Chapter 1:
Concept Learning

Adopted from Machine Learning, Mitchell

Logistics

- Course Instructor: Tanmoy Chakraborty (NLP)
  https://tanmoychak.com/

- Guest Lecture: TBD (possibly from the industry)
- TAs: Sahil, Aswini, Palash, Prottoy, Vaibhav, Soumyodeep, Anand

- Course page: https://lcs2-iitd.github.io/ELL409-2401/
- Discussion forum: Piazza (https://piazza.com/iitd.ac.in/fall2024/ell409)
  Access Code: *ell409mli*

- For assignment submission: Moodle
- Group Email: 2401-ELL409@courses.iitd.ac.in

Course Directives

- **Class Time:** Mon and Thu, 8:00 AM - 9:20 AM
- **Office Hour:** as per requirement (email me to schedule an appointment)
- **TA Hour:** (Please email at least an hour before to confirm the meeting location)
  - **Monday** 4 PM to 5 PM: Vaibhav (mt1210236@iitd.ac.in)
  - **Tuesday** 4 PM to 5 PM: Soumyodeep (aiy237526@scai.iitd.ac.in)
  - **Wednesday** 4 PM to 5 PM: Sahil (eez238354@ee.iitd.ac.in)
  - **Wednesday** 3 PM to 4 PM: Aswini (eez238359@iitd.ac.in)
  - **Thursday** 4 PM to 5 PM: Palash (sondhanil1@gmail.com)
  - **Friday** 3 PM to 4 PM: Anant (aib232068@scai.iitd.ac.in)
- **Room:** LH114

Timeline

- **Project Finalization:** 10/08/2024
- **Quiz 1:** 12/08/2024
- **Assignment 1:** 13/08/2024
- **Quiz 2:** 05/09/2024
- **Mid-Term:** 12/09/2024 - 18/09/2024
- **Assignment 2:** 20/09/2024
- **Assignment 3:** 17/10/2024
- **Quiz 3:** 21/10/2024
- **Quiz 4:** 11/11/2024
- **Major:** 16/11/2024 - 23/11/2024
- **Project assessment:** Before endsem

Some announcements

- Coding practice twice a month (led by the TA) – outside the regular lecture hours

Some announcements

- Coding practice twice a month (led by the TA) – outside the regular lecture hours
- Sample questions for practice before midterm and major

Some announcements

- Coding practice twice a month (led by the TA) – outside the regular lecture hours

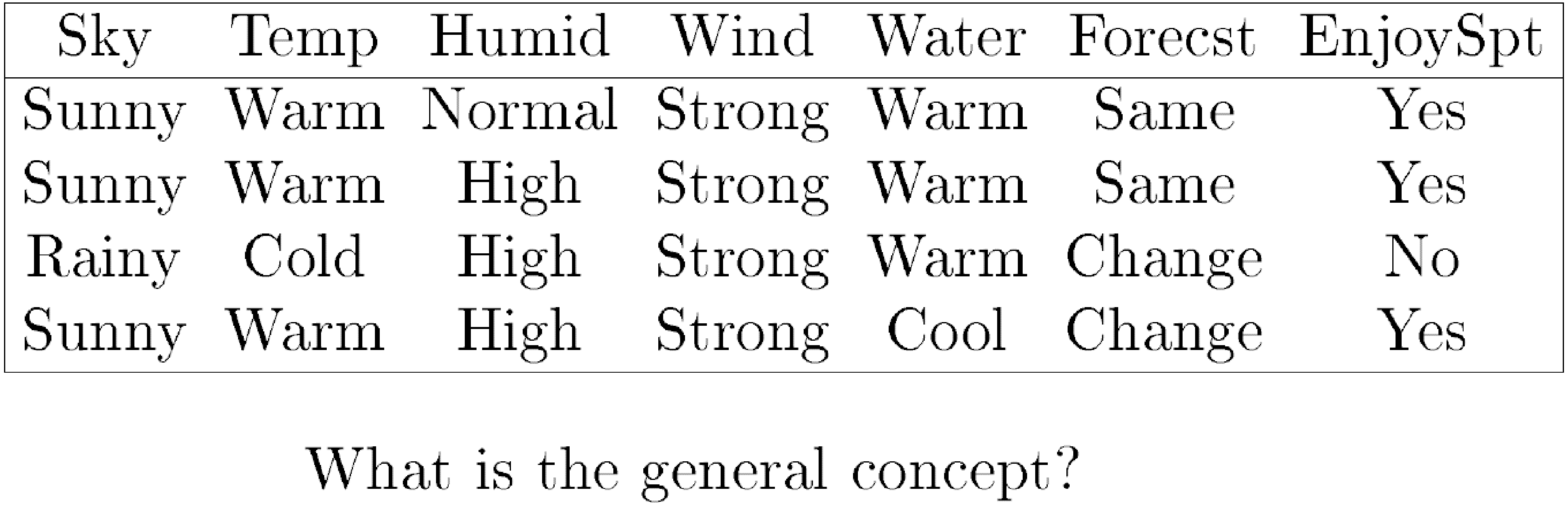
- Sample questions for practice before midterm and major

