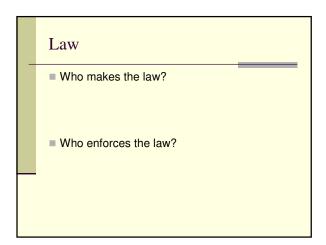


# Law "Equal justice under law" http://en.wikipedia.org/wiki/File:CourtEqualJustice.JPG Which building is that?



# Right to a Fair Trial Sixth Amendment: "In all criminal prosecutions, the accused shall enjoy the right to a speedy and public trial, by an impartial jury of the State and district where in the crime shall have been committed ..."

# Fair Trial Presumed innocence until proven guilty Burden of proof is on the prosecution Trial by jury (peers)

# Attorneys Why do we need attorneys to represent clients (the accused)?

### Attorneys

- Why do we need attorneys to represent clients (the accused)?
- Fifth Amendment:
  - "...nor shall be compelled in any criminal case to be a witness against himself..."

### Attorneys

- Why do we need attorneys to represent clients (the accused)?
- Fifth Amendment:
  - "...nor shall be compelled in any criminal case to be a witness against himself..."
- Miranda Rights:
  - "You have the right to remain silent. Anything you say can and will be used against you in a court of law. You have the right to an attorney. If you cannot afford an attorney, one will be appointed to you. ..."

# Attorney Scheduling

Problem 1

### **Attorney Scheduling**

- Real problem for Senior Projects
- http://www.cs.fit.edu/~pkc/classes/seniorProjects/opportunities/attorneyScheduling.pdf

### **Problem Formulation**

- Given (input)
- Find (output)
- Simplification

### **Problem Formulation**

- Given (input)
  - Judge/case schedule
  - Attorney availability
  - Constraints—no time conflicts
- Find (output)
- Simplification

### **Problem Formulation**

- Given (input)
  - Judge/case schedule
  - Attorney availability
  - Constraints—no time conflicts
- Find (output)
  - Attorney schedule
- Simplification
  - Days instead of hours. Specialty, even load
    - ... are ignored

	Mon	Tue	Wed	Thu	Fri
John	Case A			Case B	
Jane	Case C		Case D		
Jack			Case E		
Attorney	Mon	Tue	Wed	Thu	Fri
Alice	available	available	available	available	
<u>A</u> ndy	available	available	available	available	

### First Available Attorney

For each case, schedule the first available attorney

### First available attorney

Judge	Mon	Tue	Wed	Thu	Fri
John	Case A			Case B	
Jane	Case C		Case D		
Jack			Case E		
Attorney	Mon	Tue	Wed	Thu	Fri
Alice	Case A	available	available	available	
<u>A</u> ndy	available	available	available	available	

### First available attorney

Mon	Tue	Wed	Thu	Fri
Case A			Case B	
Case C		Case D		
		Case E		
Mon	Tue	Wed	Thu	Fri
Case A	available	available	Case B	
available	available	available	available	
	Case A Case C  Mon Case A	Case A Case C  Mon Tue Case A available	Case A         Case D           Case C         Case D           Case E         Wed           Case A         available	Case A         Case B           Case C         Case D           Case E         Case E    Mon Tue Wed Thu  Case A available available Case B

### First available attorney

Judge	Mon	Tue	Wed	Thu	Fri
John	Case A			Case B	
Jane	Case C		Case D		
Jack			Case E		
Attorney	Mon	Tue	Wed	Thu	Fri
Attorney Alice	Mon Case A	Tue available	Wed available	Thu Case B	Fri
	-				Fri

Fire	st avail	able at	torney		
Judge	Mon	Tue	Wed	Thu	Fri
John	Case A			Case B	
Jane	Case C		Case D		
Jack			Case E		
Attorney	Mon	Tue	Wed	Thu	Fri
Alice	Case A	available	Case D	Case B	
<u>A</u> ndy	Case C	available	available	available	

