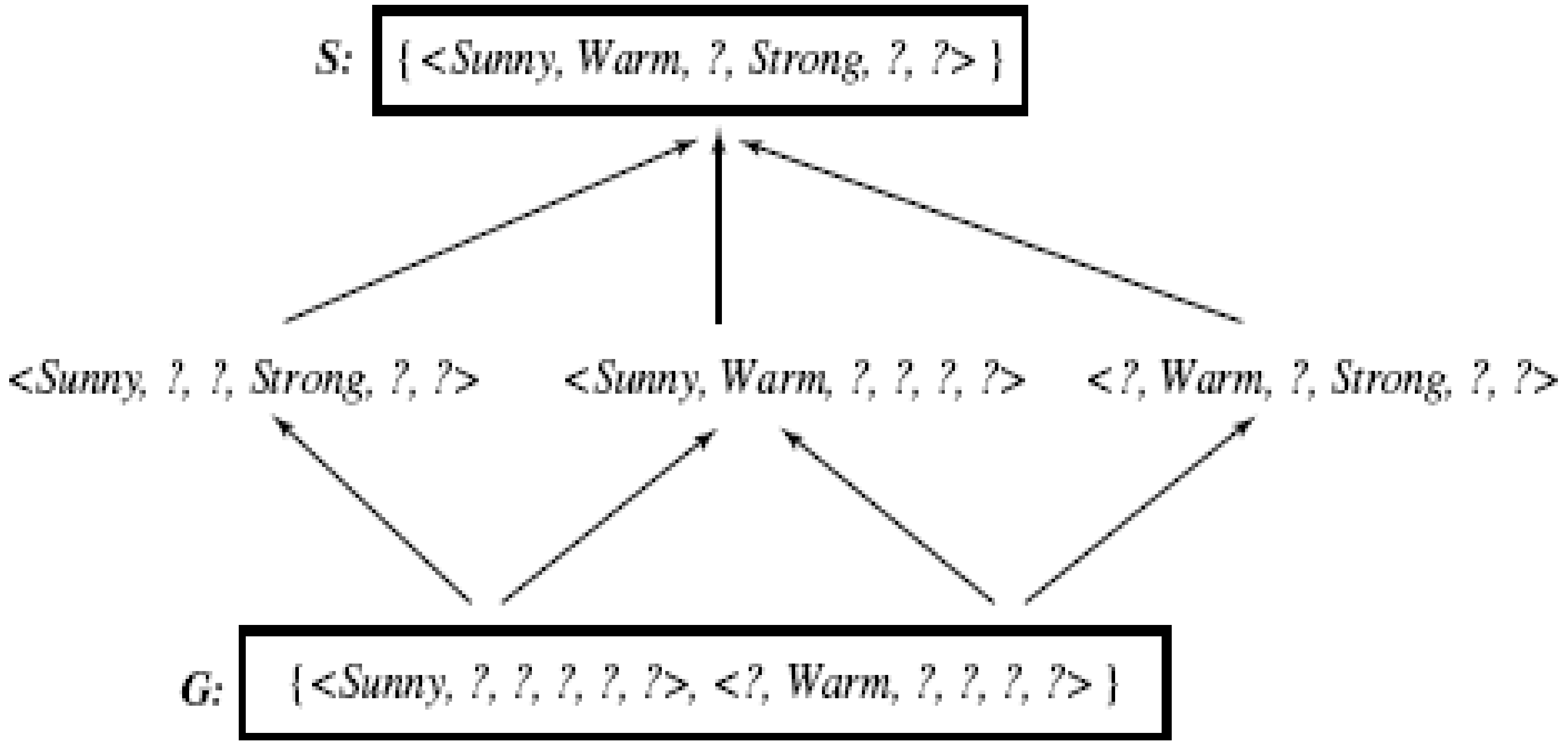
- Given an unlimited number of examples of a precisely existing concept, candidate-elimination will converge to the unique h that describes the target concept as long as:
  - the training examples have no errors
  - the hypothesis set contains the target concept

Ex. Trace candidate elim. alg.

S_0: {<Ø, Ø, Ø, Ø, Ø, Ø>}

G_0: {<?, ?, ?, ?, ?, ?>}

Ex. Trace candidate elim. alg.



