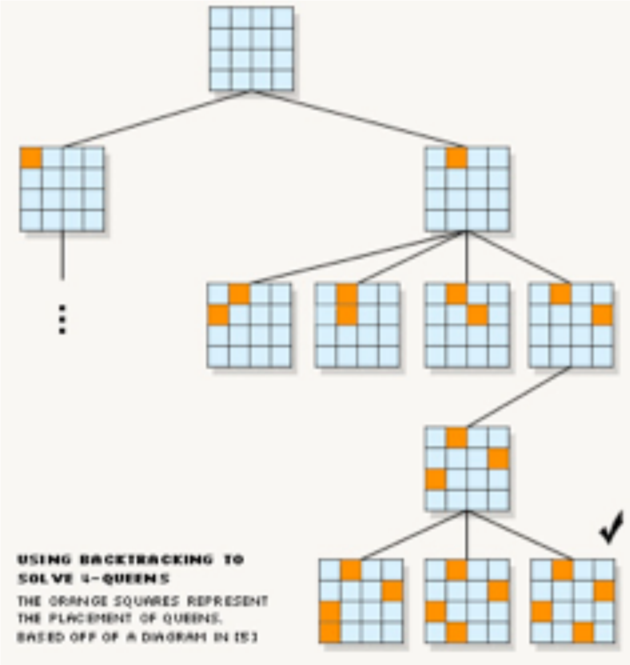
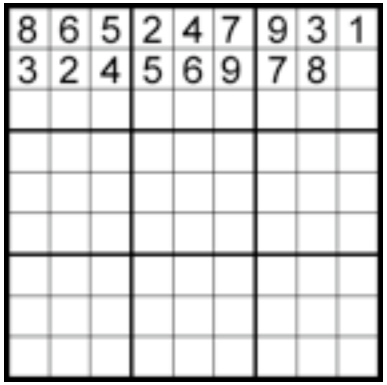
Backtracking by image search


USING BACKTRACKING TO SOLVE 4-QUEENS
THE ORANGE SQUARES REPRESENT THE PLACEMENT OF QUEENS.
BASED OFF OF A DIAGRAM IN [5]



8	6	5	2	4	7	9	3	1
3	2	4	5	6	9	7	8	

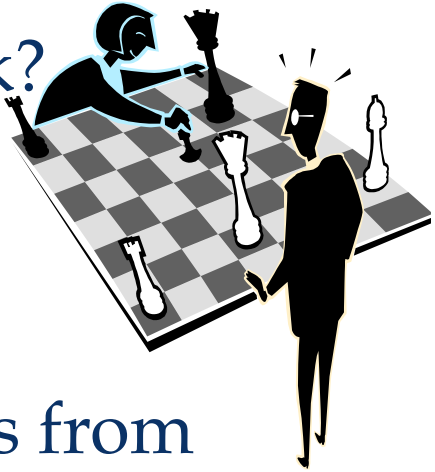
Searching with no guarantees

- Search for best move in automated game play
  - Can we explore every move?
  - Are there candidate moves ranked by "goodness"?
  - Can we explore entire tree of possible moves?

- Search with partial information
  - Autocorrect for SMS, mobile typing, …
  - Finding words on a Boggle board
  - What numbers fit in Sudoku square

- Try something, if at first you don't succeed ….

Search, Backtracking,Heuristics



- How do you find a needle in a haystack?
  - How does a computer play chess?
  - Why would you write that program?
- How does Bing/Googlemap find routes from one place to another?
  - Shortest path algorithms
  - Longest path algorithms
- Optimal algorithms and heuristic algorithms
  - When is close good enough? How do measure "closeness"?
  - When is optimality important, how much does it

