Graphs & Graph Algorithms

CSC 106

Ulrike Stege

Graphs

When do we use graphs?

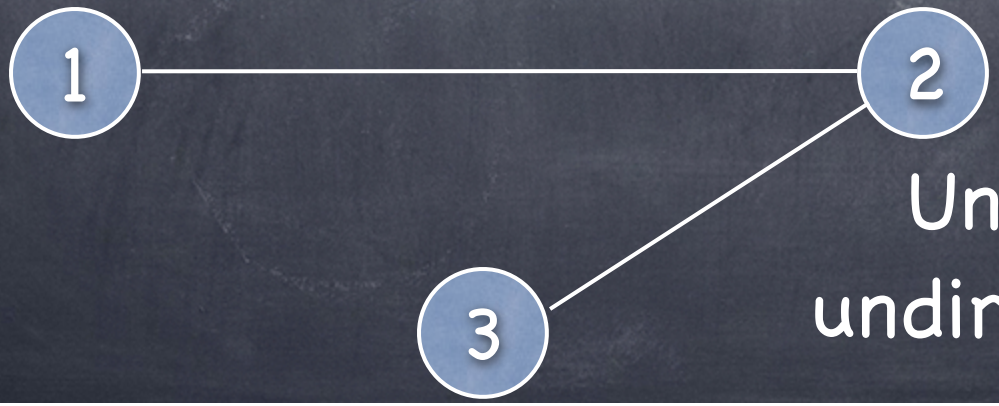
- When the data contains information about the relationship between items
- Evolutionary relationships
- Biological networks (e.g., protein interaction networks)
- Social networks
- (Computer) networks
- Visibility graphs (e.g. as support for understanding a program)
- etc

Graphs consist of

- **Vertices** (also called **nodes**)
- that are connected by **edges** or **arcs**
- Graphs can be
  - **weighted** or **unweighted**
  - **directed** or **undirected**

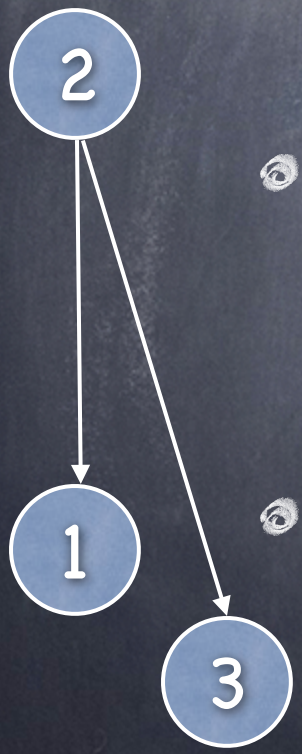
A graph G=(V,E)

- G's vertex set is denoted V
- G's edge set is denoted E
- An edge connecting vertices a and b is denoted by **pair** (a,b)
- If V = {1,2,3} and E= {(1,2), (2,3)} then



Unweighted, undirected graph

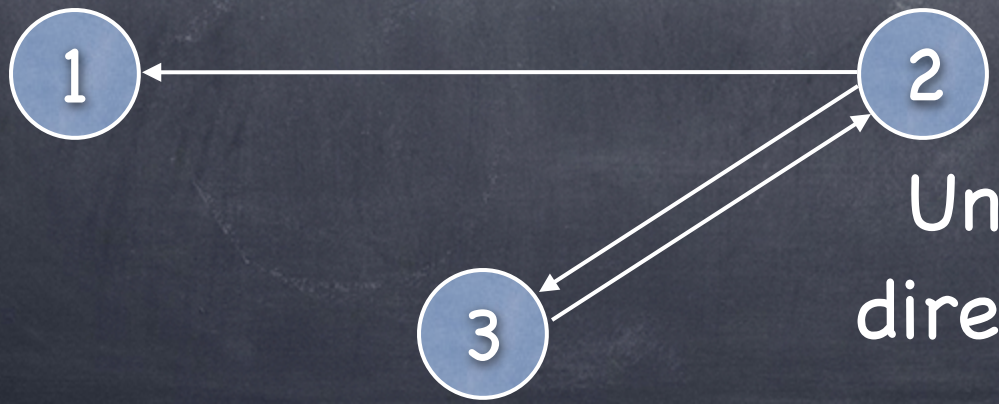
Directed Graphs



- Edges are also called **arcs**
- An arrow denotes the **direction** of the arc
- Pairs in the edge/arc set are then **ordered** pairs

A directed graph G=(V,A)

- V = {1,2,3}
- A = {(2,1),(3,2)}



Unweighted, directed graph

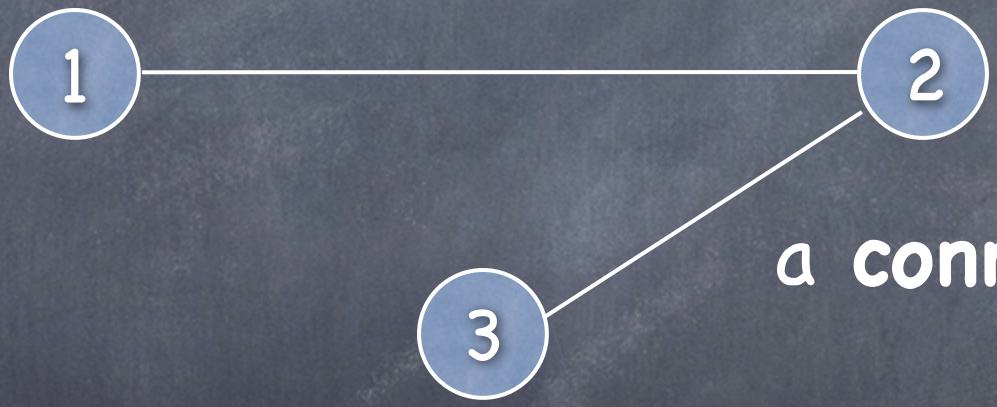
How does G=(V,E) look like?

- Let the undirected graph G = (V,E) be defined as follows.
  - V = {1,2,3,4,5}
  - E = {(1,2), (2,3), (5,4), (1,5), (3,4), (2,4)}

How does G=(V,E) look like?

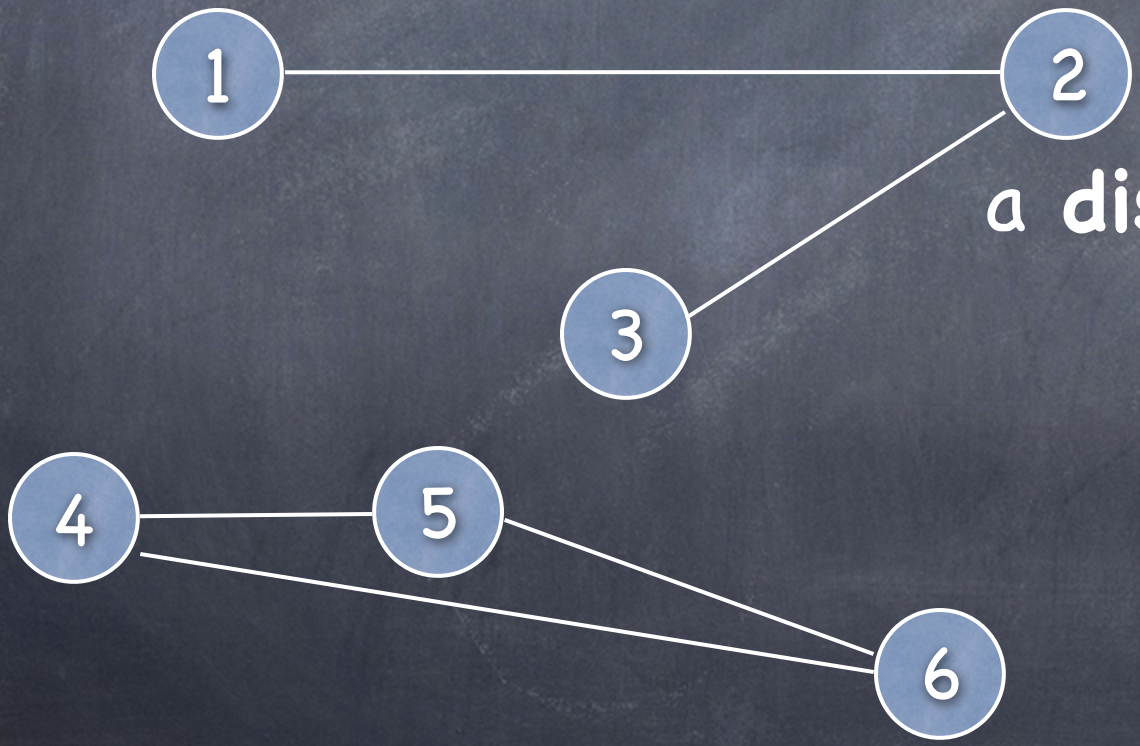
- Let the directed graph G = (V,A) be defined as follows.
  - V = {1,2,3,4,5}
  - A = {(1,2), (2,3), (5,4), (1,5), (3,4), (2,4)}

Undirected graphs can be connected or disconnected



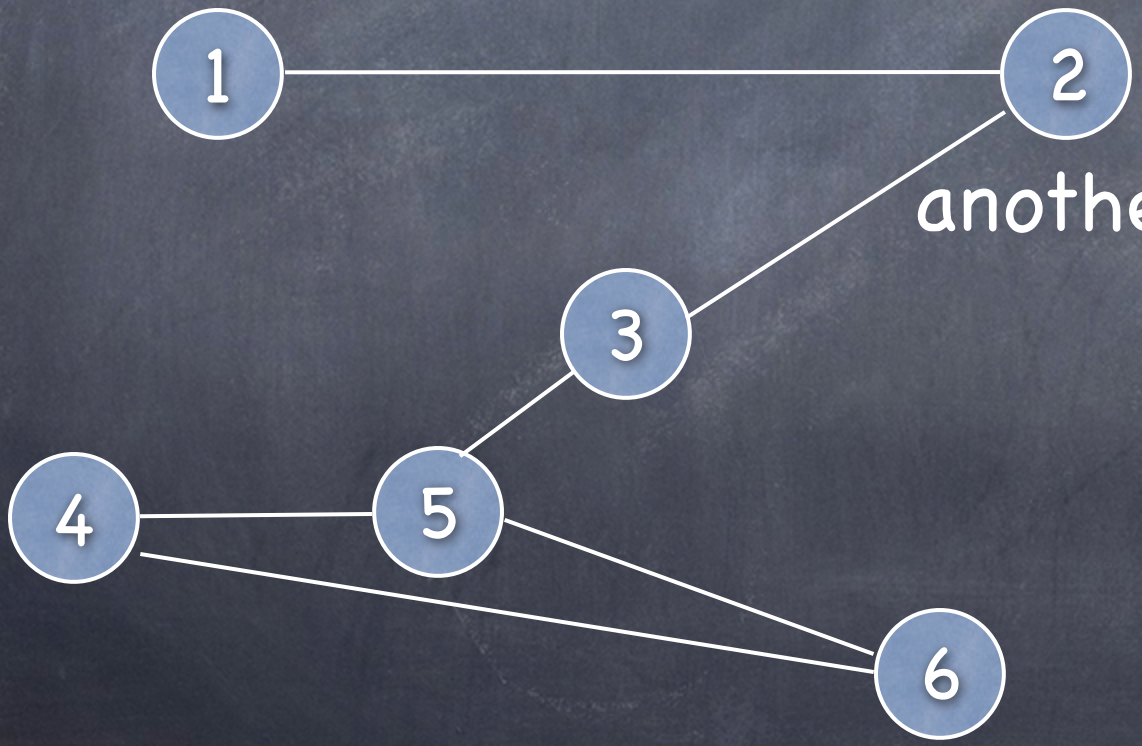
a **connected** graph

Undirected graphs can be connected or disconnected



a **disconnected** graph

Undirected graphs can be connected or disconnected



another **connected** graph

Connected and disconnected graphs

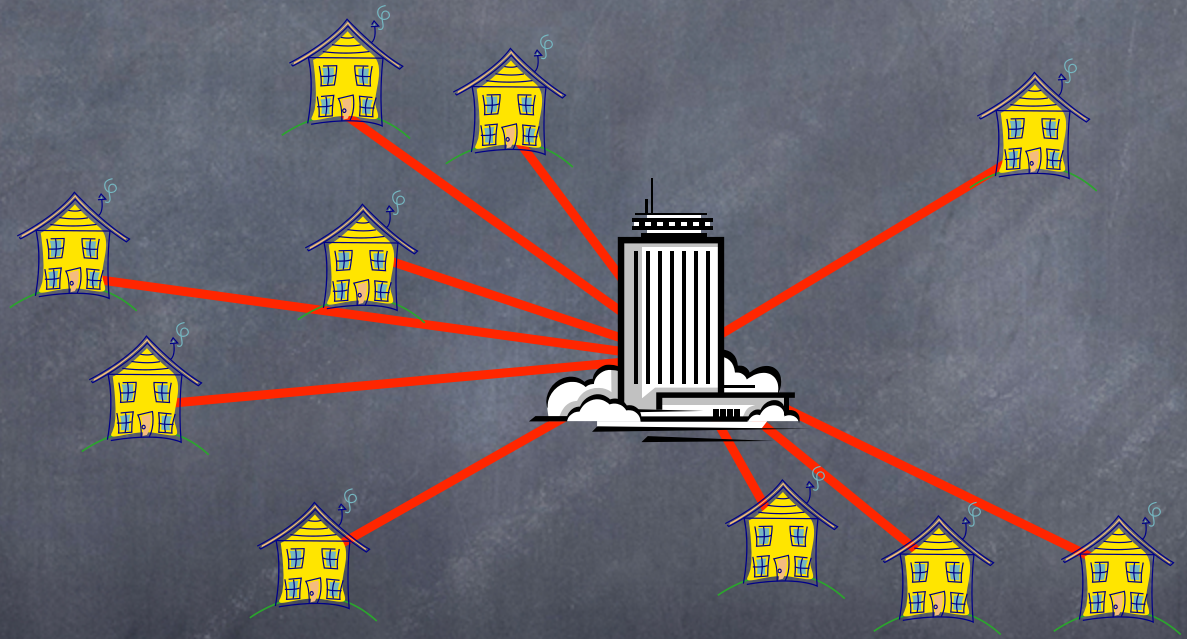
- An undirected graph is **connected** if for any pair u and v of nodes in the graph there is a path of edges that leads from u to v.
- An undirected graph is called **disconnected** if it is not connected.

Is this graph connected?

- Draw the undirected graph G = (V,E) with
  - V = {1,2,3,4,5}
  - E = {(1,3), (2,4), (1,5), (3,5)}

Laying TV Wire

Laying TV Wire

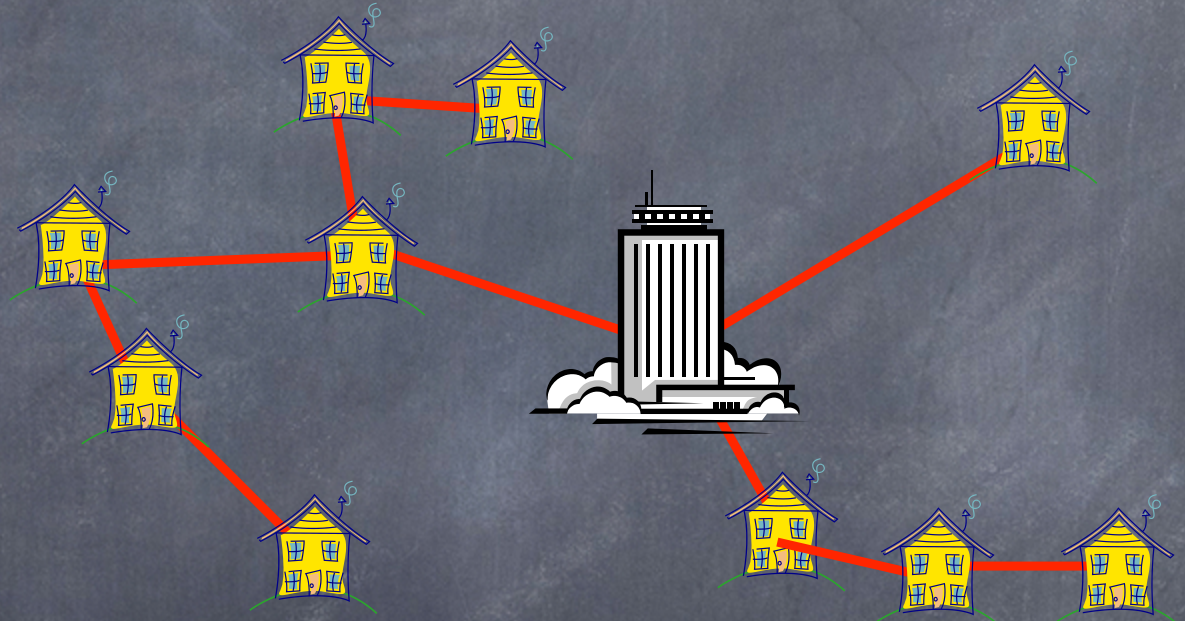


Problem: Want to minimize cost of wire, wiring, maintenance

Laying TV Wire

- Idea: Minimize the overall wire-length!

Laying TV Wire



Minimized Wiring

What network minimizes the wiring length?

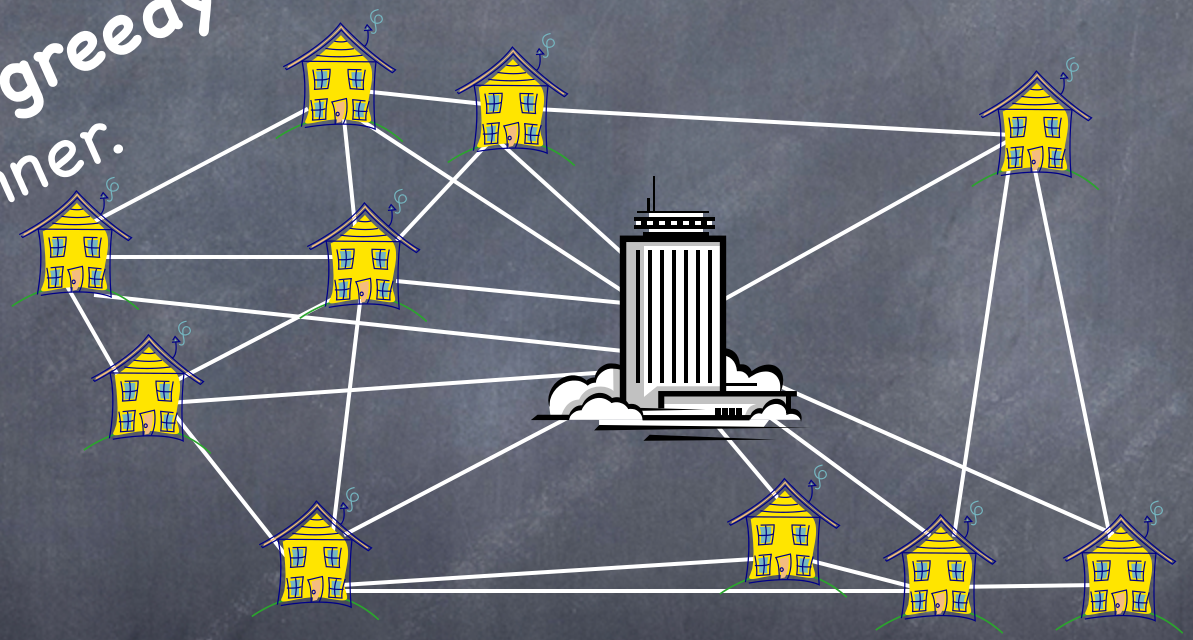
- A **shortest spanning network**
- A network or graph is **spanning** or **connected** if we can walk from every of its vertices to every other vertex
- A spanning network is **shortest** if it is a spanning network where the sum of the edge lengths is smallest

What does such a shortest spanning network look like?

- Such a network is a **tree**:
  - that is, the network/graph is cycle-free
- It is called a **minimum spanning tree**

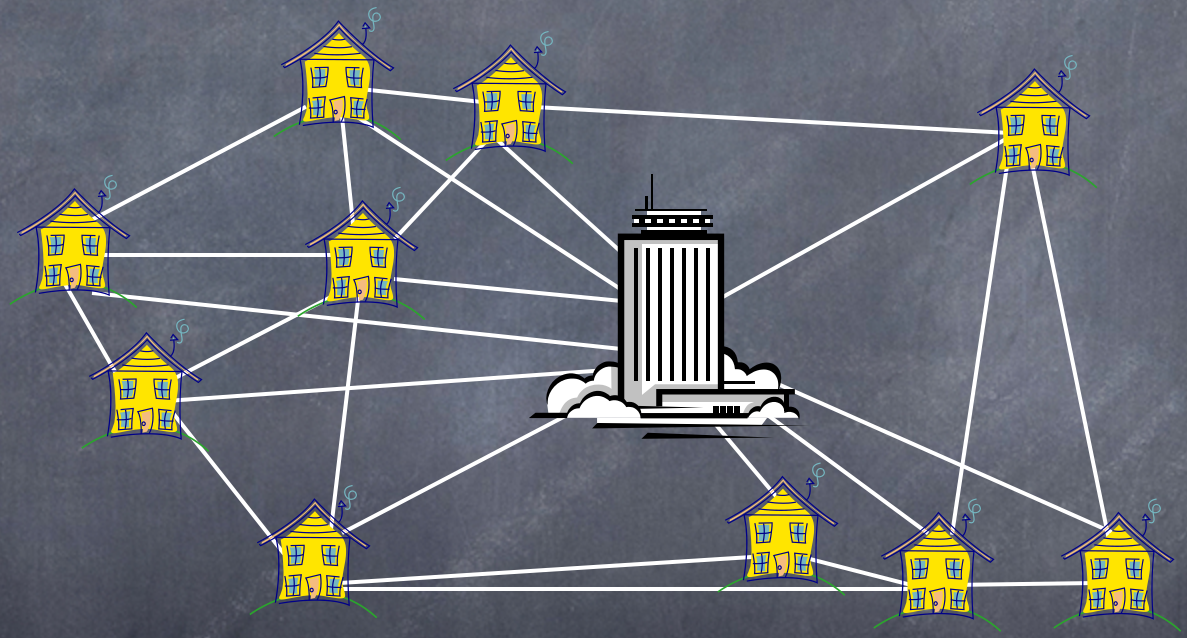
Graph Algorithm 1: How do we reach a solution?

We are approaching this in a **greedy** manner.



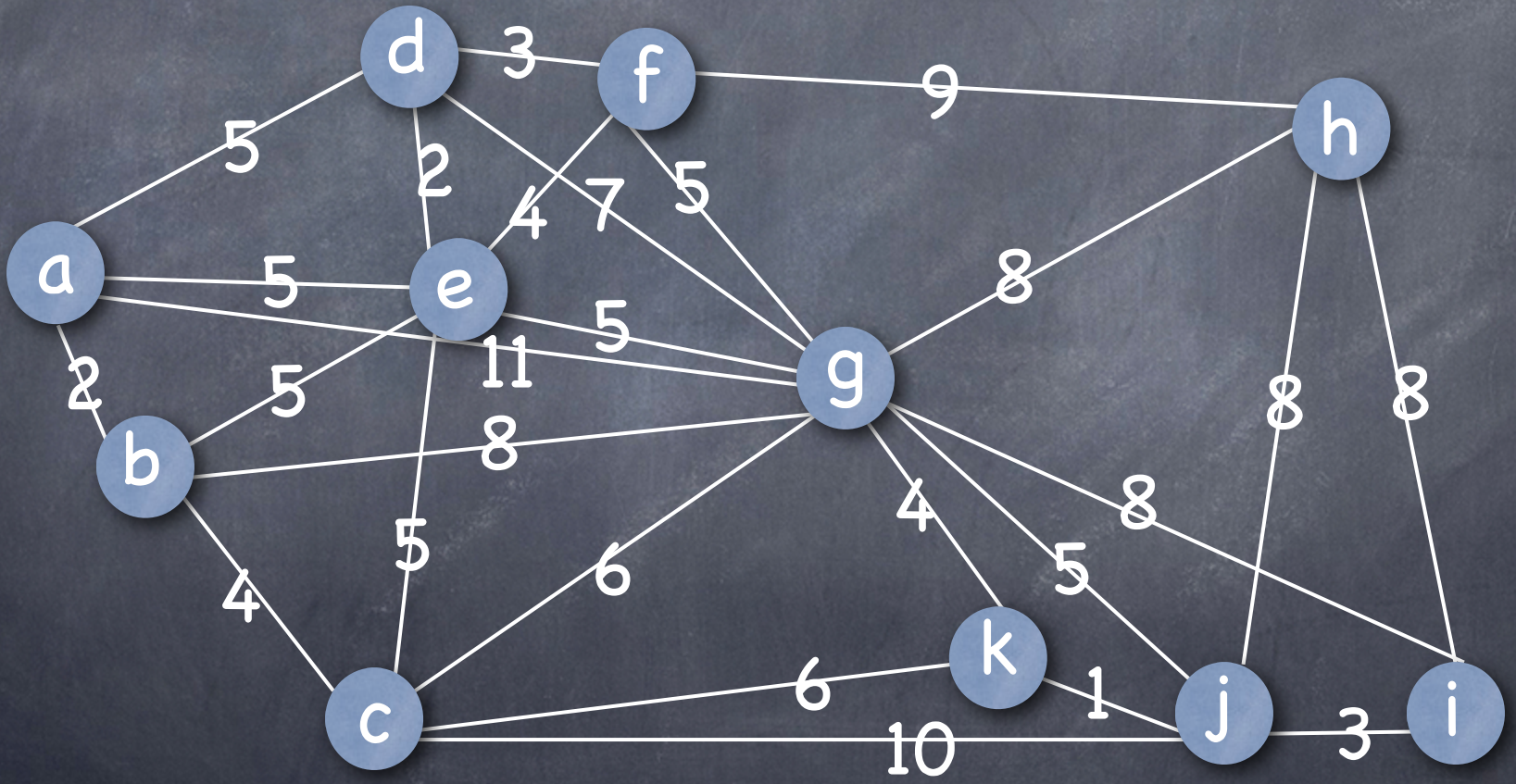
Step 1: Build graph. Vertices are the houses, edges are where connections between houses are possible.

How do we reach a solution?

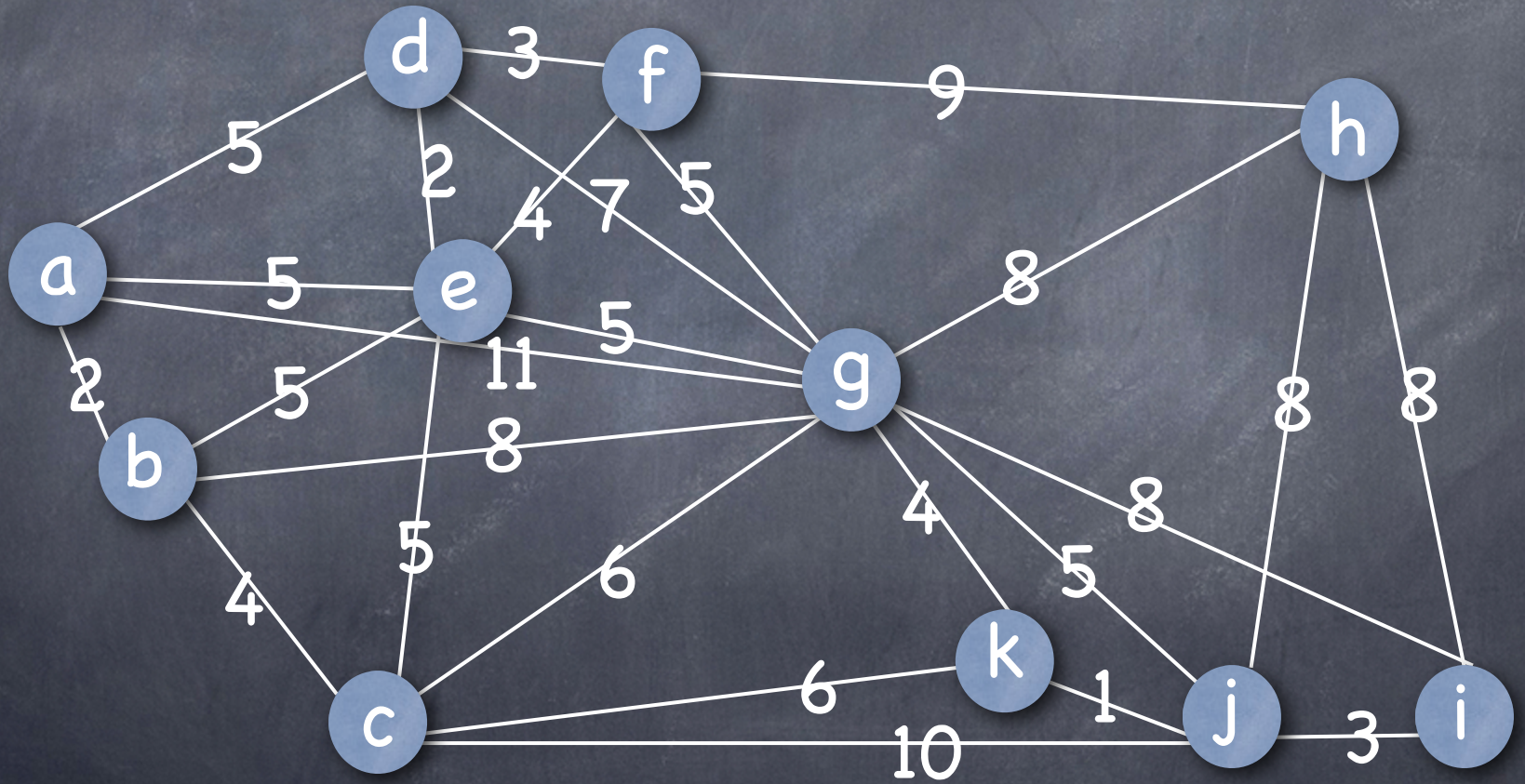


Step 2: Measure the lengths of the possible connections.

A weighted undirected graph

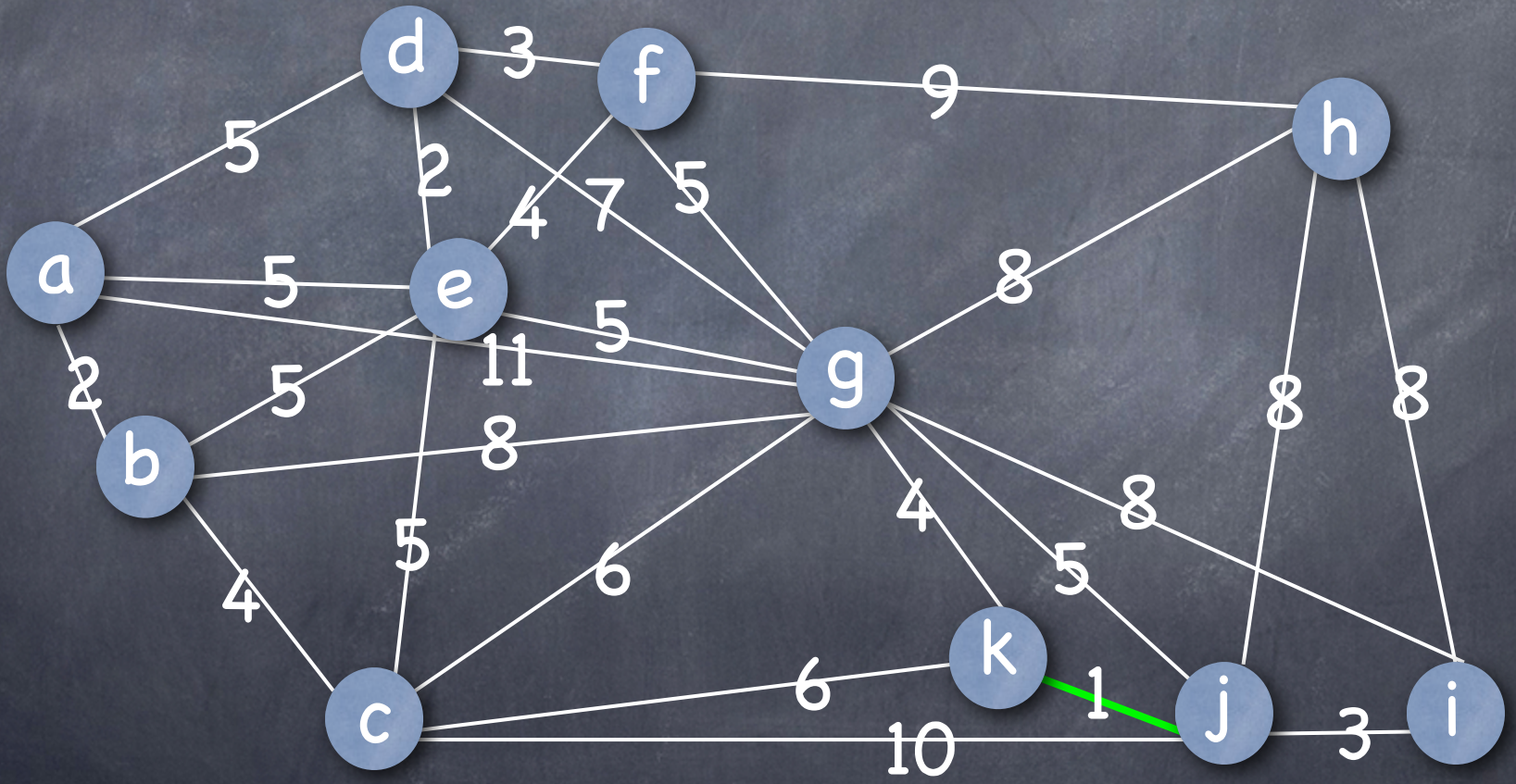


How much wire (in overall length) is necessary to have all vertices connected?

