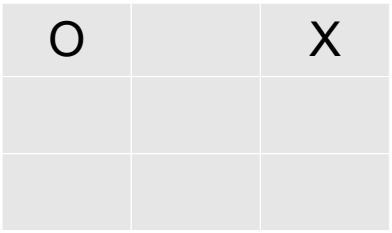
Lecture 6:

Alpha-beta Pruning and Decision Trees

6 February 2020

Adversarial Search

- Adversarial problems
  - Agent is competing against another agent
    - Could be a human or another AI agent
  - Agents are trying to accomplish conflicting tasks
    - Trying to stop other agent from reaching its goal
    - Other agent trying to stop you from reaching your goal
- Games
  - Chess
  - Checkers
  - tic tac toe
  - Computer games
    - Dota 2
- What makes adversarial problems difficult?
  - Need to predict actions of other agent



What move will opponent make?

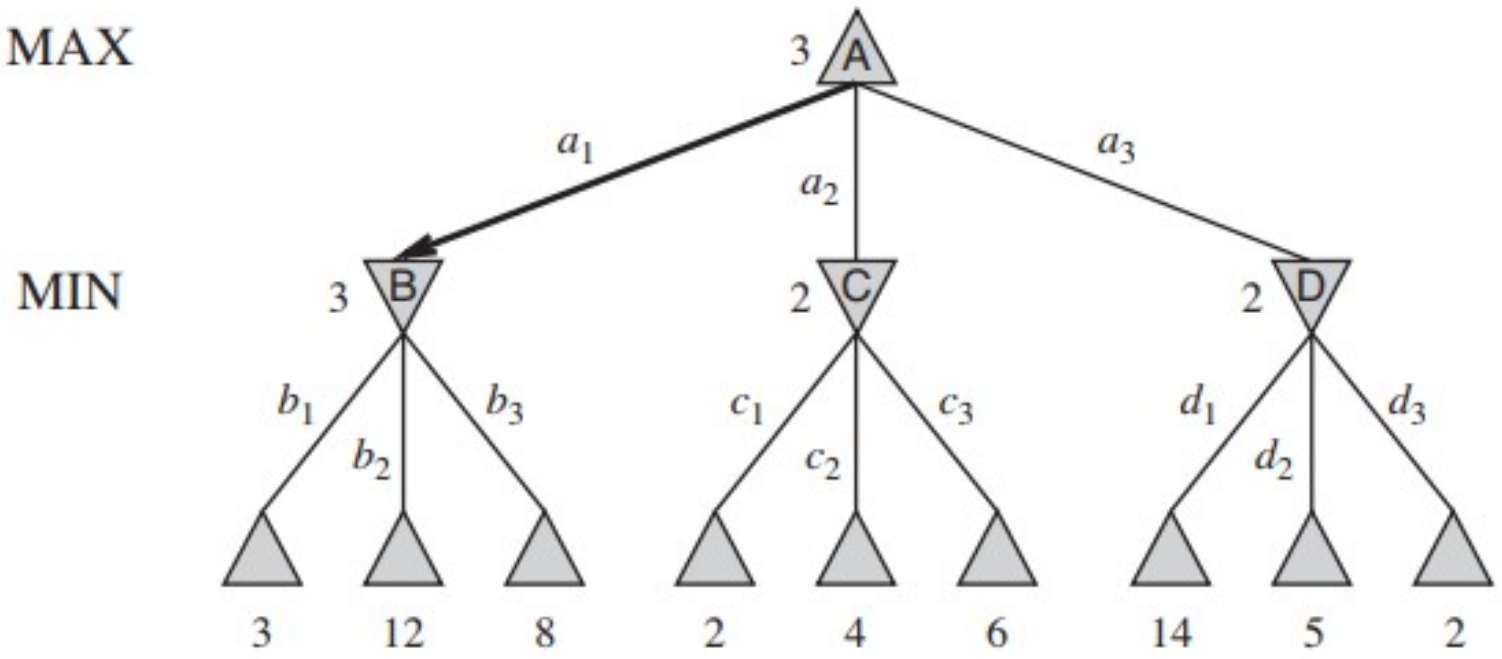
Adversarial Search

- Observation
  - A good state for you is a bad state for your opponent
  - A bad state for you is a good state for your opponent
  - Assume opponent will always pick worst state for you
  - Strategy:
    - Select action where opponent's best move will lead to best state for you
    - Select action with best worst state
    - Example: Chess
      - Opponent can mate in one
      - Select move that stops check-mate

