CS 331: Bayesian Networks 3

Designing/Learning Bayesian Networks

Bayesian networks work great!

- Represent your knowledge in the network
- Make observations of some variables, call them *E*
- Draw inferences about other variables *P(Q | E)*

But… there are a couple of nagging questions:

1. How do you come up with a Bayesian network structure in the first place?
2. How do you learn the structure and parameters automatically from data

Bayesian Network Topology

- So how do you come up with the Bayesian network structure?
- Two options:
  1. Design by hand
  2. Learn it from data

Designing Bayesian Networks By Hand

Getting an Expert to Design the Network by Hand

- Could get a domain expert to help design the Bayesian network
- Need the domain expert to come up with:
  1. Network Topology
  2. Parameters (i.e. probabilities) in the conditional probability tables

Designing the Network Topology

- Key point: Bayesian network exploits conditional independence to produce a compact representation of the full joint distribution
- Compactness is due to the fact that a Bayesian network is a locally structured system

Locally Structured Systems

- In a Bayes net, each node is directly influenced by a small number of other nodes (say k)
- This means that the CPT of each node has 2^k probabilities
- If there are n nodes overall, we need $n2^k$ probabilities
- Suppose $k = 5, n = 30$, then $n2^k$=960 probabilities but the full joint requires $> 10^9$ probabilities

What If The Network is Densely Connected?

Then your representation can't take advantage of conditional independence for compactness

- Possible but unlikely
- Could drop a few links (sacrifice accuracy for compactness)

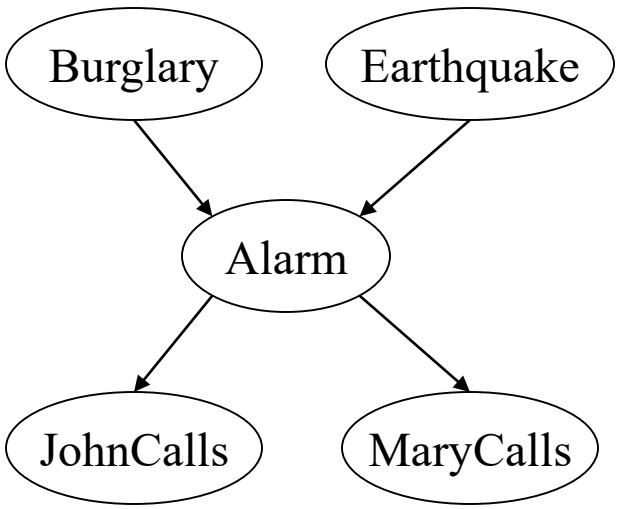
Constructing a Locally Structured Bayesian Network

- Needs:
  1. Each variable to be directly influenced by a few others
  2. Parents are the direct influences of a node
- Process:
  - Add "root causes" first (those that are not influenced by any others)
  - For each new variable, add the smallest subset of all previous variables as parents such that it is independent of the rest given the parents
  - Keep going until you reach the "leaves" which do not have a direct causal influence on the other variables

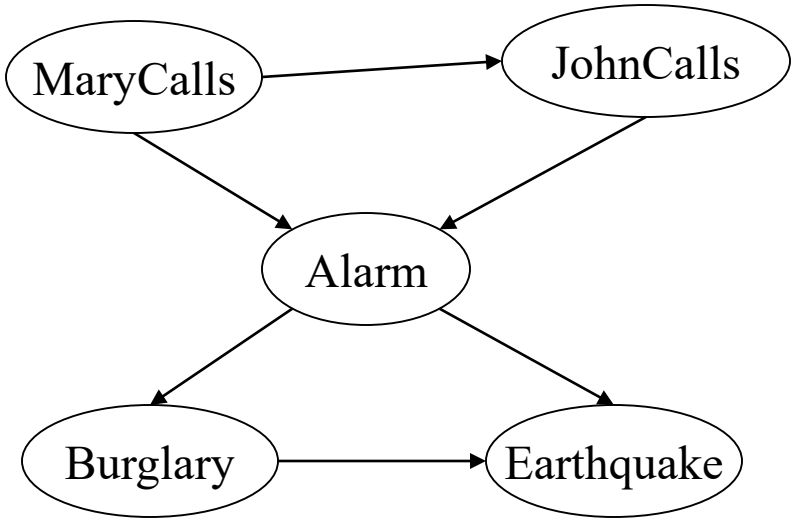
Choosing the Wrong Order

What happens if you add nodes in the wrong order?