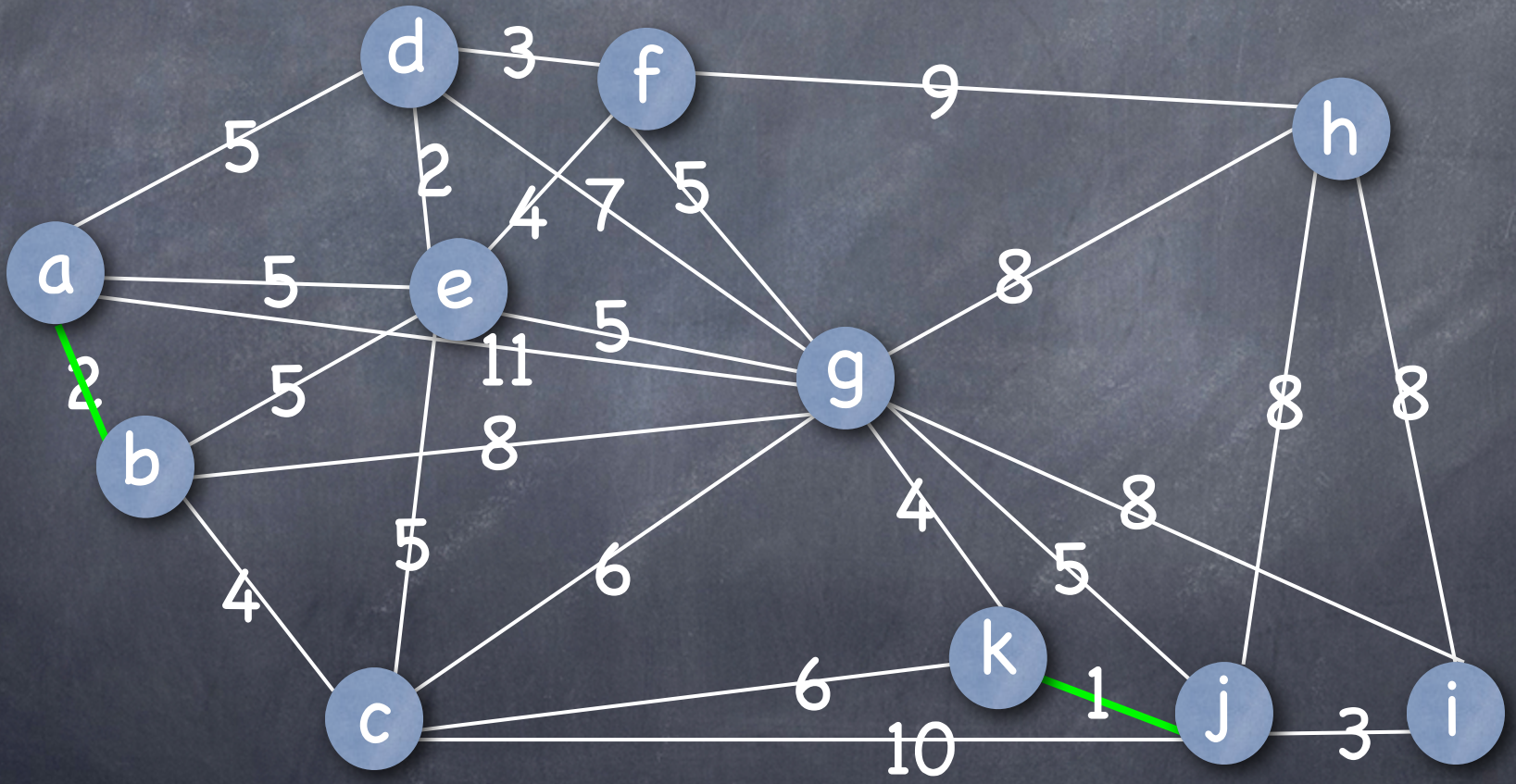


Select edges that result in such a best network!

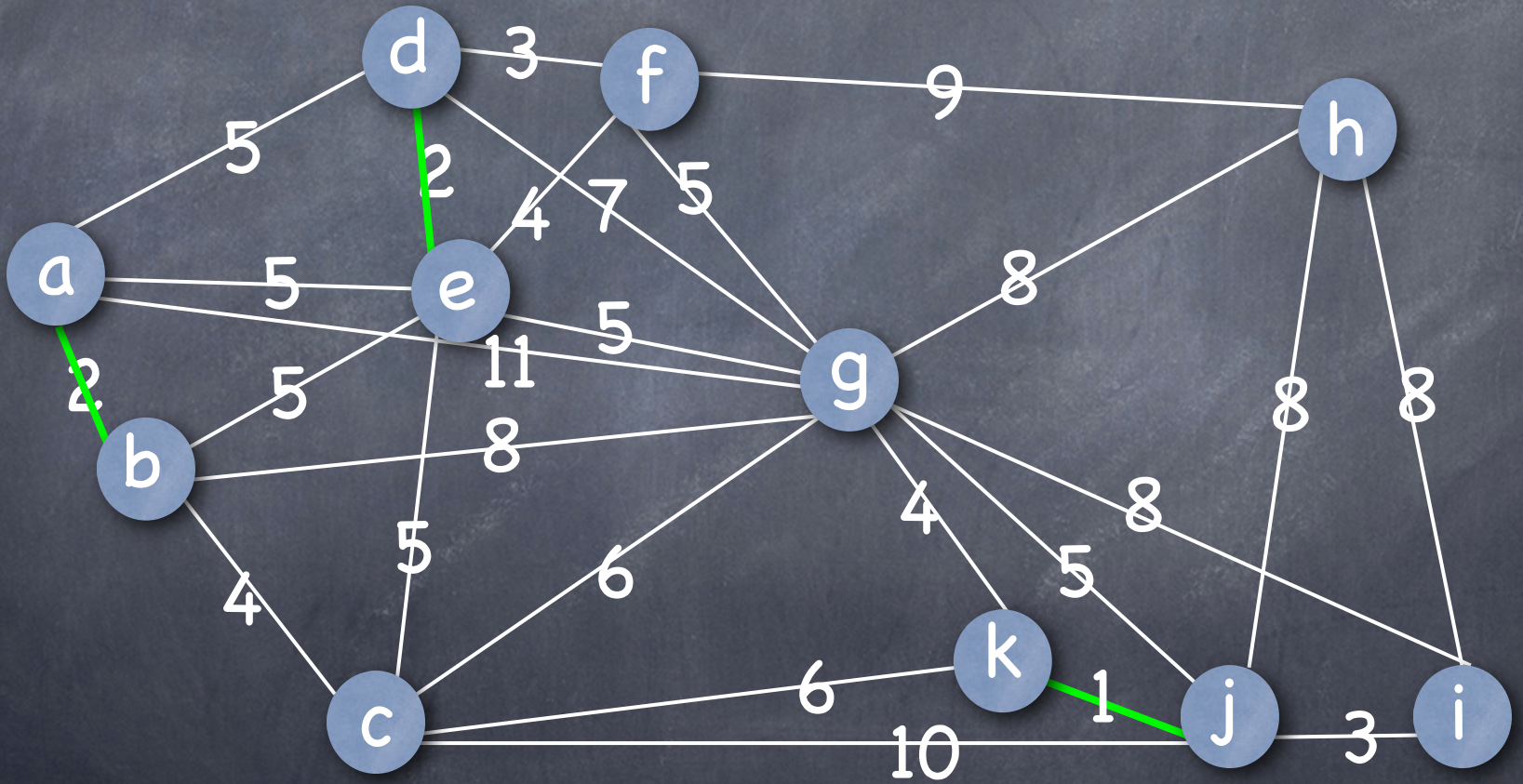
Step 3: Look for a shortest (non-selected edge) and choose it if it is a good choice



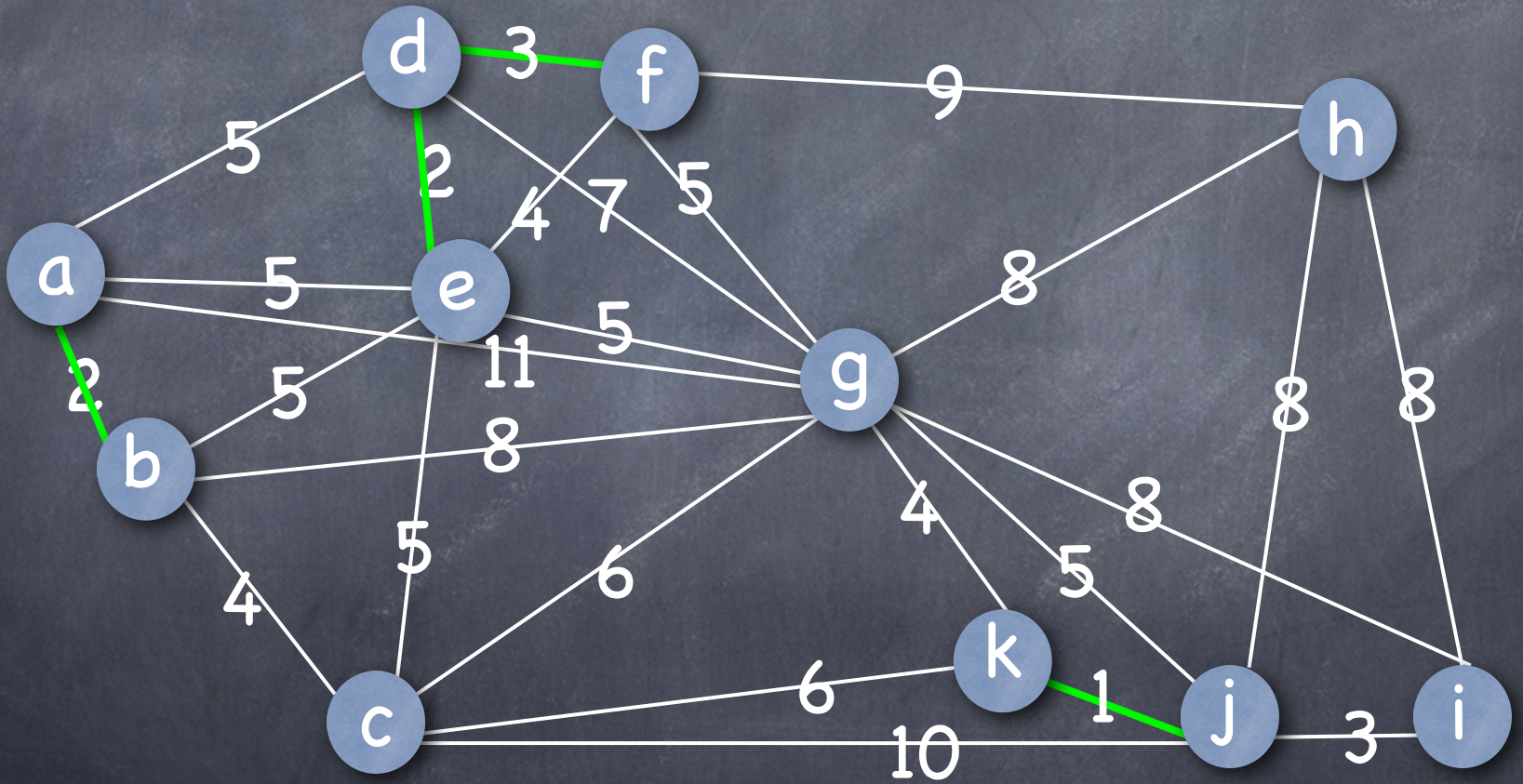
Repeat Step 3 until a spanning tree is built



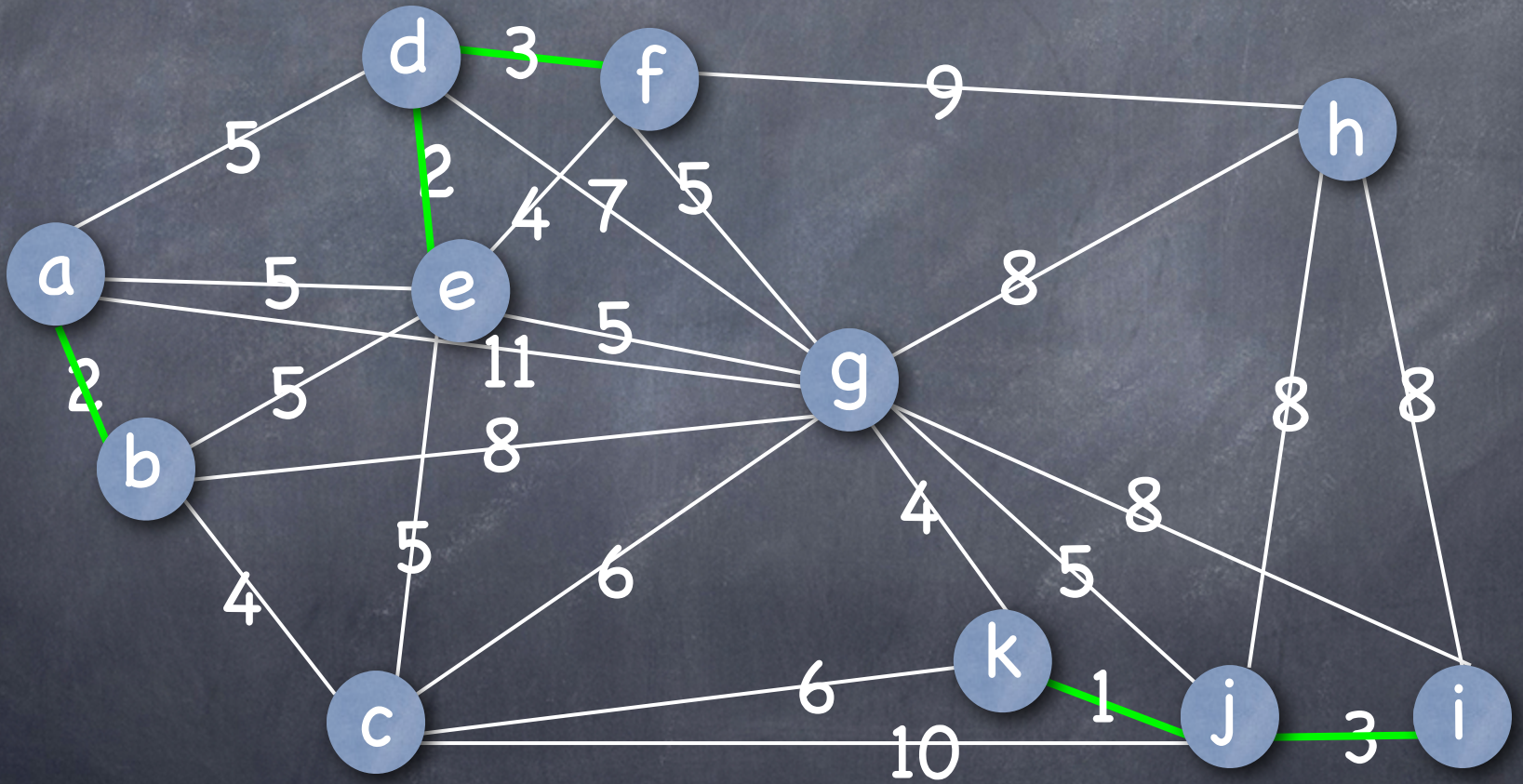
Repeat Step 3 until a spanning tree is built



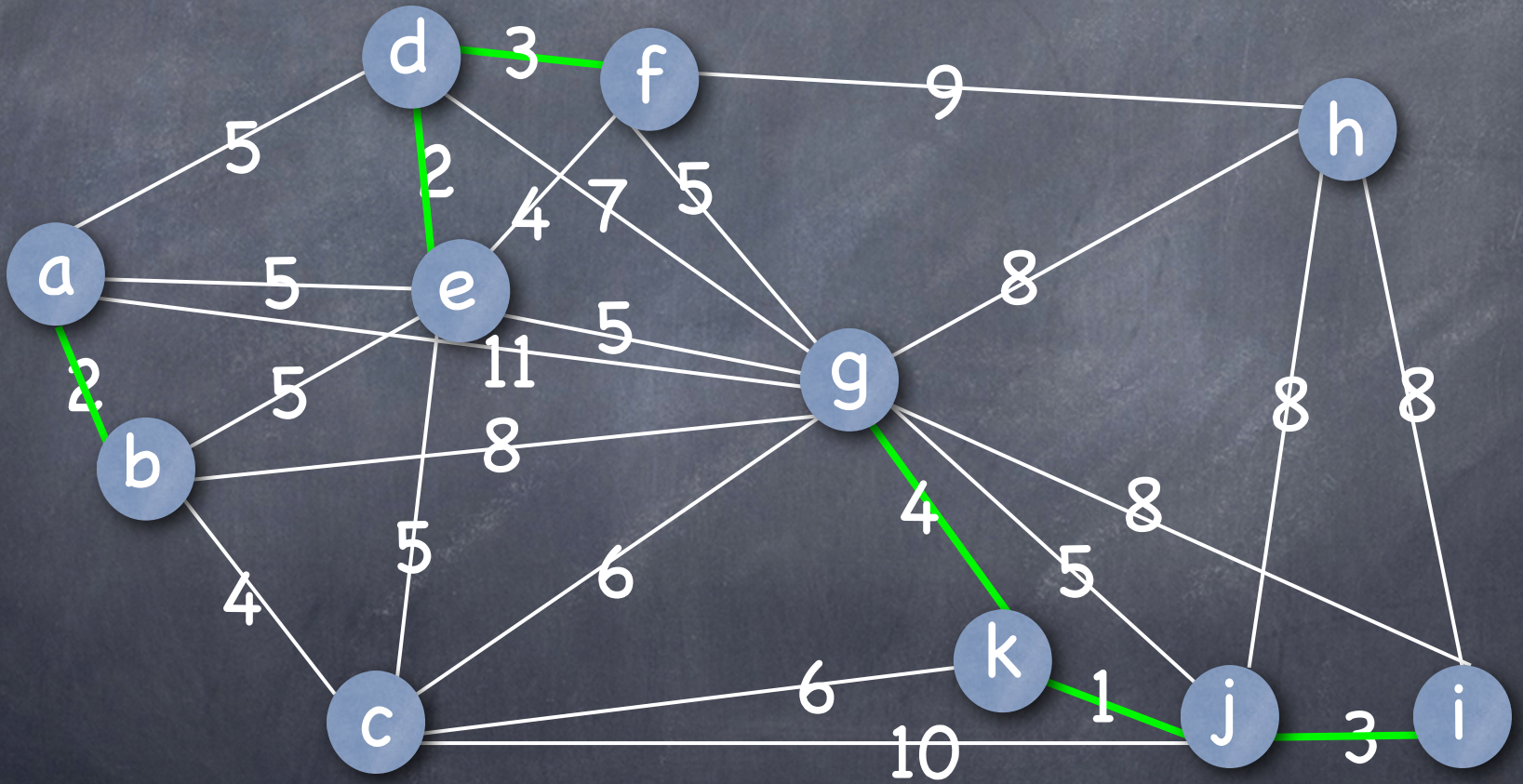
Repeat Step 3 until a spanning tree is built



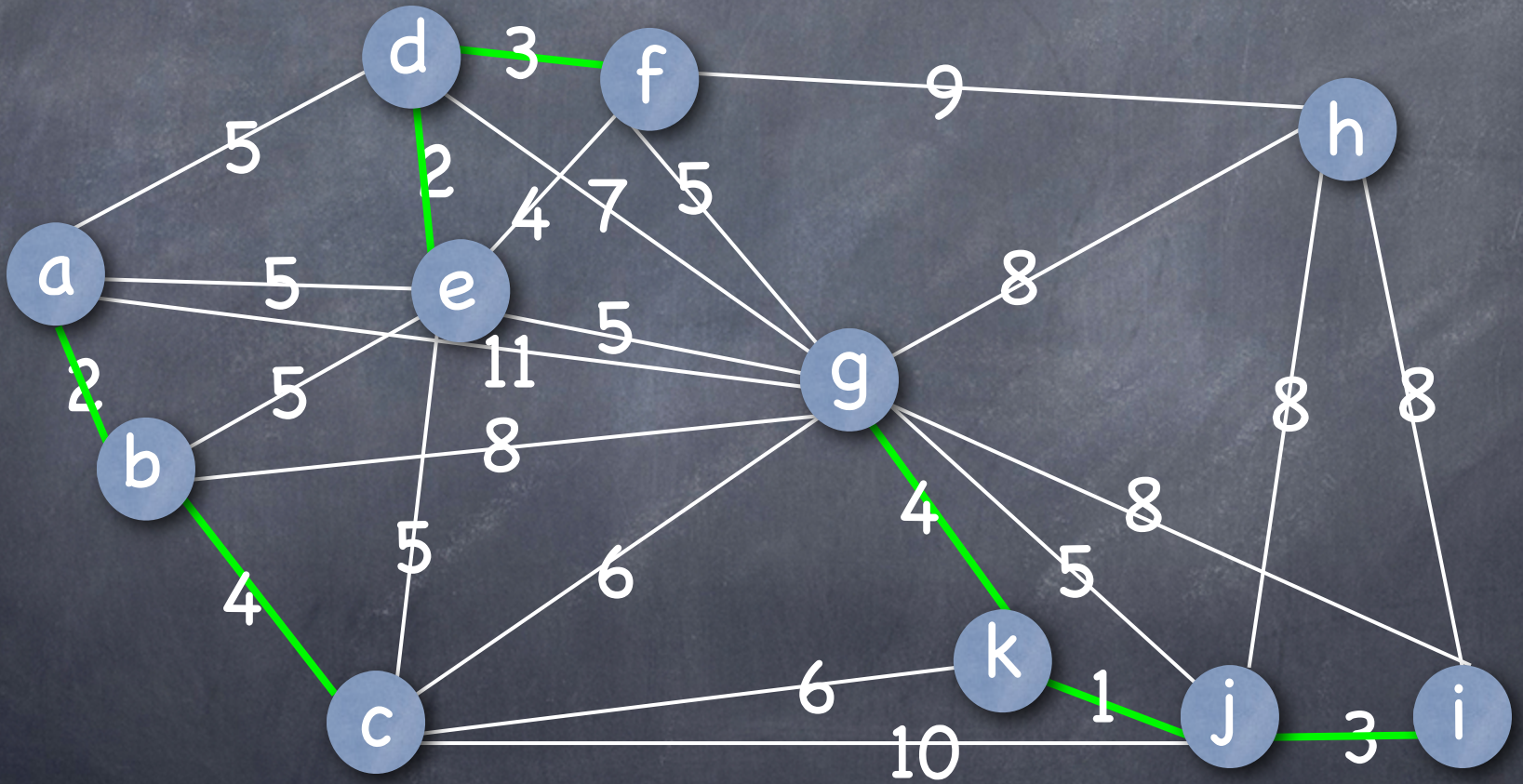
Repeat Step 3 until a spanning tree is built



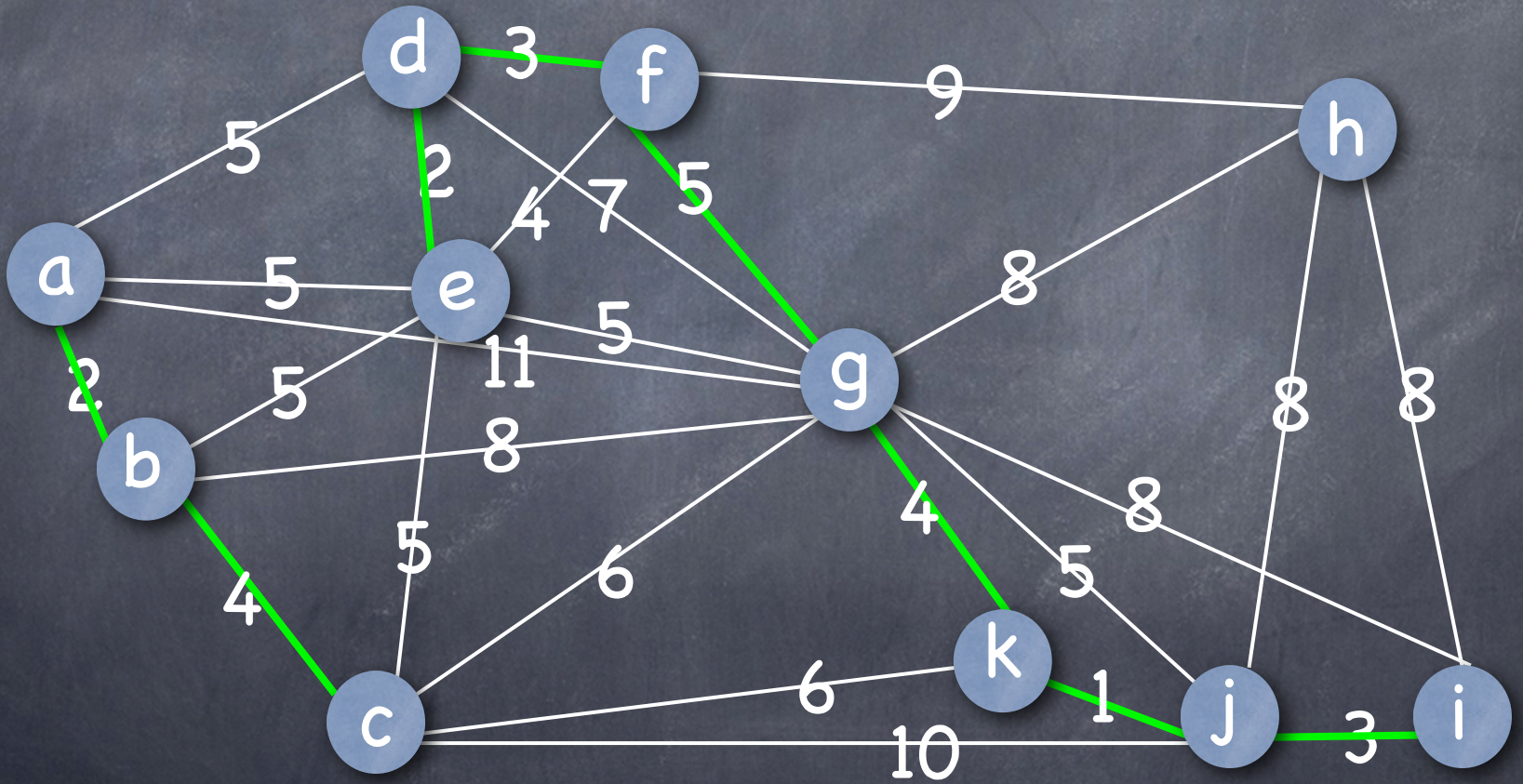
Repeat Step 3 until a spanning tree is built



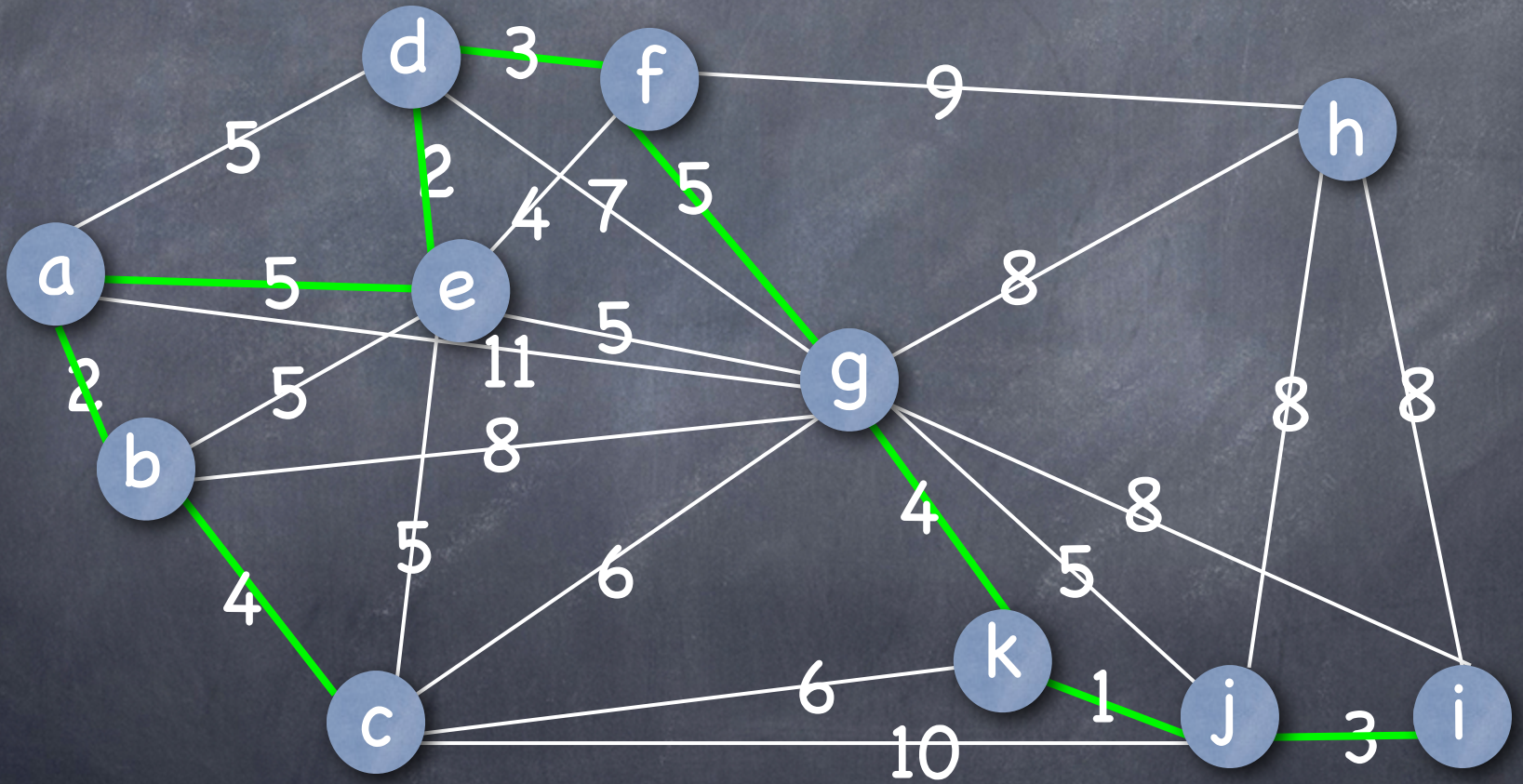
Repeat Step 3 until a spanning tree is built



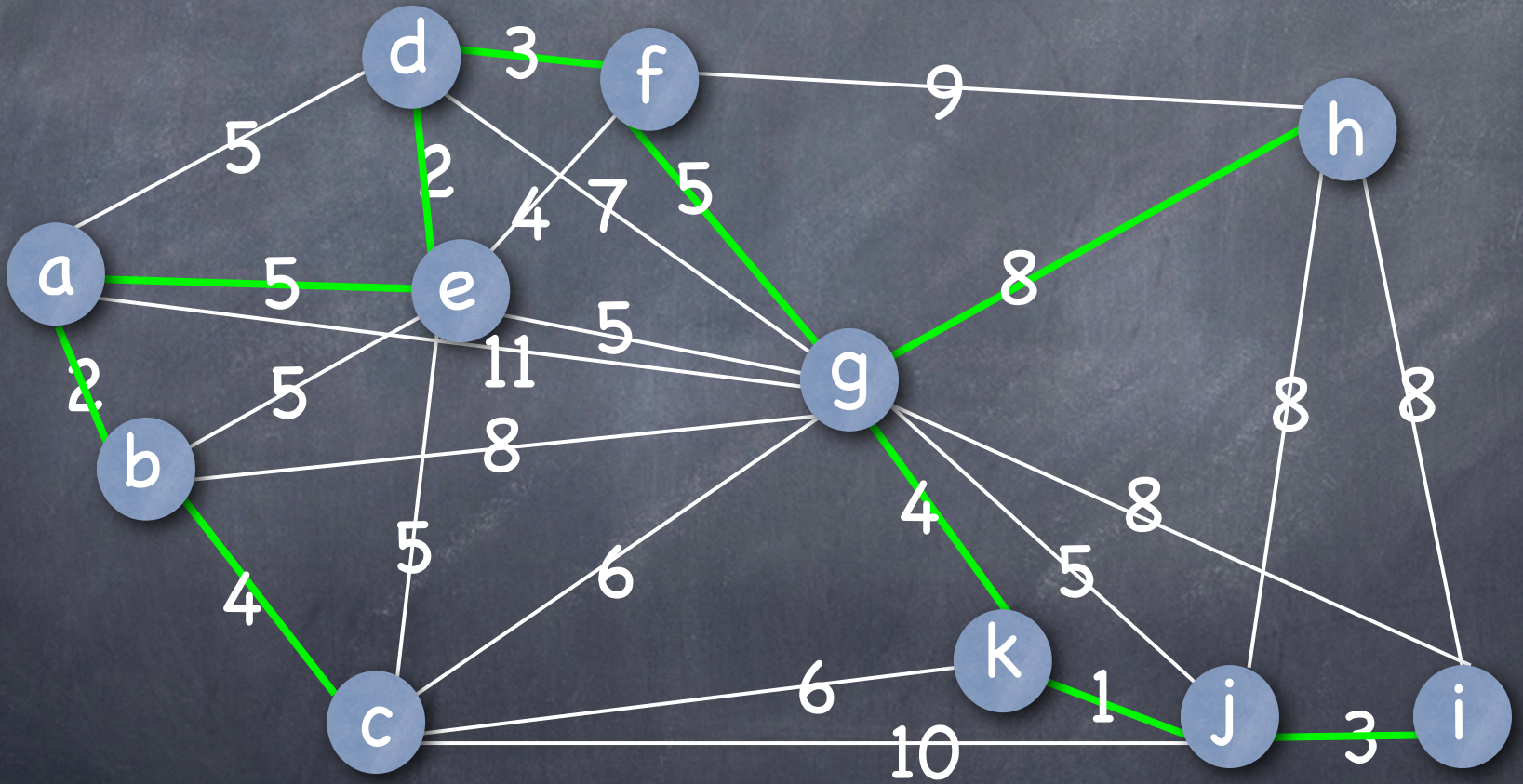
Repeat Step 3 until a spanning tree is built



Repeat Step 3 until a spanning tree is built

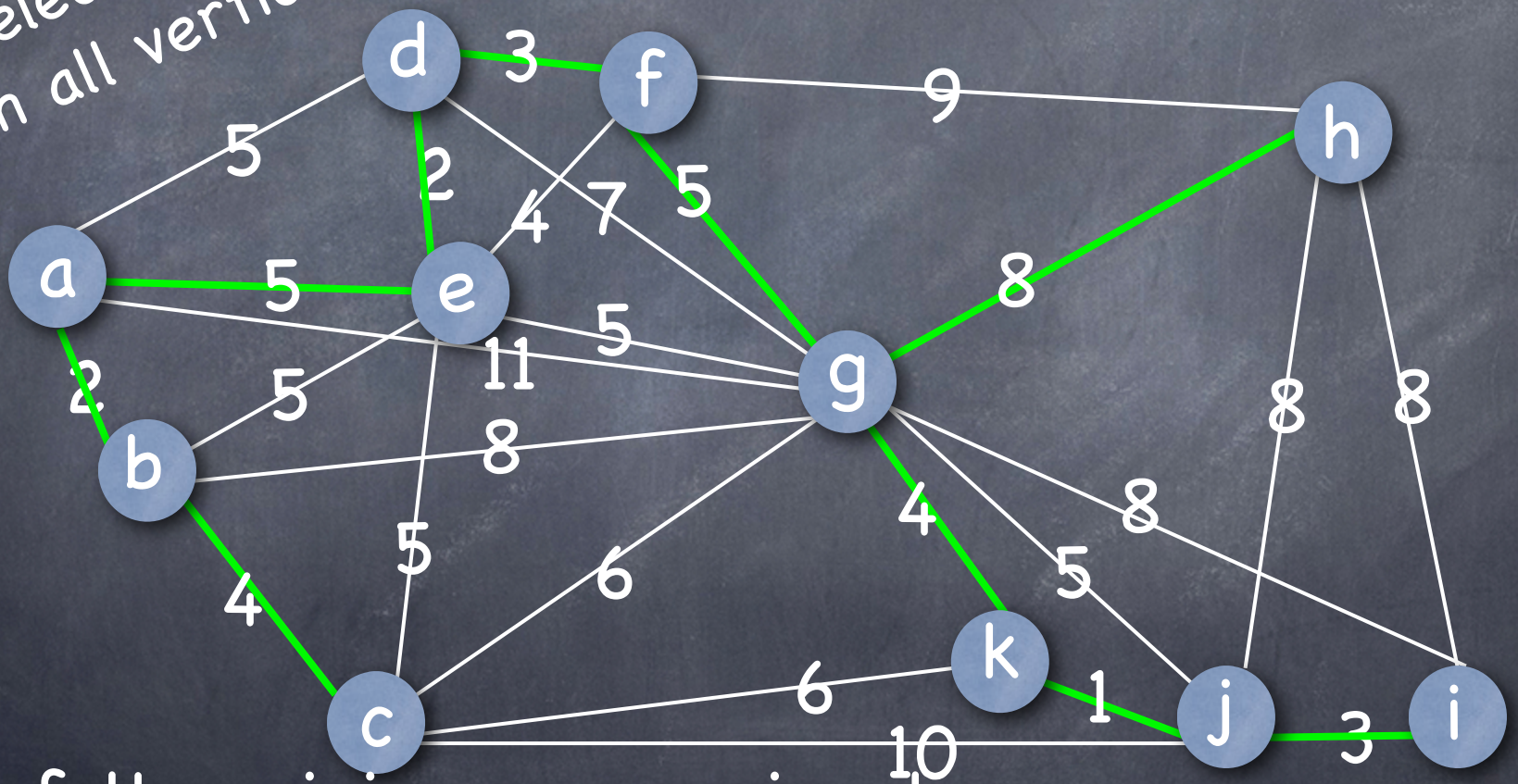


Repeat Step 3 until a spanning tree is built



Repeat Step 3 until a spanning tree is built

The selected edges span all vertices.