Judge	Mon	Tue	Wed	Thu	Fri
John	Case A			Case B	
Jane	Case C		Case D		
Jack			Case E		
Attorney	Mon	Tue	Wed	Thu	Fri
Alice	Case A	available	Case D	Case B	
<u>A</u> ndy	Case C	available	Case E	available	

# Cases with more than one day

We use day as a time unit for simplicity
Each time unit could be:
<ul><li>an hour</li></ul>
<ul><li>morning/afternoon</li></ul>

Judge	Mon	Tue	Wed	Thu	Fri
John			Case A		
Jane	Case B	Case B			
Jack			Case C	Case C	
Attorney	Mon	Tue	Wed	Thu	Fri
Alice	available	available	available	available	
<u>An</u> dy	available	available	available		

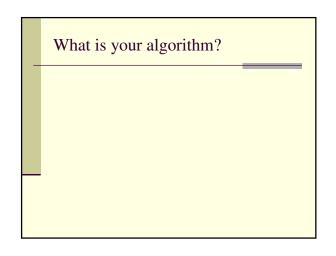
Fir	rst avail	lable at	torney		
Judge	Mon	Tue	Wed	Thu	Fri
John			Case A		
Jane	Case B	Case B			
Jack			Case C	Case C	
Attorney	Mon	Tue	Wed	Thu	Fri
Alice	available	available	Case A	available	
<u>An</u> dy	available	available	available		

Fir	st avail	able at	torney		
Judge	Mon	Tue	Wed	Thu	Fri
John			Case A		
Jane	Case B	Case B			
Jack			Case C	Case C	
Attorney	Mon	Tue	Wed	Thu	Fri
Alice	Case B	Case B	Case A	available	
<u>An</u> dy	available	available	available		
•					

### First available attorney Mon Judge Tue Wed Thu Fri John Case A Jane Case B Case B Jack Case C Case C Attorney Mon Wed Thu Fri Tue Alice Case B Case B Case A available <u>A</u>ndy available available available Cannot find an attorney for Case C

	Fire	st avail	lable at	torney		
	Judge	Mon	Tue	Wed	Thu	Fri
	John			Case A		
	Jane	Case B	Case B			
	Jack			Case C	Case C	
١,			-	\A/l	Thu	Fri
Ш	Attorney	Mon	Tue	Wed	iiiu	' ' '
	Attorney Alice	Mon Case B	Case B	Case A	available	111

Judge	Mon	Tue	Wed	Thu	Fri
John			Case A		
Jane	Case B	Case B			
Jack			Case C	Case C	
Attorney	Mon	Tue	Wed	Thu	Fri
Alice	Case B	Case B	Case C	Case C	
<u>A</u> ndy	available	available	Case A		



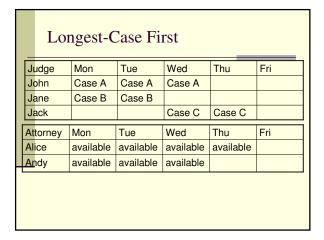
# Longest-Case First Sort the cases by length in descending order For each case, schedule the next available attorney

	Loi	ngest-C	Case Fi	rst		
	Judge	Mon	Tue	Wed	Thu	Fri
	John			Case A		
	Jane	Case B	Case B			
	Jack			Case C	Case C	
Ī	Attorney	Mon	Tue	Wed	Thu	Fri
	Alice	Case B	Case B	available	available	
Ц	<u>An</u> dy	available	available	available		
-						

Judge	Mon	Tue	Wed	Thu	Fri
John			Case A		
Jane	Case B	Case B			
Jack			Case C	Case C	
Attorney	Mon	Tue	Wed	Thu	Fri
Alice	Case B	Case B	Case C	Case C	
<u>A</u> ndy	available	available	available		