Minimax search

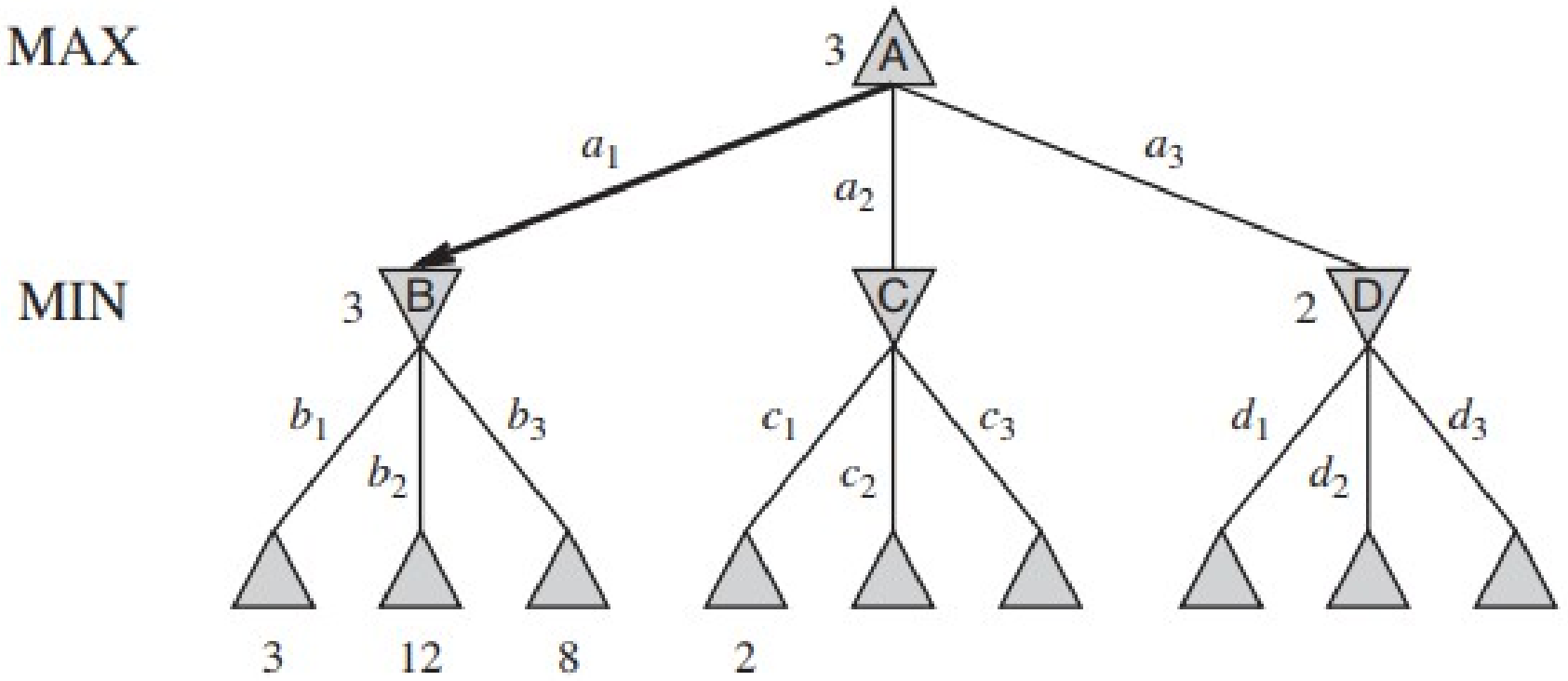
```
function minimax(s, d, bMaxPlayer)
    if d = 0 or goal(s)
        return h(s)
    if bMaxPlayer then
        v := −∞
        for each action a
            s' = τ(s,a)
            v := max(v, minimax(s, d − 1, FALSE))
        return v
    else
        for each action a
            s' = τ(s,a)
            v := min(v, minimax(s, d − 1, TRUE))
        return v
```