Cost of the minimum spanning tree:

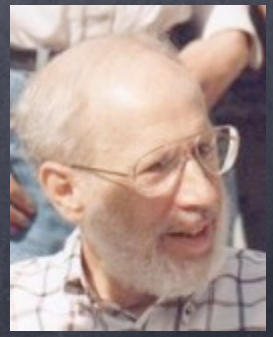
1+2+2+3+3+4+4+5+5+8 = 37

What algorithm did we use?

- We collect edges in an initially empty set M
- **while** we have not (yet) selected a spanning network **do**
  - pick a shortest edge e that has not been investigated yet
  - **if** adding this edge to M would yield a cycle **then** reject e (and don't consider it again)
  - **else** add e to M

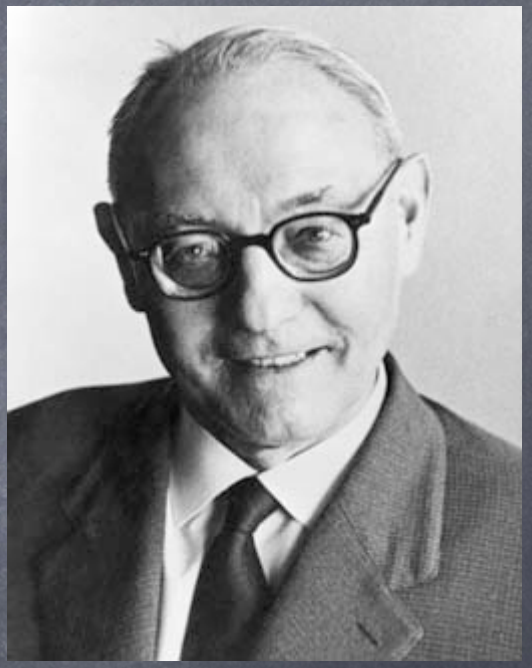
The algorithm

- builds a minimum spanning tree
- is a greedy algorithm (it uses an algorithm design technique called greedy; every step in the algorithm is composed of a local optimal choice)
- Running time for an efficient implementation: O(|E| log |V|)
- is called **Kruskal's algorithm**
- named after Joseph Kruskal (1928-2010)



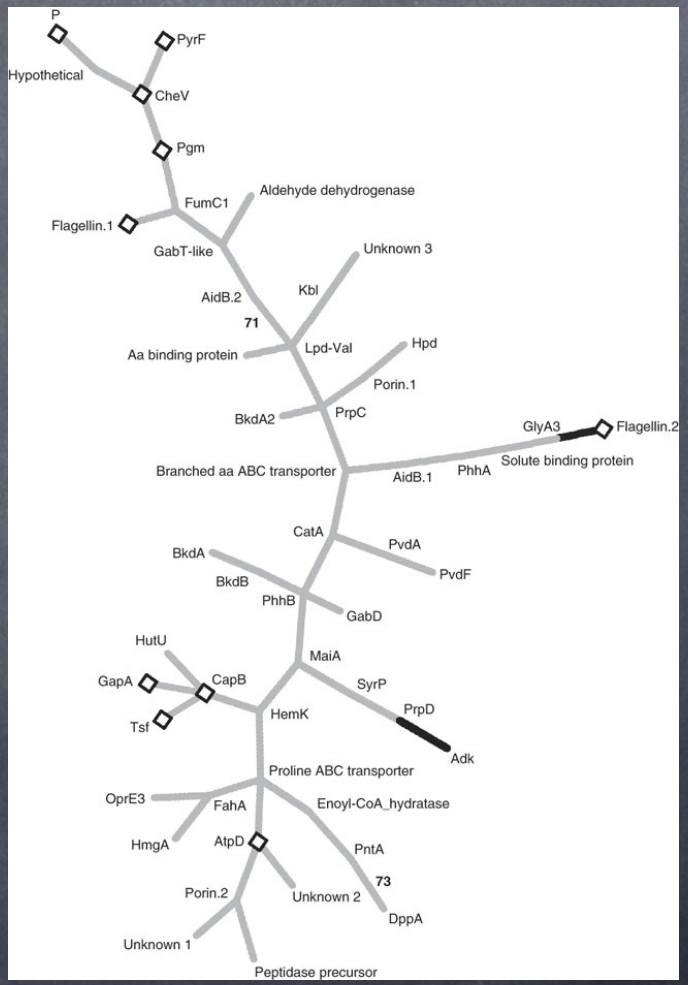
MST origin

- Otakar Borouvka (1899-1995) described in 1926 the first minimum spanning tree algorithm for finding a most economical construction for an electrical-power network
- Example where a concrete Engineering problem turned into a key problem in Computer Science



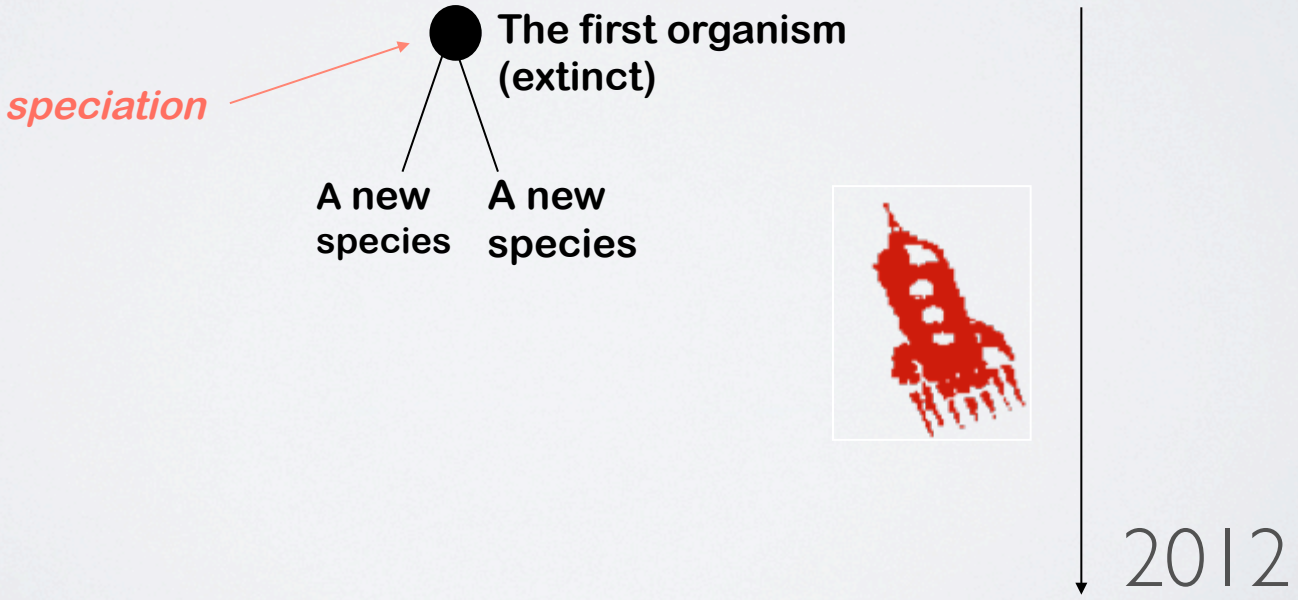
Other Minimum Spanning Tree Applications!

- Cheapest way to connect a set of terminals
- Routing problems
- Web services
- Skype network
- Music distribution
- Power distribution
- Wire routing on printed circuit boards
- Sewer pipe layout
- Road planning
- Gene evolution (gene mutations)
- Subroutine for many other algorithms

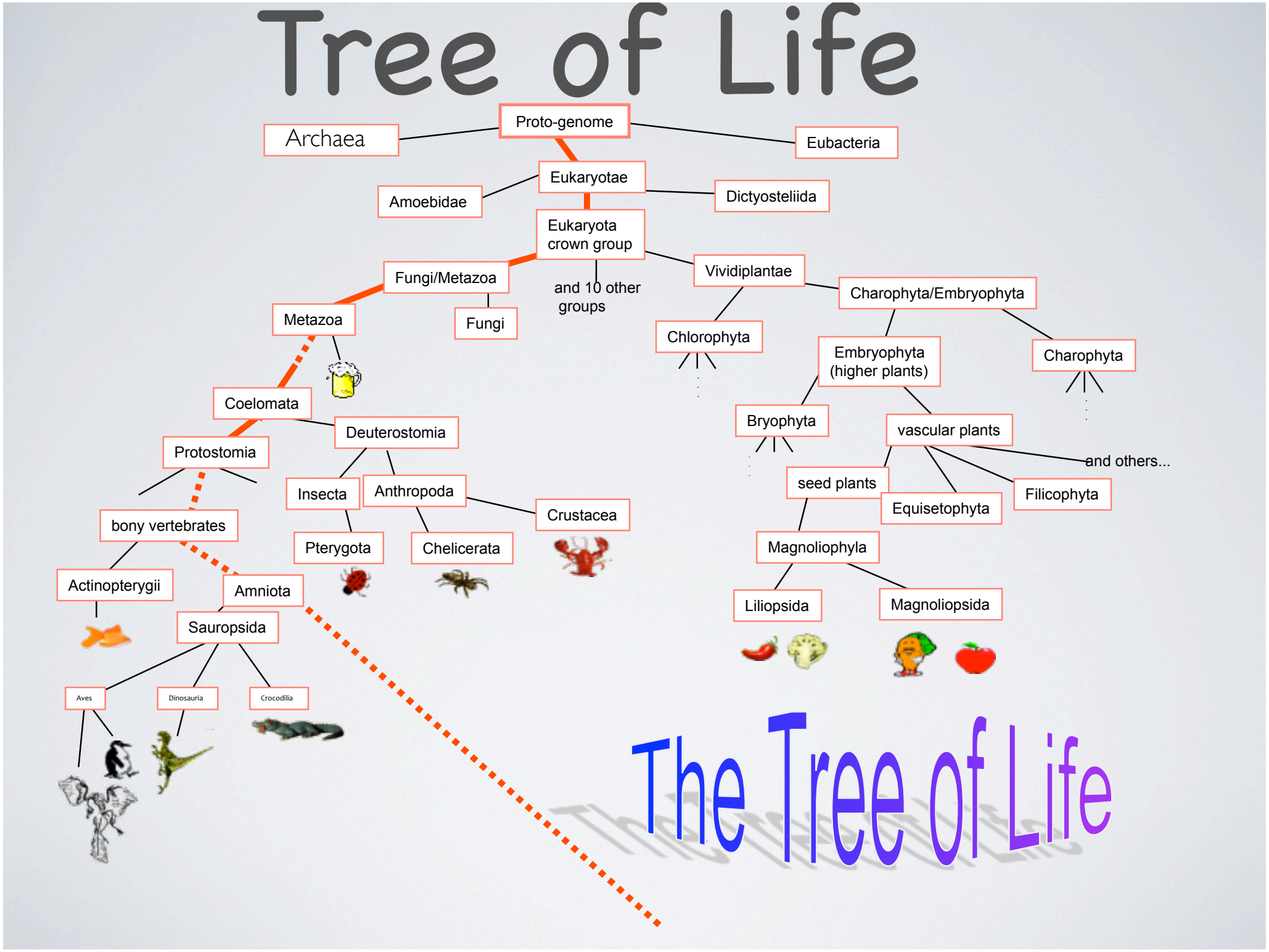


ANOTHER APPLICATION OF GRAPHS: EVOLUTIONARY TREES

HOW DID LIFE EVOLVE?



Tree of Life


Archaea
Proto-genome
Eubacteria
Eukaryotae
Amoebidae
Dictyosteliida
Eukaryota crown group
and 10 other groups
Fungi/Metazoa
Fungi
Metazoa
Vividiplantae
Chlorophyta
Charophyta/Embryophyta
Embryophyta (higher plants)
Charophyta
Bryophyta
vascular plants
and others...
seed plants
Filicophyta
Equisetophyta
Magnoliophyla
Liliopsida
Magnoliopsida
Coelomata
Deuterostomia
Protostomia
Insecta
Anthropoda
Crustacea
Pterygota
Chelicerata
bony vertebrates
Actinopterygii
Amniota
Sauropsida
Aves
Dinosauria
Crocodilia
The Tree of Life

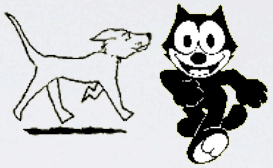
Eutheria

Artiodacyla

Proboscidea

Rodenta

Carnivora



Perissodactyla

Primates

Edentata

WHAT IS THE EVOLUTIONARY TREE FOR THESE SPECIES?

tortoise

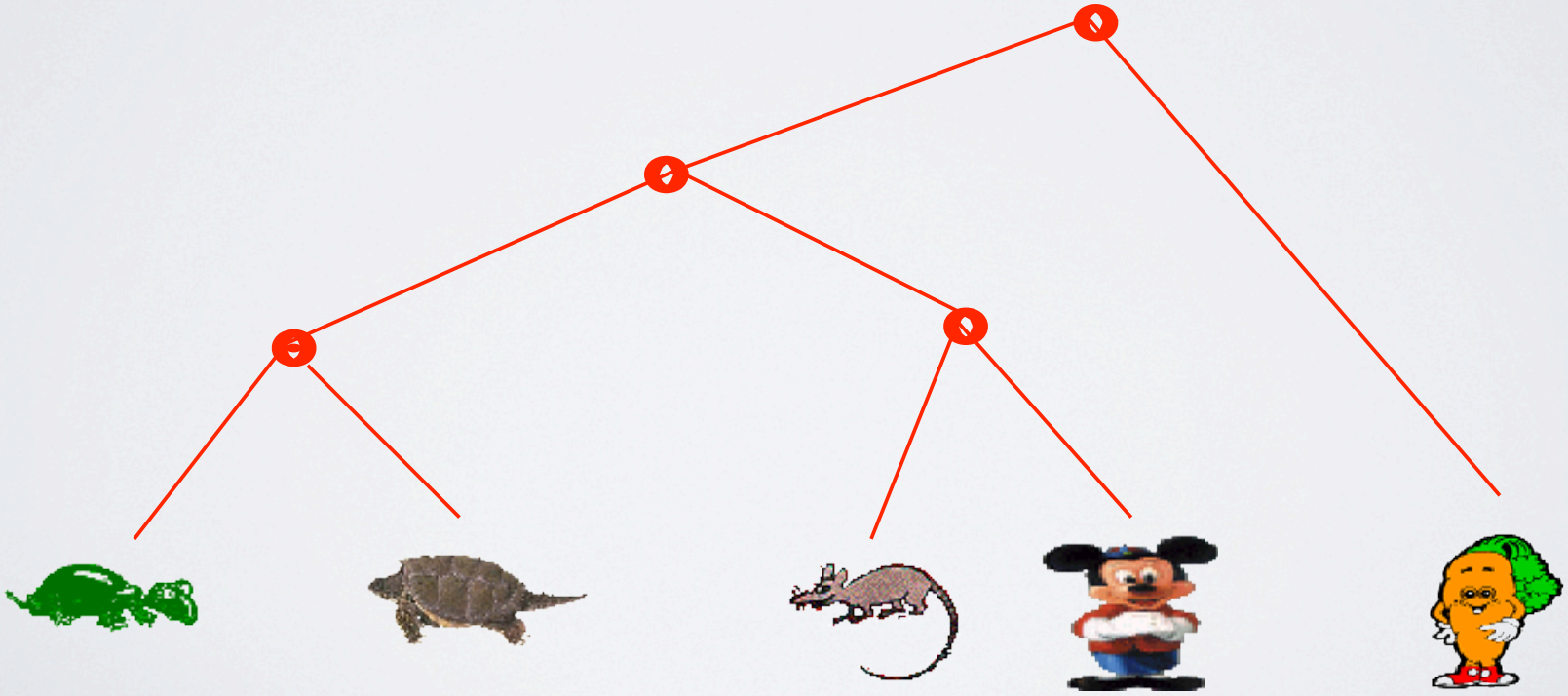
turtle

rat

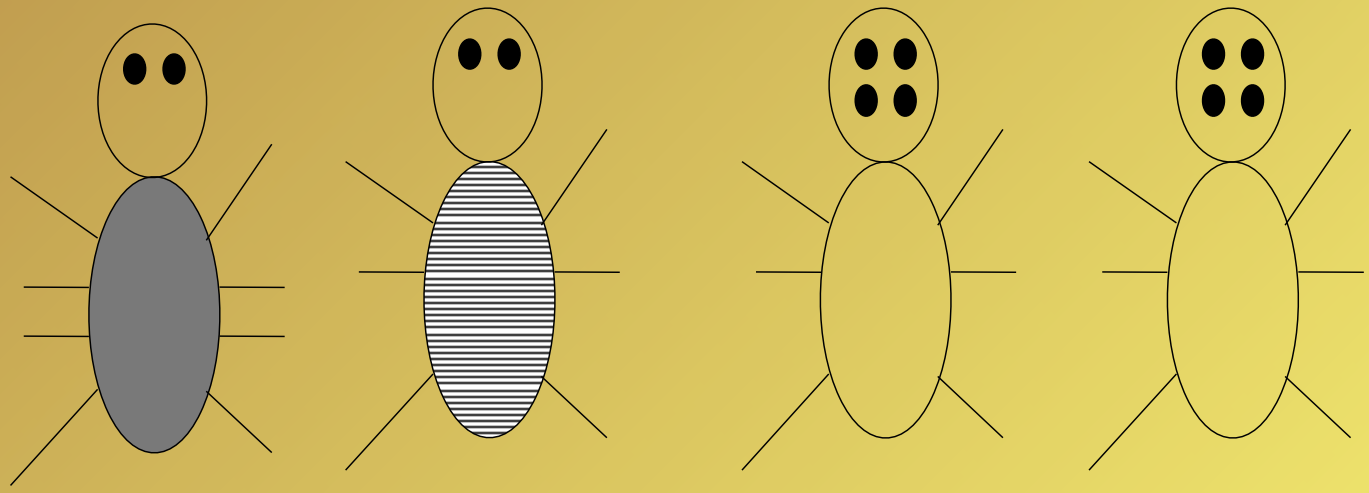
mouse

carrot

A POSSIBLE EVOLUTIONARY TREE FOR THESE SPECIES

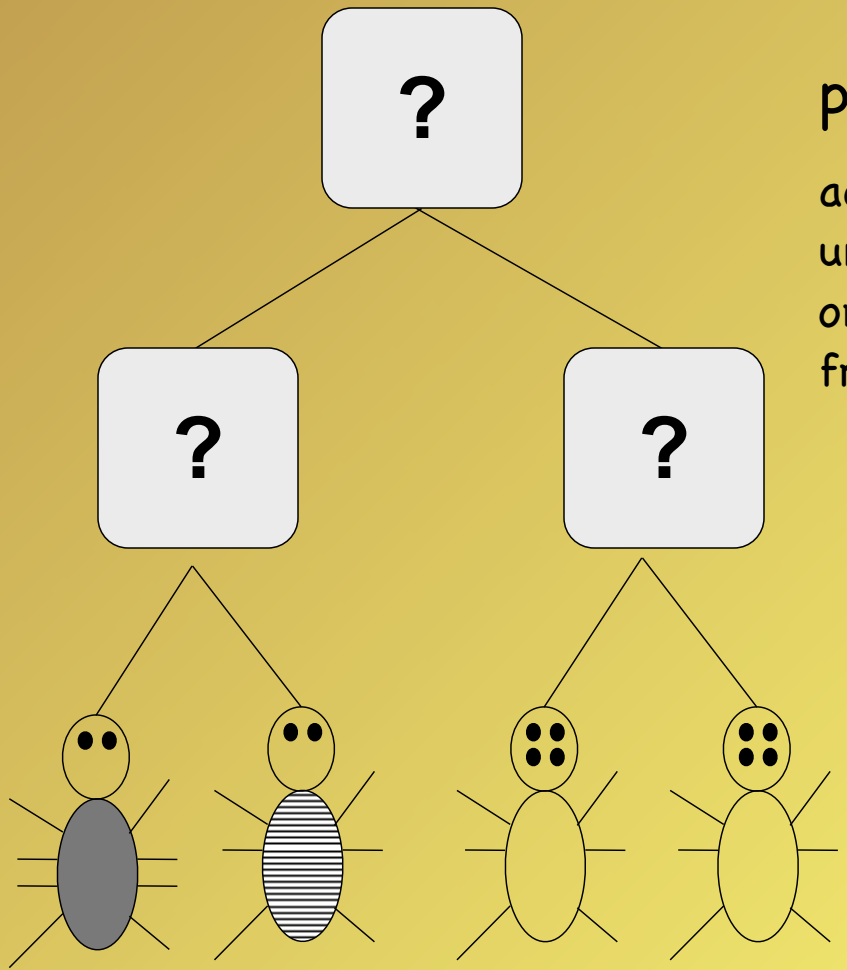


Bug Mutation: How did the bugs evolve??



Goal: Given an evolutionary "tree skeleton" for several organisms, determine the evolutionary change necessary to explain the evolution of the organisms

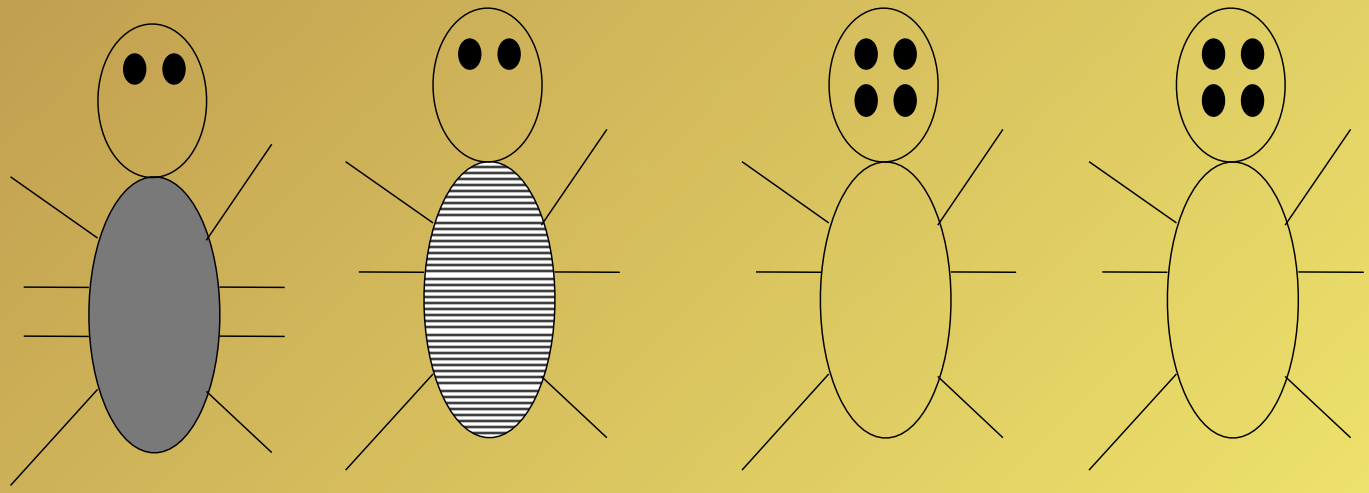
We call this problem the **small parsimony problem.**



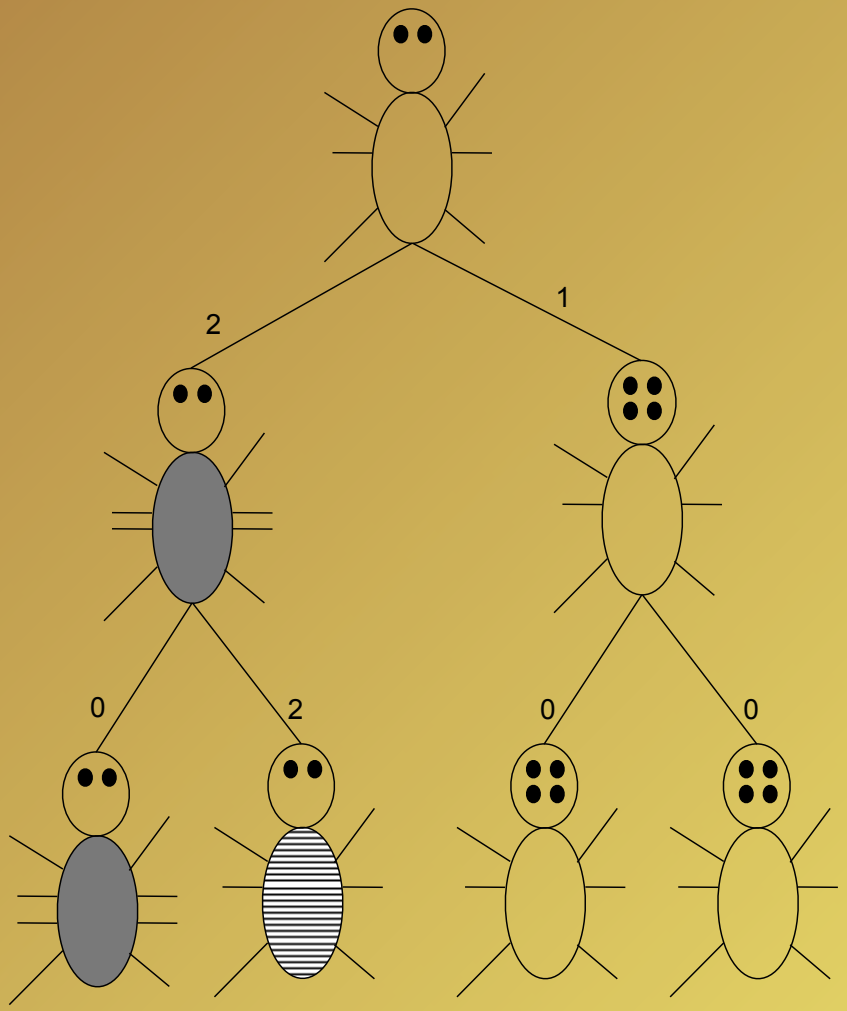
parsimonious |ˌpärsəˈmōnēəs|

adjective
unwilling to spend money or use resources; stingy or frugal

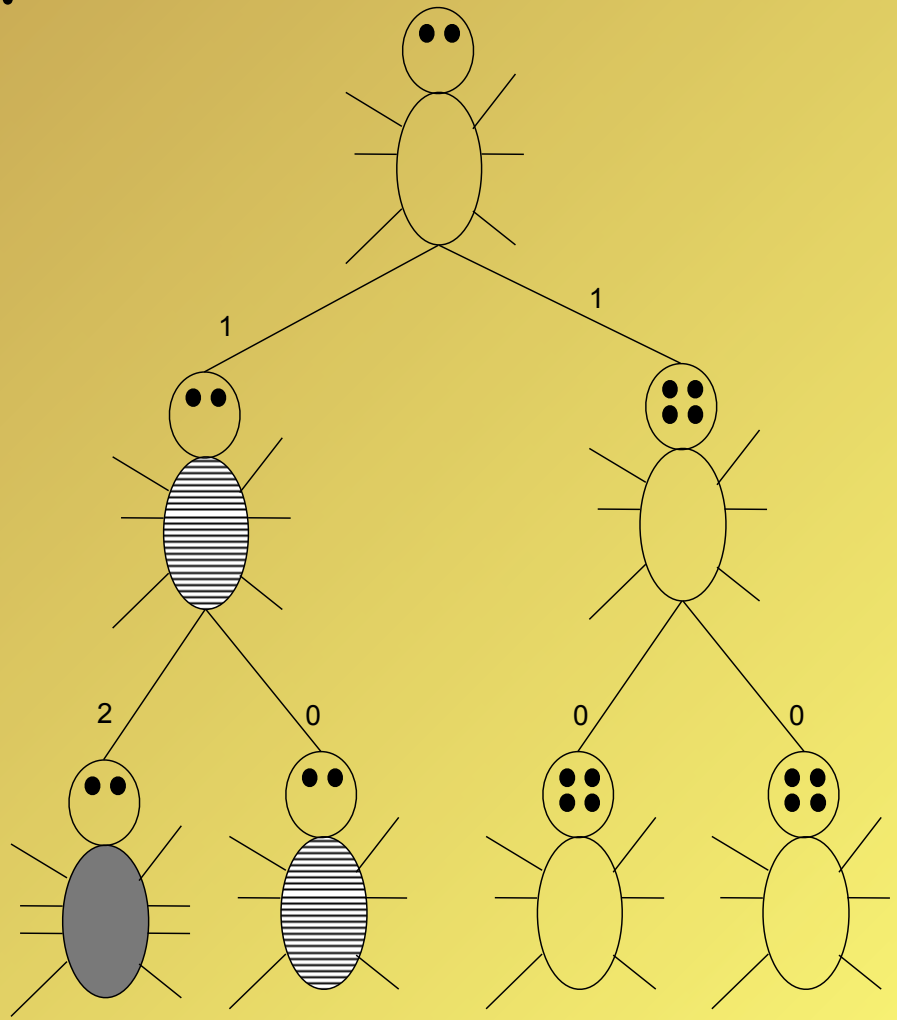
Given these bugs …



Which Evolutionary Tree is more likely (parsimonious)?



Parsimony Score: 5



Parsimony Score: 4

Clicker question: What is the score of this tree?