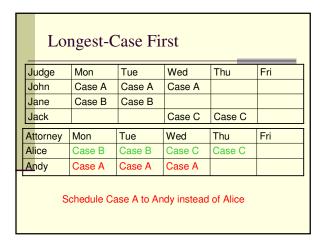
ngest-C	Case Fi	rst		
Mon	Tue	Wed	Thu	Fri
		Case A		
Case B	Case B			
		Case C	Case C	
Mon	Tue	Wed	Thu	Fri
Case B	Case B	Case C	Case C	
	aailabla	Case A		
available	available	Case A		
available	avaliable	Case A		
available	avallable	Case A		
	Mon Case B Mon	Mon Tue  Case B Case B  Mon Tue	Case A   Case B   Case C	Mon         Tue         Wed         Thu           Case A         Case A         Case B           Case B         Case C         Case C           Mon         Tue         Wed         Thu

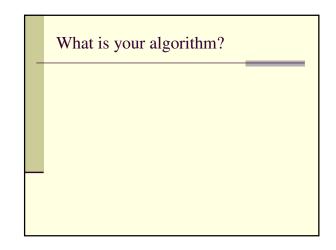
# Longest-Case First Also does not guarantee finding a solution if a solution exists



Lo	ngest-C	Case Fi	rst		
Judge	Mon	Tue	Wed	Thu	Fri
John	Case A	Case A	Case A		
Jane	Case B	Case B			
Jack			Case C	Case C	
Attorney	Mon	Tue	Wed	Thu	Fri
Alice	Case A	Case A	Case A	available	
<u>An</u> dy	available	available	available		
	•				

	ngest-(	Case Fi			
Judge	Mon	Tue	Wed	Thu	Fri
John	Case A	Case A	Case A		
Jane	Case B	Case B			
Jack			Case C	Case C	
Attorney	Mon	Tue	Wed	Thu	Fri
Alice	Case A	Case A	Case A	available	
Andy	Case B	Case B	available		
Canno	t find an at	torney for (	Case C, but	a solution	exists!





### Longest Case, Least Available Attorney Consider the longest case first

- Consider the least available attorney first

Loi	Longest Case, Least Available				
Att	orney				
Judge	Mon	Tue	Wed	Thu	Fri
John	Case A	Case A	Case A		
Jane	Case B	Case B			
Jack			Case C	Case C	
Attorney	Mon	Tue	Wed	Thu	Fri
Alice	Case A	Case A	Case A	available	
<u>A</u> ndy	available	available	available		available
	•	•		•	

### Longest Case, Least Available Attorney Judge Wed

Judge	IVIOII	i uc	· · · · · ·	11110	
John	Case A	Case A	Case A		
Jane	Case B	Case B			
Jack			Case C	Case C	
Attorney	Mon	Tue	Wed	Thu	Fri
Attorney Alice	Mon Case A	Tue Case A	Wed Case A	Thu available	Fri
					Fri available

Cannot find an attorney for Case C, but a solution exists

What is the moral of the story?

# What is the moral of the story? The first solution might not work

### What is the moral of the story?

- The first solution might not work
- It's NOT about "coding"

### What is the moral of the story?

- The first solution might not work
- It's NOT about "coding"
- Consider a counter example that breaks your solution
  - Do you need more than one counter example?

### What is the moral of the story?

- The first solution might not work
- It's NOT about "coding"
- Consider a counter example that breaks your solution
  - Do you need more than one counter example?
- Note that we started with more simplified problems first

### What is your algorithm?

### Pseudocode

- returns schedule, empty (empty schedule, no cases), or no solution (no valid schedule)
- scheduling(cases, attorneys)
  - if more cases
    - select a case C
    - schedule = no solution
    - while schedule is no solution & can select attorney A for C
      - schedule = scheduling(remainingCases, remainingAttorneys)
      - if schedule has a solution (not no solution)
      - add C:A to schedule
  - else
  - schedule = empty
  - return schedule

### Not all cases/attorneys are equal

- Consider certain cases/attorneys before others
  - \*could\* (not will) find the schedule faster
- Case ordering
  - Node ordering
- Attorney ordering
  - Branch ordering

### Case/Node Ordering

■ Which case should we consider first?

### Case/Node Ordering

■ Most difficult case first, why?

