BIL 682 – Artificial Intelligence

Week #3: Game playing

Aykut Erdem
Computer Vision Lab (CVL)
Hacettepe University

Image credit: Futurama S02E02 (Mars University)

Oily to Fatbot: "Mate in 143 moves."
Today

• Iterative Improvement Algorithms
  – Hill-climbing search
  – Simulated annealing search
  – Genetic Algorithms
  – Gradient Methods

• Game Playing
Recap: Hill-climbing Search

- Idea: keep a single “current” state and try to locally improve it
- “Like climbing mount Everest in thick fog with amnesia”
Recap: The state space “landscape”

- How to escape local maxima?
  - Random restart hill-climbing
Simulated annealing search

• Idea: escape local maxima by allowing some "bad" moves but gradually decrease their frequency
  – Probability of taking downhill move decreases with number of iterations, steepness of downhill move
    – Controlled by *annealing schedule*

• Inspired by tempering of glass, metal
Simulated annealing search

• Initialize \( \text{current} \) to starting state
• For \( i = 1 \) to \( \infty \)
  
  If \( T(i) = 0 \) return \( \text{current} \)
  
  Let \( \text{next} = \) random successor of \( \text{current} \)
  
  Let \( \Delta = \text{value}(\text{next}) - \text{value}(\text{current}) \)
  
  If \( \Delta > 0 \) then let \( \text{current} = \text{next} \)
  
  Else let \( \text{current} = \text{next} \) with probability \( \exp(\Delta/T(i)) \)
Effect of temperature

\[ \exp(\Delta/T) \]
Simulated annealing search

• One can prove: If temperature decreases slowly enough, then simulated annealing search will find a global optimum with probability approaching one

• However:
  – This usually takes impractically long
  – The more downhill steps you need to escape a local optimum, the less likely you are to make all of them in a row

• More modern techniques: general family of *Markov Chain Monte Carlo* (MCMC) algorithms for exploring complicated state spaces
Gene	algorithms use a natural selection metaphor

- Like beam search (selection), but also have pairwise crossover operators, with optional mutation
- Probably the most misunderstood, misapplied (and even maligned) technique around!

• Genetic algorithms use a natural selection metaphor
• Like beam search (selection), but also have pairwise crossover operators, with optional mutation
• Probably the most misunderstood, misapplied (and even maligned) technique around!
Example: N-Queens

• Why does crossover make sense here?
• When wouldn’t it make sense?
• What would mutation be?
• What would a good fitness function be?
Continuous Problems

- **Placing airports in Romania**
  - States \((x_1,y_1,x_2,y_2,x_3,y_3)\)
  - Cost: sum of squared distances to closest city
Gradient Methods

• How to deal with continuous (therefore infinite) state spaces?
  • Discretization: bucket ranges of values
    – e.g. force integral coordinates
• Continuous optimization
  – e.g. gradient ascent

\[ \nabla f = \left( \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial y_1}, \frac{\partial f}{\partial x_2}, \frac{\partial f}{\partial y_2}, \frac{\partial f}{\partial x_3}, \frac{\partial f}{\partial y_3} \right) \]

\[ x \leftarrow x + \alpha \nabla f(x) \]
Today

• Iterative Improvement Algorithms

• Game Playing
  – Games
  – Minimax Search
  – $\alpha$-$\beta$ Tree Pruning
  – Game Theory

The slides are mostly adopted from Dan Klein (UC Berkeley), Lana Lazebnik (UNC) and Hal Daumé III (UMD)
Adversarial Search
Game Playing

• Many different kinds of games!

• Axes:
  – Deterministic or stochastic?
  – One, two, or more players?
  – Perfect information (can you see the state)?