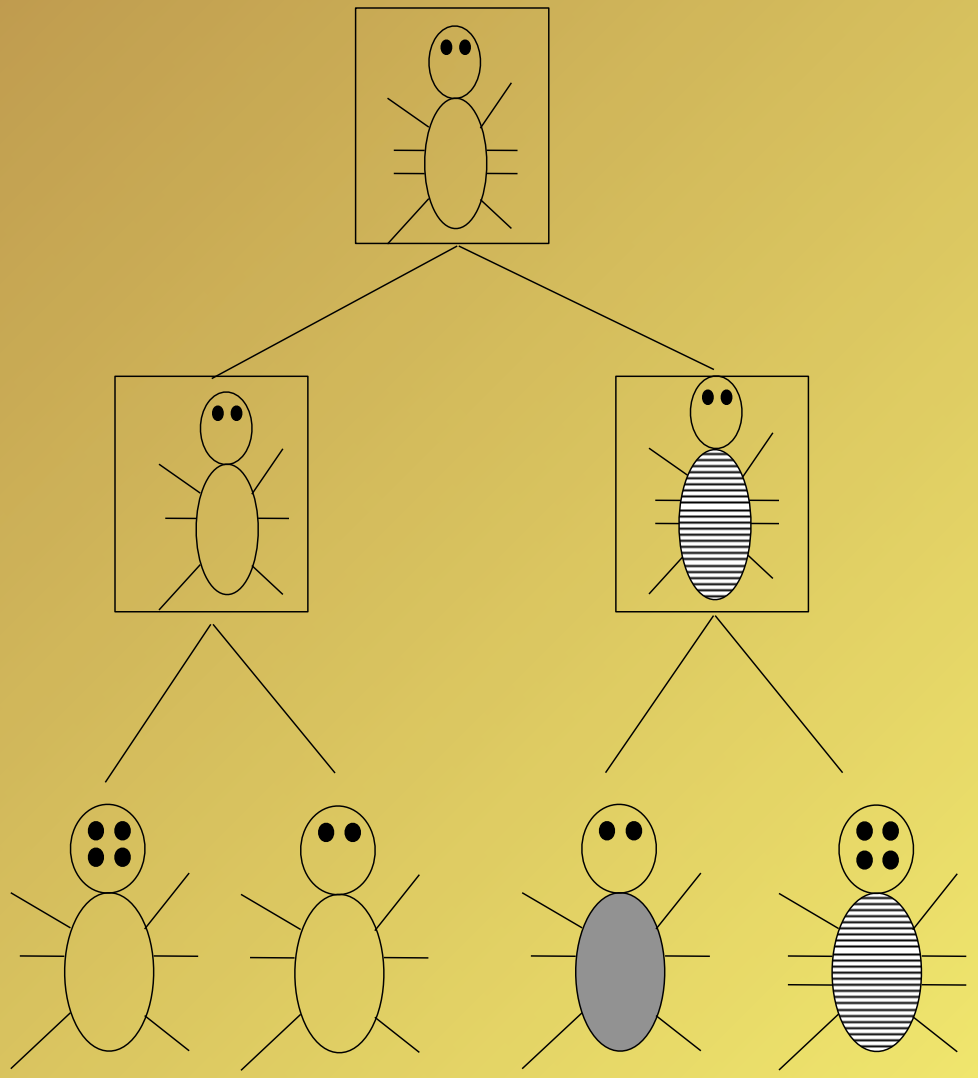
A: 3

B: 4

C: 5

D: 6

E: 7



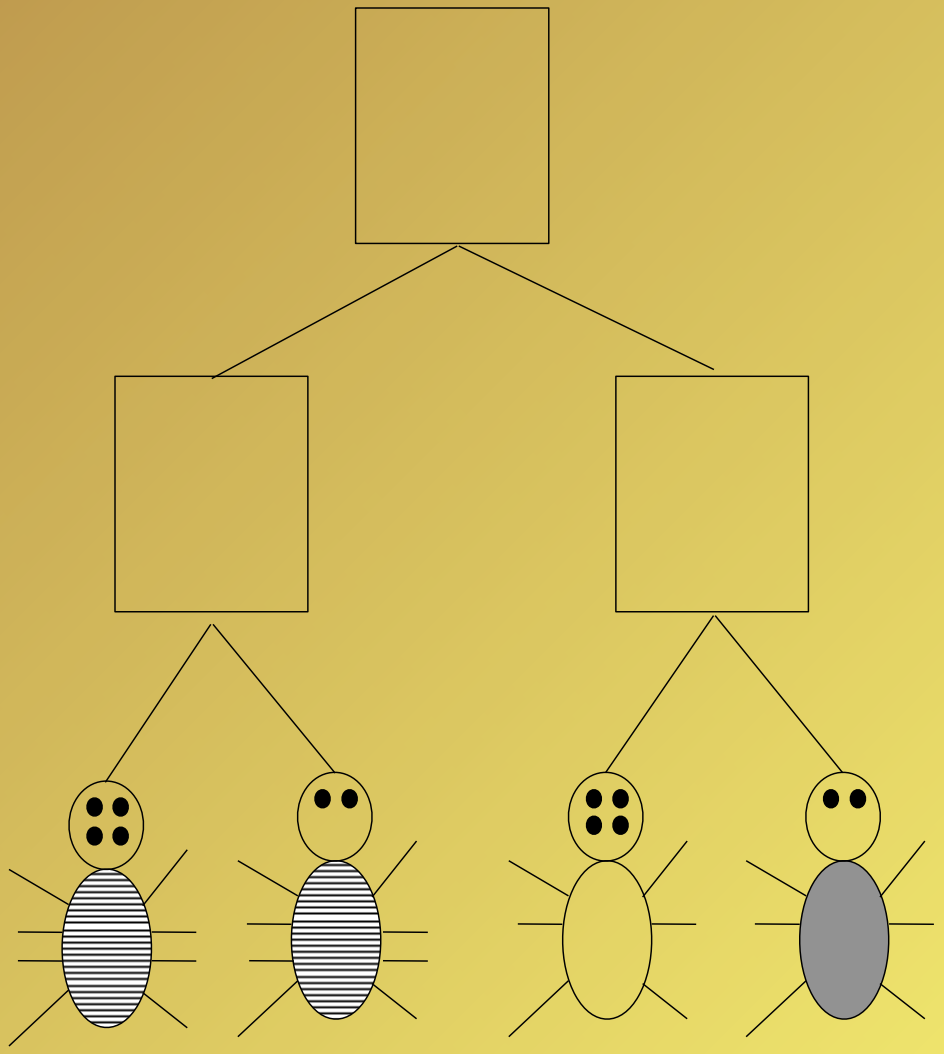
A most parsimonious tree follows the principle of **Occam's Razor**

In Science, we often follow the principle of

Occam's Razor:

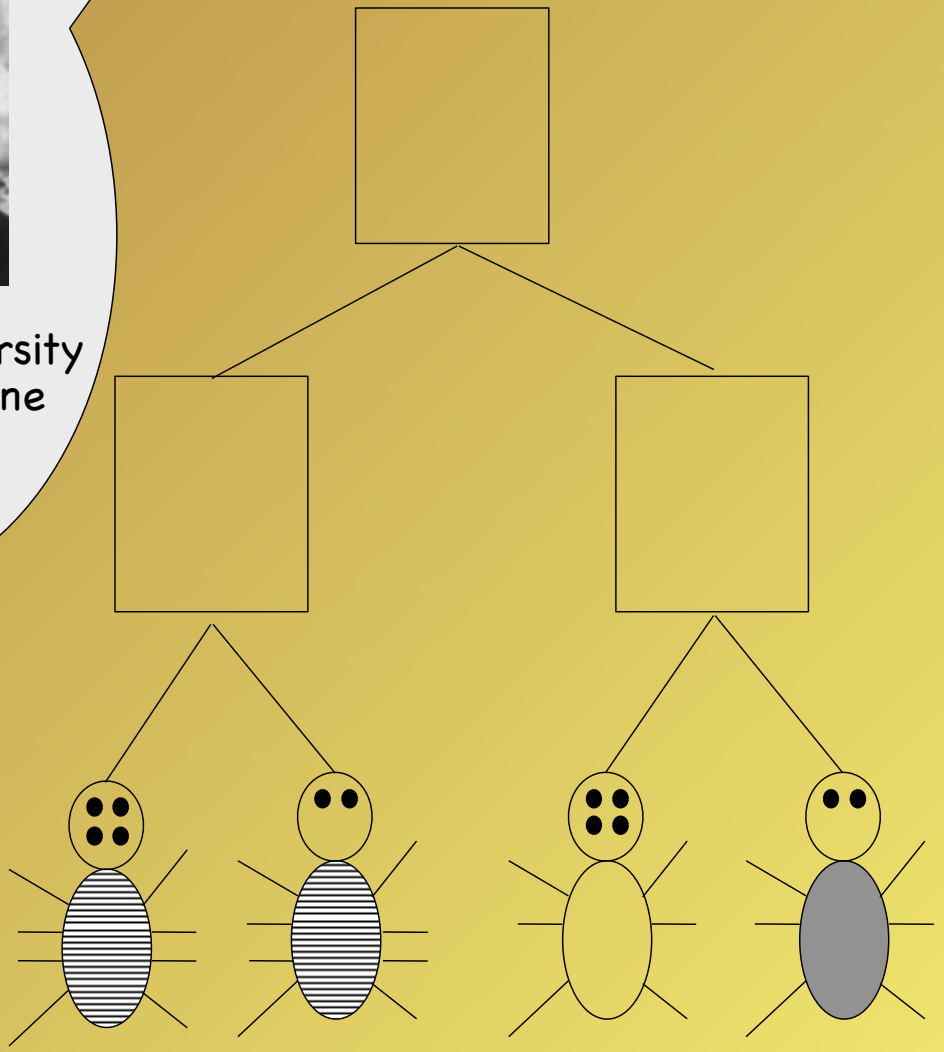
When competing hypotheses are equal in other respects, the Occam's razor principle recommends the selection of a hypothesis that introduces fewest assumptions while still sufficiently answering the question.

Find most parsimonious ancestors for this suggested tree topology!!



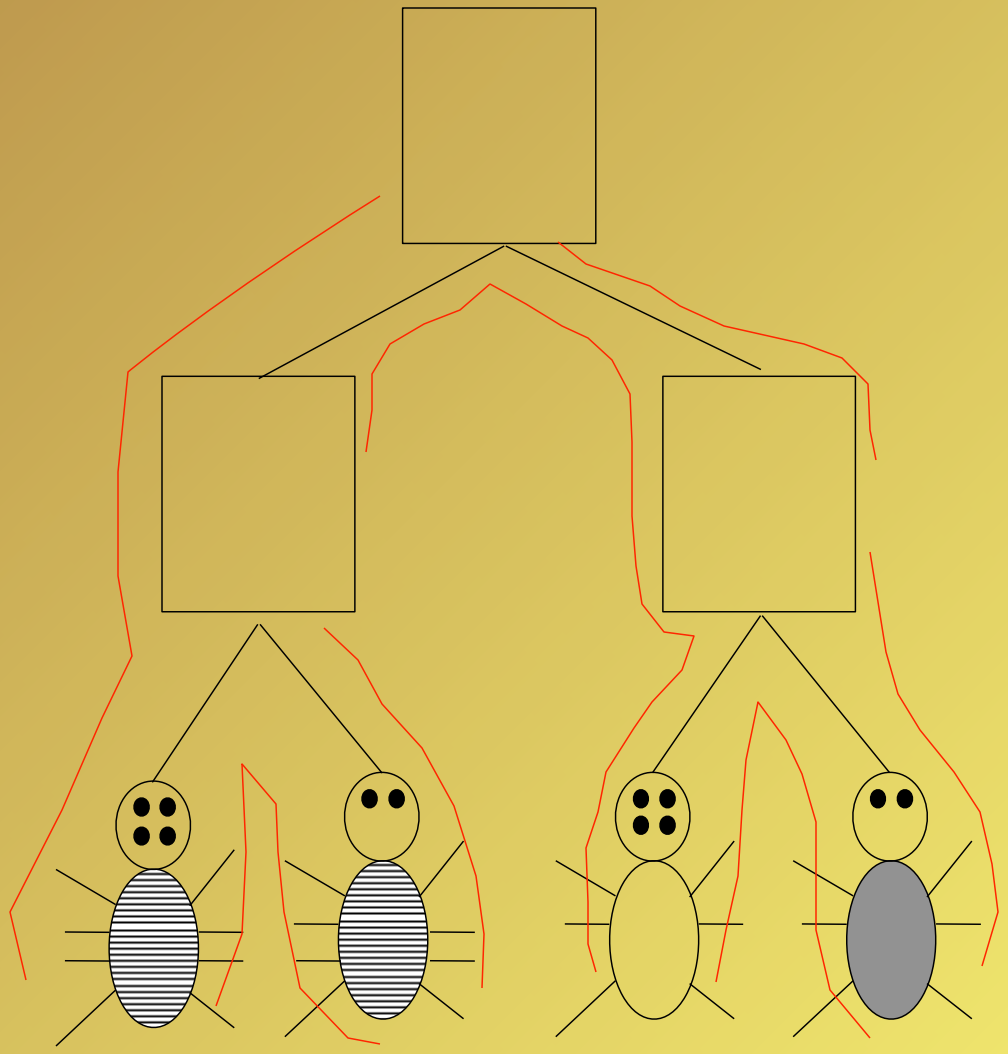
Fitch's Algorithm

Walter Fitch
(1929-2011), University of California, Irvine

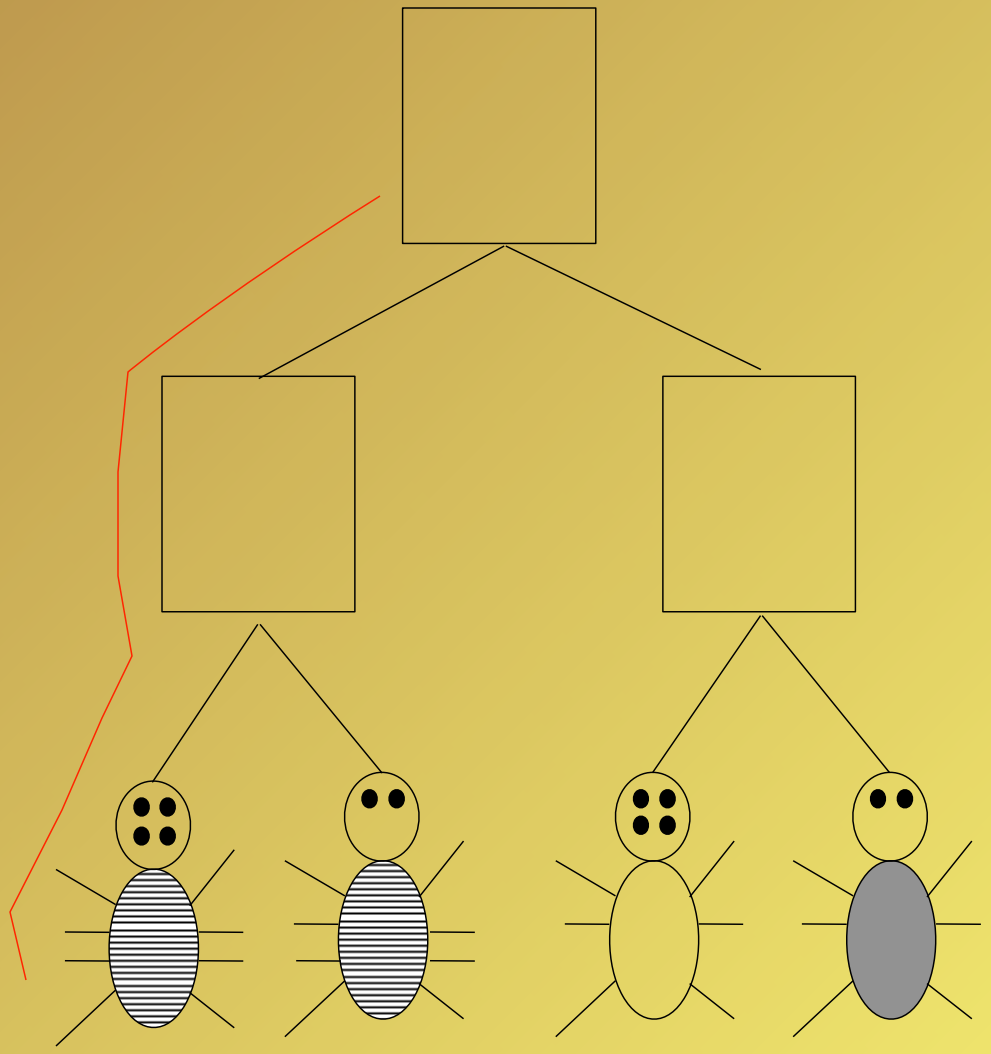


Step 1: Do a **postorder traversal** of the tree and label every node in the tree with a set of possible best features.

What is a postorder traversal of a tree?

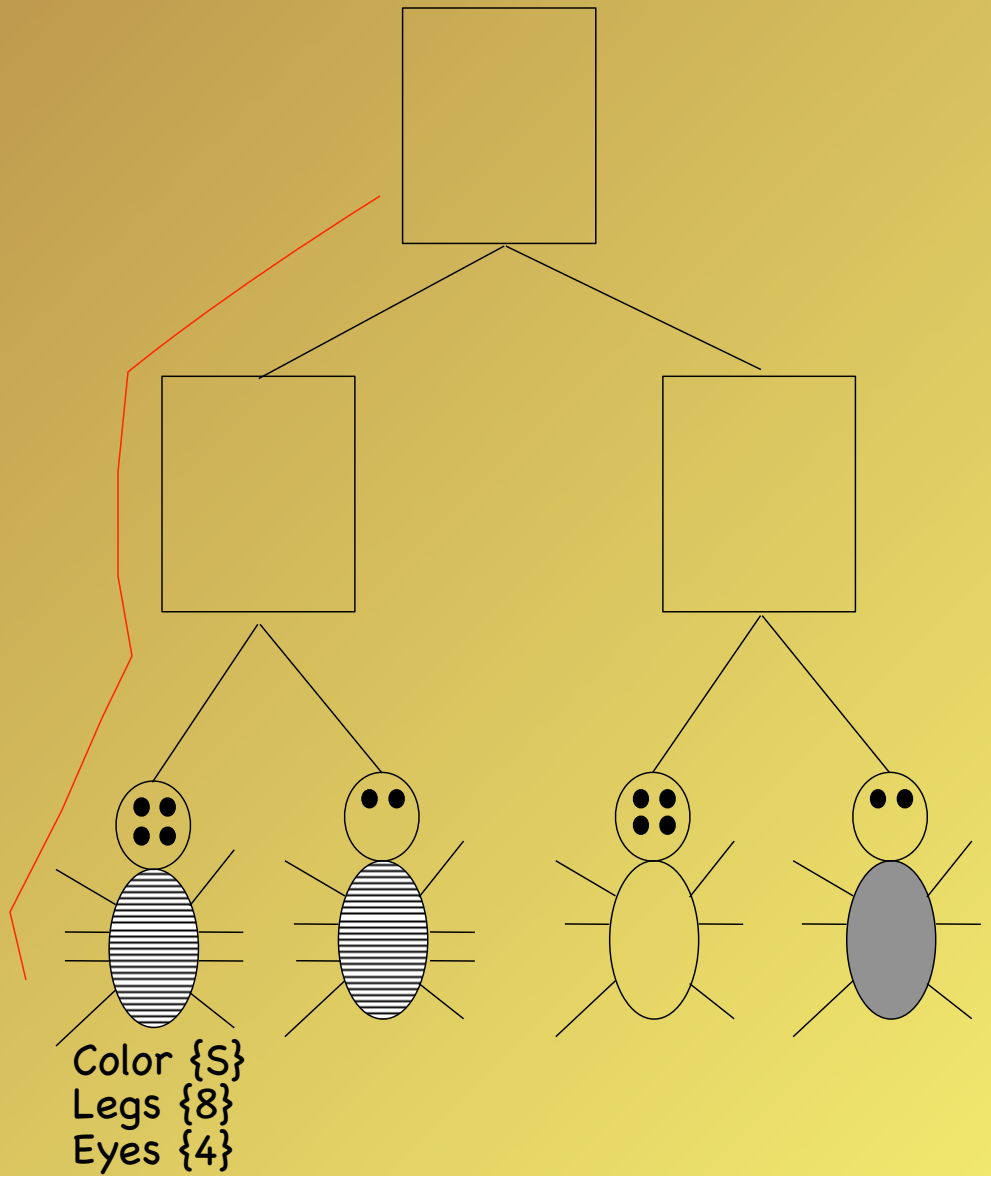


Fitch's Algorithm



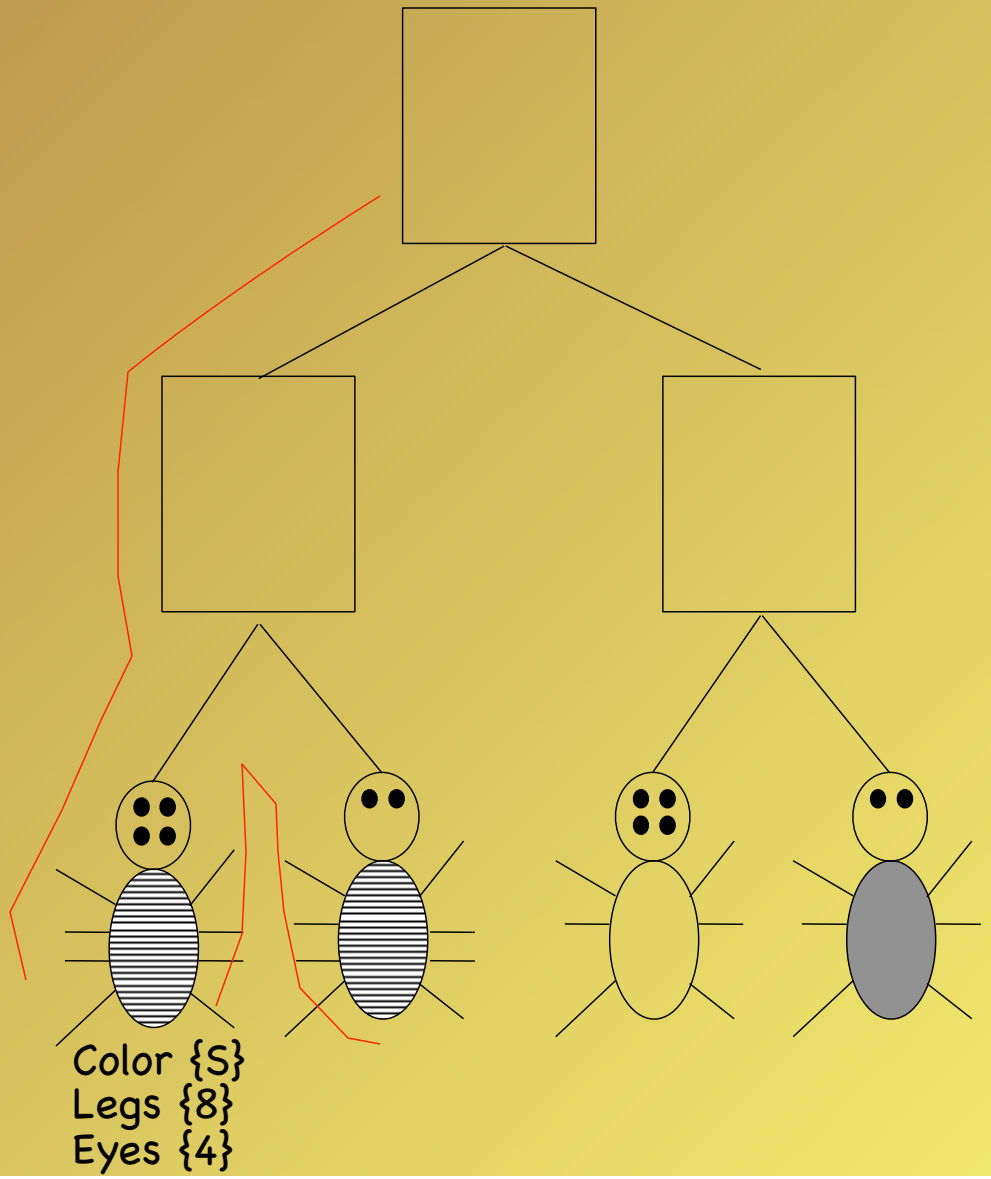
Step 1: Do a **postorder traversal** of the tree and label every node in the tree with a set of possible best features.

Fitch's Algorithm



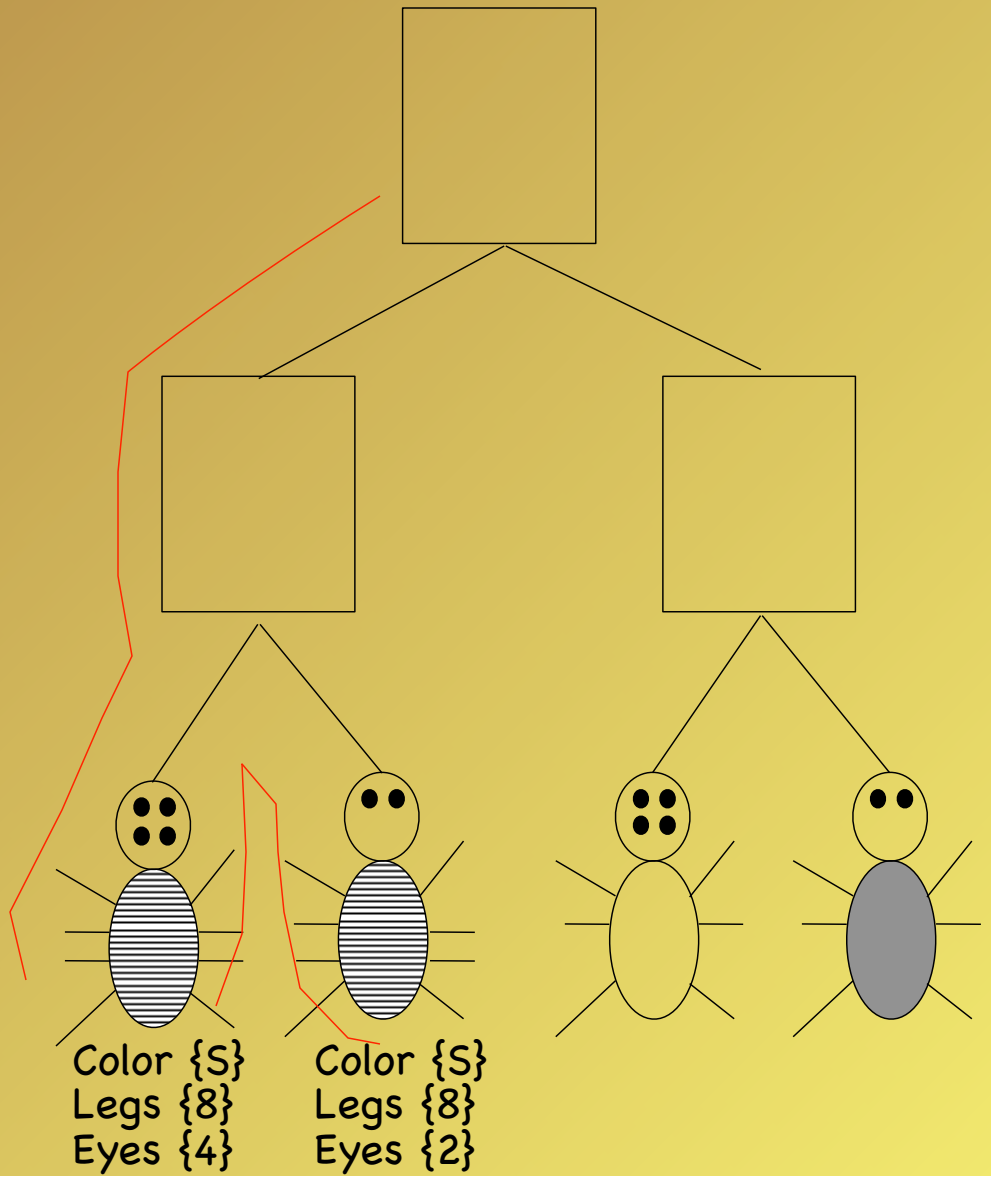
Step 1: Do a **postorder traversal** of the tree and label every node in the tree with a set of possible best features.

Fitch's Algorithm



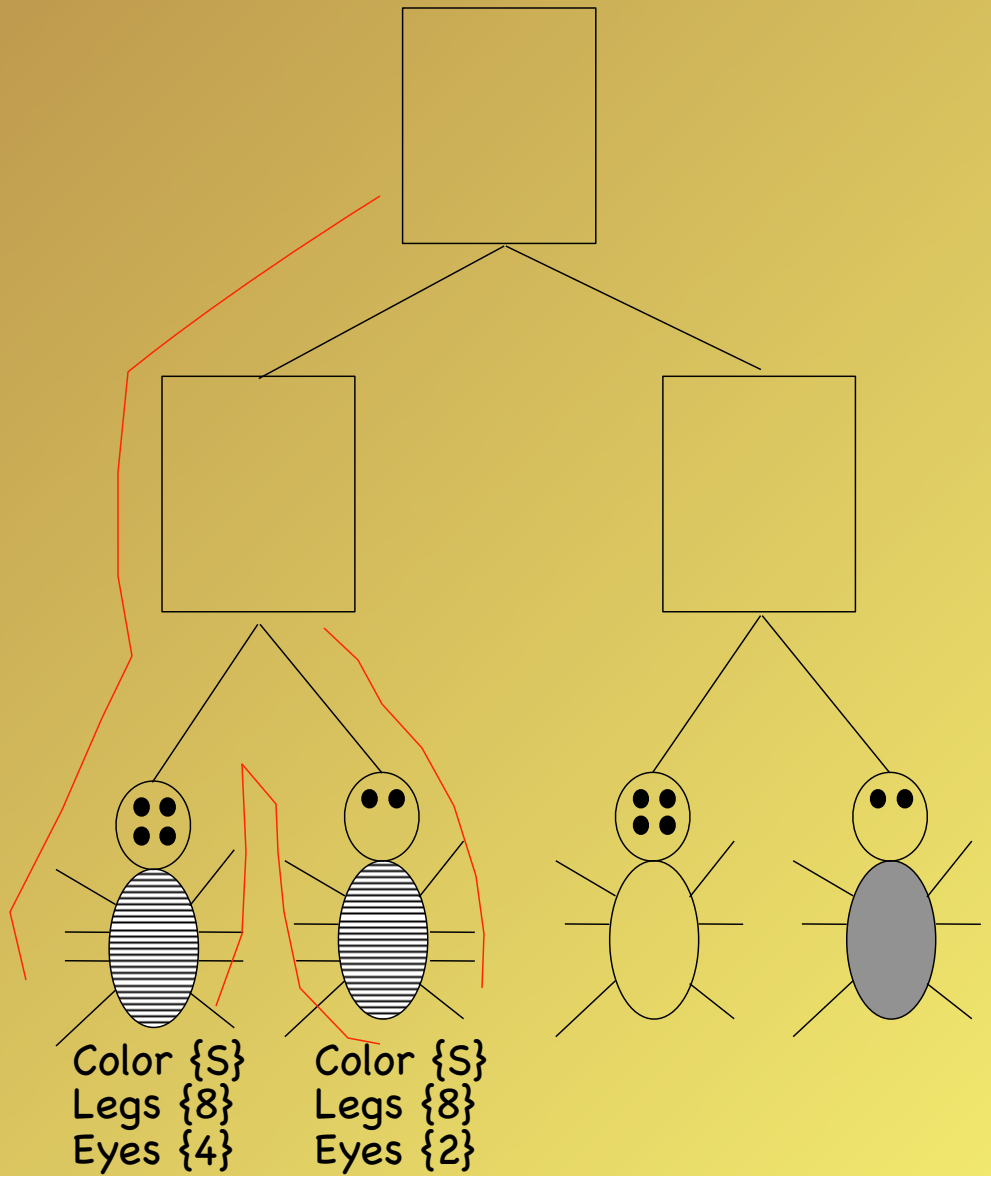
Step 1: Do a **postorder traversal** of the tree and label every node in the tree with a set of possible best features.

Fitch's Algorithm



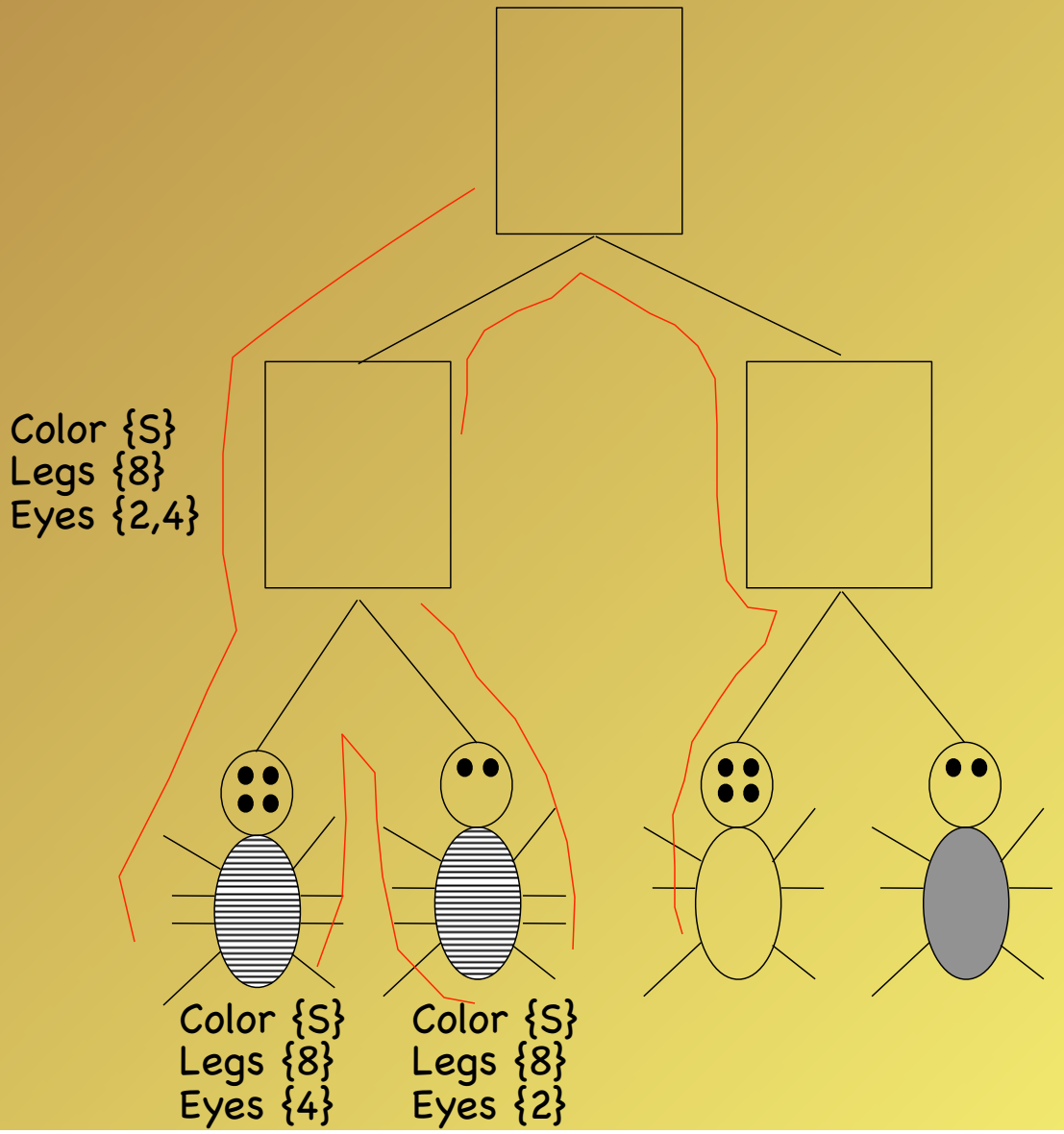
Step 1: Do a **postorder traversal** of the tree and label every node in the tree with a set of possible best features.