### Case/Node Ordering

- Most difficult case first, why?
  - if later, fewer attorneys left, more likely to get stuck

### Case/Node Ordering

- Most difficult case first
  - if later, fewer attorneys left, more likely to get stuck
- Longest case

### Case/Node Ordering

- Most difficult case first
  - if later, fewer attorneys left, more likely to get stuck
- Longest case
- What about a short case that only one attorney can be scheduled?

### Case/Node Ordering

- More difficult case first
  - if later, fewer attorneys left, more likely to get
- Longer cases
- What about a short case that only one attorney can be scheduled?
- Case/node with fewest attorneys/branches first ("the most constraining case")

### Attorney/Branch Ordering

■ Which attorney should we consider first?

### Attorney/Branch Ordering

- Which attorney should we consider first?
- Attorney results the most attorneys for the other cases
  - Consider Cases A and B
    - Schedule Alice to A =>
      - 0 attorney left for B
    - Schedule Andy to A =>
      - 1 attorney left for B

### Attorney/Branch Ordering

- Which attorney should we consider first?
- Attorney results the most attorneys for the other cases
  - Consider cases A and B
    - Schedule Alice to A =>
      - 0 attorney for B
    - Schedule Andy to A =>
      - 1 attorney for B
- Order Andy before Alice ("the least constraining attorney")

### Attorney/Branch Ordering

Case A	Case B	Case C	Case D	Case E
1	Alice?	1	3	3
4	Andy?	4	4	0
2	Amy?	2	2	3

- The numbers are available attorneys after an attorney is assigned to Case B
- How would you order them and why?

### Obvious Scenarios with No Solutions

Check before trying to schedule:

### Obvious Scenarios with No Solutions

- Check before trying to schedule:
  - For each day, number of attorneys is fewer that number of cases

### Obvious Scenarios with No Solutions

- Check before trying to schedule:
  - For each day, number of attorneys is fewer that number of cases
  - A case that no attorney can be scheduled
    - e.g. The case is Monday thru Friday, but none of the attorneys are available five days in a row

### Actions for no Solutions

- No solutions:
  - "Obvious": found before scheduling
  - "Not Obvious": found during scheduling
- Actions:

### Actions for no Solutions

- No solutions:
  - "Obvious": found before scheduling
  - "Not Obvious": found during scheduling
- Actions:
  - Report to user and exit
  - Suggest the user to reduce cases and/or increase attorneys
  - Output partial schedule

### Partial Schedule

What properties are preferred?

### Partial Schedule

Maximize the number of scheduled cases

### Partial Schedule

- Maximize the number of scheduled cases
  - "longest path", weight = 1

### Partial Schedule

- Maximize the number of scheduled cases
  - "longest path", weight = 1
- Maximize the total length of scheduled cases

### Partial Schedule

- Maximize the number of scheduled cases
  - "longest path", weight = 1
- Maximize the total length of scheduled cases
  - "longest path", weight = case length

### Implementation: Key Operations?

- returns schedule, empty (empty schedule, no cases), or no solution (no valid schedule)
- scheduling(cases, attorneys)
  - if more cases
    - select a case C
    - schedule = no solution
    - while schedule is no solution & can select attorney A for C
      - schedule = scheduling(remainingCases,
        remainingAttorneys)
      - if schedule has a solution (not no solution)
        - add C:A to schedule
  - else
    - schedule = empty
  - return schedule

### Implementation: Key Operations?

- returns schedule, empty (empty schedule, no cases), or no solution (no valid schedule)
- scheduling(cases, attorneys)
  - if more cases
    - select a case C
    - schedule = no solution
    - while schedule is no solution & can select attorney A for C
      - schedule = scheduling(remainingCases, remainingAttorneys)

      - if schedule has a solution (not no solution) add C:A to schedule

  - schedule = empty
  - return schedule

### Selecting a case or attorney

- Selecting a case
  - Most-constraining case
- Selecting an attorney
  - Least-constraining attorney
- Both involve one key operation, what is it?