• Want algorithms for calculating a strategy (policy) which recommends a move in each state

<table>
<thead>
<tr>
<th></th>
<th>Deterministic</th>
<th>Stochastic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perfect information</td>
<td>Chess, checkers, go</td>
<td>Backgammon, monopoly</td>
</tr>
<tr>
<td>(fully observable)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Imperfect information</td>
<td>Battleships</td>
<td>Scrabble, poker,</td>
</tr>
<tr>
<td>(partially observable)</td>
<td></td>
<td>bridge</td>
</tr>
</tbody>
</table>
Deterministic Games

• Many possible formalizations, one is:
  – States: $S$ (start at $s_0$)
  – Players: $P=\{1...N\}$ (usually take turns)
  – Actions: $A$ (may depend on player / state)
  – Transition Function: $SxA \rightarrow S$
  – Terminal Test: $S \rightarrow \{t,f\}$
  – Terminal Utilities: $SxP \rightarrow R$

• Solution for a player is a policy: $S \rightarrow A$
Deterministic Games / Search
Games vs. single-agent search

• We don’t know how the opponent will act
  – The solution is not a fixed sequence of actions from start state to goal state, but a *strategy* or *policy* (a mapping from state to best move in that state)

• Efficiency is critical to playing well
  – The time to make a move is limited
  – The branching factor, search depth, and number of terminal configurations are huge
    • In chess, branching factor ≈ 35 and depth ≈ 100, giving a search tree of $10^{154}$ nodes
      – Number of atoms in the observable universe ≈ $10^{80}$
  – This rules out searching all the way to the end of the game
Deterministic Single-Player?

• Deterministic, single player, perfect information:
  – Know the rules
  – Know what actions do
  – Know when you win
  – E.g. Freecell, 8-Puzzle, Rubik’s cube
• ... it’s just search!
• Slight reinterpretation:
  – Each node stores a value: the best outcome it can reach
  – This is the maximal outcome of its children (the max value)
  – Note that we don’t have path sums as before (utilities at end)
• After search, can pick move that leads to best node
Deterministic Two-Player

• E.g. tic-tac-toe, chess, checkers

• Zero-sum games
  – One player maximizes result
  – The other minimizes result

• Minimax search
  – A state-space search tree
  – Players alternate
  – Each layer, or play, consists of a round of moves*
  – Choose move to position with highest minimax value = best achievable utility against best play

* Slightly different from the book definition
Tic-tac-toe Game Tree

- A game of tic-tac-toe between two players, “max” and “min”
COMPLETE MAP OF OPTIMAL TIC-TAC-TOE MOVES

Your move is given by the position of the largest red symbol on the grid. When your opponent picks a move, zoom in on the region of the grid where they went. Repeat.

MAP FOR X:

http://xkcd.com/832/
MAP FOR Q:

http://xkcd.com/832/
Minimax Search

function $\text{Max-Value}(state)$ returns a utility value
  if $\text{Terminal-Test}(state)$ then return $\text{Utility}(state)$
  $v \leftarrow -\infty$
  for $a, s$ in $\text{Successors}(state)$ do $v \leftarrow \text{Max}(v, \text{Min-Value}(s))$
  return $v$

function $\text{Min-Value}(state)$ returns a utility value
  if $\text{Terminal-Test}(state)$ then return $\text{Utility}(state)$
  $v \leftarrow \infty$
  for $a, s$ in $\text{Successors}(state)$ do $v \leftarrow \text{Min}(v, \text{Max-Value}(s))$
  return $v$
A more abstract game tree

Terminal utilities (for MAX)

A two-player game
A more abstract game tree

- **Minimax value of a node**: the utility (for MAX) of being in the corresponding state, assuming perfect play on both sides
- **Minimax strategy**: Choose the move that gives the best worst-case payoff
Computing the minimax value of a state

- **Minimax**(state) =
  - Utility(state) if state is terminal
  - max **Minimax**(successors(state)) if player = MAX
  - min **Minimax**(successors(state)) if player = MIN
The minimax strategy is optimal against an optimal opponent

- If the opponent is sub-optimal, the utility can only be higher
- A different strategy may work better for a sub-optimal opponent, but it will necessarily be worse against an optimal opponent
Minimax Properties

- Optimal against a perfect player.
- Time complexity?
  - \( O(b^m) \)
- Space complexity?
  - \( O(bm) \)
- For chess, \( b \approx 35, m \approx 100 \)
  - Exact solution is completely infeasible
  - But, do we need to explore the whole tree?
Resource Limits

• Cannot search to leaves
• Depth-limited search
  – Instead, search a limited depth of tree
  – Replace terminal utilities with an eval function for non-terminal positions
• Guarantee of optimal play is gone
• More plies makes a BIG difference
• Example:
  – Suppose we have 100 seconds, can explore 10K nodes / sec
  – So can check 1M nodes per move
  – reaching about depth 8 – decent chess program
Evaluation Functions