Fitch's Algorithm



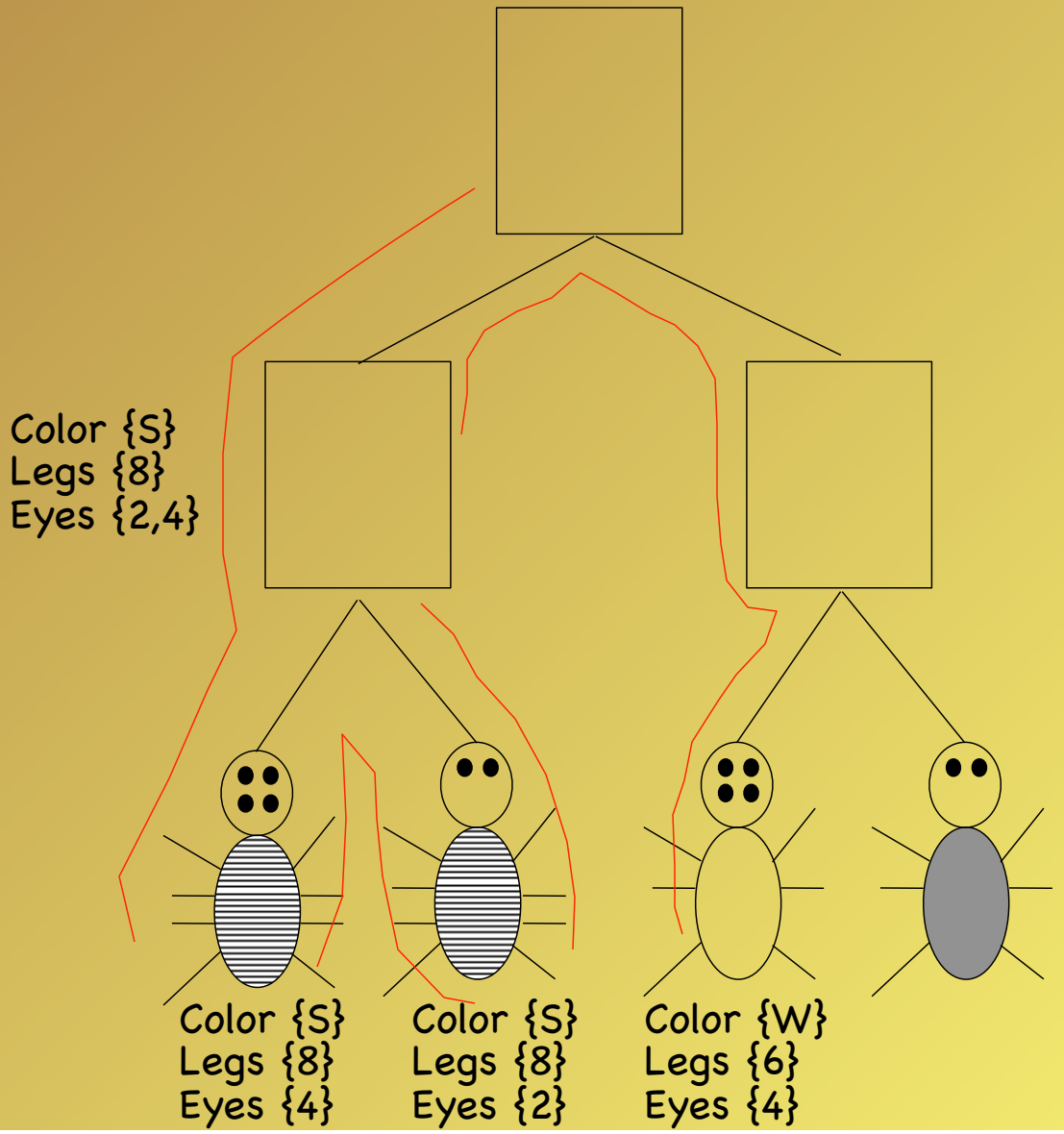
Step 1: Do a **postorder traversal** of the tree and label every node in the tree with a set of possible best features.

Fitch's Algorithm



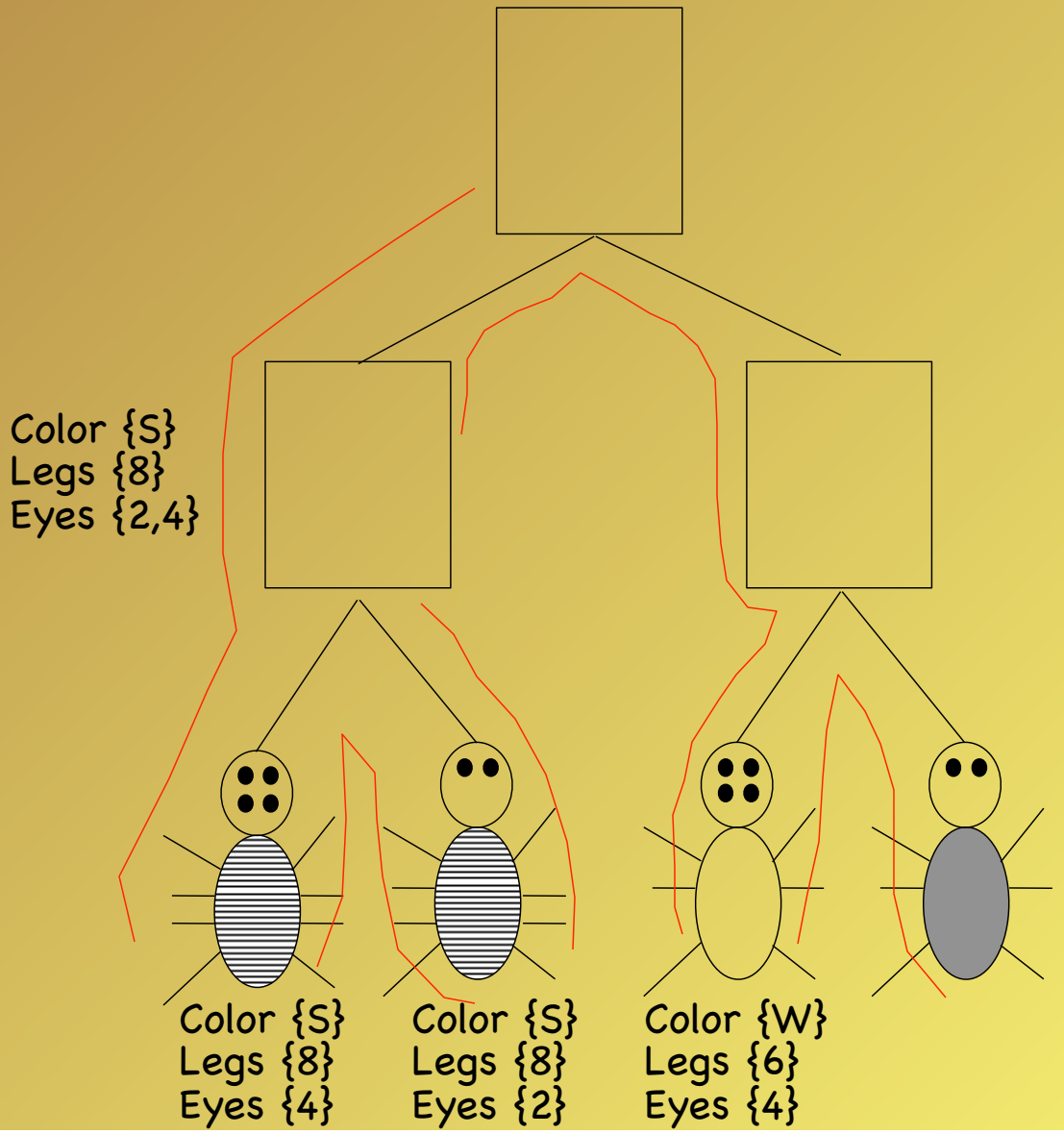
Step 1: Do a **postorder traversal** of the tree and label every node in the tree with a set of possible best features.

Fitch's Algorithm



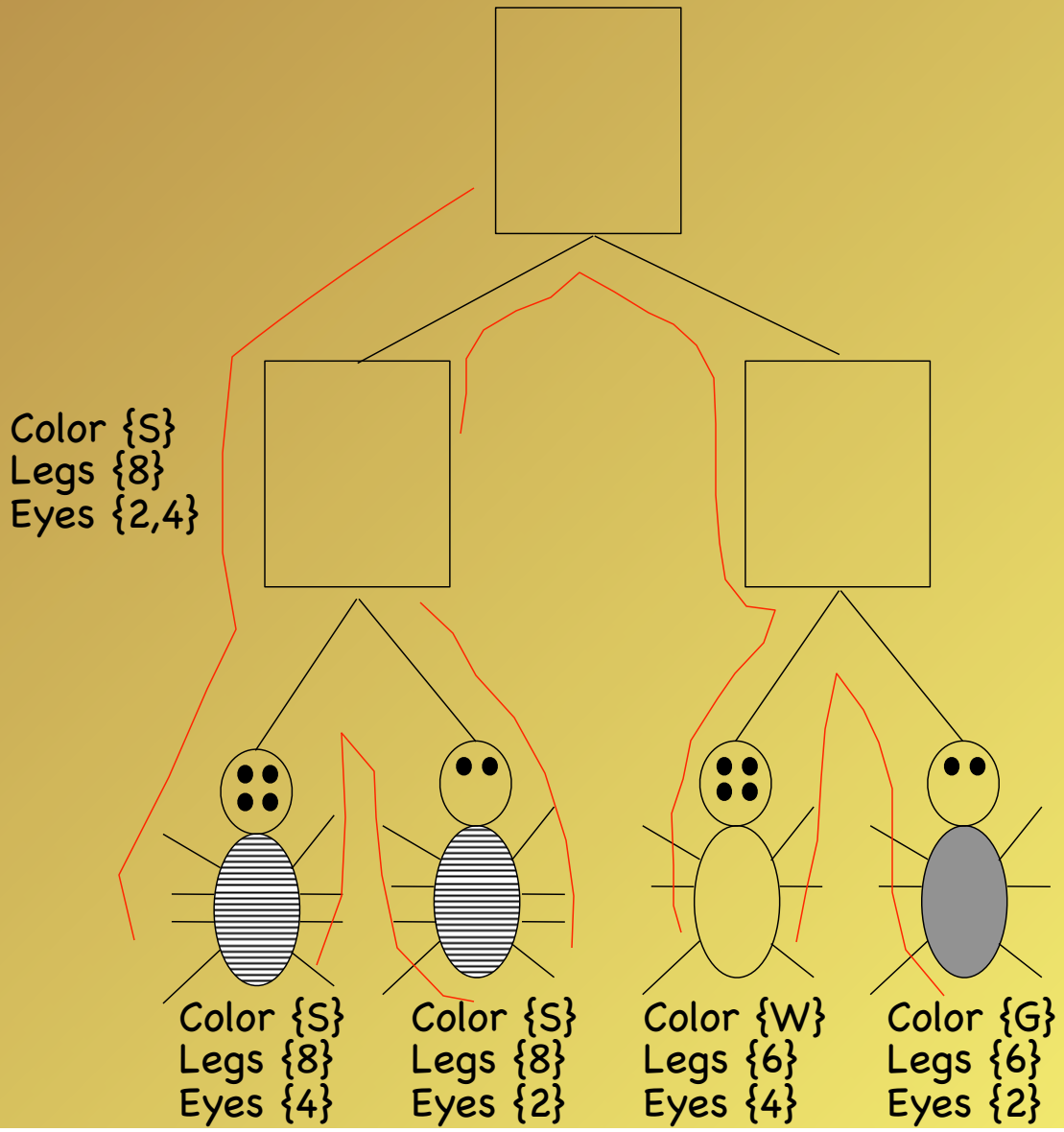
Step 1: Do a **postorder traversal** of the tree and label every node in the tree with a set of possible best features.

Fitch's Algorithm



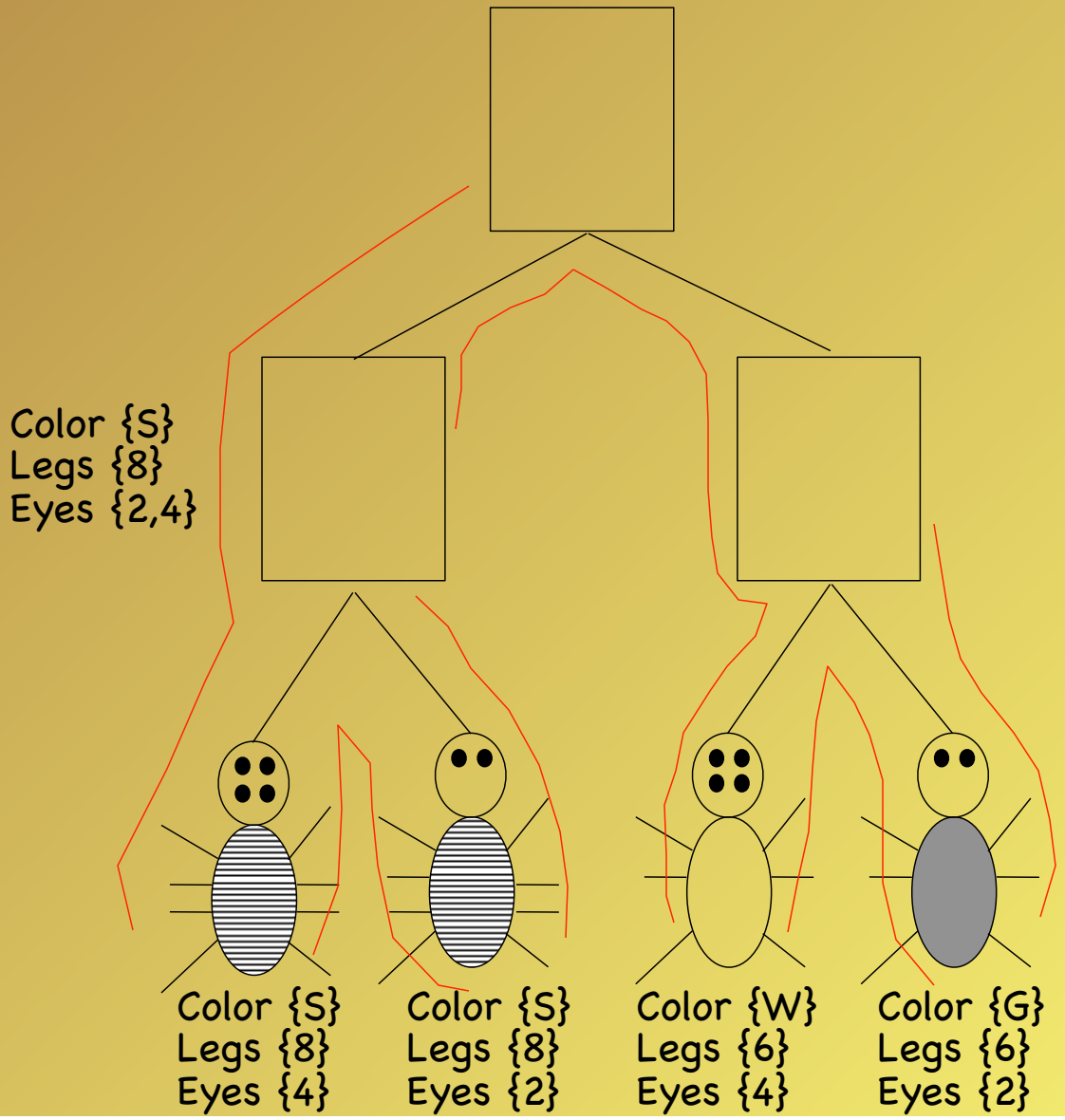
Step 1: Do a **postorder traversal** of the tree and label every node in the tree with a set of possible best features.

Fitch's Algorithm



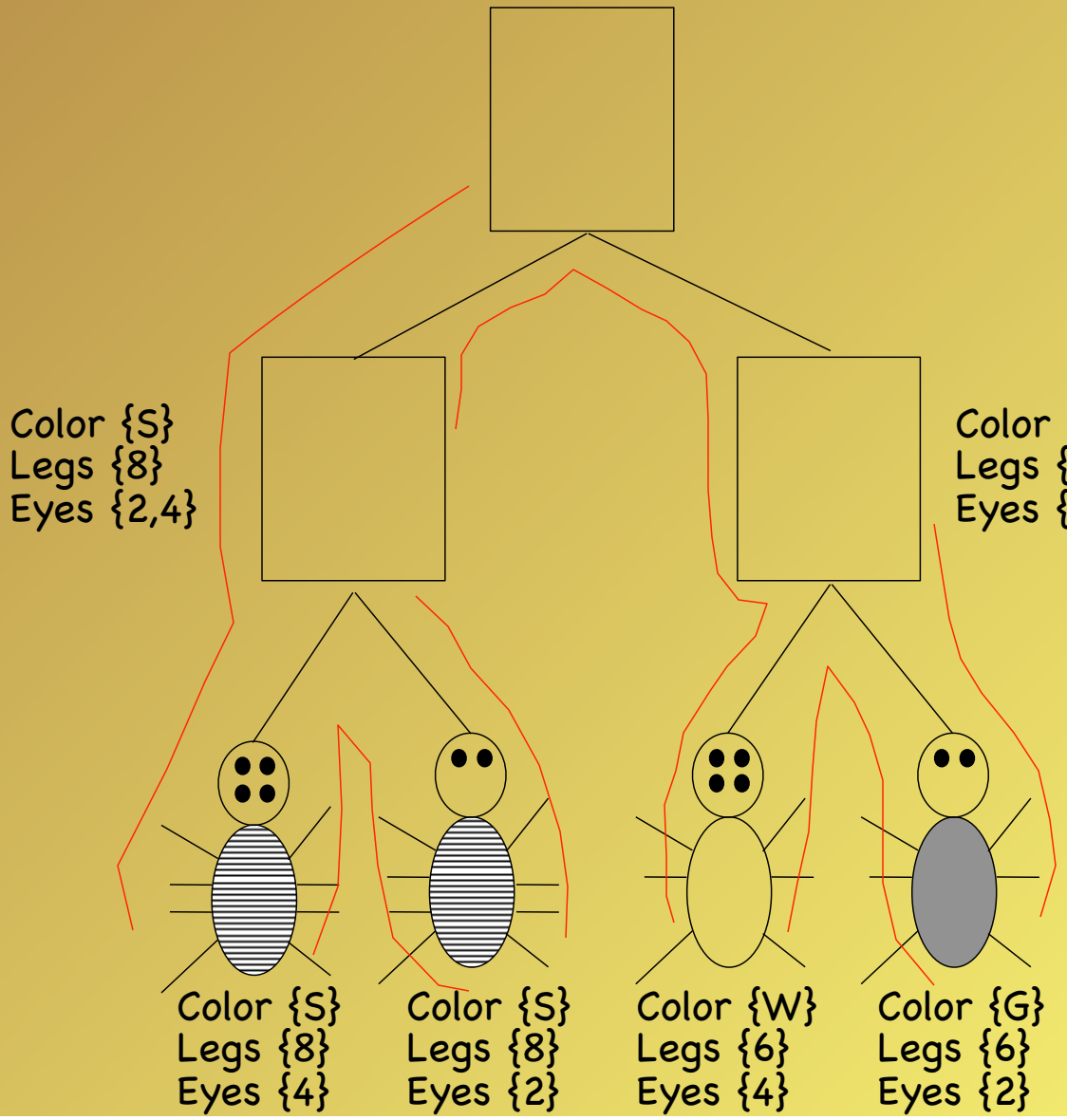
Step 1: Do a **postorder traversal** of the tree and label every node in the tree with a set of possible best features.

Fitch's Algorithm



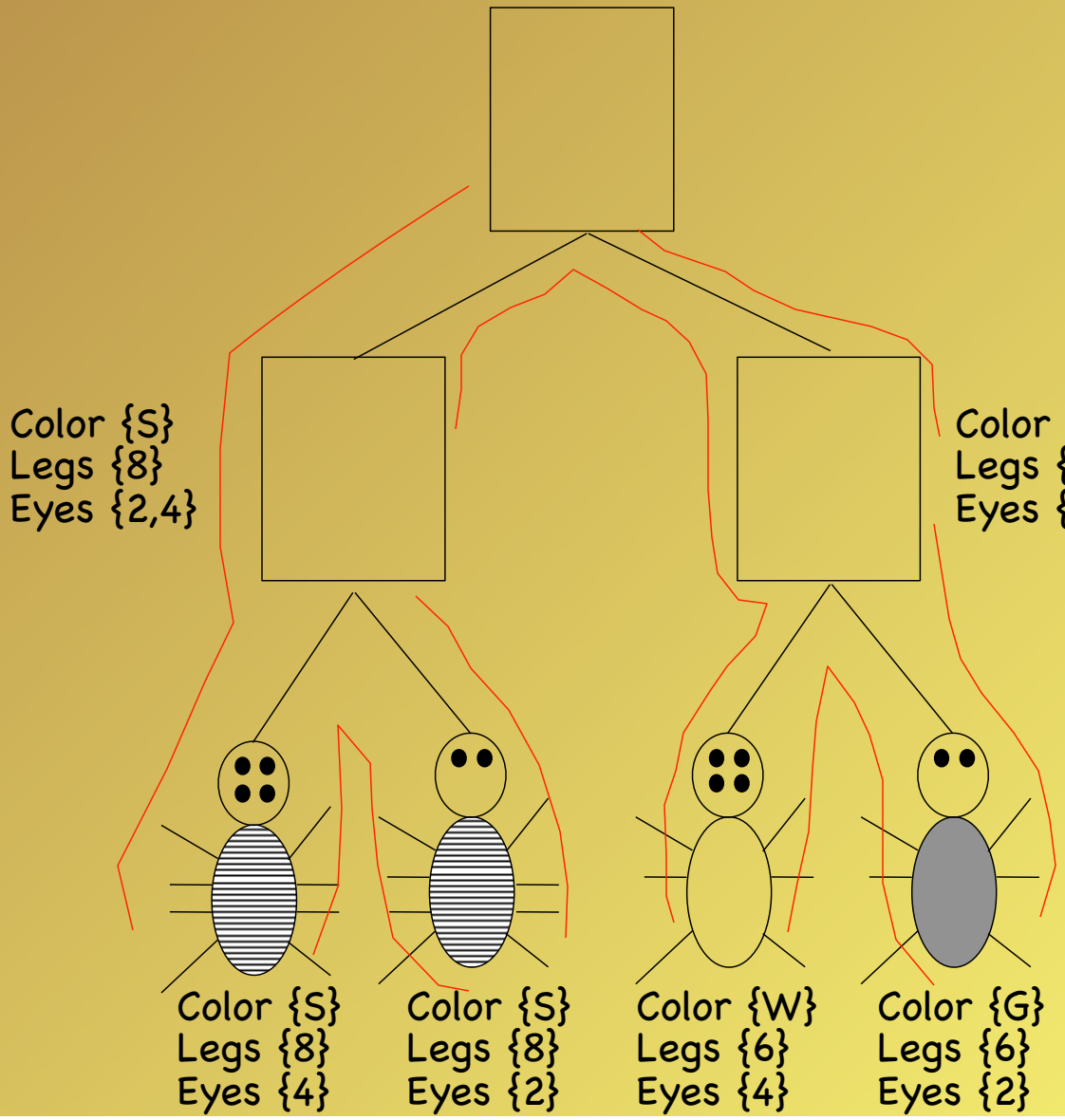
Step 1: Do a **postorder traversal** of the tree and label every node in the tree with a set of possible best features.

Fitch's Algorithm



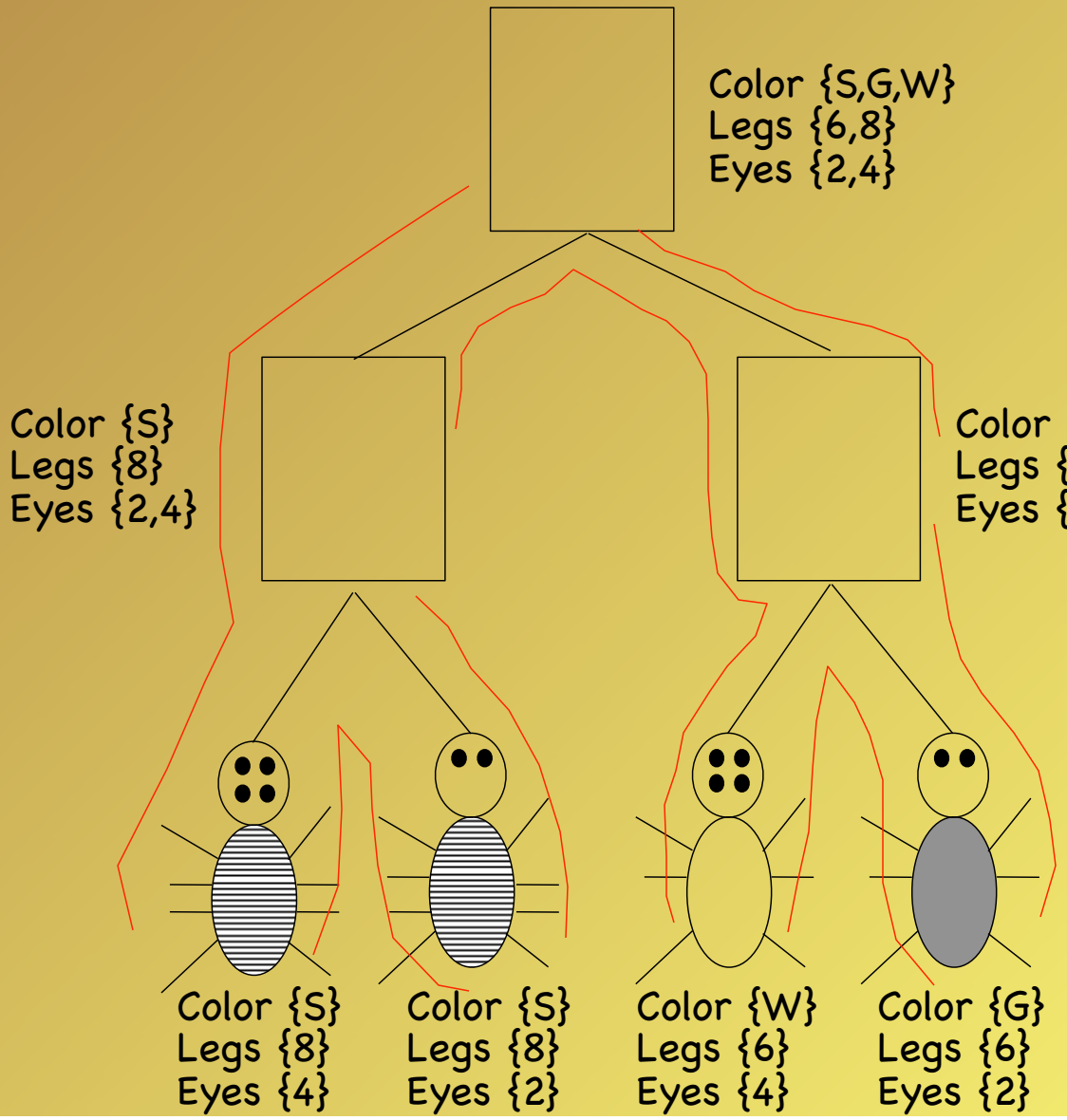
Step 1: Do a **postorder traversal** of the tree and label every node in the tree with a set of possible best features.

Fitch's Algorithm



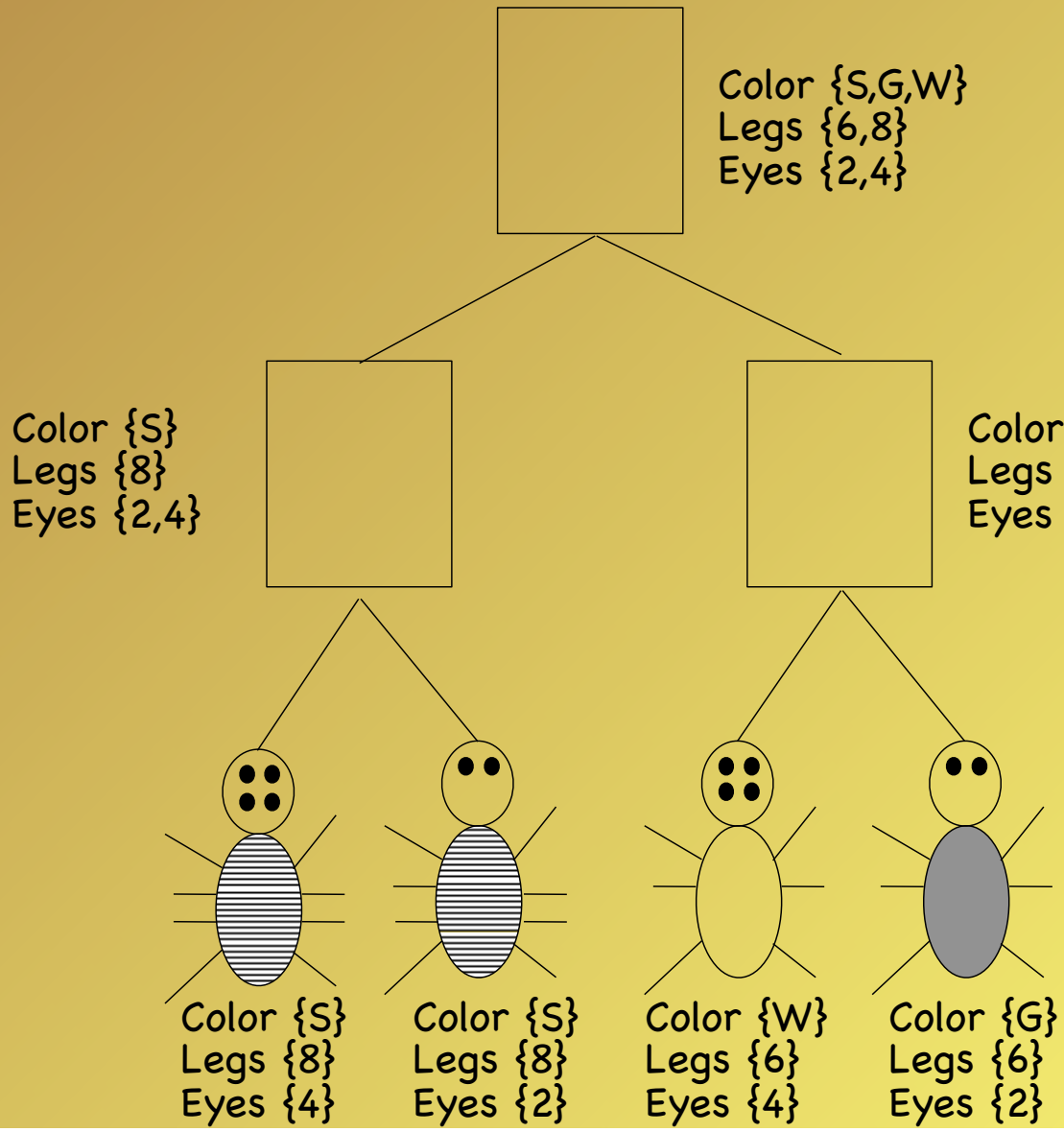
Step 1: Do a **postorder traversal** of the tree and label every node in the tree with a set of possible best features.

Fitch's Algorithm



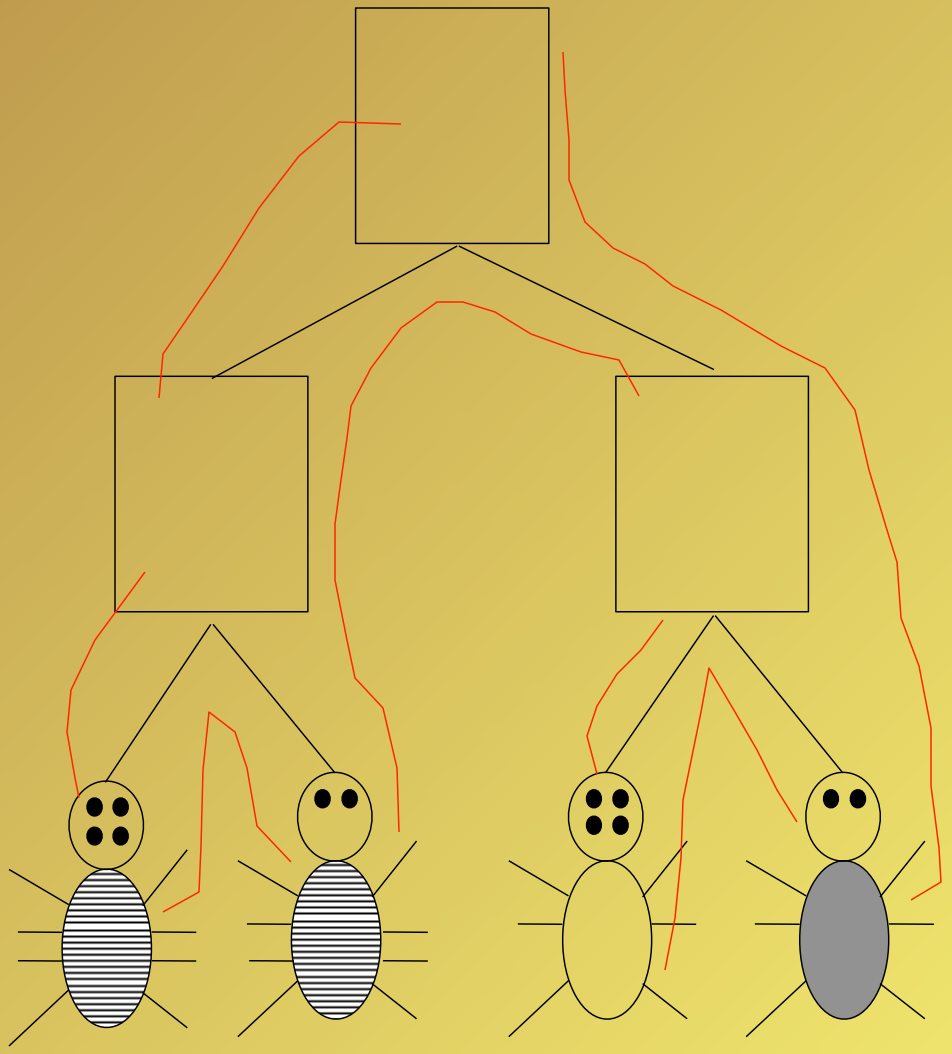
Step 1: Do a **postorder traversal** of the tree and label every node in the tree with a set of possible best features.