- Quiz every class - 8:00 AM to 8:05 AM

Outline

- Learning from examples
- General-to-specific ordering over hypotheses
- Version spaces and candidate elimination algorithm
- Picking new examples
- The need for inductive bias

Training Examples for *EnjoySport*



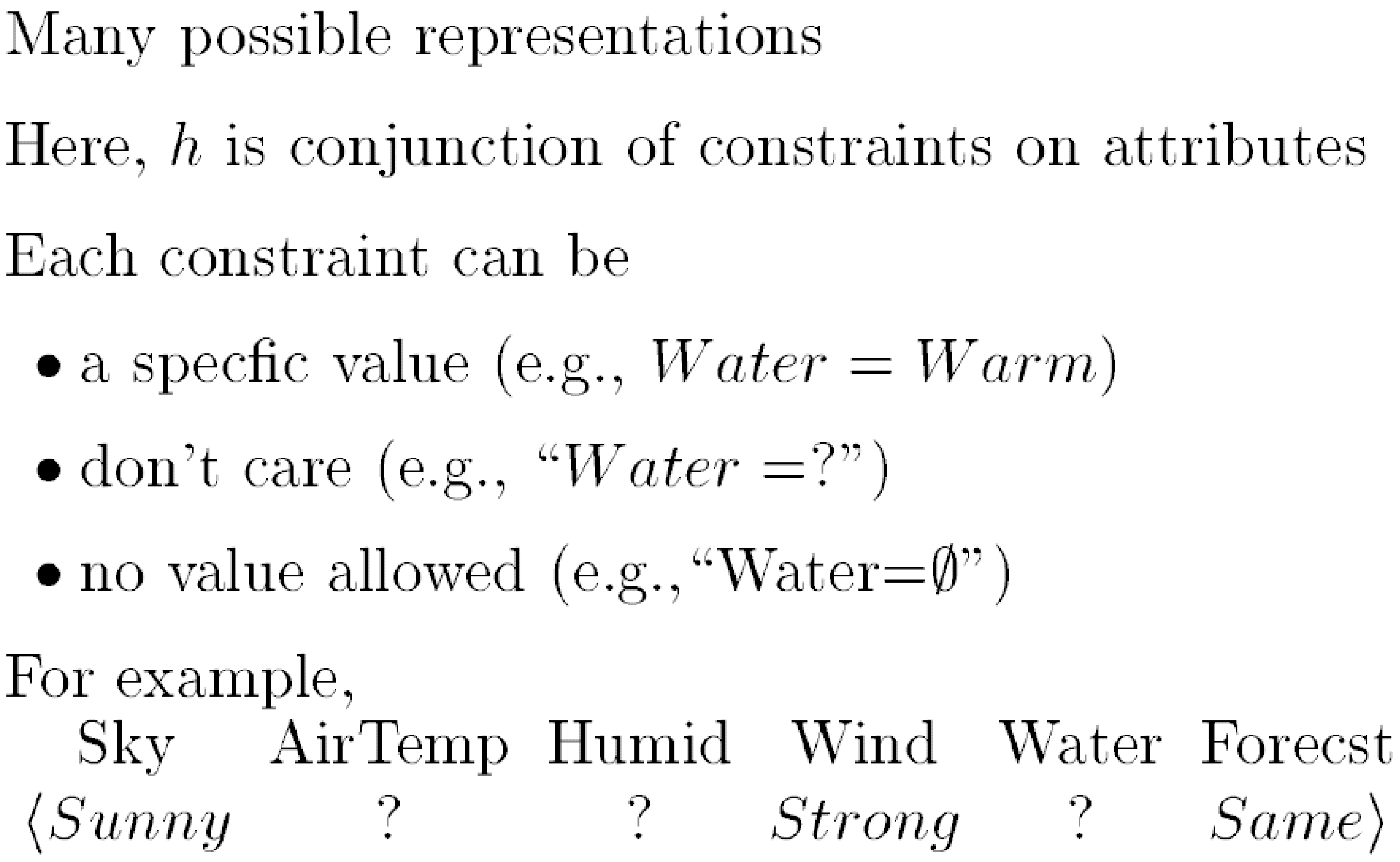
Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