Minimax search example



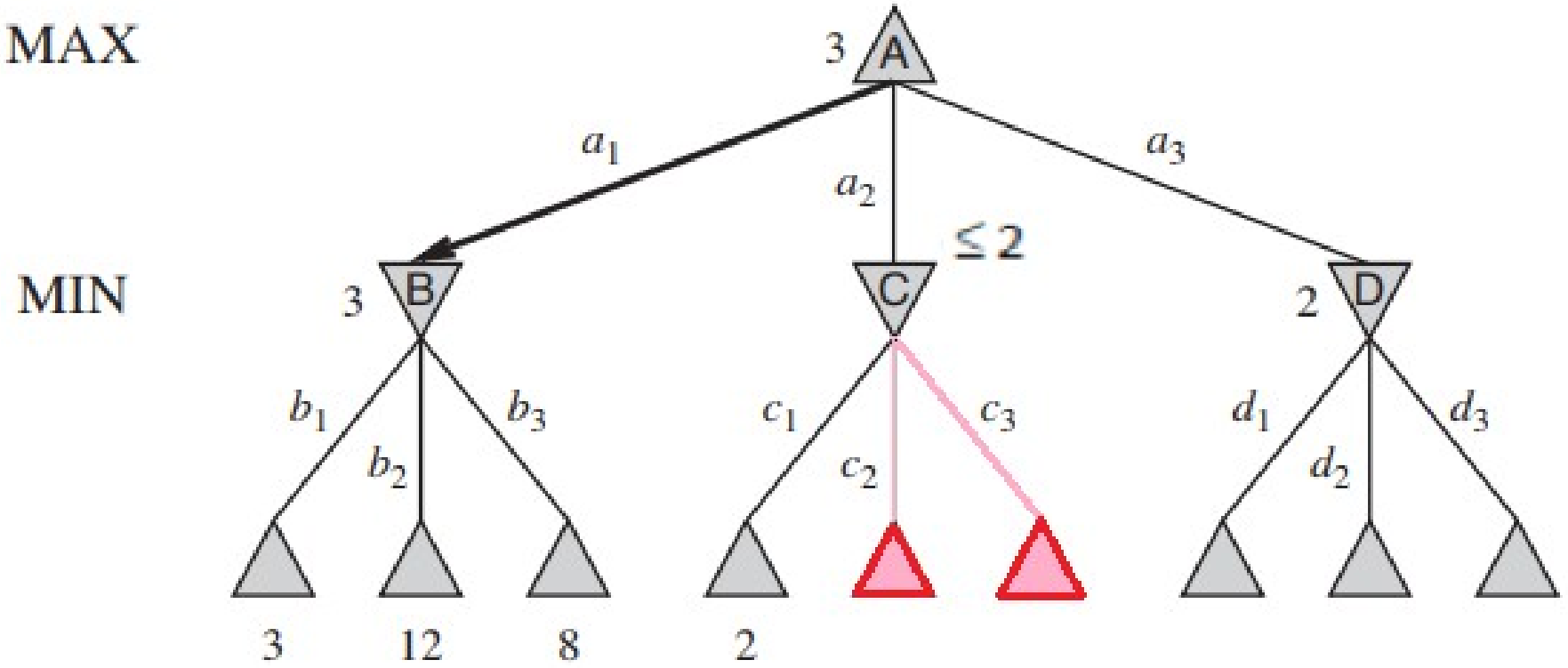
- Look ahead *d* moves
- Compute h of all leaves
- Inductively compute values for internal nodes from values of children
  - Max Step (agent's turn): Set values to be max value of children
  - Min Step (opponent's turn): Set value to min value of children
- Select child of root that has highest value
  - This is the action the agent should make