• Function which scores non-terminals

• Ideal function: returns the utility of the position
• In practice: typically weighted linear sum of features:

\[ \text{Eval}(s) = w_1 f_1(s) + w_2 f_2(s) + \ldots + w_n f_n(s) \]

• e.g. \( f_1(s) = (\text{num white queens} - \text{num black queens}) \), etc.
Iterative Deepening uses DFS as a subroutine:

1. Do a DFS which only searches for paths of length 1 or less. (DFS gives up on any path of length 2)
2. If “1” failed, do a DFS which only searches paths of length 2 or less.
3. If “2” failed, do a DFS which only searches paths of length 3 or less.
   ....and so on.

Why do we want to do this for multiplayer games?
α-β Tree-Pruning
\( \alpha-\beta \) Pruning Example

\[
\begin{array}{c}
A_1 \\
A_2 \\
A_3 \\
A_{11} \\
A_{12} \\
A_{13} \\
A_{21} \\
A_{22} \\
A_{23} \\
A_{31} \\
A_{32} \\
A_{33} \\
3 \\
12 \\
8 \\
2 \\
4 \\
6 \\
14 \\
5 \\
2
\end{array}
\]
Pruning in Minimax Search

![Diagram of a Minimax search tree with pruning]

- Leaf nodes with values:
  - 3
  - 12
  - 8
  - 2
  - 14
  - 5
  - 2

- Pruned branches:
  - From leaf node with value 8
  - From leaf node with value 14
\(\alpha-\beta\) Pruning

- General configuration
  - \(\alpha\) is the best value that MAX can get at any choice point along the current path
  - If \(n\) becomes worse than \(\alpha\), MAX will avoid it, so can stop considering \(n\)'s other children
  - Define \(\beta\) similarly for MIN

![Diagram of \(\alpha-\beta\) Pruning](image_url)
\[ \alpha - \beta \] Pruning Pseudocode

**function Max-Value(state) returns a utility value**

if Terminal-Test(state) then return Utility(state)

\[ v \leftarrow -\infty \]

for \( a, s \) in Successors(state) do \( v \leftarrow \max(v, \text{Min-Value}(s)) \)

return \( v \)

**function Max-Value(state, \( \alpha, \beta \)) returns a utility value**

*inputs: state, current state in game*

\( \alpha \), the value of the best alternative for \( \max \) along the path to \( state \)

\( \beta \), the value of the best alternative for \( \min \) along the path to \( state \)

if Terminal-Test(state) then return Utility(state)

\[ v \leftarrow -\infty \]

for \( a, s \) in Successors(state) do

\[ v \leftarrow \max(v, \text{Min-Value}(s, \alpha, \beta)) \]

if \( v \geq \beta \) then return \( v \)

\( \alpha \leftarrow \max(\alpha, v) \)

return \( v \)
\(\alpha-\beta\) Pruning Properties

- Pruning has no effect on final result
- Good move ordering improves effectiveness of pruning
- With “perfect ordering”:  
  - Time complexity drops to \(O(b^{m/2})\)
  - Doubles solvable depth
  - Full search of, e.g. chess, is still hopeless!
- A simple example of metareasoning, here reasoning about which computations are relevant
Non-Zero-Sum Games

• Similar to minimax:
  – Utilities are now tuples
  – Each player maximizes their own entry at each node
  – Propagate (or back up) nodes from children
Stochastic Single-Player

- What if we don’t know what the result of an action will be? E.g.,
  - In solitaire, shuffle is unknown
  - In minesweeper, mine locations
  - In pacman, ghosts!
- Can do expectimax search
  - Chance nodes, like actions except the environment controls the action chosen
  - Calculate utility for each node
  - Max nodes as in search
  - Chance nodes take average (expectation) of value of children

Stochastic Two-Player

- E.g. backgammon
  - Expectiminimax
  - Environment is an extra player that moves after each agent
  - Chance nodes take expectations, otherwise like minimax

What's Next?