Exhaustive Search/Heuristics

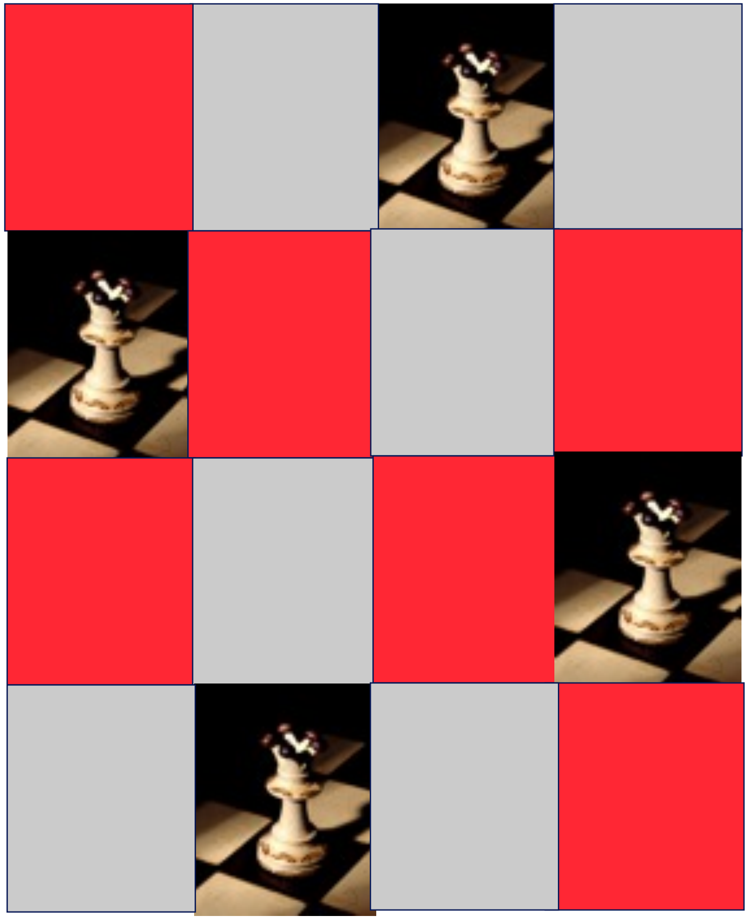
- We can probably explore entire game tree for tic-tac-toe, but not for chess
  - How many tic-tac-toe boards are there?
  - How many chess boards are there?

- What do we do when the *search space* is huge?
  - Brute-force/exhaustive won't work, need *heuristics*?
  - What about google-maps/Garmin finding routes?

- Backtracking can use both concepts
  - Game tree pruning a good idea most of the time

Classic problem: N queens

- Can queens be placed on a chess board so that no queens attack each other?
  - Easily place two queens
  - What about 8 queens?
- Make the board NxN, this is the N queens problem
  - Place one queen/column
  - Horiz/Vert/Diag attacks
- Backtracking
  - Tentative placement
  - Recurse, if ok done!
  - If fail, undo tentative, retry
- wikipedia-n-queens



Backtracking idea with N queens

- For each column C, tentatively place a queen
    - Try first row in column C, if ok, move onto next column
        - Typically "move on" is recursive
    - If solved, done, otherwise try next row in column C
        - Must unplace queen when failing/unwind recursion

- Each column C "knows" what row R it's on
    - If first time, that's row zero, but might be an attack
    - Unwind recursion/backtrack, try "next" location

- Backtracking: record an attempt go forward
    - Move must be "undoable" on backtracking/unwinding

N queens backtracking:

```
public boolean solve(int col){
  if (col == mySize) return true;
        // try each row until all are tried

  for(int r=0; r < mySize; r++){
      if (myBoard.safeToPlace(r,col)){
          myBoard.setQueen(r,col,true);
          if (solve(col+1)){
              return true;
          }
          myBoard.setQueen(r,col,false);
      }
  }
  return false;
}
```

Basic ideas in backtracking search

- Enumerate all possible choices/moves
    - We try these choices in order, committing to a choice
    - If the choice doesn't pan out we must undo the choice
        - Backtracking step, choices must be undoable

- Inherently recursive, when to stop searching?
    - When all columns tried in N queens
    - When we have found the exit in a maze
    - When every possible moved tried in Tic-tac-toe or chess?
        - Is there a difference between these games?

- Summary: enumerate choices, try a choice, undo a choice, this is *brute force* search: try everything

Queens Details

- How do we know when it’s safe to place a queen?
  - No queen in same row, or diagonal
  - For each column, store the row that a queen is in
  - See QBoard.java for details

- Advanced OO: For GUI version, we use a *decorator*
  - The QBoardGUI is an IQueenState class and it has an IQueenState object in it
  - Appears as an IQueenState to client, but uses an existing one to help do its work
  - One of many object oriented design patterns, seen in java.io.* for example

Backtracking Practice

- GridGame APT: how many winning moves?
  - It's my turn and there are no places to go: return 0, why?
  - Try to place an 'X', count opponent wins. If 0 then …
    - After placing 'X' and counting, unplace the 'X', why?
- http://bit.ly/201-btrack