Compact network

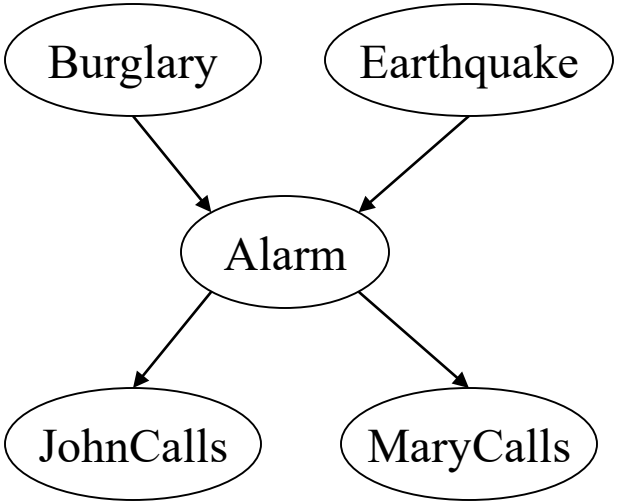


Not-So-Compact Network

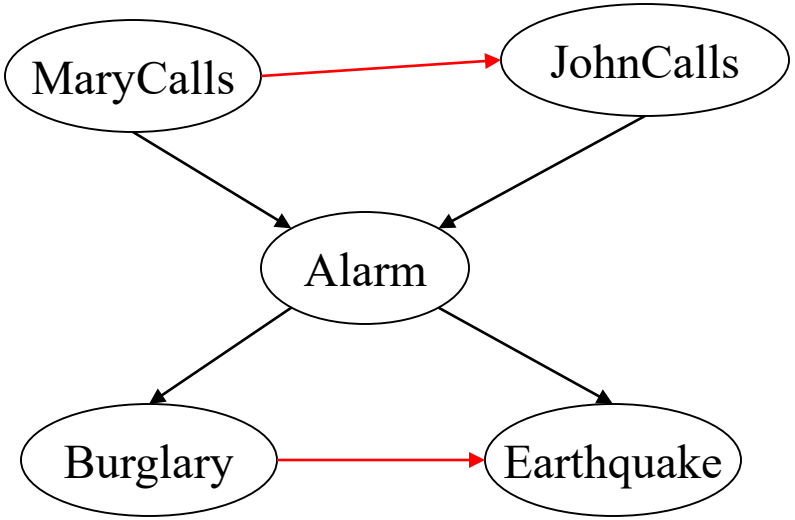


Choosing the Wrong Order

Compact network



Not-So-Compact Network

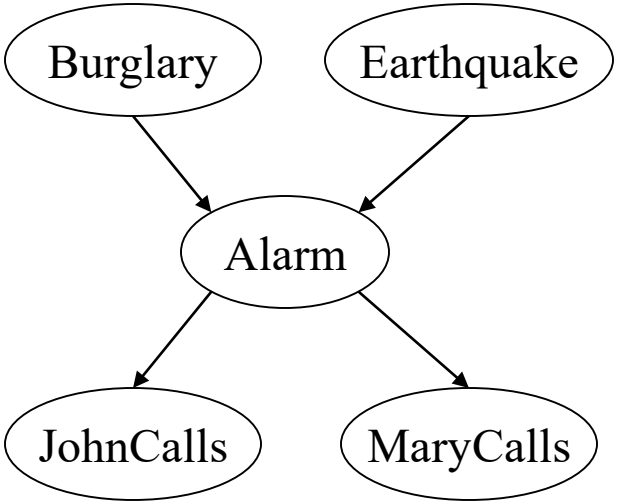


Two more links

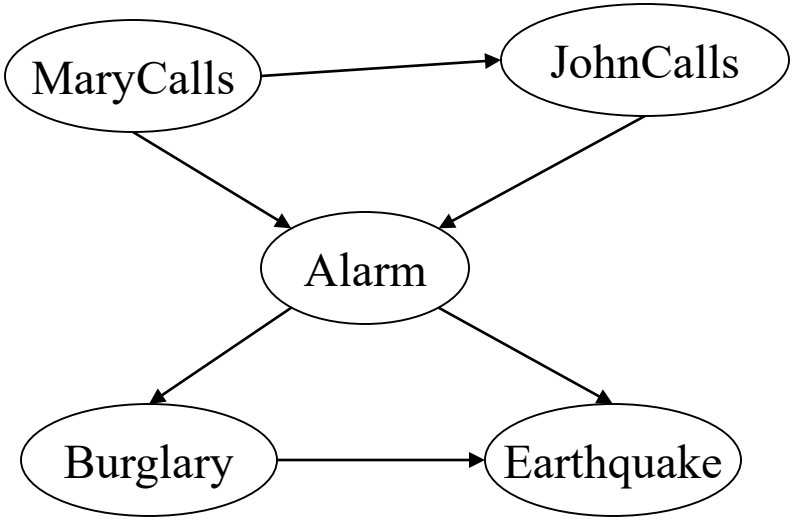
Some links result in conditional probability tables that require unnatural/difficult probability judgments eg. P(Earthquake | Burglary, Alarm)

Choosing the Wrong Order

Compact network



Not-So-Compact Network



Note: Both networks can represent the same joint probability distribution. The problem is that the one on the right doesn't represent all the conditional independence relationships and some links need not be there

Diagnostic versus Causal models

- Build causal models i.e. a link from Node X to Node Y indicates X causes Y
- Don't build diagnostic models i.e. Links go from symptoms to causes
- Diagnostic models result in additional dependencies between otherwise independent causes
- Causal models result in fewer parameters and easier parameters to come up with

Designing the Parameters in the Bayesian Network

- As was mentioned previously, make sure the probabilities in the CPT are natural and easy for an expert to come up with
- E.g. P(Earthquake | Burglary, Alarm) is not natural but P(Alarm | Burglary, Earthquake) is
- In general, coming up with these probabilities can be tricky
- E.g. A physician can't tell you exactly what P(Headache | Flu) is.

Designing the Parameters of the Bayesian Network

- Possible solutions:
  - Specify a range of values for that probability
  - Specify a distribution for the probability with a known form