What is the general concept?

Concept Learning

Inferring a Boolean-valued function from training examples of its input and output

Representing Hypotheses



Many possible representations

Here, h is conjunction of constraints on attributes

Each constraint can be

- a specfic value (e.g., $Water = Warm$)
- don't care (e.g., "$Water =?$")
- no value allowed (e.g.,"Water=∅")

For example,

Sky	AirTemp	Humid	Wind	Water	Forecst
⟨*Sunny*	?	?	*Strong*	?	*Same*⟩

Notations

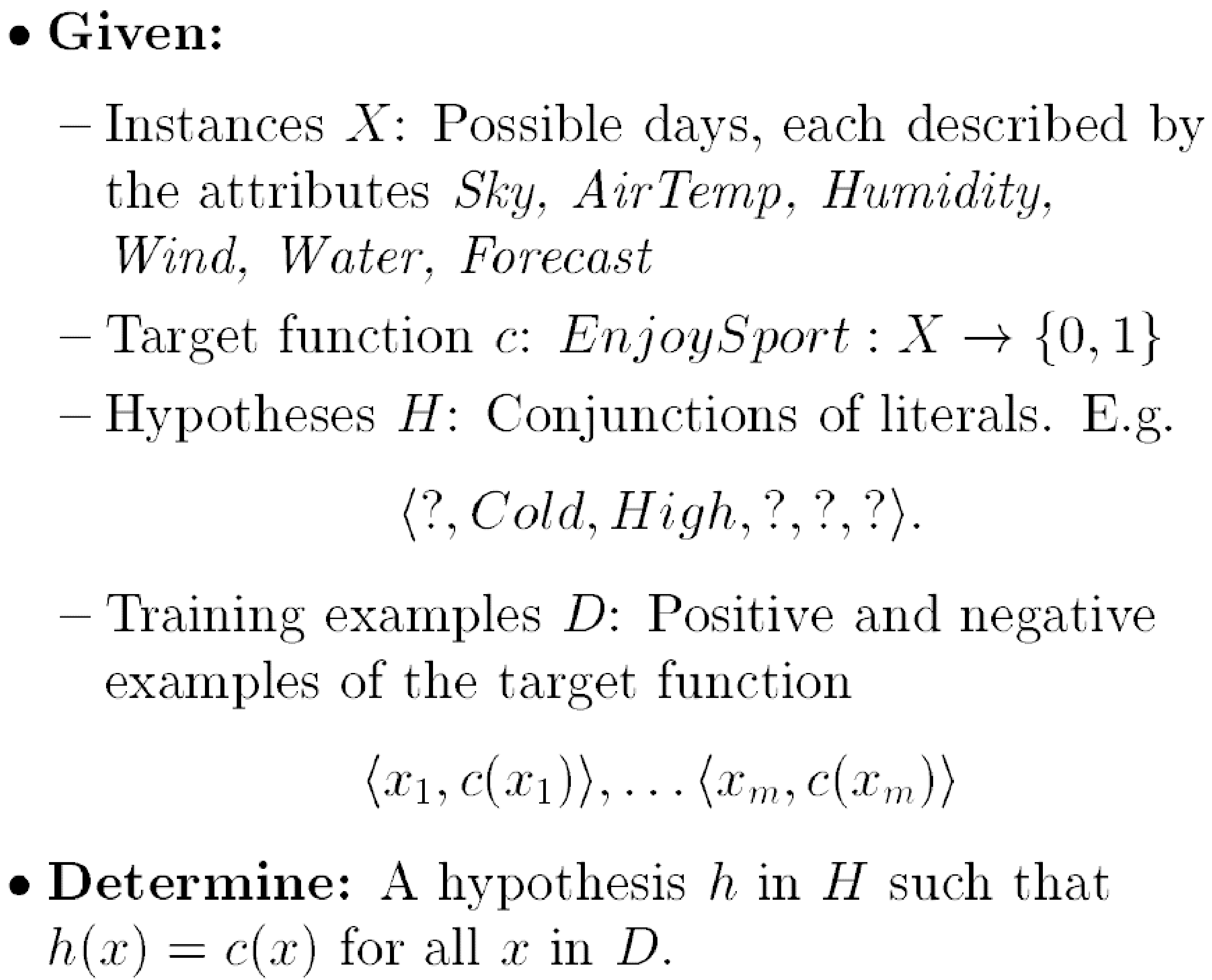
- Instances: The set of items over which the concept is defined