Alpha/beta pruning



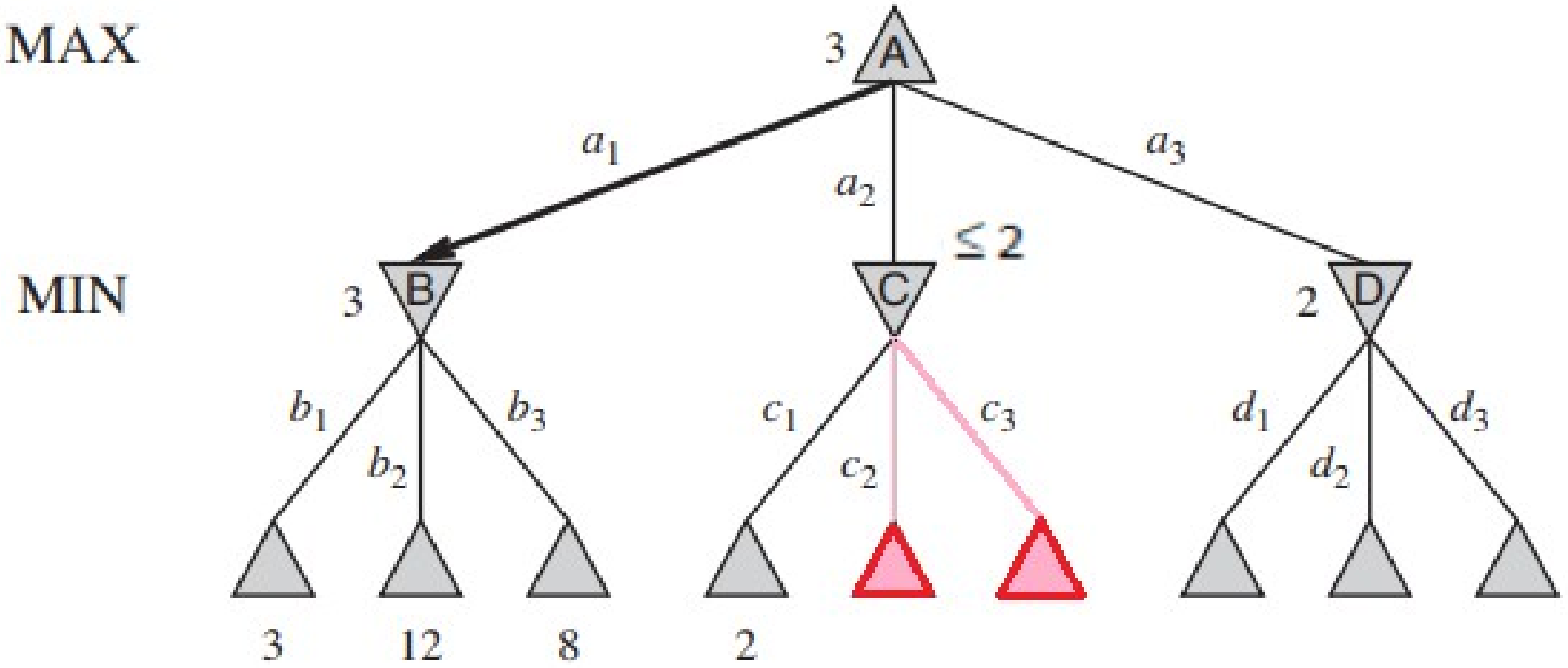
- Expand noes in depth first ordering
- Expand action c_1 and get 2
- What can you say about about state C and action a_2?

Alpha/beta pruning



- Expand noes in depth first ordering
- Expand action c_1 and get 2
- What can you say about about state C and action a_2?
  - At state C opponent will be able to get to a state with a value ≤ 2
  - Already know that a_1 gives you a value of 3
  - Never want to take action a_2
- Don't even need to consider other children of C

Alpha/beta pruning



- Expand noes in depth first ordering
- Expand action c_1 and get 2
- What can you say about about state C and action a_2?
  - At state C opponent will be able to get to a state with a value ≤ 2
  - Already know that a_1 gives you a value of 3
  - Never want to take action a_2
- Don't even need to consider other children of C

Alpha/beta pruning

- Best case complexity
  - $O(|A|^{d/2})$
- Worst case complexity
  - $O(|A|^{d})$
- What impacts how long search takes?