Fitch's Algorithm

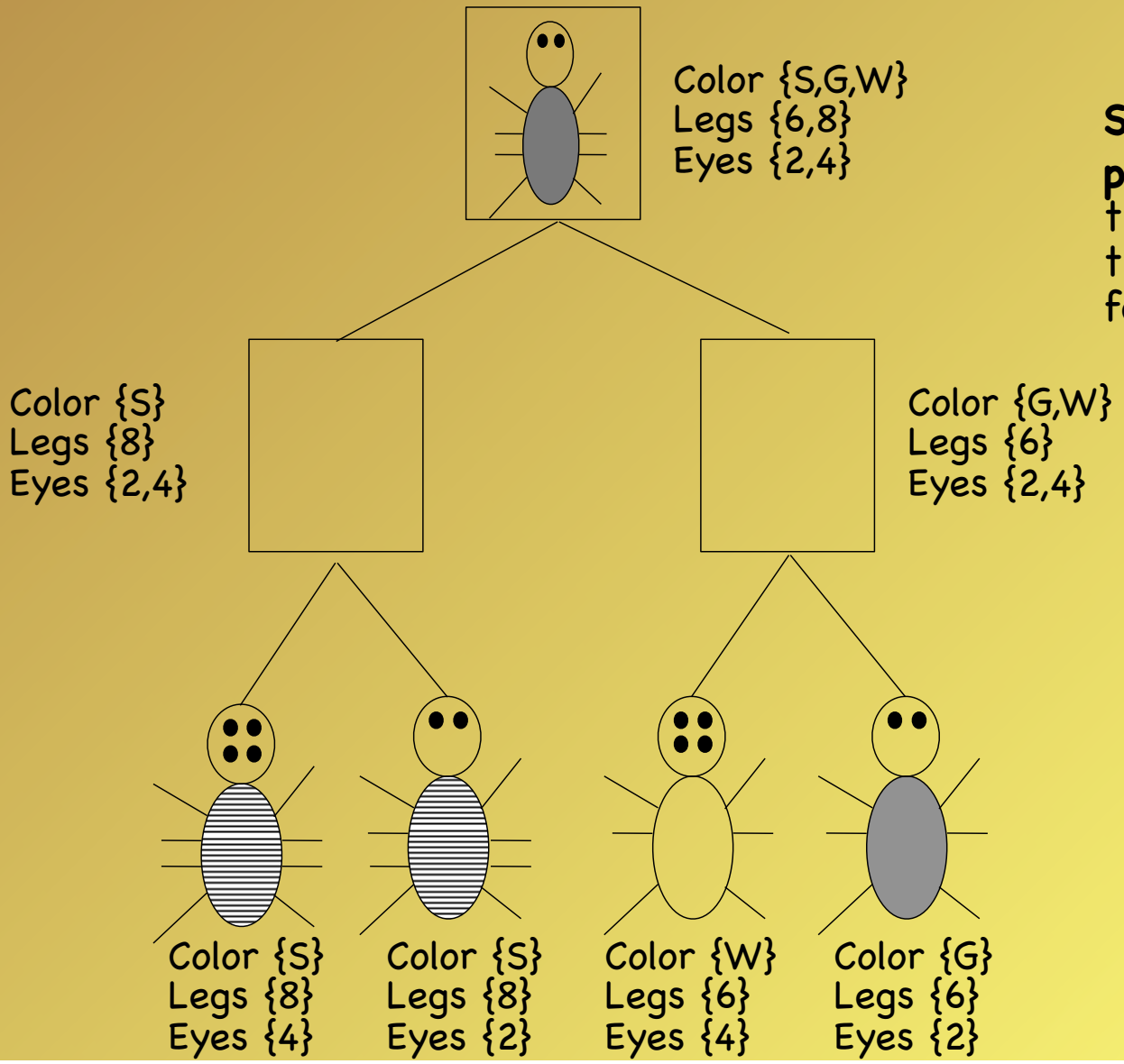


Step 2: Do a **preorder** traversal of the tree and decide on features.

What is a preorder traversal?

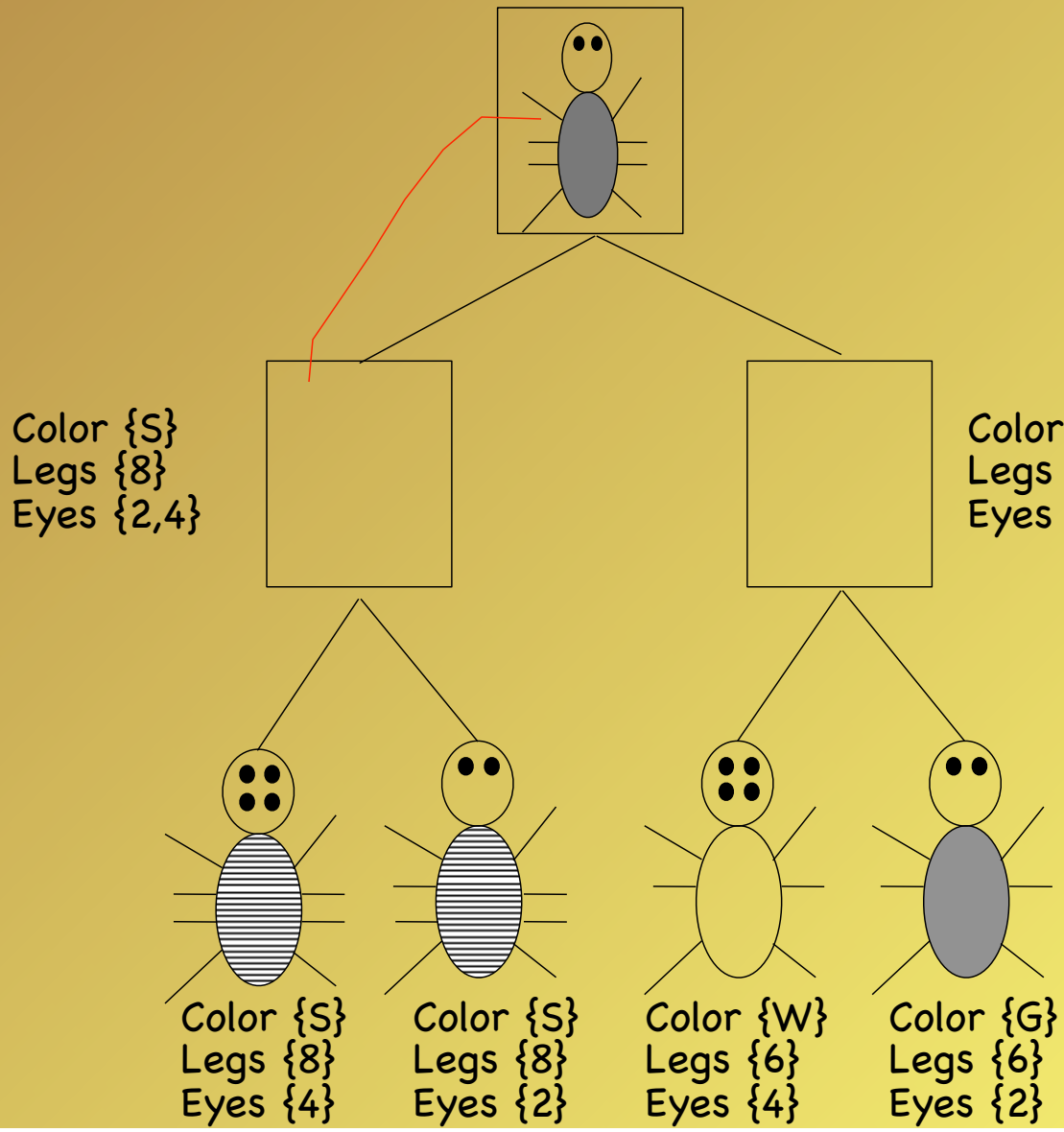


Fitch's Algorithm



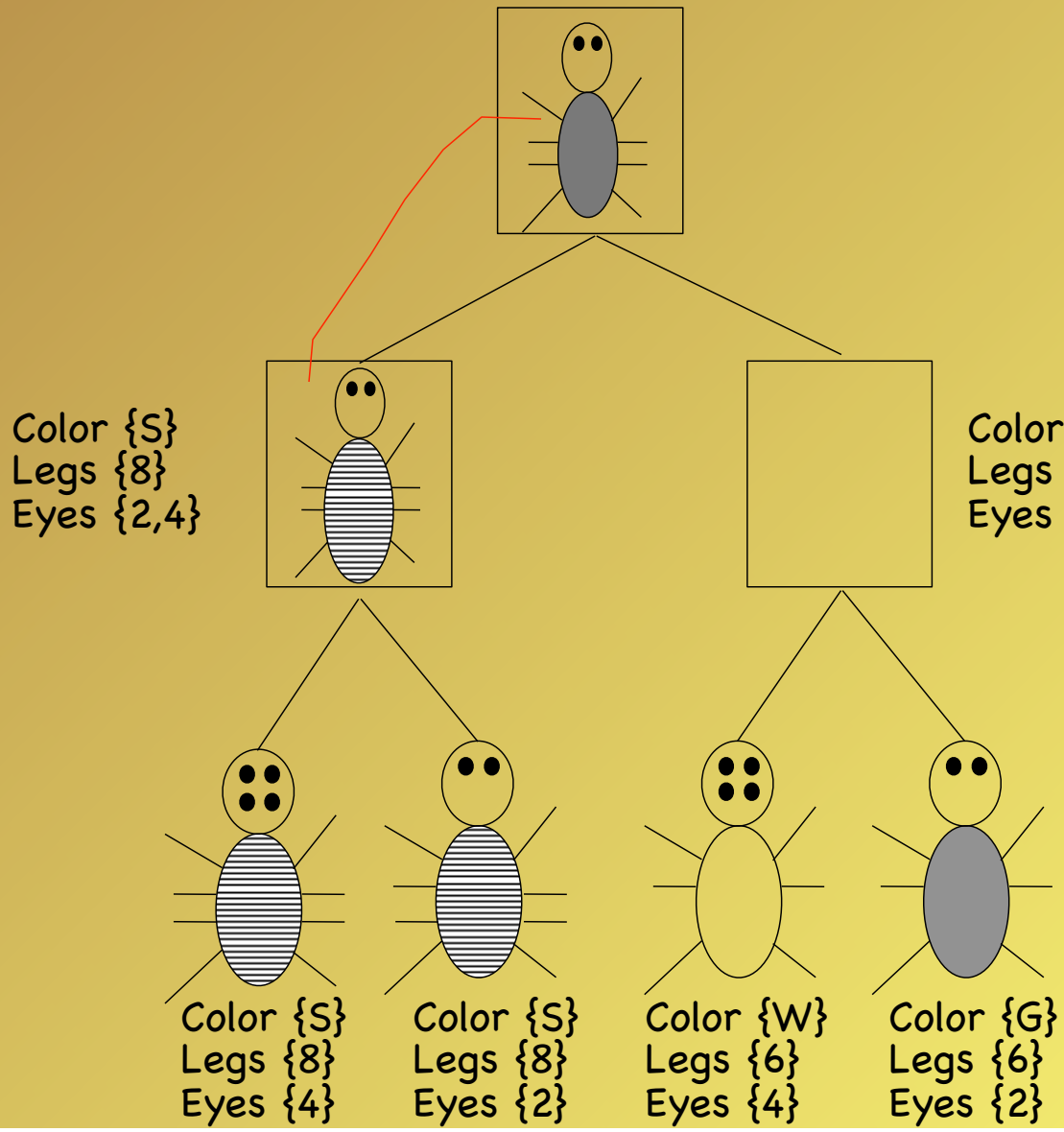
Step 2: Do a **preorder** traversal of the tree and decide on features.

Fitch's Algorithm



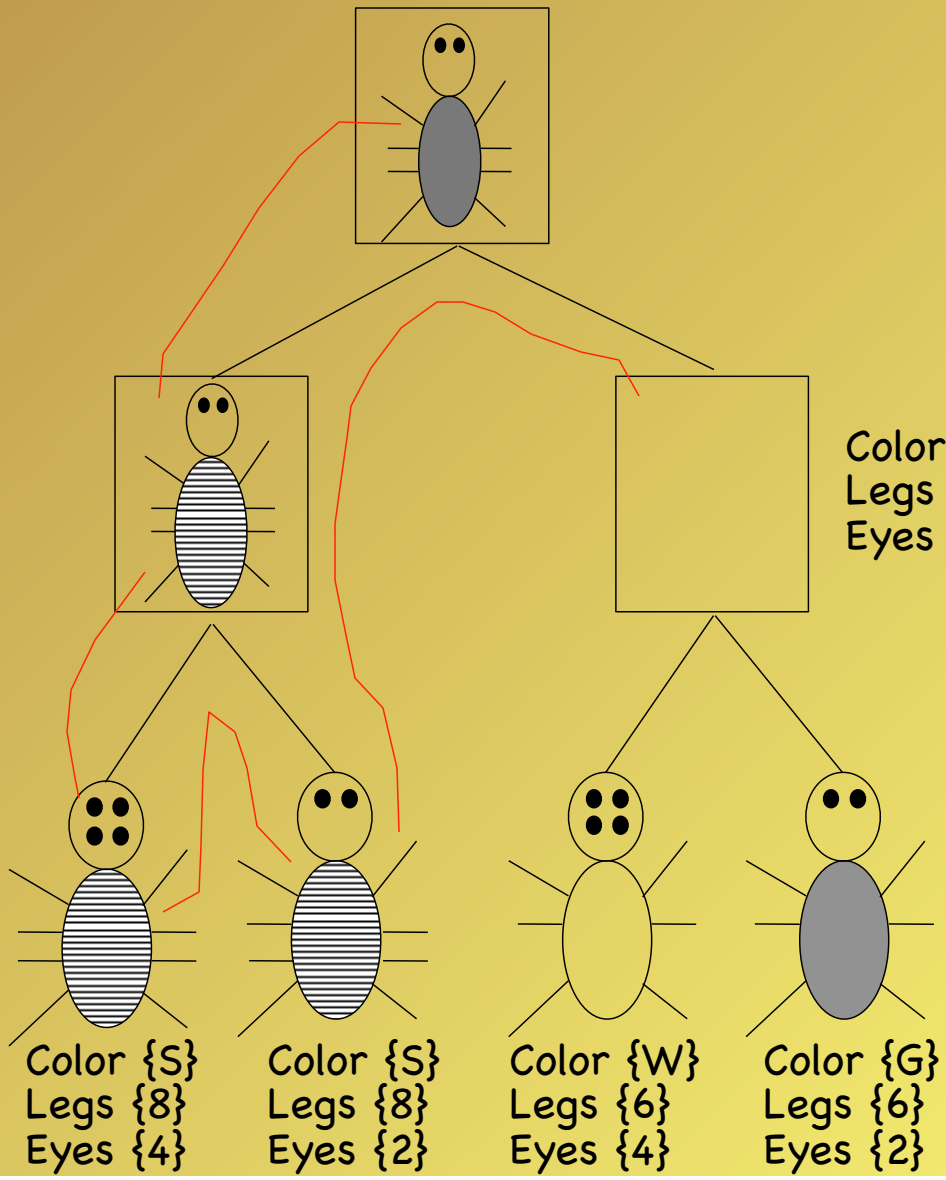
Step 2: Do a **preorder** traversal of the tree and decide on features.

Fitch's Algorithm



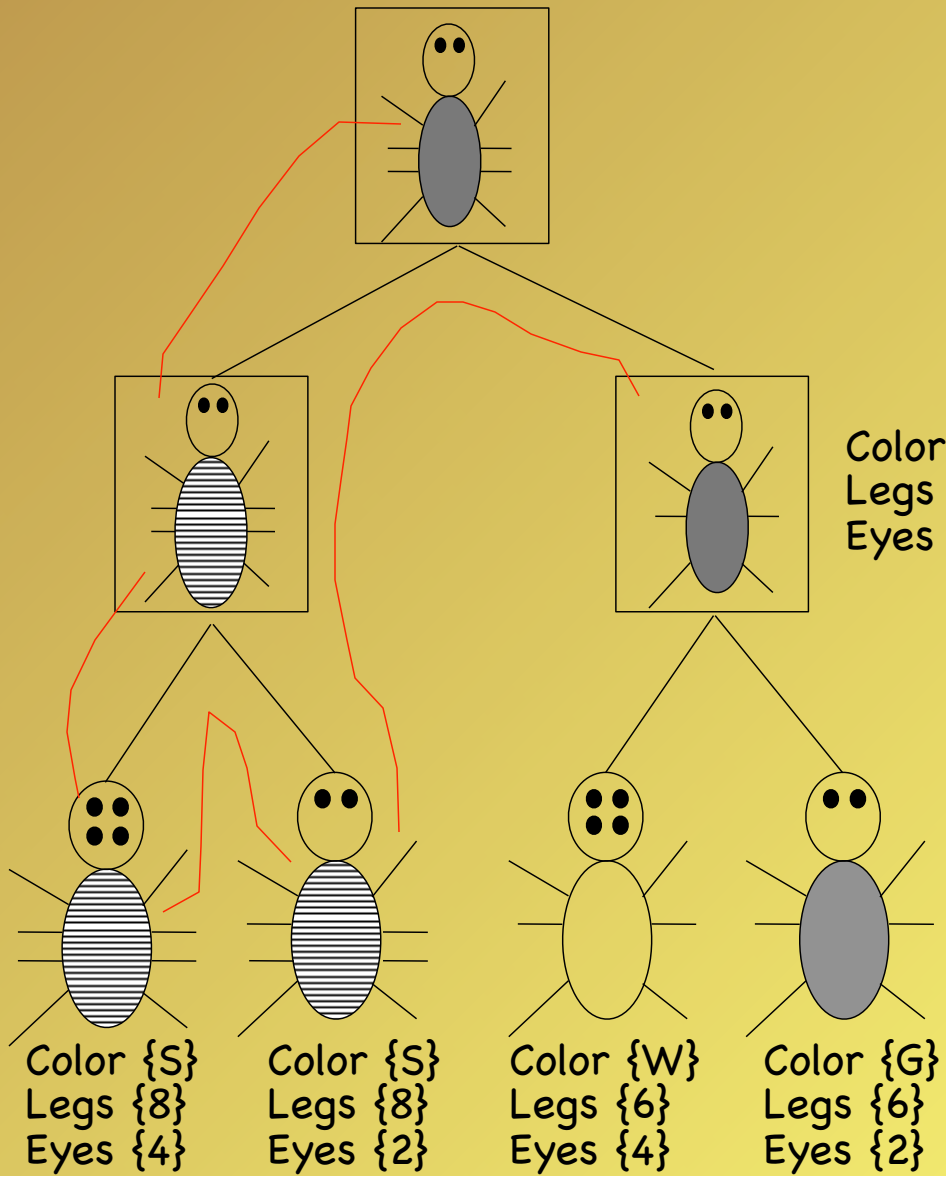
Step 2: Do a **preorder** traversal of the tree and decide on features.

Fitch's Algorithm



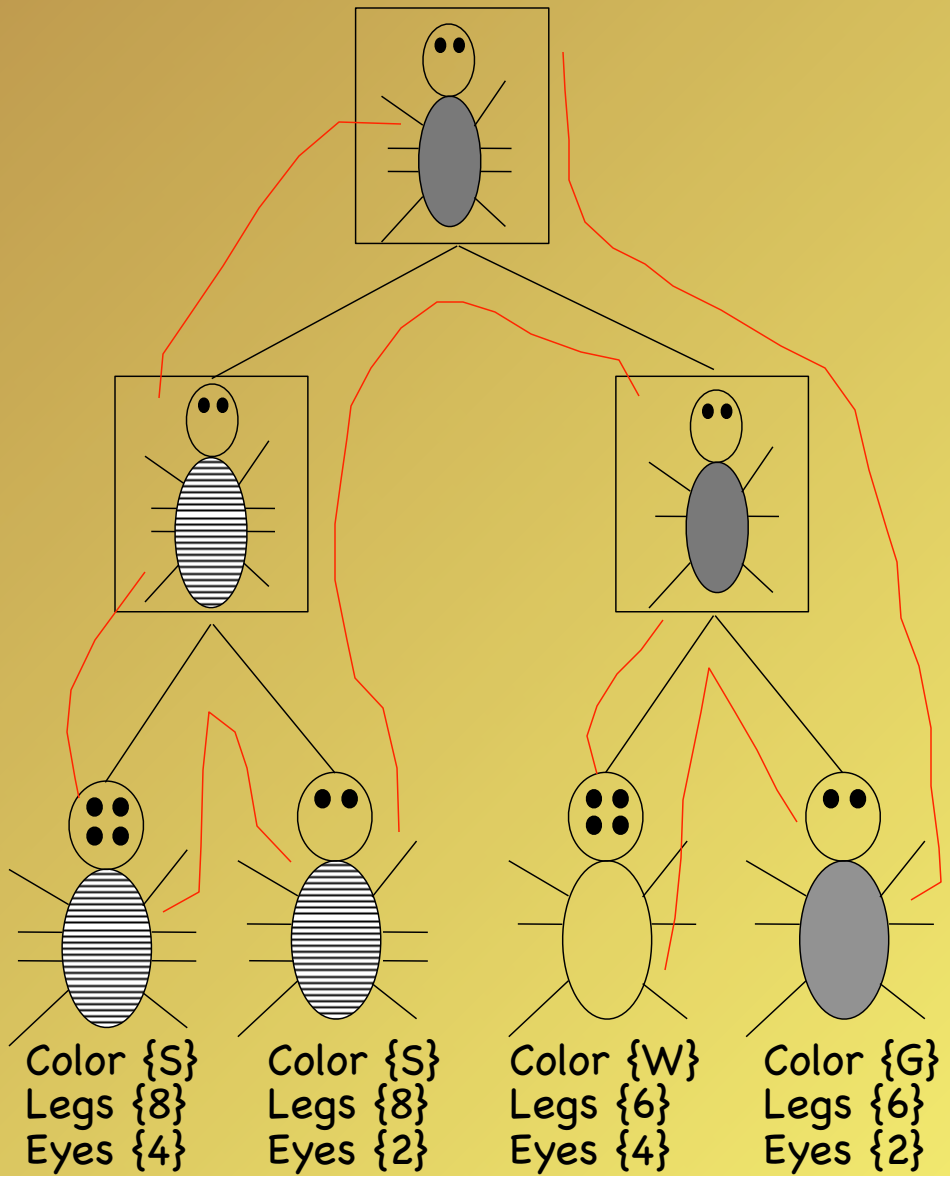
Step 2: Do a **preorder** traversal of the tree and decide on features.

Fitch's Algorithm



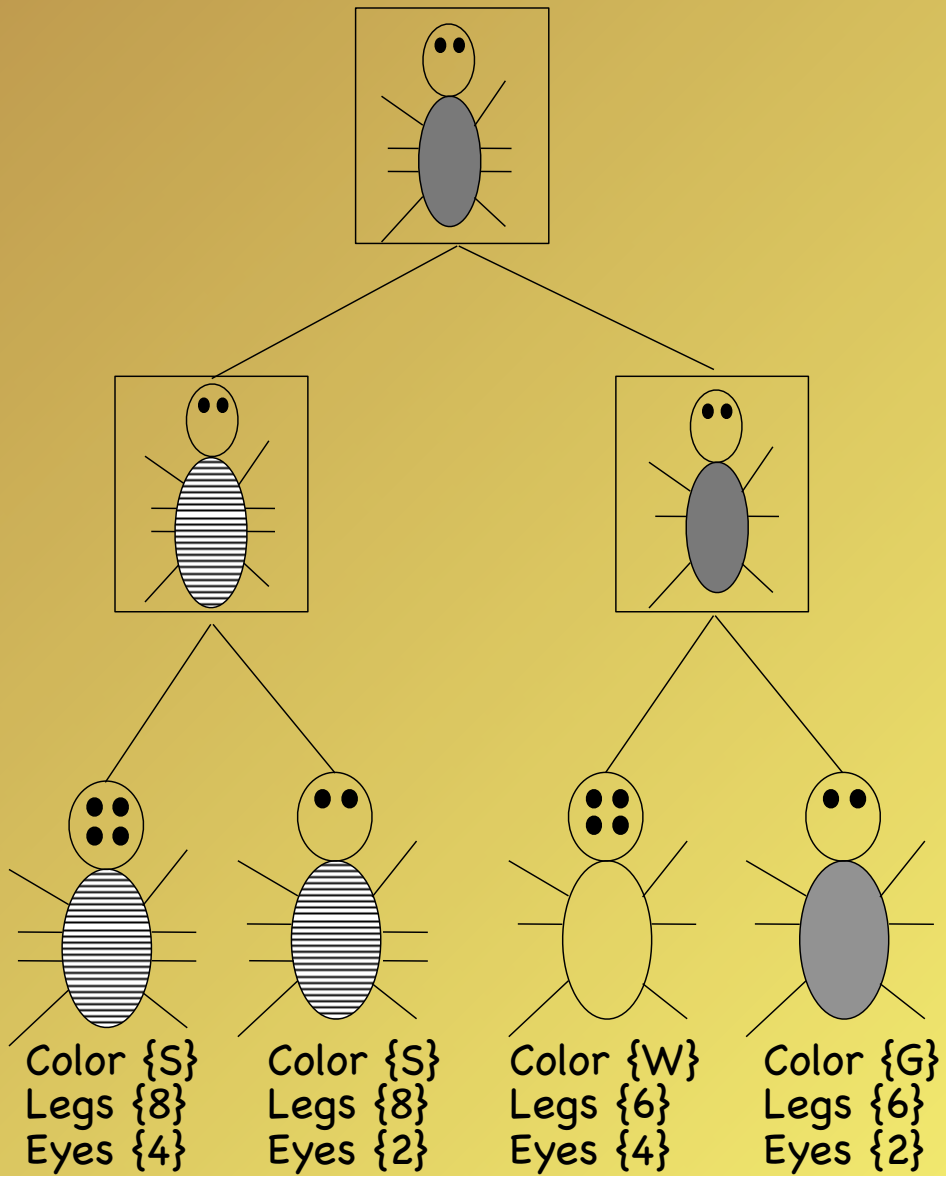
Step 2: Do a **preorder** traversal of the tree and decide on features.

Fitch's Algorithm

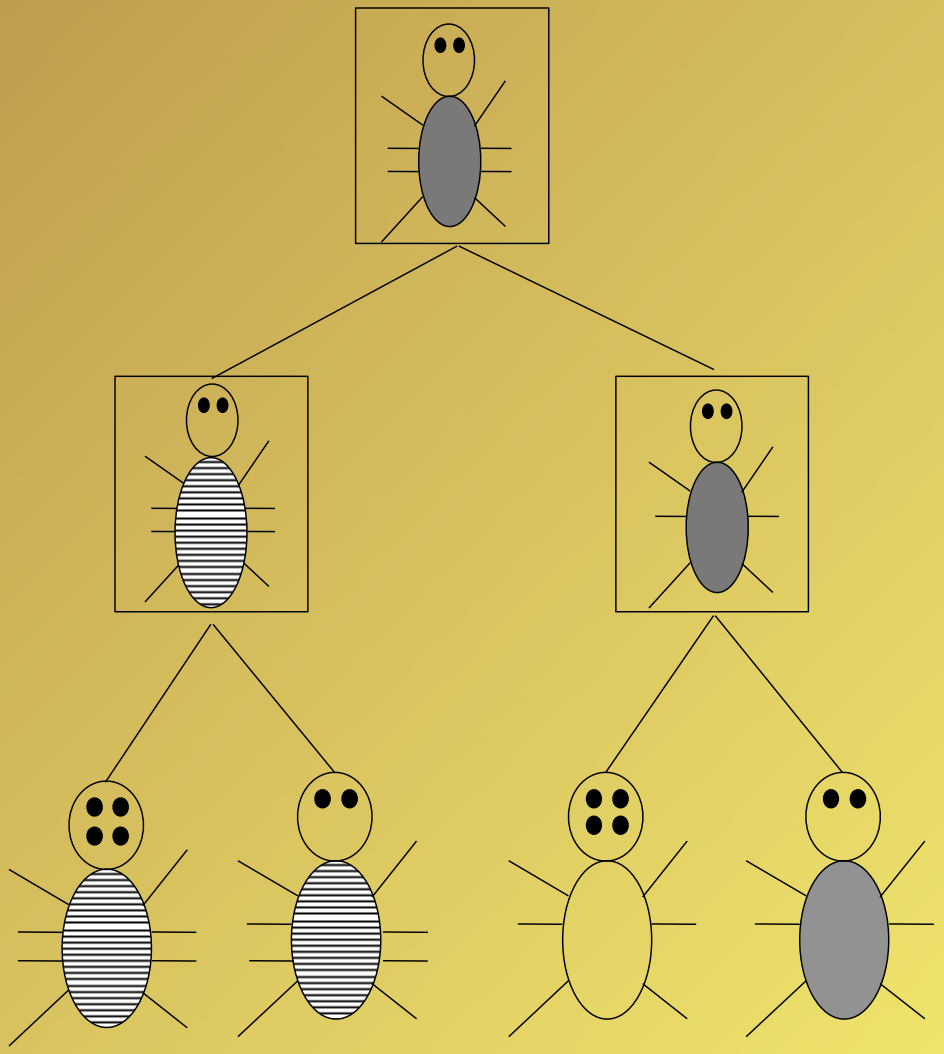


Step 2: Do a **preorder** traversal of the tree and decide on features.

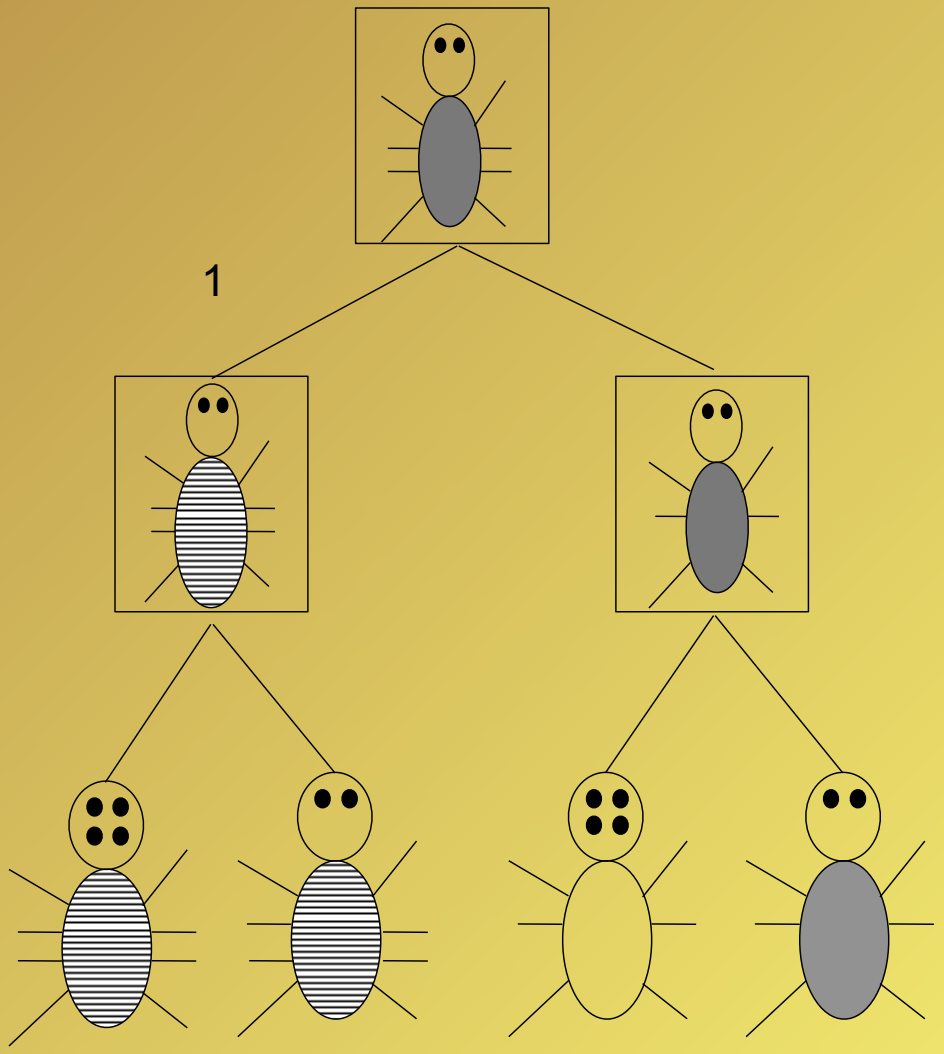
Fitch's Algorithm



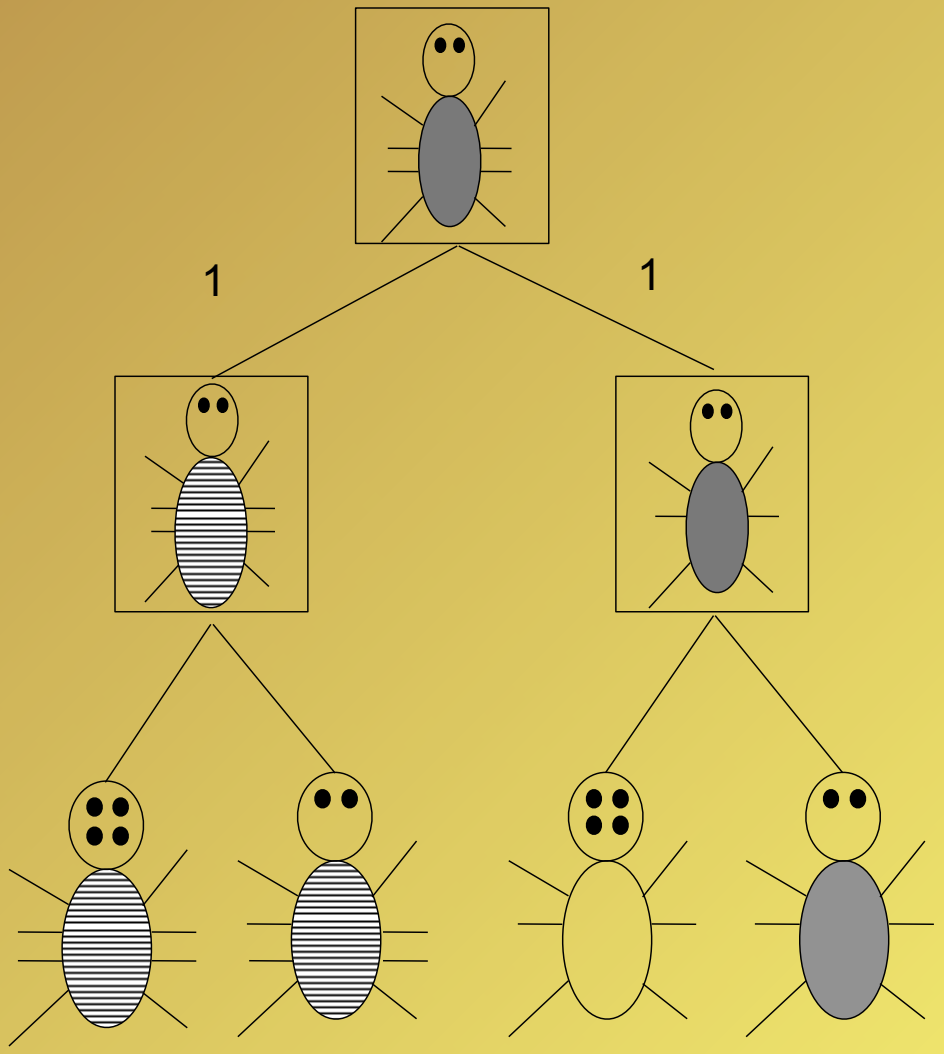
What is the score of this tree?