- Make sure you know what:
  - Probabilities are
  - Expectations are
- Next topics:
  - Dealing with uncertainty
  - How to learn evaluation functions
  - Markov Decision Processes
Stochastic Two-Player

- E.g. backgammon
- Expectiminimax (!)
  - Environment is an extra player that moves after each agent
  - Chance nodes take expectations, otherwise like minimax

```
if state is a Max node then
    return the highest ExpectiMinimax-Value of Successors(state)
if state is a Min node then
    return the lowest ExpectiMinimax-Value of Successors(state)
if state is a chance node then
    return average of ExpectiMinimax-Value of Successors(state)
```
Stochastic Two-Player

• Dice rolls increase $b$: 21 possible rolls with 2 dice
  – Backgammon $\approx$ 20 legal moves
  – Depth $4=20 \times (21 \times 20)^3 \approx 1.2 \times 10^9$

• As depth increases, probability of reaching a given node shrinks
  – So value of lookahead is diminished
  – So limiting depth is less damaging
  – But pruning is less possible...

• TDGammon uses depth-2 search + very good eval function + reinforcement learning: world-champion level play
Game playing algorithms today

• Computers are better than humans:
  – **Checkers**: solved in 2007
  – **Chess**: IBM Deep Blue defeated Kasparov in 1997

• Computers are competitive with top human players:
  – **Backgammon**: TD-Gammon system used reinforcement learning to learn a good evaluation function
  – **Bridge**: top systems use Monte Carlo simulation and alpha-beta search

• Computers are not competitive:
  – **Go**: branching factor 361. Existing systems use Monte Carlo simulation and pattern databases
DIFFICULTY OF VARIOUS GAMES FOR COMPUTERS

EASY

- TIC-TAC-TOE
- NIM
- GHOST (1989)
- CONNECT FOUR (1995)

SOLVED FOR ALL POSSIBLE POSITIONS

SOLVED FOR STARTING POSITIONS

COMPUTERS CAN PLAY PERFECTLY

COMPUTERS CAN BEAT TOP HUMANS

- GOMOKU
- CHECKERS (2007)
- SCRABBLE
- COUNTERSTRIKE
- REVERSI
- BEER PONG
- CHESS

FEBRUARY 10, 1996: FIRST WIN BY COMPUTER AGAINST TOP HUMAN
NOVEMBER 21, 2005: LAST WIN BY HUMAN AGAINST TOP COMPUTER

HARD

COMPUTERS STILL LOSE TO TOP HUMANS
(BUT FOCUSED R&D COULD CHANGE THIS)

- JEOPARDY!
- POKER
- STARCRAFT
- ARIMAA
- GO
- SNAKES AND LADDERS
- MAO
- SEVEN MINUTES IN HEAVEN
- CALVINBALL

http://xkcd.com/1002/
Game theory

- **Game theory** deals with systems of interacting agents where the outcome for an agent depends on the actions of all the other agents
  - Applied in sociology, politics, economics, biology, and, of course, AI

- **Agent design:** determining the best strategy for a rational agent in a given game

- **Mechanism design:** how to set the rules of the game to ensure a desirable outcome
Modelling behaviour

Game theory in practice

Computing: Software that models human behaviour can make forecasts, outfox rivals and transform negotiations

Sep 3rd 2011 | From the print edition

http://www.economist.com/node/21527025
Simultaneous single-move games

- Players must choose their actions at the same time, without knowing what the others will do
  - Form of partial observability

**Normal form** representation:

<table>
<thead>
<tr>
<th>Player 1</th>
<th>0,0</th>
<th>1,-1</th>
<th>-1,1</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1,1</td>
<td>0,0</td>
<td>1,-1</td>
<td></td>
</tr>
<tr>
<td>1,-1</td>
<td>-1,1</td>
<td>0,0</td>
<td></td>
</tr>
</tbody>
</table>

**Payoff matrix**

- (Player 1’s utility is listed first)

Is this a zero-sum game?
Rock-Paper-Scissors Championship
Prisoner’s dilemma

- Two criminals have been arrested and the police visit them separately.
- If one player testifies against the other and the other refuses, the one who testified goes free and the one who refused gets a 10-year sentence.
- If both players testify against each other, they each get a 5-year sentence.
- If both refuse to testify, they each get a 1-year sentence.