  - Could get expert to encode relative relationships e.g. “This value is twice as likely as the other one”
  - Get probabilities from studies or census

Learning Bayesian Network Structure From Data

Learning Structure From Data

- You can think of the structure and parameters of the Bayesian network as representing causal knowledge about the domain
- If you don't have an expert, you can learn both the structure and parameters from data

Learning Structure From Data

- There are other good reasons for learning the structure/parameters from data
- The actual causal model may be unavailable or unknown
- The actual causal model may be subject to dispute (maybe because of a subjective bias by the domain expert)

Learning the Structure from Data

Two cases:

1. Complete data
2. Incomplete data

We will describe what these mean!

Complete Data

- Your domain is fully observable (i.e. you can observe the values of all the random variables in the data)
- Your data has no missing values

No missing values

Age	**Gender**	**Home Zip**
50-60	Male	97330
20-30	Female	97333
40-50	Female	97331

Has 3 missing values

Age	**Gender**	**Home Zip**
?	Male	97330
20-30	Female	?
?	Female	97331

Parameter Learning From Complete Data

- Let's first assume that the Bayesian network structure is fixed
- Learning the parameters from complete data is easy:

$$P(X = x|Pa(X) = y) = \frac{N(X = x, Pa(X) = y)}{N(Pa(X) = y)}$$

Number of examples where $X = x \; and \; Pa(X) = y$

Number of examples where $Pa(X) = y$

- We have already seen this approach in Naïve Bayes
- We won't deal with incomplete data in this class