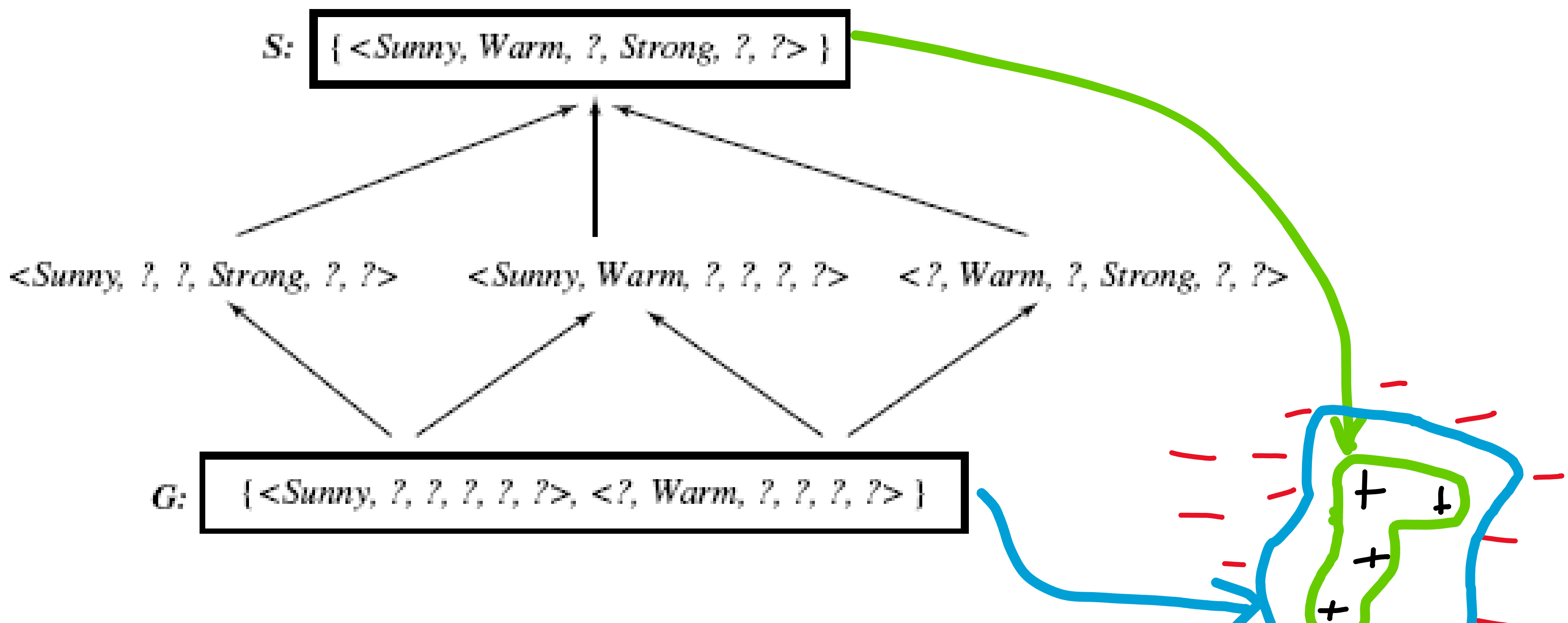
Obs. on VS candidate elimination

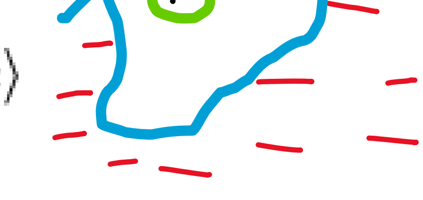
- The learned VS is independent of the order of tr. data
- As more tr. data arrive ➔ S and G move toward each other
- Target concept c is learned when S and G converge to a single h
- Err in data➔ ∅ VS
- ➔ Candidate elim. alg. converges to true h = c if
  1. There are no err. in train data
  2. h = c is in H

Partially learned concepts (p.38)

- Its VS contains more than one hypothesis
- ➔ an instance must fit the concept if every h of the max. specific hypot. S classifies it as pos.
- Classification rule for partially learned candidate elimination
  - Classify as pos. any ex. satisfying every h in S
  - Classify as neg. any ex. eliminated by every h in G
  - Do not classify anything else

How should these be classified?





$\langle Sunny\ Warm\ Normal\ Strong\ Cool\ Change \rangle$

$\langle Rainy\ Cool\ Normal\ Light\ Warm\ Same \rangle$

$\langle Sunny\ Warm\ Normal\ Light\ Warm\ Same \rangle$ no classification (not rejected by G and not accepted by S)

Inductive bias (p.39)

- **Def.1** Is preferring one h over equally good ones

- Types of inductive bias
  - Restrictive H – e.g. use only "and"-ing representation of solution (in EnjoySports ex.)
  - Preference for smaller h

- For EnjoySports ex. – it is the choice of a hypothesis set (hypothesis representation) e.g. using conjunction repres. (and)

- **Def.2** Is the set of assumptions that together with the training set guarantees that the learned hypothesis makes only valid predictions about c.

- Ex. of S and G in an <u>unbiased</u> hypothesis space (this is a bad solution unable to generalize):
  - Training data:
    - x_1, x_2, x_3 - pos. ex.
    - x_4, x_5 – neg. ex.
  - Apply candidate elimination

    ➔S = x_1 or x_2 or x_3

    ➔G = NOT(x_4 or x_5)
  - Test: x_6 will satisfy G but not S ➔ unable to classify any new ex.

- OBS. Without an inductive bias the obtained VS provides no generalization!!
- OBS. The restriction to conjunctions in EnjoySport ex. is a strength not a weakness!!

Three learners with different biases

- Rote learner: store ex. and classify x if has been seen before
  - No inductive bias: besides the tr. ex. there is no additional assumption
- Candidate elimination alg.
  - Inductive bias: target c can be represented in its H
- Find S
  - Stronger inductive bias:
    - Inductive bias: target c can be represented in its H
    - All instances are neg. unless they satisfy S

Summary

- Concept learning as search through H
- General to specific ordering over H
- Find-S alg.
- Version space candidate elimination alg. – finds VS
- S and G boundaries characterize learner's uncertainty
- Generalization possible only if learner is biased