- Target concept (c): The concept to be learned

- Hypothesis (h): A supposition or proposed explanation made on the basis of limited evidence (training set)

- Hypotheses Space (H): The set of all possible hypotheses

Prototypical concept learning task



- **Given:**
  - Instances X: Possible days, each described by the attributes *Sky, AirTemp, Humidity, Wind, Water, Forecast*
  - Target function c: $EnjoySport : X \rightarrow \{0, 1\}$
  - Hypotheses H: Conjunctions of literals. E.g.

    $$\langle ?, Cold, High, ?, ?, ? \rangle.$$

  - Training examples D: Positive and negative examples of the target function

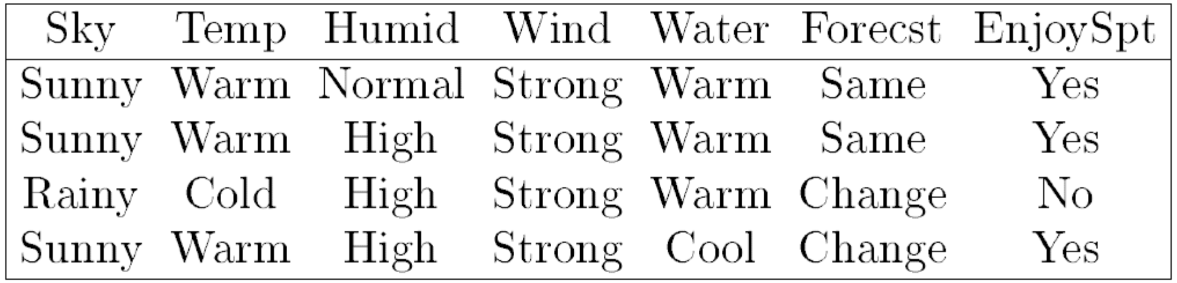
    $$\langle x_1, c(x_1) \rangle, \ldots \langle x_m, c(x_m) \rangle$$

- **Determine:** A hypothesis h in H such that $h(x) = c(x)$ for all x in D.

The inductive learning hypothesis: Any hypothesis found to approximate the target function well over a sufficiently large set of training examples will also approximate the target function well over other unobserved examples.

Concept Learning as a search

- The task of searching through a large space of hypotheses
- Goal: Find the hypothesis that best fits the training example
- *How many distinct instances are possible?*