Learning the Structure

- Involves a search over possible directed acyclic graph structures to find the best fitting one
- However, for *n* nodes, there are the following number of possible structures [Robinson, 1973]:

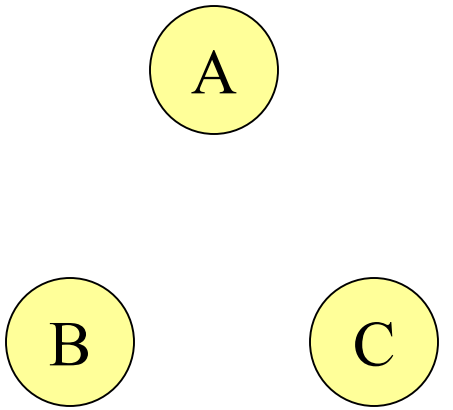
$$O(n!2^{\binom{n}{2}})$$

Learning the Structure

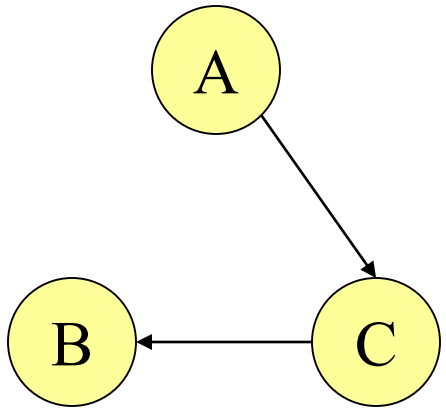
- This is clearly impossible to do an exhaustive search to find the optimal structure
- Need to resort to local search methods e.g. hill-climbing, simulated annealing
- We'll illustrate this using a 3 node example.

Local Search Methods

Initial State:



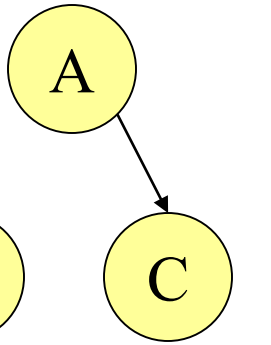
Start with no links



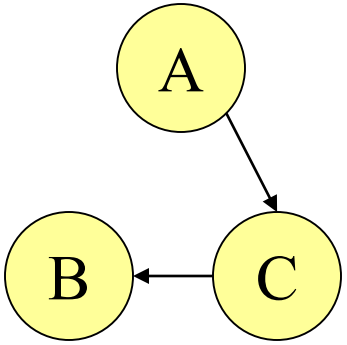
Start with a random set of links

Local Search Methods

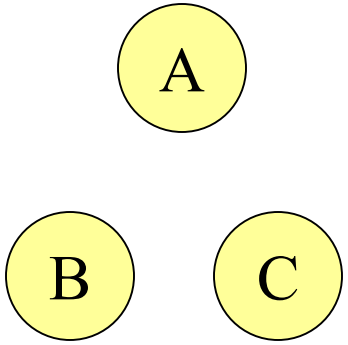
Neighborhood:



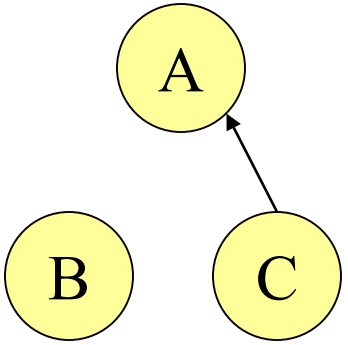
Current State



Add a link



Remove a link



Reverse a link

Things to Watch Out For

- Need to avoid introducing cycles
- Need to re-estimate parameters everytime you modify a link in the Bayes net
  - Do you need to re-estimate the parameters for all nodes?
  - No, just the ones that are affected by the modified link
- Lots of local optima problems. Use random restarts.