### Selecting a case or attorney

- Selecting a case
  - Most-constraining case
- Selecting an attorney
  - Least-constraining attorney
- Both involve one key operation
  - Finding available attorneys for each case
    - Comparison of time segments of attorneys for each case

### **Comparing Time Segments**

- Case X: Mon, Tue
  - Alice: Mon, Tue, Wed, Thu
  - Andy: Mon, Wed, Fri
  - Amy: Mon, Tue, Thu, Fri

### Which is More Important?

- Efficient data structure for
  - Case schedule
  - Attorney availability?

# Data Structure for Attorney Availability

### Data Structure for Attorney Availability

- 2D-array (table similar to previous slides)
  - Column=day, row=attorney, cell=available

### Data Structure for Attorney Availability

- 2D-array (table similar to previous slides)
  - Column=day, row=attorney, cell=available
- Interval
  - Start day, end day

### Data Structure for Attorney Availability

- 2D-array (table similar to previous slides)
  - Column=day, row=attorney, cell=available
- Interval
  - Start day, end day
  - Indexed by start day: (attorney, end day)

## Data Structure for Attorney Availability

- 2D-array (table similar to previous slides)
  - Column=day, row=attorney, cell=available
- Interva
  - Start day, end day
  - Indexed by start day: (attorney, end day)
- Duration
  - Start day, duration

## Data Structure for Attorney Availability

- 2D-array (table similar to previous slides)
  - Column=day, row=attorney, cell=available
- Interval
  - Start day, end day
  - Indexed by start day: (attorney, end day)
- Duration
  - Start day, duration
  - Indexed by start day: (attorney, duration)

### Data Structure for Attorney Availability

- 2D-array (table similar to previous slides)
  - Column=day, row=attorney, cell=available
- Interval
  - Start day, end day
  - Indexed by start day: (attorney, end day)
- Duration
  - Start day, duration
  - Indexed by start day: (attorney, duration)
  - Indexed by (start day, duration): attorney

### Designing Tables in Databases

- Consider storing attorney availability in a DB
  - Does the data structure discussion affect how you would design the DB tables?

### Designing Tables in Databases

- Consider storing attorney availability in a DB
  - Does the data structure discussion affect how you would design the DB tables?
- What is the moral of the story?

### Designing Tables in Databases

- Consider storing attorney availability in a DB
  - Does the data structure discussion affect how you would design the DB tables?
- What is the moral of the story?
  - Considering the key operations is important

## Multiple Segments in Attorney Availability

- Assume days in cases are consecutive
- Days in attorney availability might not be consecutive

## Multiple Segments in Attorney Availability

- Assume days in cases are consecutive
- Days in attorney availability might not be consecutive
  - Initial Andy's availability: Mon-Fri
  - Case: Wed-Thu
  - Updated Andy's availability: Mon-Tue, Fri
- Can the data structure accommodate it?

## Multiple Segments in Attorney Availability

- Assume days in cases are consecutive
- Days in attorney availability might not be consecutive
  - Initial Andy's availability: Mon-Fri
  - Case: Wed-Thu
  - Updated Andy's availability: Mon-Tue, Fri
- Can the data structure accommodate it?
  - Mon: (Andy, 2)
  - Fri: (Andy, 1)

### Checking Attorney Availability

- Case starts on Mon
  - Do you want to check availability starting:
    - Tue, ..., Fri?

### Checking Attorney Availability

- Case starts on Mon
  - Do you want to check availability starting:
    - Tue, ..., Fri? No
- Case starts on Wed
  - Do you want to check availability starting:
    - Before Wed: Mon &Tue?
    - After Wed: Thu & Fri?

# Checking Attorney Availability

- Case starts on Mon
  - Do you want to check availability starting:
    - Tue, ..., Fri? No
- Case starts on Wed
  - Do you want to check availability starting:
    - Before Wed: Mon &Tue? Yes
    - After Wed: Thu & Fri? No

### Checking Attorney Availability

- Case: starts on Wed, length 1 (ie Wed only)
  - Mon: (Andy, 3)
  - Tue: (Alice, 3)
  - Wed: (Amy, 3)
  - Who do you prefer and why--attorney/branch ordering?