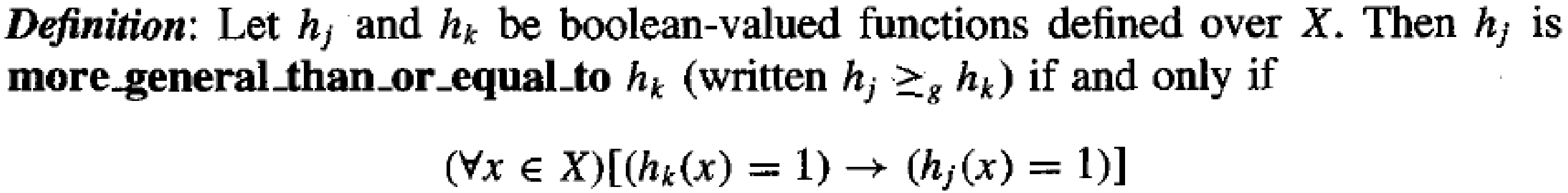
Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

- *How many systematically distinct hypotheses are possible?*
- *How many* ***semantically*** *distinct hypotheses are possible?*

General-to-specific ordering of hypotheses

- h1= <Sunny, ?, ?, Strong, ?, ?>
- h2= <Sunny, ?, ?, ?, ?, ?>

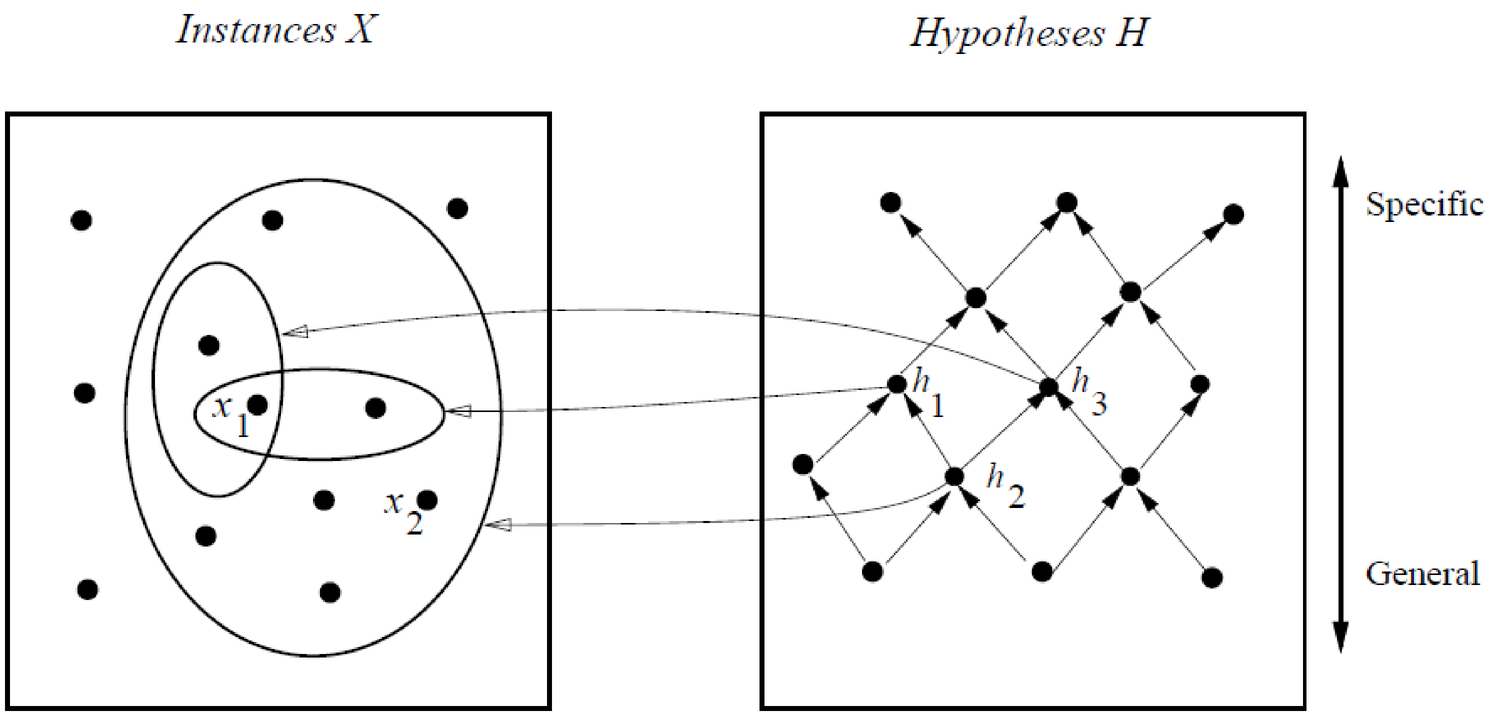
- For any instance *x* in *X* and hypothesis *h* in *H*, we say that x ***satisfies*** h if and only if h(x) = 1.