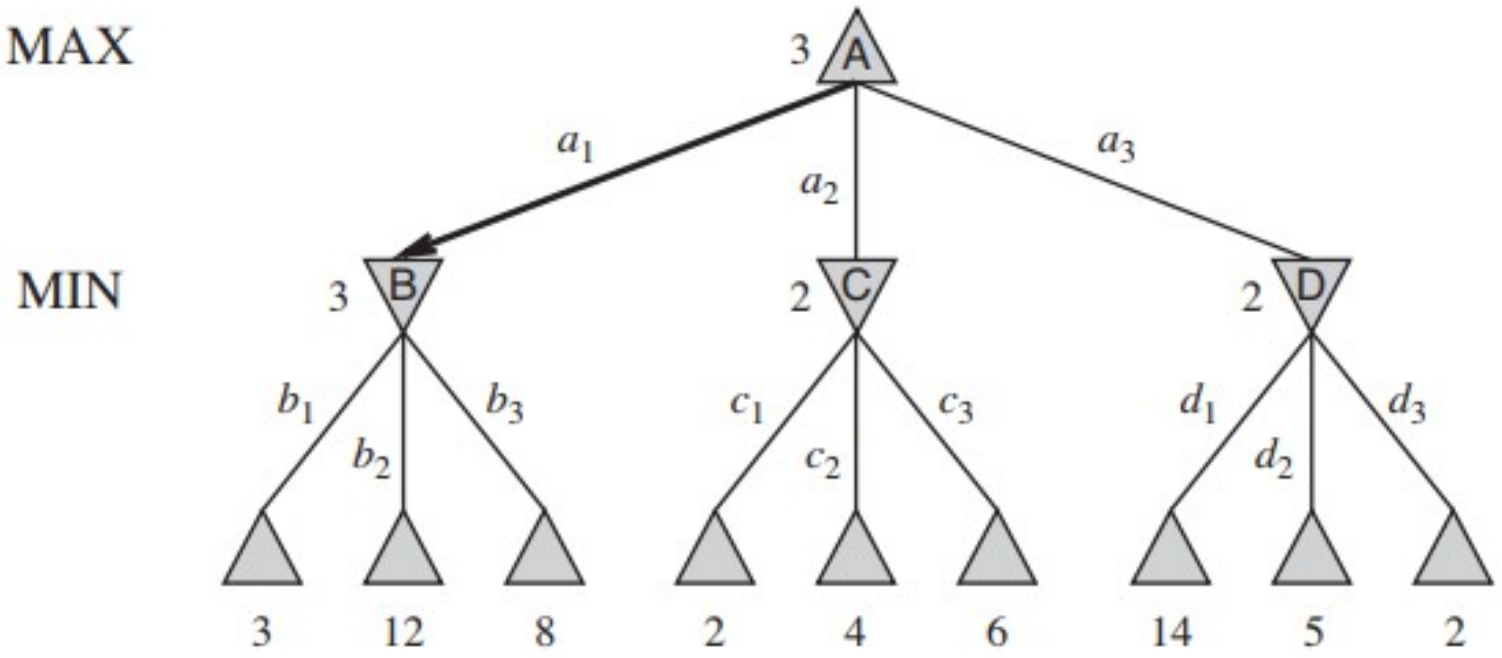
Alpha/beta pruning

- Best case complexity
  - $O(|A|^{d/2})$
- Worst case complexity
  - $O(|A|^{d})$
- What impacts how long search takes?
  - Order which you explore children
  - Maximizing steps: Want to expand children with higher value first
  - Minimizing step: Want to expand children with lower value first
- How can you accomplish this?

Alpha/beta pruning

- Best case complexity
  - $O(|A|^{d/2})$
- Worst case complexity
  - $O(|A|^{d})$
- What impacts how long search takes?
  - Order which you explore children
  - Maximizing steps: Want to expand children with higher value first
  - Minimizing step: Want to expand children with lower value first
- How can you accomplish this?
  - Order children based on heuristic

Minimax search example



Example: tic tac toe

- Agent is O
- Agent moves next
- h = number of chains of 2 Os and a blank

  - number of chains of 2 Xs and a blank

  - Below example would have a h value of -1

O		X
	X	
		O

Decision Trees

- Nodes are states
  - At each state you need to make a decision specific to that state
  - Edge for each choice
  - Edge leads to state obtained by selecting choice
- Effectively, actions are choices
  - Different from other examples because actions are different for each state
- Example - navigating a road map
  - At intersection of Main and Oak
    - Turn right on Oak
    - Turn Left on Oak
    - Stay on Main Street
  - Options available different for different intersections

Decision Tree Example

- Plant and animal classification

Building Decision Trees

- Can we build a decision tree from a sample set of data?

Building Decision Trees

- Problem
  - Given a list of choices construct a decision tree

Building Decision Trees: Example

- Problem
  - Given a list of choices construct a decision tree
  - Given a list of restaurants
    - Construct a decision tree
      - Decide what type of restaurant to go to

	Type	Price	Crowded	Time
McDonald's	Fast Food	$	Yes	< 10 min
Burger King	Fast Food	$	Yes	< 10 min
Chipotle	Mexican	$$	Yes	< 10 min
Jennie's	Diner	$$$	Yes	< 10 min
Fred's Fried Fish	Seafood	$	No	< 10 min
Captain John's	Seafood	$$$	No	20 min
Henri's	French	$$$	No	30 min
Pinocchio's	Italian	$$	Yes	20 min
Taste of Italy	Italian	$$	No	20 min