Daphne Koller

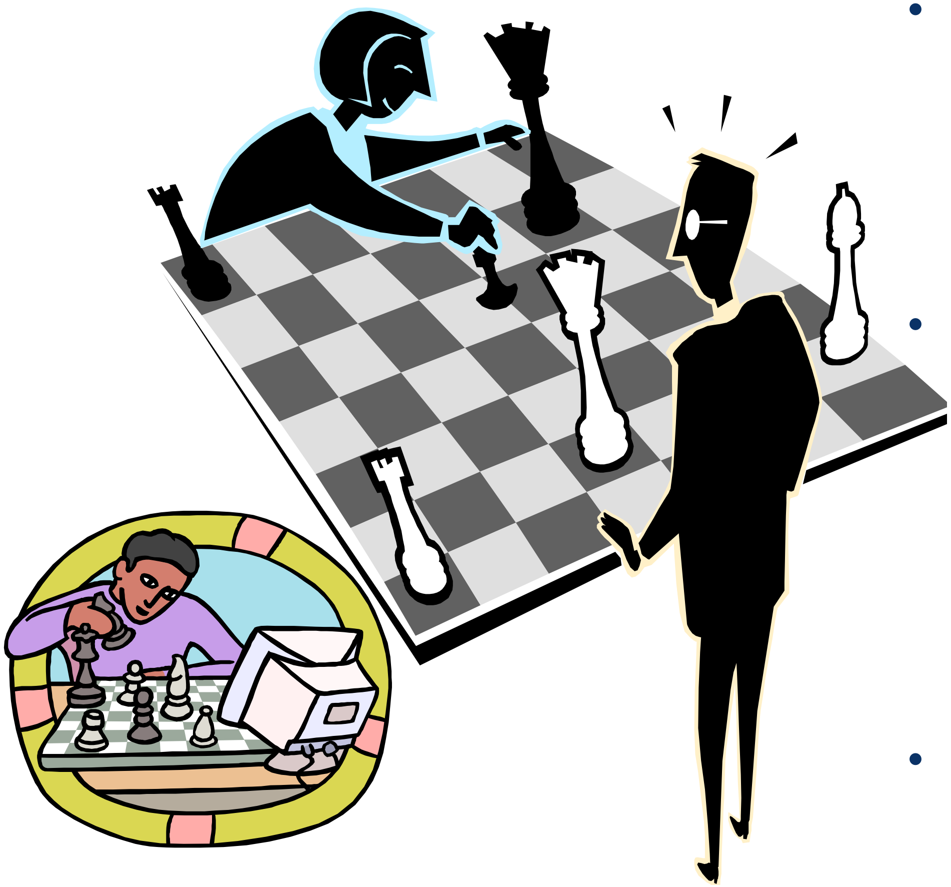
- 2004, Macarthur
- 2008, first ACM/Infosys

"The world is noisy and messy …You need to deal with the noise and uncertainty."

- http://tinyurl.com/3tdlug

"I find it distressing that the view of the field is that you sit in your office by yourself surrounded by old pizza boxes and cans of Coke, hacking away at the bowels of the Windows operating system," she said. "I spend most of my time thinking about things like how does a cell work or how do we understand images in the world around us?"

Computer v. Human in Games



- Computers can explore a large search space of moves quickly
  - How many moves possible in chess, for example?
- Computers cannot explore every move (why) so must use heuristics
  - Rules of thumb about position, strategy, board evaluation
  - Try a move, undo it and try another, track the best move
- What do humans do well in these games? What about computers?
  - What about at Duke?

Games at Duke

- Alan Biermann
  - Natural language processing
  - Compsci 1: Great Ideas
  - Duchess, checkers, chess
- Tom Truscott
  - Duke undergraduate working with/for Biermann
  - Usenet: online community
- Second EFF Pioneer Award (with Vint Cerf!)

Heuristics

- A heuristic is a rule of thumb, doesn't always work, isn't guaranteed to work, but useful in many/most cases
    - Search problems that are "big" often can be approximated or solved with the right heuristics

- What heuristic is good for Sudoku?
    - Is there always a no-reasoning move, e.g., 5 goes here?
    - What about "if I put a 5 here, then…"
    - Do something else? http://en.wikipedia.org/wiki/Algorithmics_of_sudoku

- What other optimizations/improvements can we make?
    - For chess, checkers: good heuristics, good data structures

Barbara Liskov

- First woman to earn PhD from compsci dept
  - Stanford
- Turing award in 2008
  - OO, SE, PL

"It's much better to go for the thing that's exciting. But the question of how you know what's worth working on and what's not separates someone who's going to be really good at research and someone who's not. There's no prescription. It comes from your own intuition and judgment."