Definition: Let h_j and h_k be boolean-valued functions defined over X. Then h_j is **more_general_than_or_equal_to** h_k (written $h_j \geq_g h_k$) if and only if

$$(\forall x \in X)[(h_k(x) = 1) \rightarrow (h_j(x) = 1)]$$

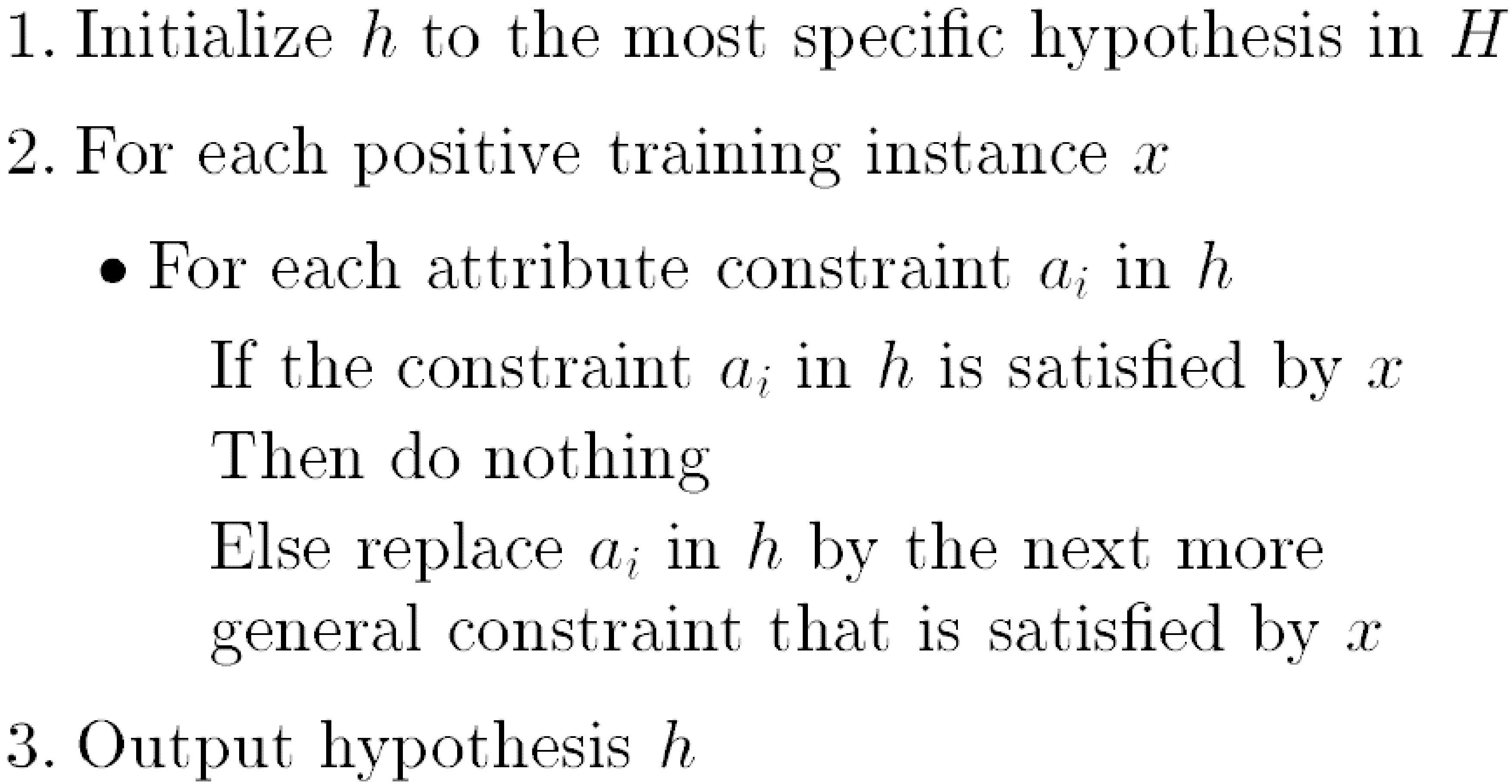
Hasse diagram



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

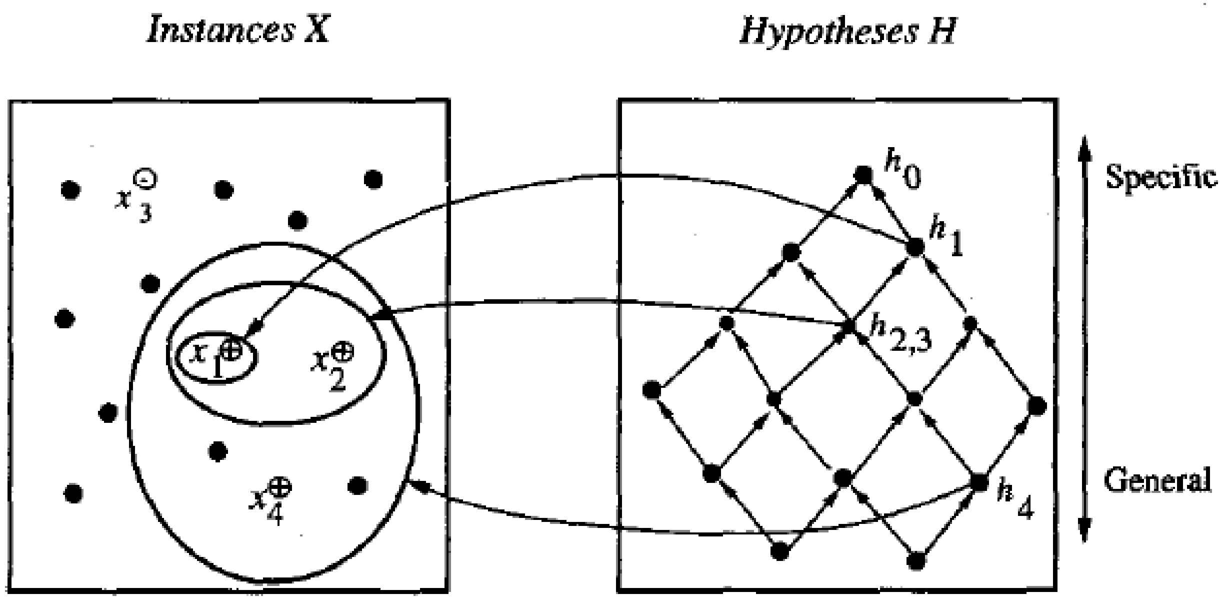
h_1 = *<Sunny, ?, ?, Strong, ?, ?>*
h_2 = *<Sunny, ?, ?, ?, ?, ?>*
h_3 = *<Sunny, ?, ?, ?, Cool, ?>*

Find-S Algorithm



```
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 Algorithm



h_0 = <∅, ∅, ∅, ∅, ∅, ∅>

x_1 = <Sunny Warm Normal Strong Warm Same>, +
h_1 = <Sunny Warm Normal Strong Warm Same>

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 ? ? >

At each step, h is the most/least specific/general hypothesis consistent with the training examples observed to this step

Find-S Algorithm – ignore negative instances

- Ignores every -ve training instances!
- However, the current hypothesis is already consistent with the -ve example
- As long as we assume that H contains a hypothesis that describes target concept and the training data is correct, it never requires to consider –ve examples

- *Why?*

Complaints about Find-S

- **Has the learner converged to the current target concept?**
  - No way to determine if it has found the *only hypothesis* that is consistent with the target concept
  - *Or* there are many other consistent hypotheses as well
- **Why prefer the most specific hypothesis?**
  - In case of multiple hypotheses consistent with the target concept, why to consider the most specific one?
- **Are the training example consistent?**
  - What if a few training instances are corrupted?
- **What if there are several maximally specific consistent hypotheses?**
  - Find-S should be backtracked to generalize the hypothesis