### Multiple Time Segments

- Do we prefer
  - Fewer segments
  - More segments

### Multiple Time Segments

- Do we prefer
  - Fewer segments
  - More segments
- Prefer fewer longer segments

### Data Structure for Case Schedule

### Data Structure for Case Schedule

- 2D array (like previous slides)
  - row=judge, column=day, cell=caseID

### Data Structure for Case Schedule

- 2D array (like previous slides)
  - row=judge, column=day, cell=caseID
- Interval
  - start day, end day

### Data Structure for Case Schedule

- 2D array (like previous slides)
  - row=judge, column=day, cell=caseID
- Interval
  - start day, end day
  - array of (start day, end day, caseID, judge)

### Data Structure for Case Schedule

- 2D array (like previous slides)
  - row=judge, column=day, cell=caseID
- Interval
  - start day, end day
  - array of (start day, end day, caseID, judge)
- Duration
  - start day, duration

### Data Structure for Case Schedule

- 2D array (like previous slides)
  - row=judge, column=day, cell=caseID
- Interval
  - start day, end day
  - array of (start day, end day, caseID, judge)
- Duration
  - start day, duration
  - array of (start day, duration, caseID, judge)

### Additional Constraints/Preferences

### Additional Constraints/Preferences

■ Time unit in hours instead of days

### Additional Constraints/Preferences

- Time unit in hours instead of days
  - More "columns" in our tables for input

### Additional Constraints/Preferences

- Time unit in hours instead of days
  - More "columns" in our tables for input
- Specialty in certain cases

### Additional Constraints/Preferences

- Time unit in hours instead of days
  - More "columns" in our tables for input
- Specialty in certain cases
  - Only consider those attorneys (fewer branches)

### Additional Constraints/Preferences

- Time unit in hours instead of days
  - More "columns" in our tables for input
- Specialty in certain cases
  - Only consider those attorneys (fewer branches)
- Prefer certain judges

### Additional Constraints/Preferences

- Time unit in hours instead of days
  - More "columns" in our tables for input
- Specialty in certain cases
  - Only consider those attorneys (fewer branches)
- Prefer certain judges
  - Consider those attorneys first (branch ordering)

### Additional Constraints/Preferences

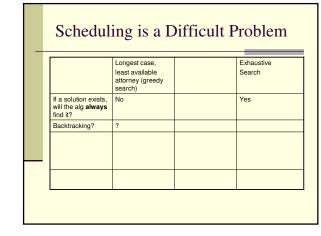
- Time unit in hours instead of days
  - More "columns" in our tables for input
- Specialty in certain cases
  - Only consider those attorneys (fewer branches)
- Prefer certain judges
  - Consider those attorneys first (branch ordering)
- More even workload for each attorney

### Additional Constraints/Preferences

- Time unit in hours instead of days
  - More "columns" in our tables for input
- Specialty in certain cases
  - Only consider those attorneys (fewer branches)
- Prefer certain judges
  - Consider those attorneys first (branch ordering)
- More even workload for each attorney
  - Consider attorneys with lighter load first (branch ordering)

	Longest case,	Exhaustive
	least available attorney (greedy search)	Search
If a solution exists, will the alg <b>always</b> find it?	?	

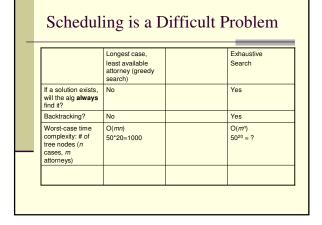
# 



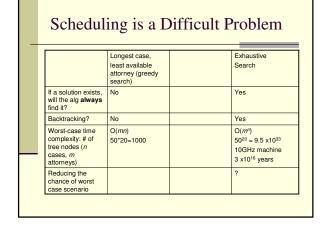
## 

If a solution exists, No	
will the alg always find it?	Yes
Backtracking? No	Yes
Worst-case time complexity: # of tree nodes (n cases, m attorneys)	