<table>
<thead>
<tr>
<th>Bob: Testify</th>
<th>Alice: Testify</th>
<th>Alice: Refuse</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>-10</td>
</tr>
<tr>
<td>Bob: Refuse</td>
<td>-10</td>
<td>-1</td>
</tr>
</tbody>
</table>

Alice: Testify
Alice: Refuse
Prisoner’s dilemma

• Alice’s reasoning:
  – Suppose Bob testifies. Then I get 5 years if I testify and 10 years if I refuse. So I should testify.
  – Suppose Bob refuses. Then I go free if I testify, and get 1 year if I refuse. So I should testify.

• Dominant strategy: A strategy whose outcome is better for the player regardless of the strategy chosen by the other player

<table>
<thead>
<tr>
<th></th>
<th>Alice: Testify</th>
<th>Alice: Refuse</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob: Testify</td>
<td>-5,-5</td>
<td>-10,0</td>
</tr>
<tr>
<td>Bob: Refuse</td>
<td>0,-10</td>
<td>-1,-1</td>
</tr>
</tbody>
</table>
Prisoner’s dilemma

- **Nash equilibrium**: A pair of strategies such that no player can get a bigger payoff by switching strategies, provided the other player sticks with the same strategy
  - *(Testify, testify)* is a *dominant strategy equilibrium*

- **Pareto optimal outcome**: It is impossible to make one of the players better off without making another one worse off

- In a non-zero-sum game, a Nash equilibrium is not necessarily Pareto optimal!

<table>
<thead>
<tr>
<th></th>
<th>Alice: Testify</th>
<th>Alice: Refuse</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob: Testify</td>
<td>-5, -5</td>
<td>-10, 0</td>
</tr>
<tr>
<td>Bob: Refuse</td>
<td>0, -10</td>
<td>-1, -1</td>
</tr>
</tbody>
</table>
Recall: Multi-player, non-zero-sum game
Prisoner’s dilemma in real life

- Price war
- Arms race
- Steroid use
- Pollution control
- **Diner’s dilemma**

![Prisoner's Dilemma Table]

http://en.wikipedia.org/wiki/Prisoner’s_dilemma
Is there any way to get a better answer?

- **Superrationality**
  - Assume that the answer to a symmetric problem will be the same for both players
  - Maximize the payoff to each player while considering only identical strategies
  - Not a conventional model in game theory

- **Repeated games**
  - If the number of rounds is fixed and known in advance, the equilibrium strategy is still to defect
  - If the number of rounds is unknown, cooperation may become an equilibrium strategy
Is there a dominant strategy for either player?
Is there a Nash equilibrium?
   – (Stag, stag) and (hare, hare)
Model for cooperative activity
Prisoner’s dilemma vs. stag hunt

<table>
<thead>
<tr>
<th></th>
<th>Cooperate</th>
<th>Defect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cooperate</td>
<td>Win – win</td>
<td>Win big – lose big</td>
</tr>
<tr>
<td>Defect</td>
<td>Lose big – win big</td>
<td>Lose – lose</td>
</tr>
</tbody>
</table>

Players can gain by defecting unilaterally

<table>
<thead>
<tr>
<th></th>
<th>Cooperate</th>
<th>Defect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cooperate</td>
<td>Win big – win big</td>
<td>Win – lose</td>
</tr>
<tr>
<td>Defect</td>
<td>Lose – win</td>
<td>Win – win</td>
</tr>
</tbody>
</table>

Players lose by defecting unilaterally
Game of Chicken

- Is there a dominant strategy for either player?
- Is there a Nash equilibrium? (Straight, chicken) or (chicken, straight)
- Anti-coordination game: it is mutually beneficial for the two players to choose different strategies
  - Model of escalated conflict in humans and animals (hawk-dove game)
- How are the players to decide what to do?
  - Pre-commitment or threats
  - Different roles: the “hawk” is the territory owner and the “dove” is the intruder, or vice versa

• **Mixed strategy**: a player chooses between the moves according to a probability distribution

• Suppose each player chooses S with probability \( \frac{1}{10} \). Is that a Nash equilibrium?