Entropy

- Entropy or Information Gain
  - define p(x) to be the proportion of elements of class x
    - Example: Proportion of restaurants of type fast food
  - E(*S*) = entropy of a set

$$\mathrm{H}(S) = \sum_{x \in X} -p(x) \log_2 p(x)$$

Iterative Dichotomiser 3 (ID3) Algorithm

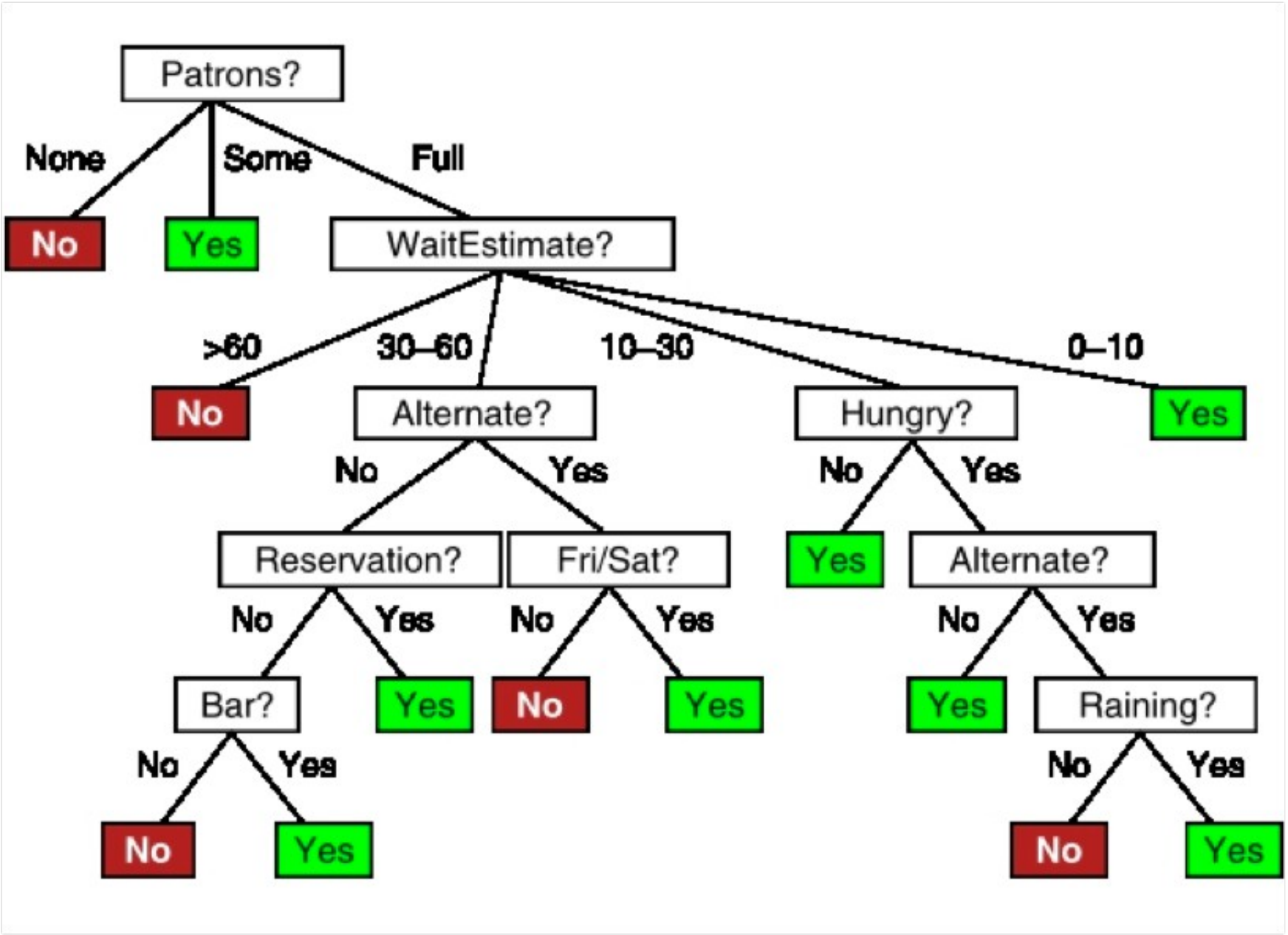
- Recursively select trait that results in groups with lowest entropy