### 



	I I	le contra
	Longest case, least available attorney (greedy search)	Exhaustive Search
If a solution exists, will the alg <b>always</b> find it?	No	Yes
Backtracking?	No	Yes
Worst-case time complexity: # of tree nodes (n cases, m attorneys)	O(mn) 50*20=1000	$O(m^{0})$ $50^{20} = 9.5 \times 10^{33}$ 10GHz machine ? years



	Ι.		1=
	Longest case, least available attorney (greedy search)	?	Exhaustive Search
If a solution exists, will the alg <b>always</b> find it?	No		Yes
Backtracking?	No		Yes
Worst-case time complexity: # of tree nodes (n cases, m attorneys)	O( <i>mn</i> ) 50*20=1000		O(m²) 50²0 = 9.5 x10³3 10GHz machine 3 x10¹6 years
Reducing the chance of worst case scenario			case/node & attorney/branch ordering is generally effective

	Beam Search
_	<ul> <li>At each level</li> <li>only top k paths are selected (to be explored to the next level)</li> </ul>

## Beam Search

- At each level
  - only top k paths are selected (to be explored to the next level)
- Greedy
  - k = ?

### Beam Search

- At each level
  - only top k paths are selected (to be explored to the next level)
- Greedy
- k = 1
- Exhaustive
  - k = ?

### Beam Search

- At each level
  - only top k paths are selected (to be explored to the next level)
- Greedy
  - k = 1
- Exhaustive
  - k = k = max

### Scheduling is a Difficult Problem

		Longest case, least available attorney (greedy search)	Beam Search	Exhaustive Search
	tion exists, alg <b>always</b>	No	?	Yes
Backtrad	cking?	No		Yes
Worst-ca complex tree nod cases, n attorney	les (n n	O( <i>mn</i> ) 50*20=1000		O(m <sup>n</sup> ) 50 <sup>20</sup> = 9.5 x10 <sup>33</sup> 10GHz machine 3 x10 <sup>16</sup> years
Reducin chance case sce	of worst			case/node & attorney/branch ordering is generally effective

### Scheduling is a Difficult Problem

_				
		Longest case, least available attorney (greedy search)	Beam Search	Exhaustive Search
	If a solution exists, will the alg <b>always</b> find it?	No	No, but node/branch ordering helps	Yes
	Backtracking?	No	?	Yes
	Worst-case time complexity: # of tree nodes (n cases, m attorneys)	O( <i>mn</i> ) 50*20=1000		O(m <sup>n</sup> ) 50 <sup>20</sup> = 9.5 x10 <sup>33</sup> 10GHz machine 3 x10 <sup>16</sup> years
	Reducing the chance of worst case scenario			case/node & attorney/branch ordering is generally effective

### Scheduling is a Difficult Problem

	Longest case, least available attorney (greedy search)	Beam Search	Exhaustive Search
If a solution exists, will the alg <b>always</b> find it?	No	No, but node/branch ordering helps	Yes
Backtracking?	No	No	Yes
Worst-case time complexity: # of tree nodes (n cases, m attorneys)	O(mn) 50*20=1000	?	O(m <sup>n</sup> ) 50 <sup>20</sup> = 9.5 x10 <sup>33</sup> 10GHz machine 3 x10 <sup>16</sup> years
Reducing the chance of worst case scenario			case/node & attorney/branch ordering is generally effective

_				Problem
		Longest case, least available attorney (greedy search)	Beam Search	Exhaustive Search
	If a solution exists, will the alg <b>always</b> find it?	No	No, but node/branch ordering helps	Yes
	Backtracking?	No	No	Yes
	Worst-case time complexity: # of tree nodes (n cases, m attorneys)	O( <i>mn</i> ) 50*20=1000	O( <i>kmn</i> ) 100*50*20=100K	$O(m^{0})$ $50^{20} = 9.5 \times 10^{33}$ 10 GHz machine $3 \times 10^{16}$ years
	Reducing the chance of worst case scenario			case/node & attorney/branch ordering is generally effective