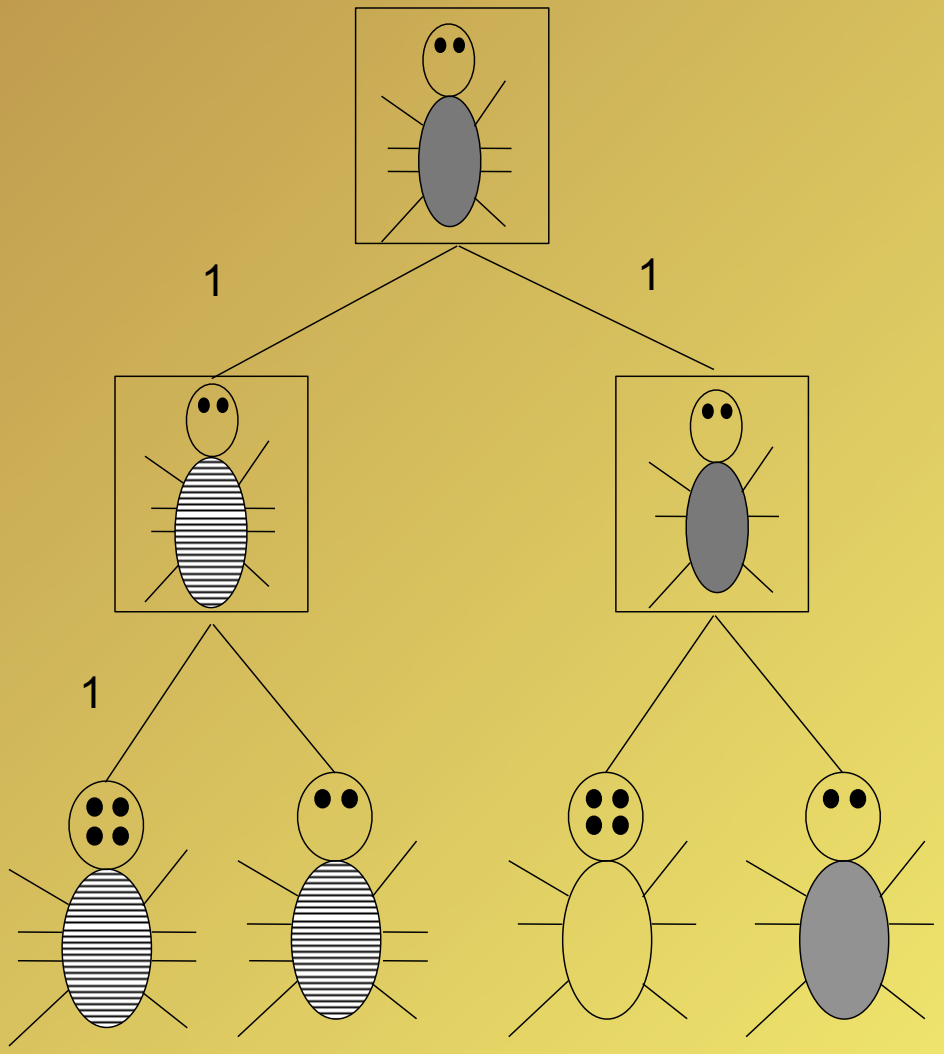
What is the score?



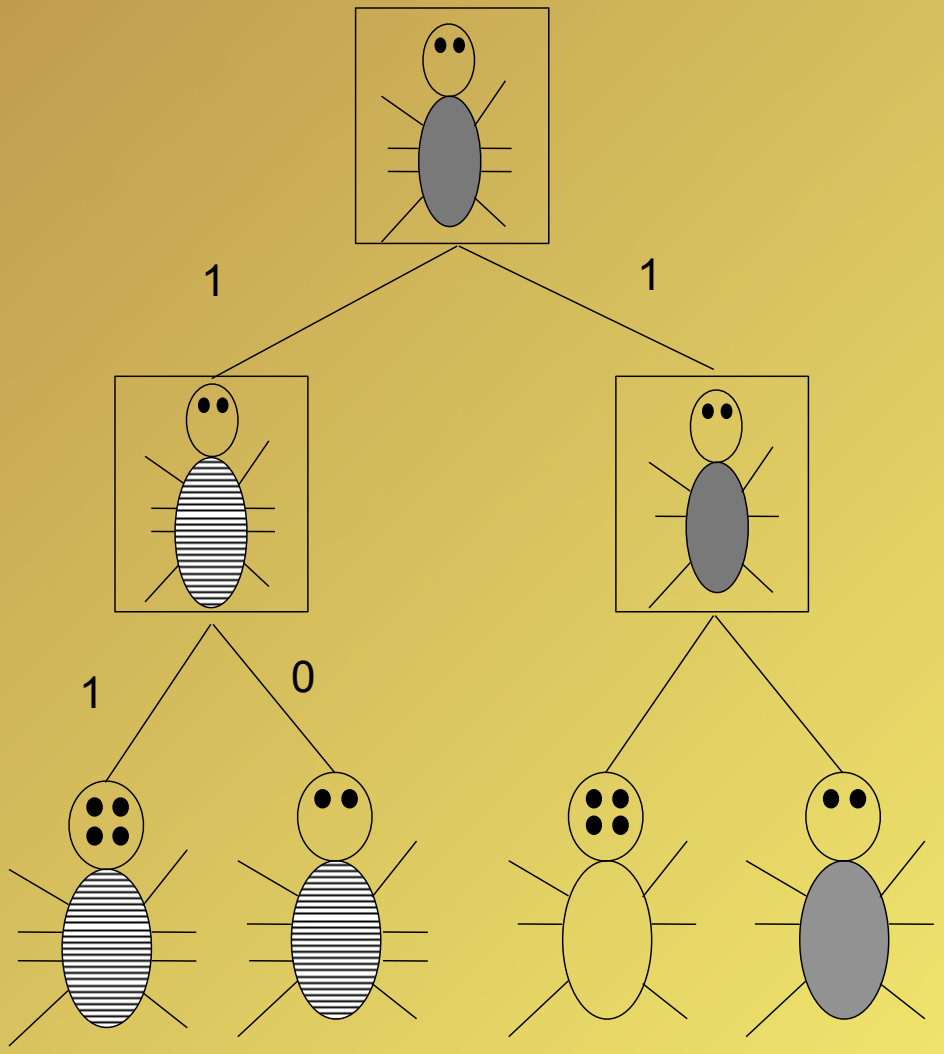
What is the score of this tree?



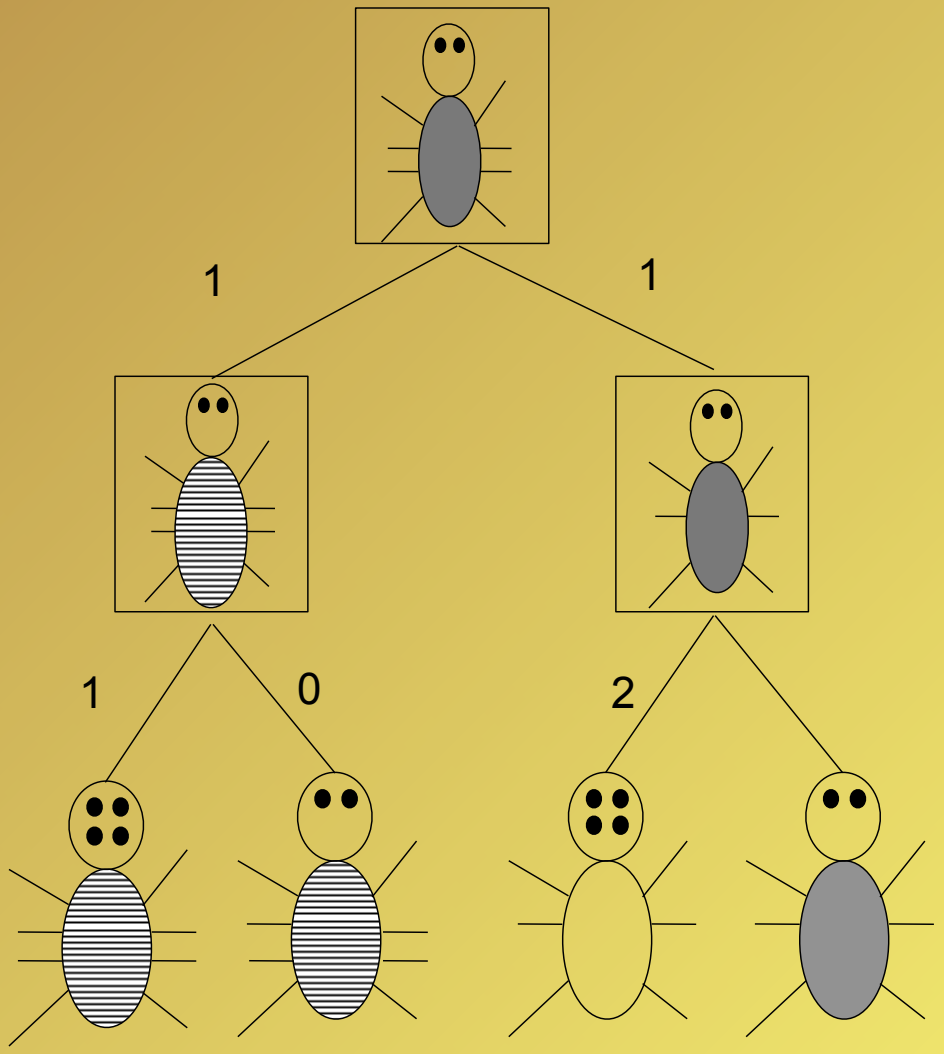
What is the score of this tree?



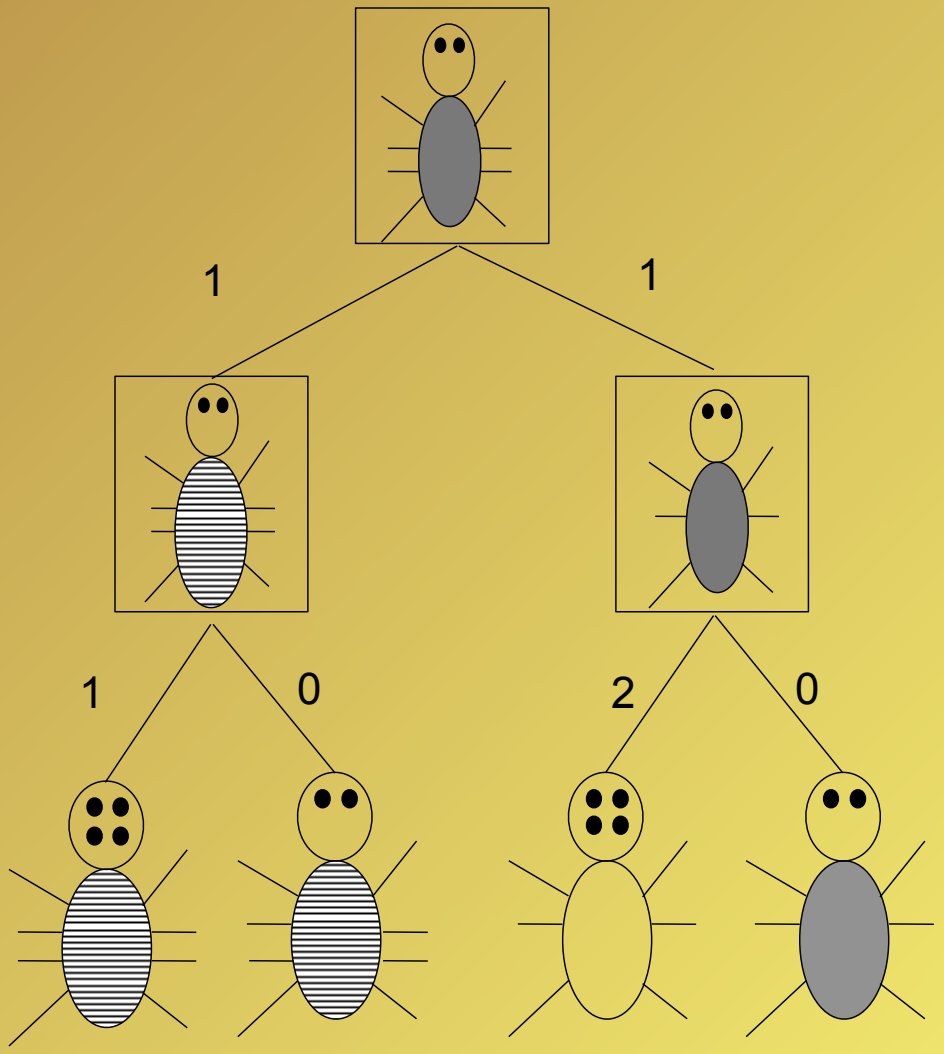
What is the score of this tree?



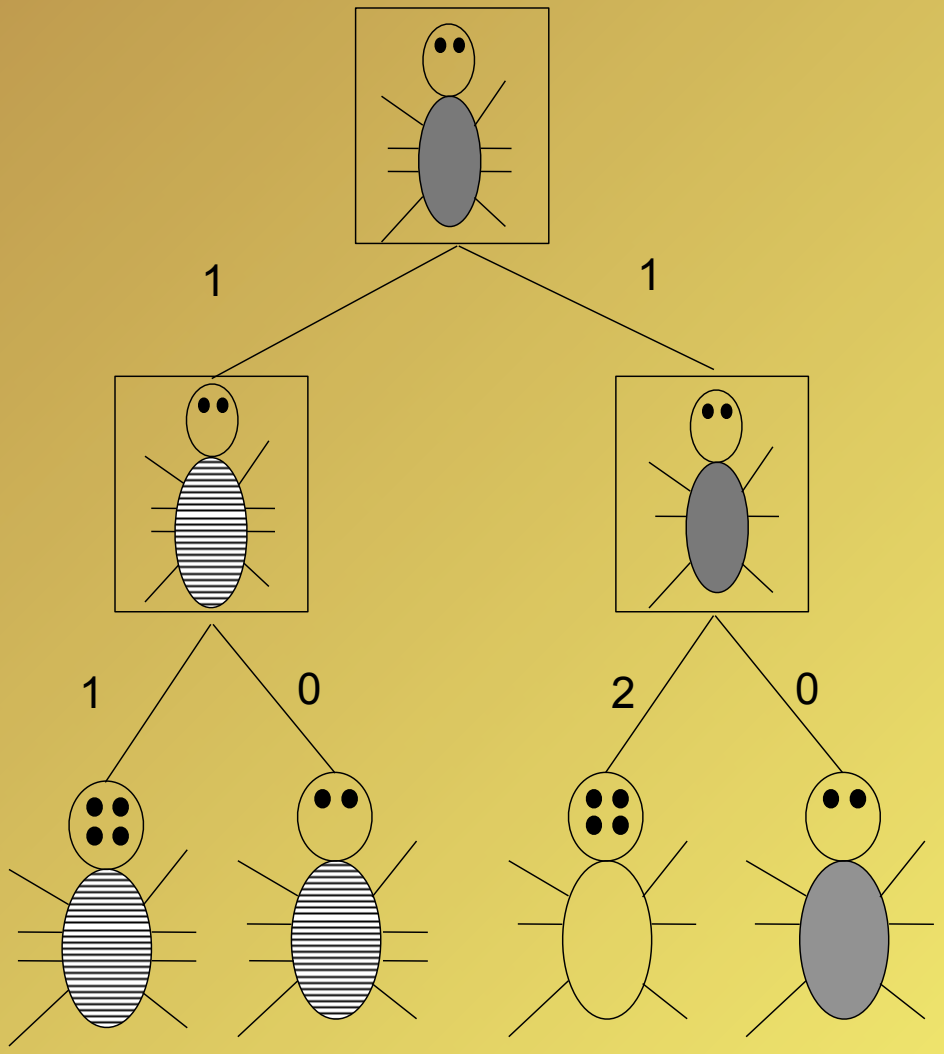
What is the score of this tree?



What is the score of this tree?



What is the score of this tree?

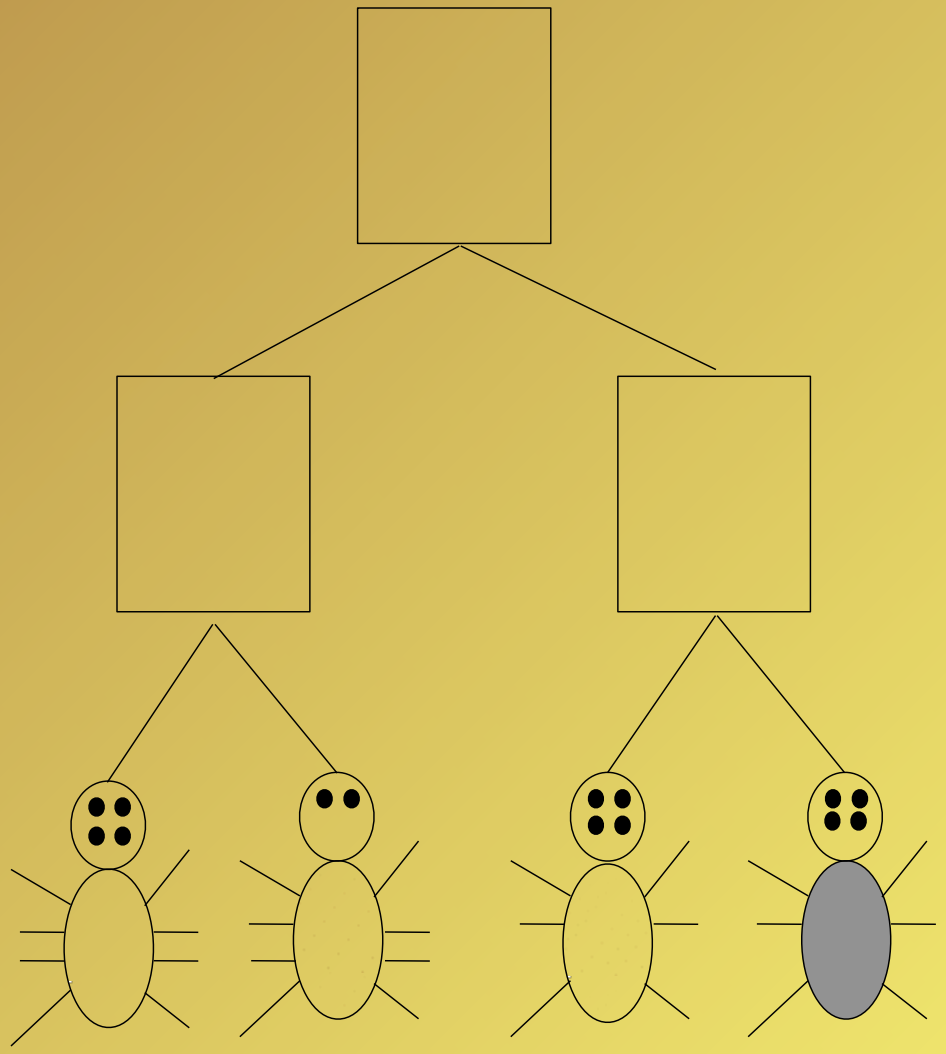


Score: 5

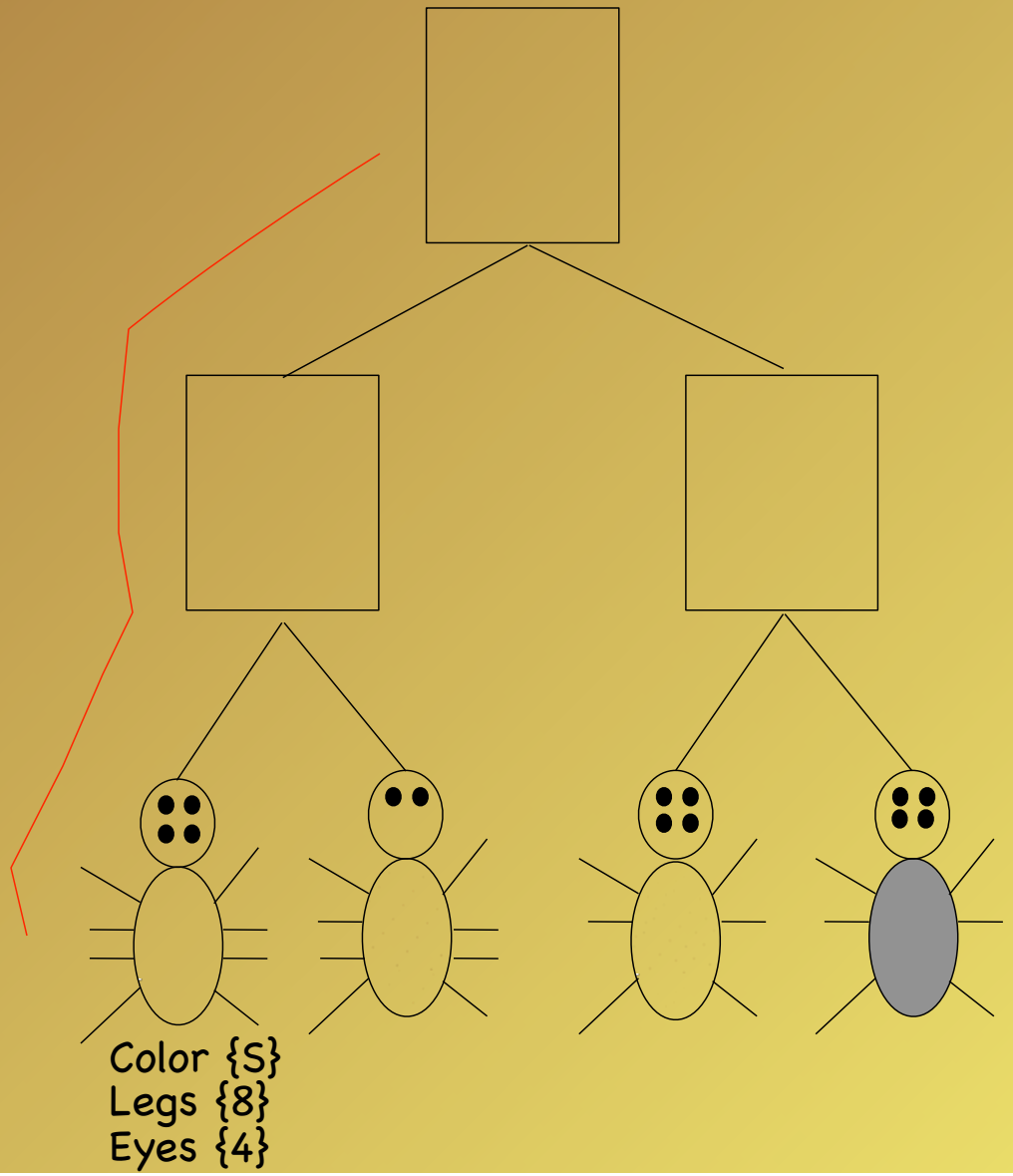
Fitch's Algorithm

- Traverse tree in postorder
  - Leafs:
    - label with the organism's features
  - Ancestors:
    - if the features of children intersect, label with intersection
    - else label with union of children's features
- Traverse tree in preorder
  - root:
    - pick any labelled feature arbitrarily
  - others:
    - pick label that matches parent feature
    - if none, pick any arbitrarily

Your turn! Use Fitch's Algorithm to determine the ancestors

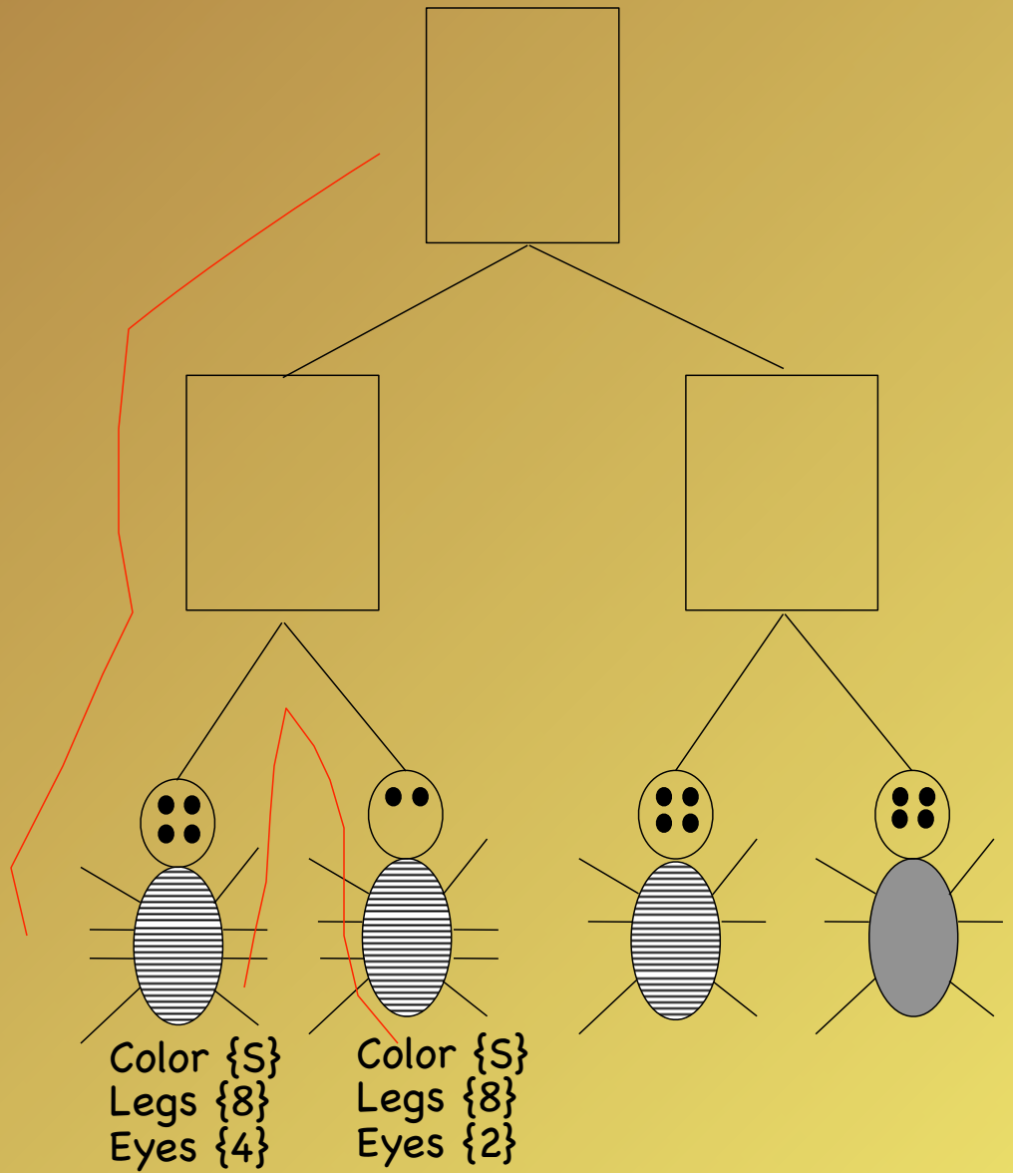


Step 1: Traverse tree in postorder



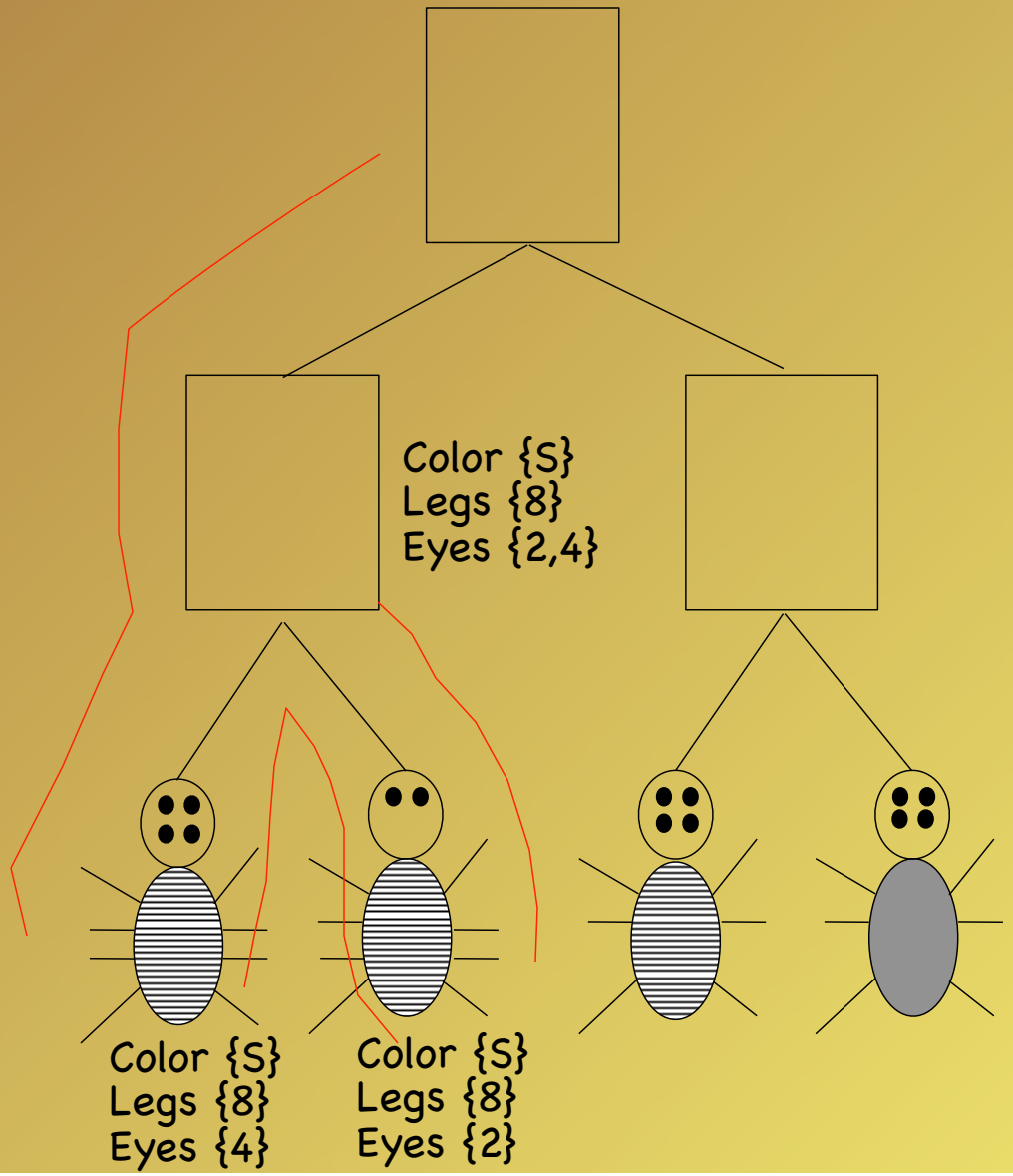
- Leafs: label with the organism's features
- Ancestors:
  - if the features of children intersect, label with intersection
  - else label with union of children's features