The Evaluation Function

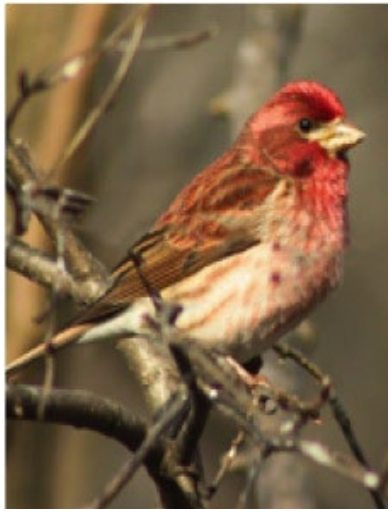
- How do we know if a Bayes net structure is good?
- Two types of evaluation functions:
  1. Evaluate if conditional independence relationships in the learned network match those in the data
  2. Evaluate how well the learned network explains the data (in the probabilistic sense).

What You Need To Know

- How to get an expert to design a Bayesian network by hand
- Briefly describe how you would use local search to learn the structure of a Bayesian network

Example: Citizen scientists may confuse two species of finch

Purple Finch

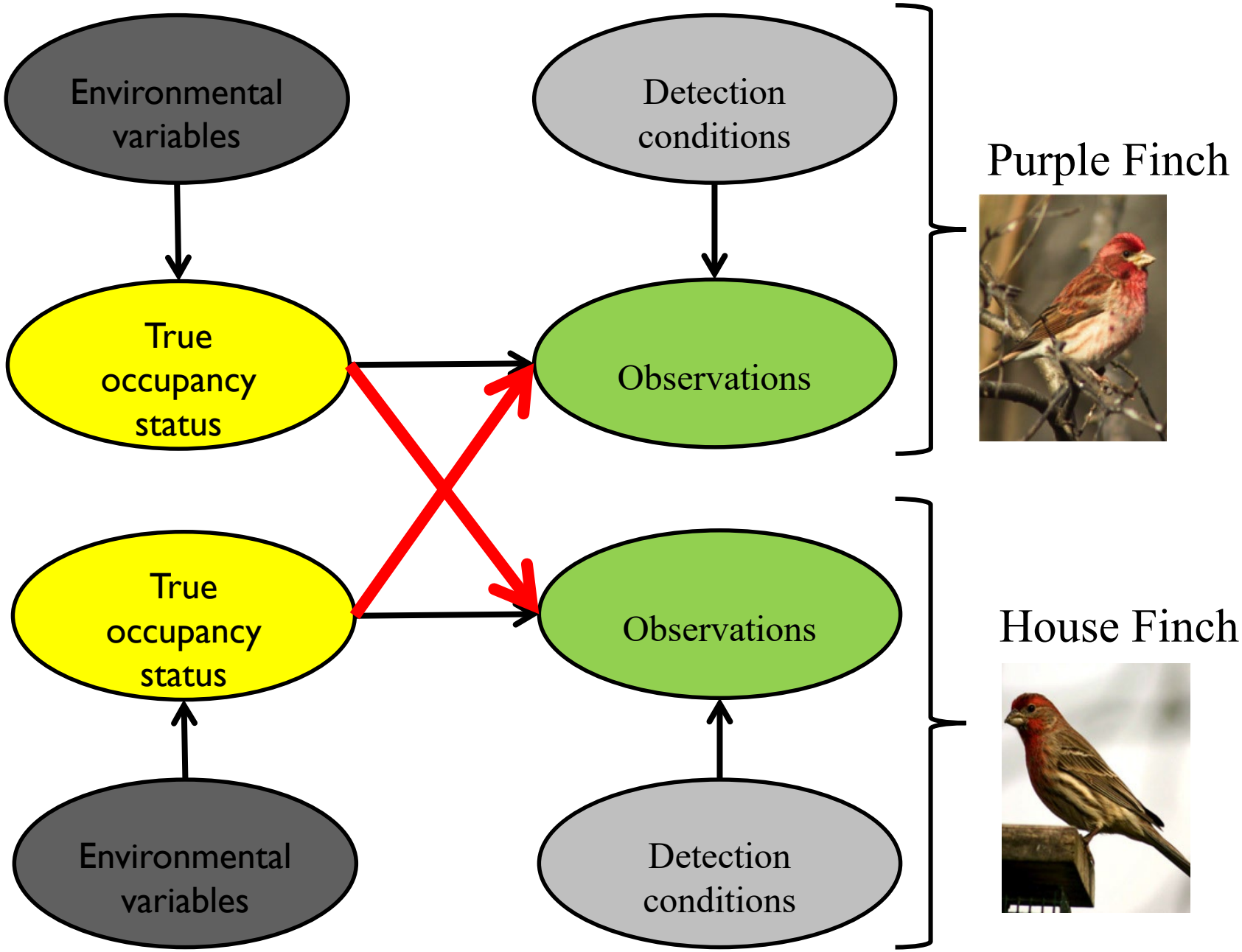


- ***Habitat:*** Mixed and coniferous woodlands; ornamental conifers in gardens.

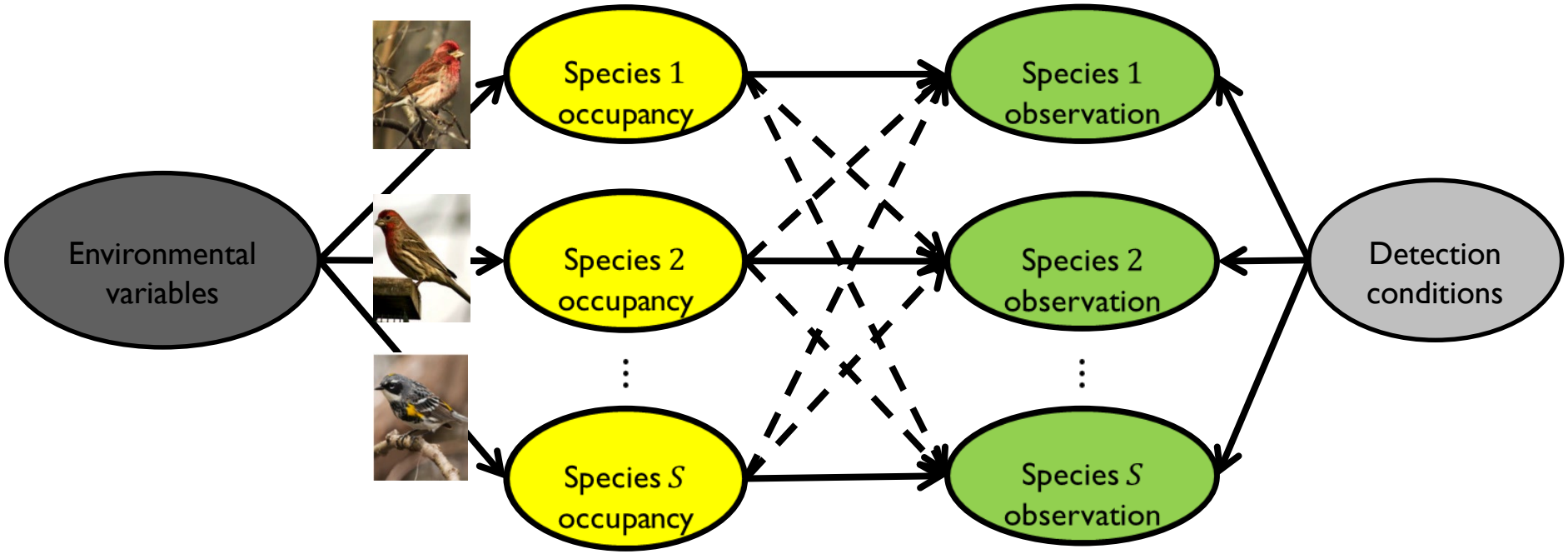
House Finch

- ***Habitat:*** cities and residential areas; coastal valleys that have become suburban.