RUTGERS Iterative Dichotomiser 3 (ID3) Algorithm

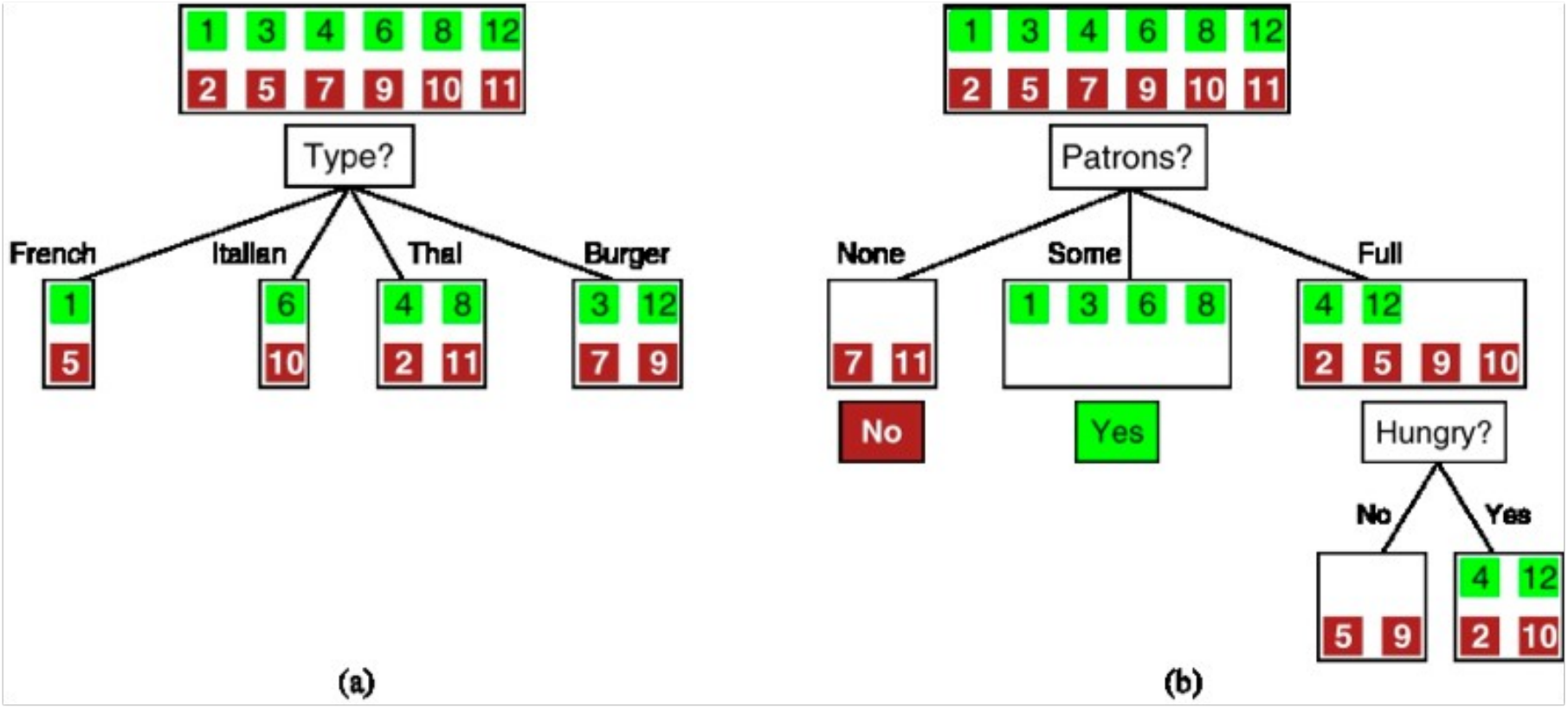
- Recursively select trait that results in groups with lowest entropy

	Type	Price	Crowded	Time
McDonald's	Fast Food	$	Yes	< 10 min
Burger King	Fast Food	$	Yes	< 10 min
Chipotle	Mexican	$$	Yes	10 min
Jennie's	Diner	$$$	Yes	10 min
Fred's Fried Fish	Seafood	$	No	< 10 min
Captain John's	Seafood	$$$	No	20 min
Henri's	French	$$$	No	30 min
Pinocchio's	Italian	$$	Yes	20 min
Taste of Italy	Italian	$$	No	20 min

Decision Tree Example



Testing on Attributes



Induced Decision Tree