Step 1: Traverse tree in postorder



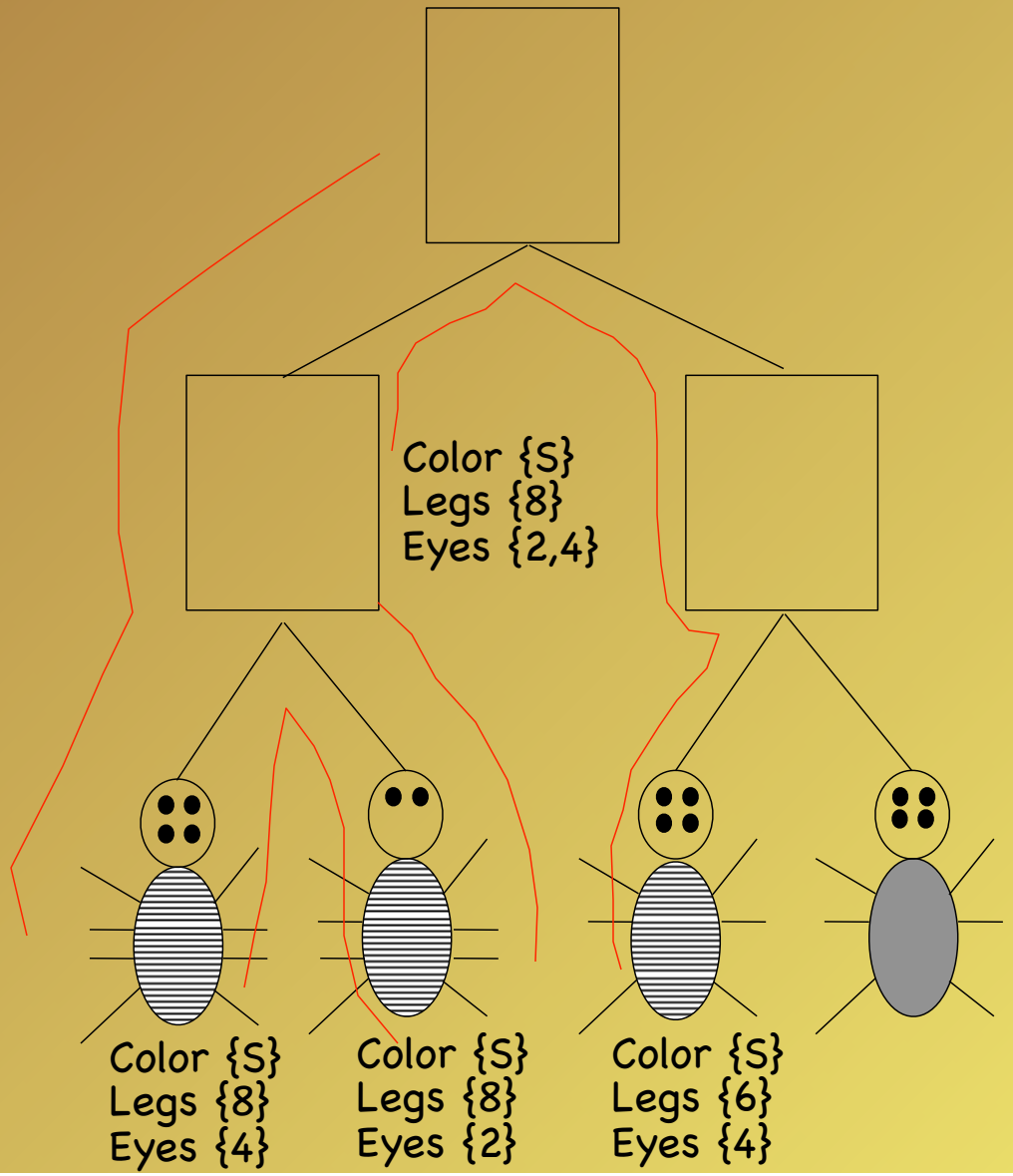
- Leafs: label with the organism's features
- Ancestors:
  - if the features of children intersect, label with intersection
  - else label with union of children's features

Step 1: Traverse tree in postorder



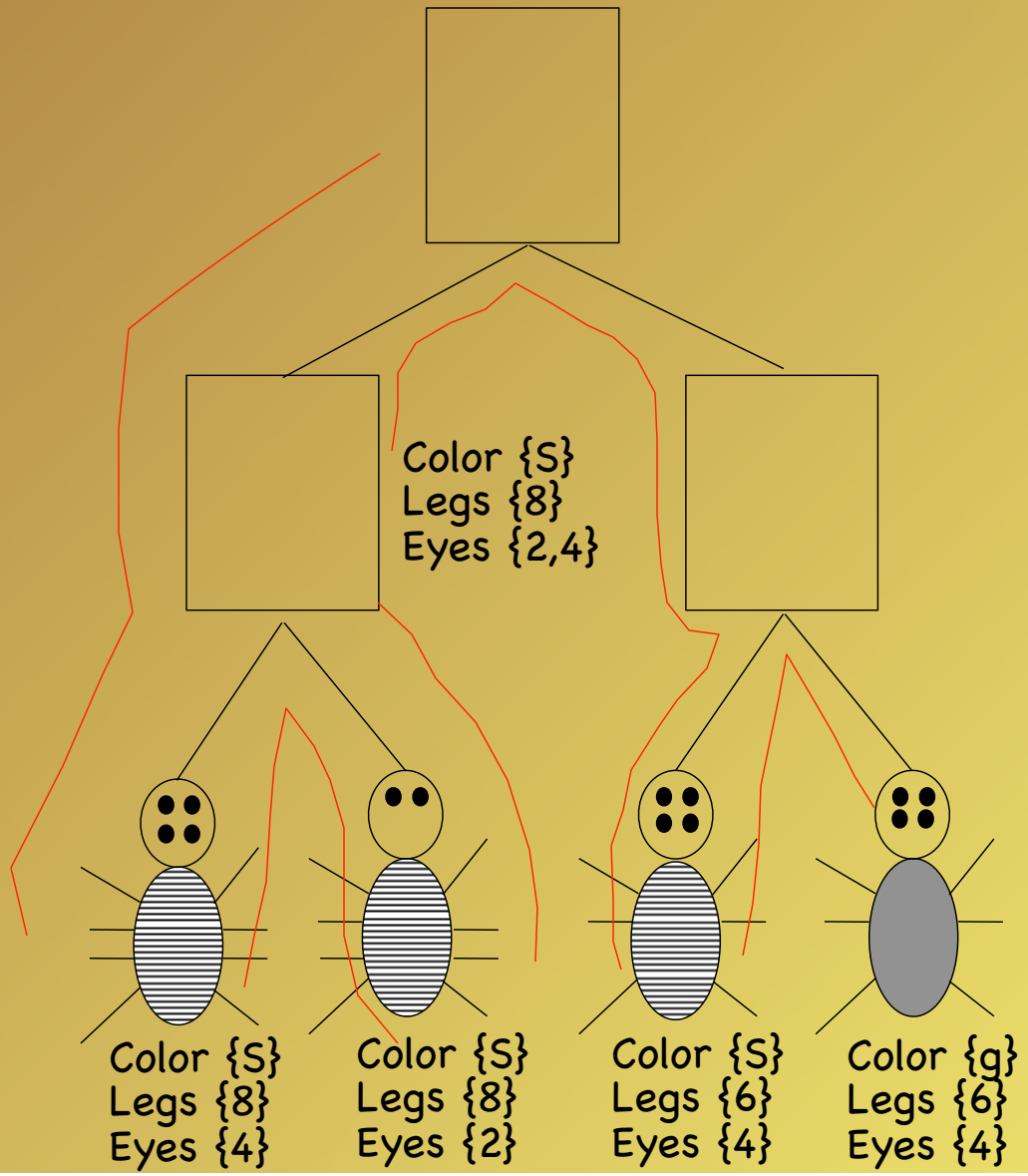
- Leafs: label with the organism's features
- Ancestors:
  - if the features of children intersect, label with intersection
  - else label with union of children's features

Step 1: Traverse tree in postorder



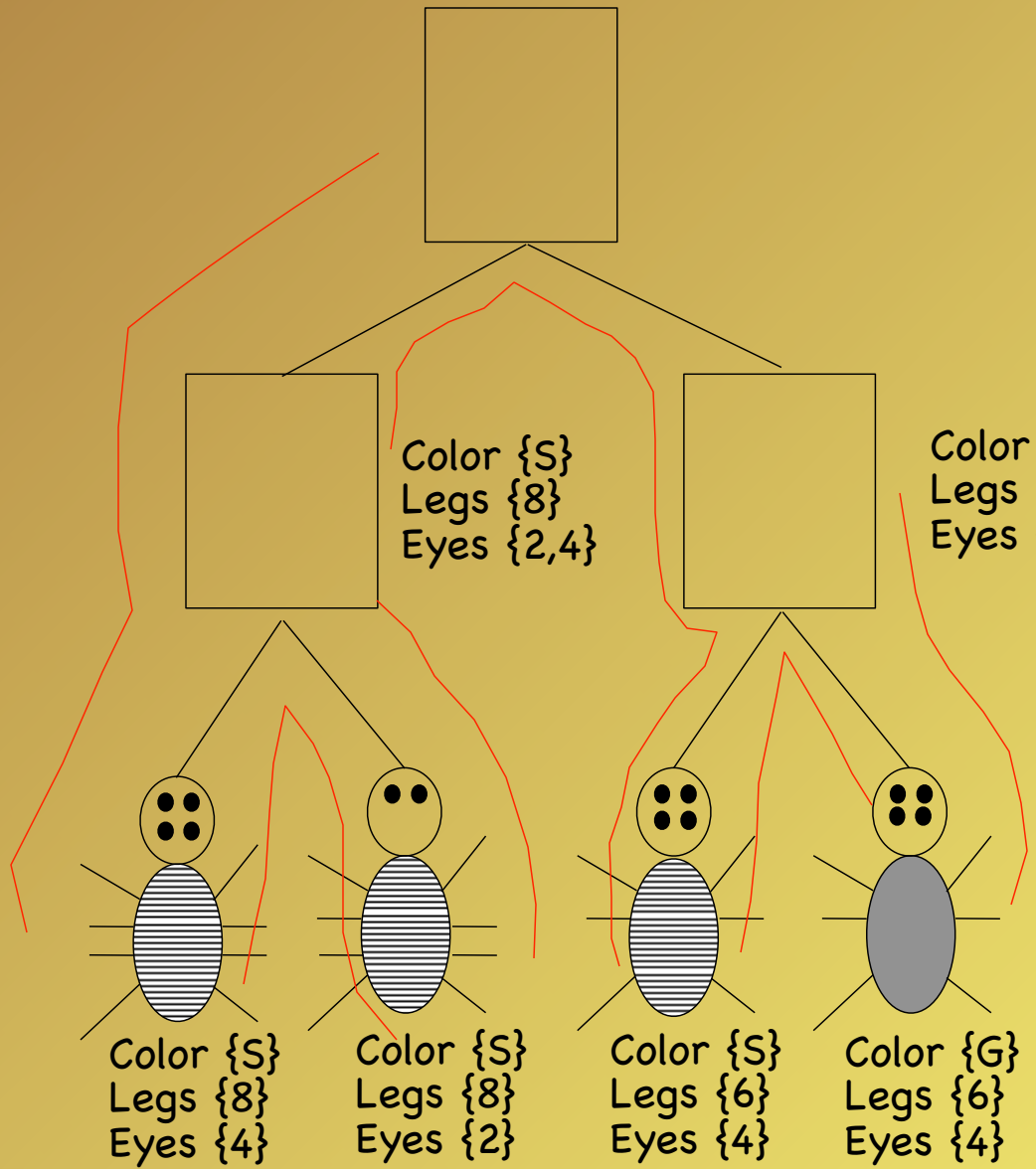
- Leafs: label with the organism's features
- Ancestors:
  - if the features of children intersect, label with intersection
  - else label with union of children's features

Step 1: Traverse tree in postorder



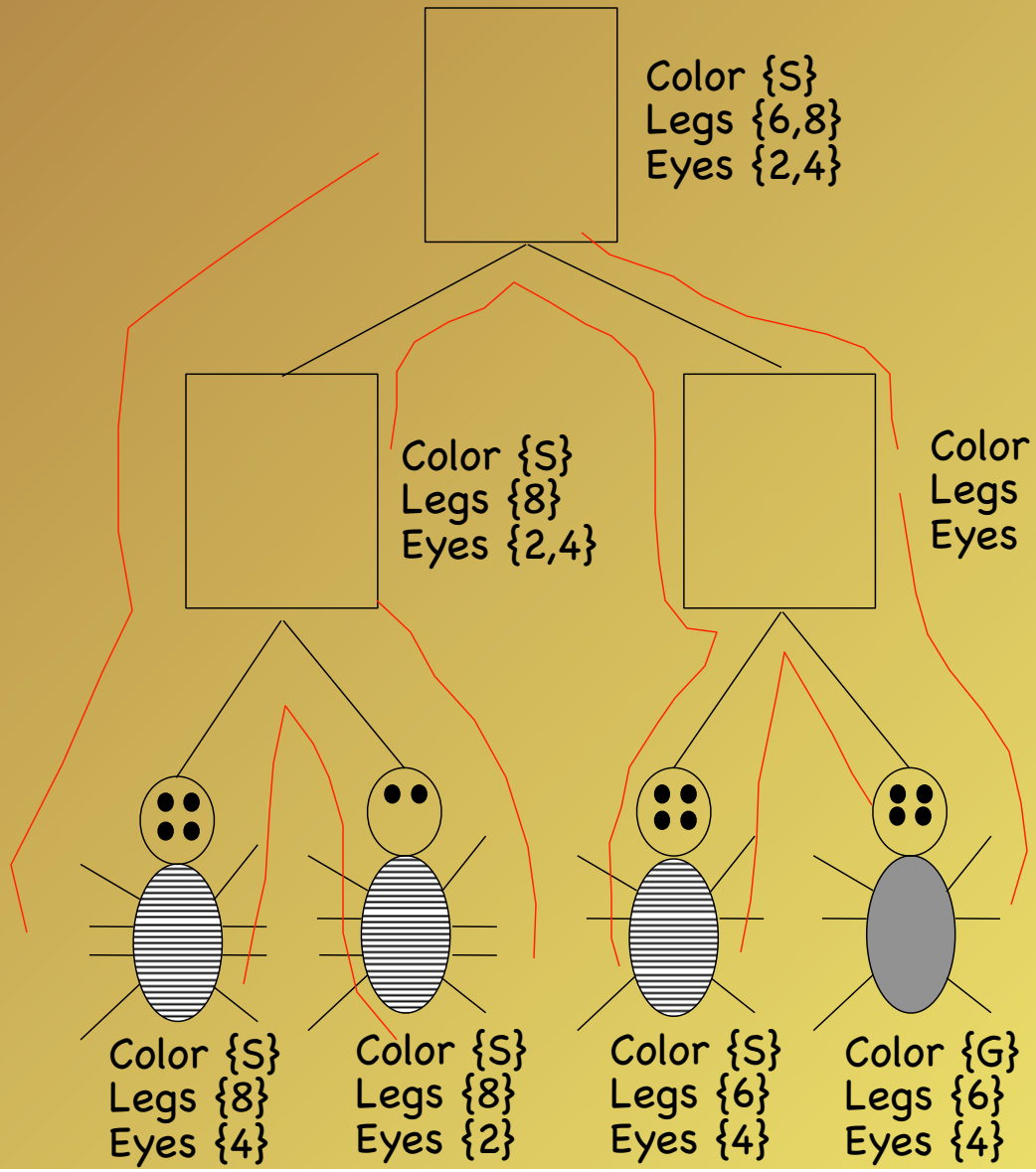
- Leafs: label with the organism's features
- Ancestors:
  - if the features of children intersect, label with intersection
  - else label with union of children's features

Step 1: Traverse tree in postorder



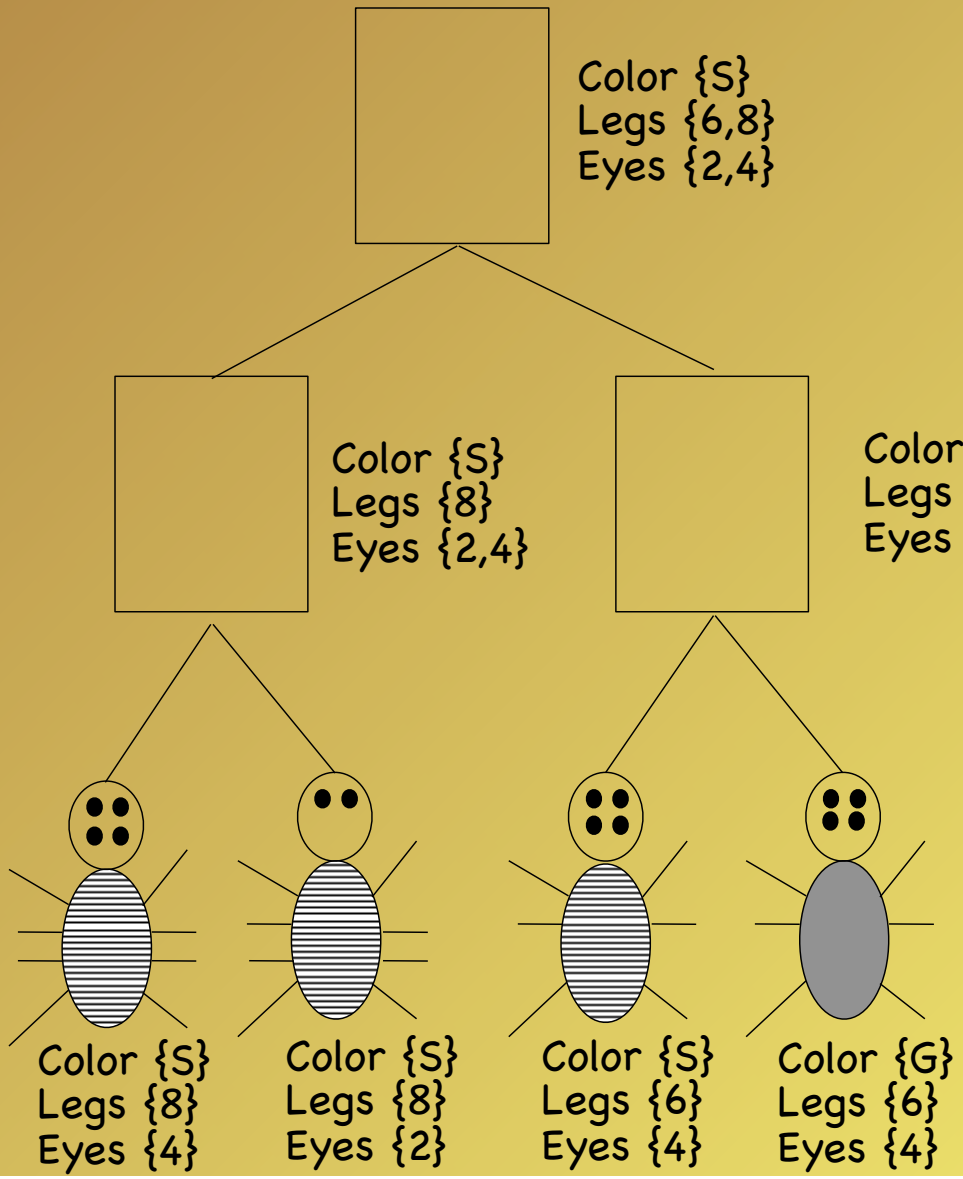
- Leafs: label with the organism's features
- Ancestors:
  - if the features of children intersect, label with intersection
  - else label with union of children's features

Step 1: Traverse tree in postorder



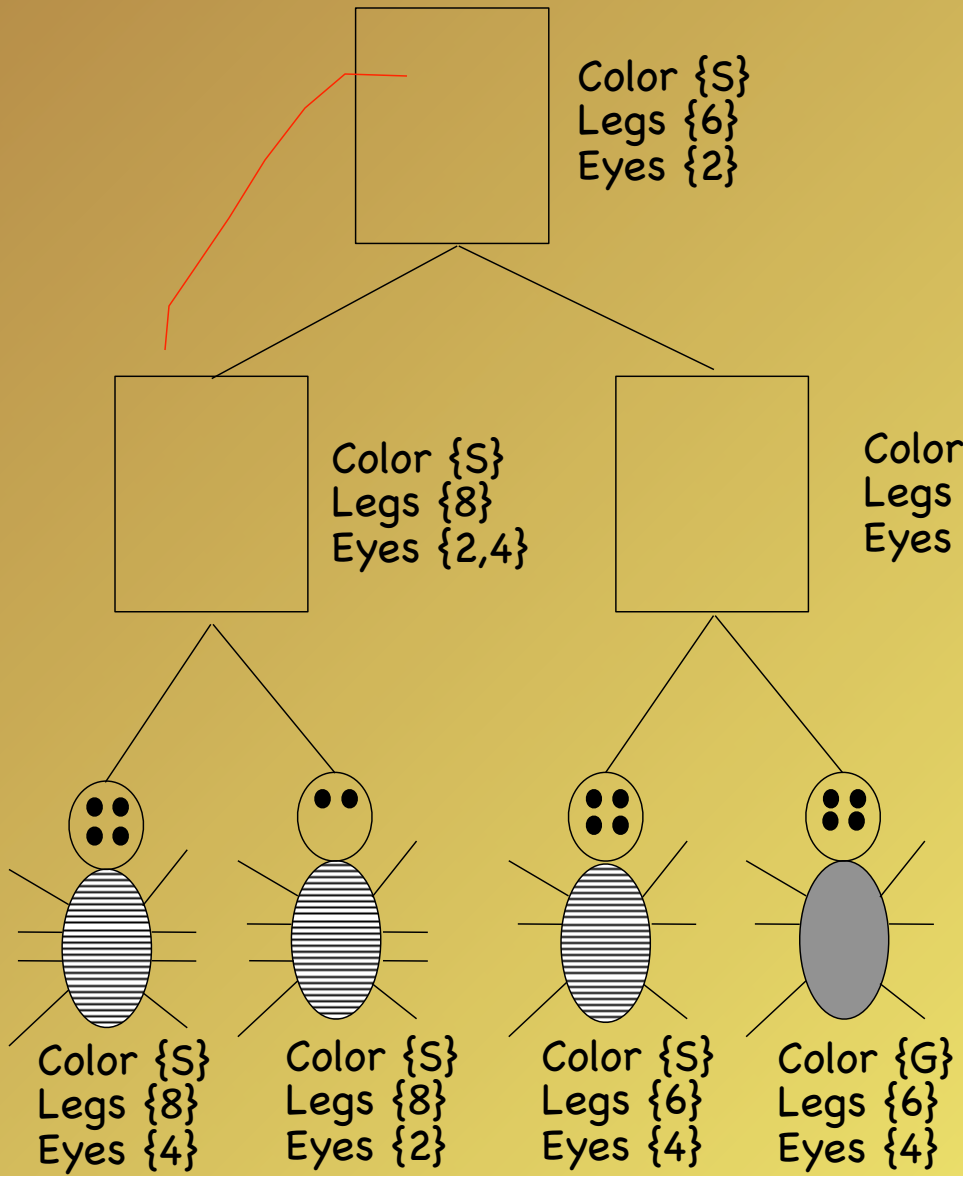
- Leafs: label with the organism's features
- Ancestors:
  - if the features of children intersect, label with intersection
  - else label with union of children's features

Step 2: Traverse tree in preorder



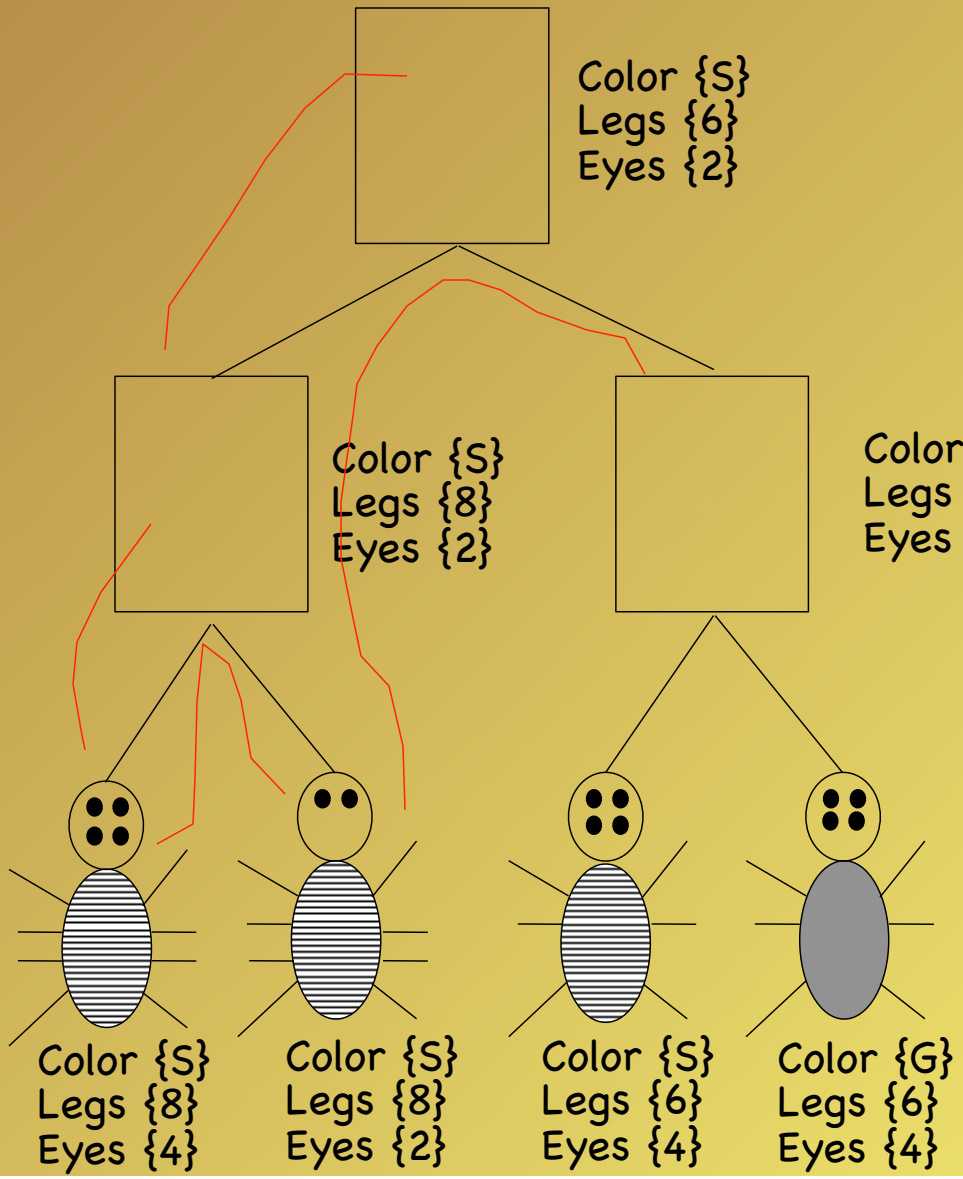
- root: pick any labelled feature arbitrarily
- others:
  - pick label that matches parent feature
  - if none, pick any arbitrarily else label with union of children's features

Step 2: Traverse tree in preorder



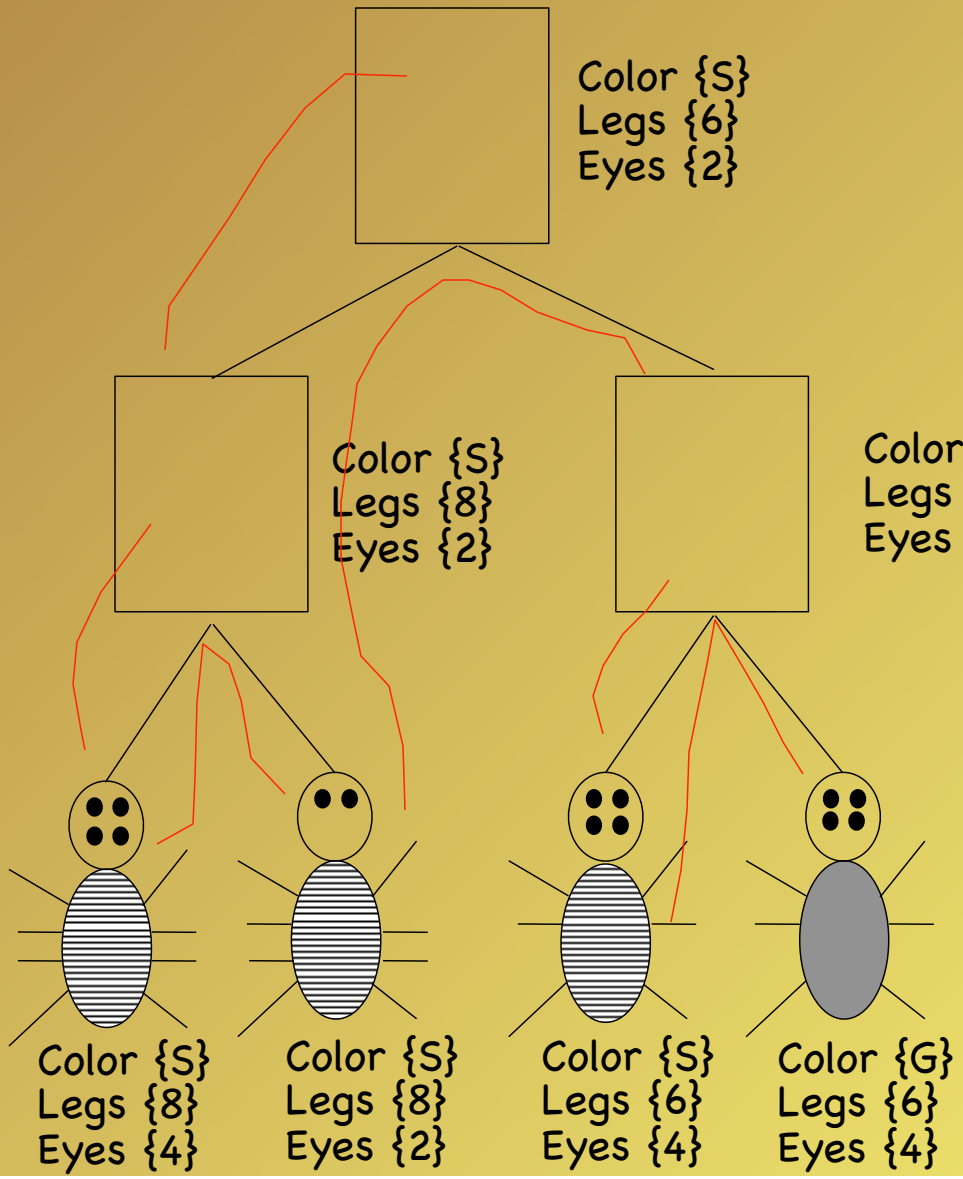
- root: pick any labelled feature arbitrarily
- others:
  - pick label that matches parent feature
  - if none, pick any arbitrarily else label with union of children's features

Step 2: Traverse tree in preorder



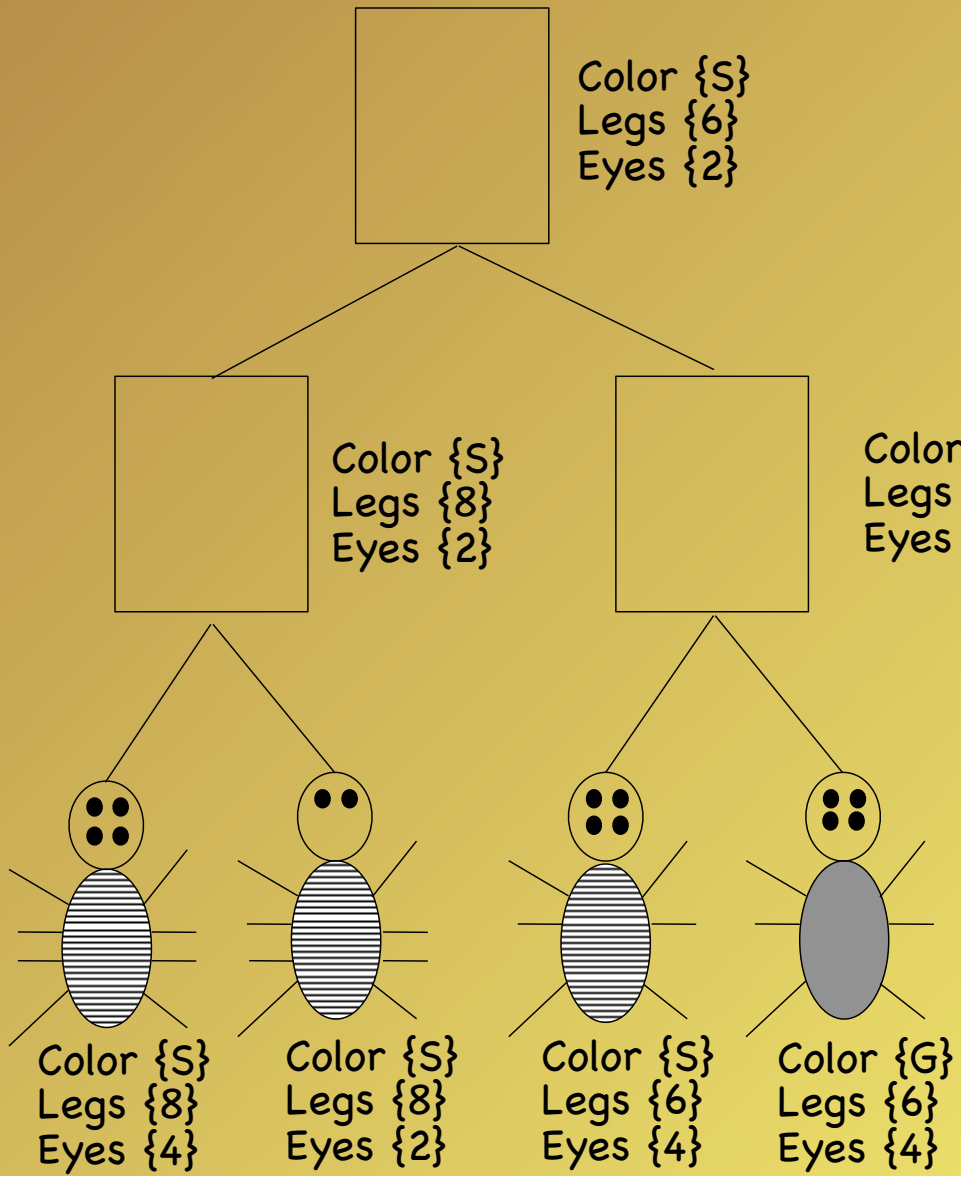
- root: pick any labelled feature arbitrarily
- others:
  - pick label that matches parent feature
  - if none, pick any arbitrarily else label with union of children's features

Step 2: Traverse tree in preorder



- root: pick any labelled feature arbitrarily
- others:
  - pick label that matches parent feature
  - if none, pick any arbitrarily else label with union of children's features

What is the score?



Parsimony problems

- Small Parsimony problem (as above)
  - tree topology is known
  - Goal: How many evolutionary changes did occur?
  - solvable in polynomial time (e.g., Fitch's algorithm)

- Large Parsimony problem
  - tree topology is unknown
  - Goal: Find Most Parsimonious Tree
  - NP-complete, that is it's not solvable in polynomial time unless P=NP

We'll talk about P, NP, and NP-completeness (and how you can win $1,000,000) on Monday

Thank you!

My email: stege@cs.uvic.ca
My office: ECS 624