• Consider payoffs to P1 while keeping P2’s strategy fixed
  – The payoff of P1 choosing S is \( \frac{1}{10}(-10) + \frac{9}{10}(1) = -1/10 \)
  – The payoff of P1 choosing C is \( \frac{1}{10}(-1) + \frac{9}{10}(0) = -1/10 \)
  – Is there a different strategy that can improve P1’s payoff?
  – Similar reasoning applies to P2
Ultimatum game

- Alice and Bob are given a sum of money $S$ to divide
  - Alice picks $A$, the amount she wants to keep for herself
  - Bob picks $B$, the smallest amount of money he is willing to accept
  - If $S - A \geq B$, Alice gets $A$ and Bob gets $S - A$
  - If $S - A < B$, both players get nothing

- What is the Nash equilibrium?
  - Alice offers Bob the smallest amount of money he will accept: $S - A = B$
  - Alice and Bob both want to keep the full amount: $A = S$, $B = S$
    (both players get nothing)

- How would humans behave in this game?
  - If Bob perceives Alice’s offer as unfair, Bob will be likely to refuse
  - Is this rational?
    - Maybe Bob gets some positive utility for “punishing” Alice?
Existence of Nash equilibria

• Any game with a finite set of actions has at least one Nash equilibrium (which may be a mixed-strategy equilibrium)

• If a player has a dominant strategy, there exists a Nash equilibrium in which the player plays that strategy and the other player plays the best response to that strategy

• If both players have strictly dominant strategies, there exists a Nash equilibrium in which they play those strategies
Computing Nash equilibria

- For a two-player zero-sum game, simple linear programming problem
- For non-zero-sum games, the algorithm has worst-case running time that is exponential in the number of actions
- For more than two players, and for sequential games, things get pretty hairy
Nash equilibria and rational decisions

• If a game has a *unique* Nash equilibrium, it will be adopted if each player
  – is rational and the payoff matrix is accurate
  – doesn’t make mistakes in execution
  – is capable of computing the Nash equilibrium
  – believes that a deviation in strategy on their part will not cause the other players to deviate
  – there is *common knowledge* that all players meet these conditions

Mechanism design (inverse game theory)

• Assuming that agents pick rational strategies, how should we design the game to achieve a socially desirable outcome?

• We have multiple agents and a center that collects their choices and determines the outcome
Auctions

• Goals
  – Maximize revenue to the seller
  – Efficiency: make sure the buyer who values the goods the most gets them
  – Minimize transaction costs for buyer and sellers
Ascending-bid auction

• What’s the optimal strategy for a buyer?
  – Bid until the current bid value exceeds your *private value*

• Usually revenue-maximizing and efficient, unless the reserve price is set too low or too high

• Disadvantages
  – Collusion
  – Lack of competition
  – Has high communication costs
Sealed-bid auction

- Each buyer makes a single bid and communicates it to the auctioneer, but not to the other bidders
  - Simpler communication
  - More complicated decision-making: the strategy of a buyer depends on what they believe about the other buyers
  - Not necessarily efficient

Sealed-bid second-price auction: the winner pays the price of the second-highest bid

- Let $V$ be your private value and $B$ be the highest bid by any other buyer
- If $V > B$, your optimal strategy is to bid above $B$ – in particular, bid $V$
- If $V < B$, your optimal strategy is to bid below $B$ – in particular, bid $V$
- Therefore, your dominant strategy is to bid $V$
- This is a truth revealing mechanism
Dollar auction

• A dollar bill is auctioned off to the highest bidder, but the second-highest bidder has to pay the amount of his last bid
  – Player 1 bids 1 cent
  – Player 2 bids 2 cents
  – ...
  – Player 2 bids 98 cents
  – Player 1 bids 99 cents
    • If Player 2 passes, he loses 98 cents, if he bids $1, he might still come out even
  – So Player 2 bids $1
    • Now, if Player 1 passes, he loses 99 cents, if he bids $1.01, he only loses 1 cent
  – ...

• What went wrong?
  – When figuring out the expected utility of a bid, a rational player should take into account the future course of the game

• What if Player 1 starts by bidding 99 cents?
Tragedy of the commons

• States want to set their policies for controlling emissions
  – Each state can reduce their emissions at a cost of -10
  or continue to pollute at a cost of -5
  – If a state decides to pollute, -1 is added to the utility of every other state

• What is the dominant strategy for each state?
  – Continue to pollute
  – Each state incurs cost of -5-49 = -54
  – If they all decided to deal with emissions, they would incur a cost of only -10 each

• Mechanism for fixing the problem:
  – Tax each state by the total amount by which they reduce the global utility (externality cost)
  – This way, continuing to pollute would now cost -54
Reading Assignments


• Due next week!