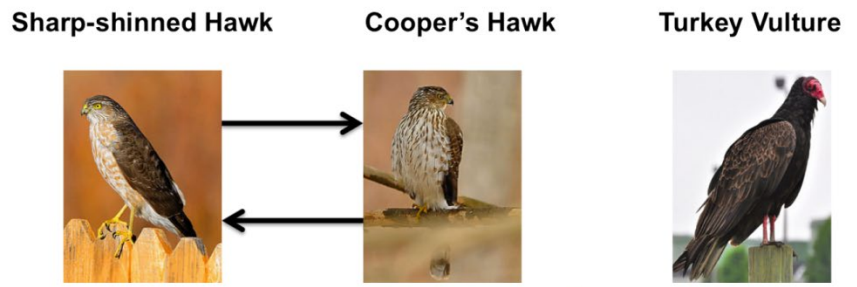
Photo credits: Chris Wood


Environmental variables
Detection conditions
Purple Finch
True occupancy status
Observations
True occupancy status
Observations
House Finch
Environmental variables
Detection conditions

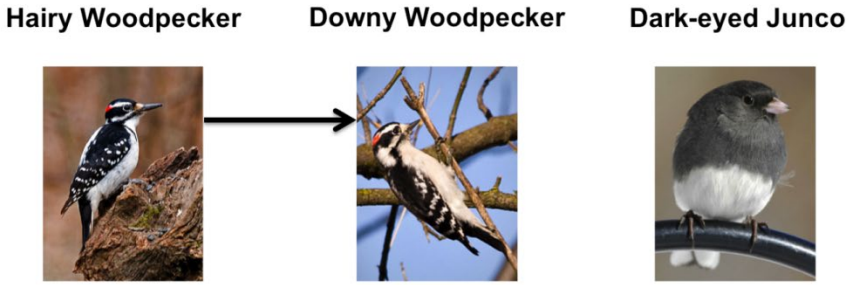
Solution: Multi-species occupancy modeling



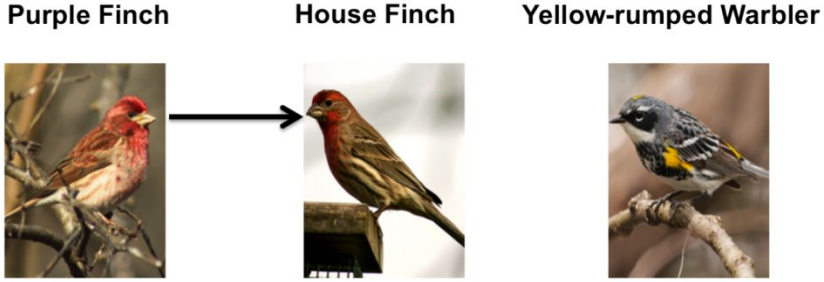
Result: Species confused by eBirders



(a) Hawks case study



(b) Woodpeckers case study

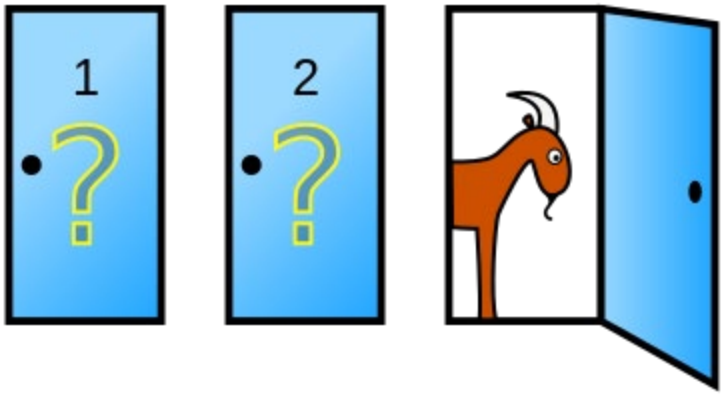


(c) Finches case study

Photo credits: Chris Wood

Example

- Monty Hall problem – there is a treasure behind one of the doors
  - Contestant chooses 1
  - Host opens one of the other doors with no treasure
  - Should the contest change his choice?



Monty Hall Problem

- P(A) = P(B) = P(C) = 1/3 – equal probability
- Contestant chooses A
- Host opens B which has no treasure
  - P(Open B|C) = 1; P(Open B|B) =0
  - P(Open B|A) = ½; P(Open C|A) = ½
- P(A| Open B)= P(A)* P(Open B|A)/P(Open B)
- P(C| Open B) = P(C)* P(Open B|C)/P(Open B)

  = 2 P(A |Open B)