A little background on computer science

Topics of Study in Computer Science
- Engineering Computer Systems
  - computer architecture
  - operating systems and distributed systems
  - databases
  - programming languages
  - software engineering
  - networking
A little background on computer science

- Algorithms
  - design and analysis of algorithms
  - computational complexity
  - fundamental limitations of computation

- Modeling and Interaction
  - computer-human interaction
  - social computing
  - ubiquitous computing
  - artificial intelligence (including speech, vision, learning, natural language, robotics)
  - computer graphics
Computer “Science” and Real Science

- Computational Science: high-performance computing for large-scale computations
  - computational physics
  - computational chemistry
- Data Exploration Science
  - bio-informatics
  - computational astronomy
  - ecosystem informatics
 Contributions of Computer Science to Real Science

- High-performance computing
  - quantum computations
  - n-body problems
  - observable consequences of unified theories
- Algorithms
  - assembly of genome from shotgun sequencing
  - similarity matching of DNA and protein sequences
- Data management and data mining
  - web access to data (including GIS)
  - finding patterns in data
  - semantic web
- Visualization and Graphics
- Model construction and analysis
  - software engineering of models
  - calibration, computation, sensitivity analysis, uncertainty
Unit 3

Introduce Bayesian Networks and Influence Diagrams

- Language for representing decision-making problems
- Algorithms for solving these problems
- Statistical “learning” methods for fitting probability models
Probability Review

Random Variables
- Boolean: \( W_{1,2} \) Two possible values \{true, false\}
- Discrete: Weather \( \in \{\text{sunny, cloudy, rainy, snow}\} \)
- Continuous: Temperature \( \in \mathbb{R} \)

Propositions (also called "events")
- \( W_{1,2} = \text{true}, \text{Weather} = \text{sunny}, \text{Temperature} = \text{65} \)
Consider a car described by 3 random variables:

- Gas $\in \{\text{true, false}\}$: There is gas in the tank
- Meter $\in \{\text{empty, full}\}$: The gas gauge shows the tank is empty or full
- Starts $\in \{\text{yes, no}\}$: The car starts when you turn the key in the ignition
Joint Probability Distribution

- Each row is called a “primitive event”
- Rows are mutually exclusive and exhaustive
- Corresponds to an “8-sided coin” with the indicated probabilities

<table>
<thead>
<tr>
<th>Gas</th>
<th>Meter</th>
<th>Starts</th>
<th>P(G,M,S)</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>empty</td>
<td>no</td>
<td>0.1386</td>
</tr>
<tr>
<td>false</td>
<td>empty</td>
<td>yes</td>
<td>0.0014</td>
</tr>
<tr>
<td>false</td>
<td>full</td>
<td>no</td>
<td>0.0594</td>
</tr>
<tr>
<td>false</td>
<td>full</td>
<td>yes</td>
<td>0.0006</td>
</tr>
<tr>
<td>true</td>
<td>empty</td>
<td>no</td>
<td>0.0240</td>
</tr>
<tr>
<td>true</td>
<td>empty</td>
<td>yes</td>
<td>0.0560</td>
</tr>
<tr>
<td>true</td>
<td>full</td>
<td>no</td>
<td>0.2160</td>
</tr>
<tr>
<td>true</td>
<td>full</td>
<td>yes</td>
<td>0.5040</td>
</tr>
</tbody>
</table>
Any Query Can Be Answered from the Joint Distribution

- $P(\text{Gas} = \text{false} \land \text{Meter} = \text{full} \land \text{Starts} = \text{yes}) = 0.0006$
- $P(\text{Gas} = \text{false}) = 0.2$, this is the sum of all cells where Gas = false
- In general: To compute $P(Q)$, for any proposition $Q$, add up the probability in all cells where $Q$ is true

<table>
<thead>
<tr>
<th>Gas</th>
<th>Meter</th>
<th>Starts</th>
<th>$P(G,M,S)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>empty</td>
<td>no</td>
<td>0.1386</td>
</tr>
<tr>
<td>false</td>
<td>empty</td>
<td>yes</td>
<td>0.0014</td>
</tr>
<tr>
<td>false</td>
<td>full</td>
<td>no</td>
<td>0.0594</td>
</tr>
<tr>
<td>false</td>
<td>full</td>
<td>yes</td>
<td>0.0006</td>
</tr>
<tr>
<td>true</td>
<td>empty</td>
<td>no</td>
<td>0.0240</td>
</tr>
<tr>
<td>true</td>
<td>empty</td>
<td>yes</td>
<td>0.0560</td>
</tr>
<tr>
<td>true</td>
<td>full</td>
<td>no</td>
<td>0.2160</td>
</tr>
<tr>
<td>true</td>
<td>full</td>
<td>yes</td>
<td>0.5040</td>
</tr>
</tbody>
</table>
Notations

- $P(G,M,S)$ denotes the entire joint distribution. It is a table or function that maps from $G$, $M$, and $S$ to a probability.
- $P(\text{true, empty, no})$ denotes a single probability value:
  
  $$P(\text{Gas=true} \land \text{Meter=empty} \land \text{Starts=no})$$
Operations on Probability Tables

(1)

Marginalization
("summing away")

\[ \sum_{M,S} P(G,M,S) = P(G) \]

\( P(G) \) is called a “marginal probability” distribution.
It consists of two probabilities:

<table>
<thead>
<tr>
<th>Gas</th>
<th>P(G)</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>( P(\text{false,empty,\text{yes}}) + )  |</td>
</tr>
<tr>
<td></td>
<td>( P(\text{false,empty,\text{no}}) + )  |</td>
</tr>
<tr>
<td></td>
<td>( P(\text{false,full,\text{yes}}) + )  |</td>
</tr>
<tr>
<td></td>
<td>( P(\text{false,full,\text{no}}) )</td>
</tr>
<tr>
<td>true</td>
<td>( P(\text{true,empty,\text{yes}}) + )  |</td>
</tr>
<tr>
<td></td>
<td>( P(\text{true,empty,\text{no}}) + )  |</td>
</tr>
<tr>
<td></td>
<td>( P(\text{true,full,\text{yes}}) + )  |</td>
</tr>
<tr>
<td></td>
<td>( P(\text{true,full,\text{no}}) )</td>
</tr>
</tbody>
</table>
Conditional Probability

- Suppose we observe that \( M = \text{full} \). What is the probability that the car will start?
  \[ P(S = \text{yes} \mid M = \text{full}) \]
- Definition: \( P(A \mid B) = \frac{P(A \cap B)}{P(B)} \)

\[
P(S = \text{yes} \mid M = \text{full}) = \frac{P(S = \text{yes} \cap M = \text{full})}{P(M = \text{full})} = \frac{0.5040 + 0.0006}{0.5040 + 0.0006 + 0.2160 + 0.0594} = 0.6469
\]
Conditional Probability

- Select cells that match the condition (M=full)
- Delete remaining cells and M column
- Renormalize the table to obtain $P(S,G|M=\text{full})$
- Sum away Gas: $\sum_G P(S,G \mid M=\text{full}) = P(S\mid M=\text{full})$
- Read answer from $P(S=\text{yes} \mid M=\text{full})$ cell

<table>
<thead>
<tr>
<th>Gas</th>
<th>Meter</th>
<th>Starts</th>
<th>$P(G,M,S)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>full</td>
<td>no</td>
<td>0.0594</td>
</tr>
<tr>
<td>false</td>
<td>full</td>
<td>yes</td>
<td>0.00006</td>
</tr>
<tr>
<td>true</td>
<td>full</td>
<td>no</td>
<td>0.2160</td>
</tr>
<tr>
<td>true</td>
<td>full</td>
<td>yes</td>
<td>0.5040</td>
</tr>
<tr>
<td>true</td>
<td>full</td>
<td>no</td>
<td>0.2160</td>
</tr>
<tr>
<td>true</td>
<td>full</td>
<td>yes</td>
<td>0.5040</td>
</tr>
</tbody>
</table>
Operations on Probability Tables (2): Conditionalizing

Construct $P(G, S | M)$ by normalizing the subtable corresponding to $M=\text{full}$ and normalizing the subtable corresponding to $M=\text{empty}$

<table>
<thead>
<tr>
<th>Gas</th>
<th>Meter</th>
<th>Starts</th>
<th>$P(G, M, S)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>empty</td>
<td>no</td>
<td>0.0064</td>
</tr>
<tr>
<td>false</td>
<td>empty</td>
<td>yes</td>
<td>0.6300</td>
</tr>
<tr>
<td>false</td>
<td>full</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>false</td>
<td>full</td>
<td>yes</td>
<td></td>
</tr>
<tr>
<td>true</td>
<td>empty</td>
<td>no</td>
<td>0.1091</td>
</tr>
<tr>
<td>true</td>
<td>empty</td>
<td>yes</td>
<td>0.2545</td>
</tr>
<tr>
<td>true</td>
<td>full</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>true</td>
<td>full</td>
<td>yes</td>
<td></td>
</tr>
</tbody>
</table>
Chain Rule of Probability

\[ P(A,B,C) = P(A|B,C) \cdot P(B|C) \cdot P(C) \]

Proof:

\[
P(A, B, C) = \frac{P(A, B, C)}{P(B, C)} \cdot P(B, C)
\]

\[
= P(A | B, C) \cdot P(B, C)
\]

\[
= P(A | B, C) \cdot \frac{P(B, C)}{P(C)} \cdot P(C)
\]

\[
= P(A | B, C) \cdot P(B | C) \cdot P(C)
\]
Chain Rule (2)

Holds for distributions too:
\[ P(A, B, C) = P(A | B, C) \cdot P(B | C) \cdot P(C) \]

This means that for each setting of A, B, and C, we can substitute into the equation, and it is true.
Belief Networks (1): Independence

Defn: Two random variables $X$ and $Y$ are \textit{independent} iff
$$P(X,Y) = P(X) \cdot P(Y)$$

Example:
- $X$ is a coin with $P(X=\text{heads}) = 0.4$
- $Y$ is a coin with $P(Y=\text{heads}) = 0.8$
- Joint distribution:

<table>
<thead>
<tr>
<th>$P(X,Y)$</th>
<th>$X=\text{heads}$</th>
<th>$X=\text{tails}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Y=\text{heads}$</td>
<td>0.32</td>
<td>0.48</td>
</tr>
<tr>
<td>$Y=\text{tails}$</td>
<td>0.08</td>
<td>0.12</td>
</tr>
</tbody>
</table>
Belief Networks (2)
Conditional Independence

Defn: Two random variables $X$ and $Y$ are *conditionally independent* given $Z$ iff

$$P(X,Y \mid Z) = P(X \mid Z) \cdot P(Y \mid Z)$$

Example:

$$P(S,M \mid G) = P(S \mid G) \cdot P(M \mid G)$$

Intuition: $G$ independently causes $S$ and $M$
Operations on Probability Tables (3): Conformal Product

Allocate space for resulting table and then fill in each cell with the product of the corresponding cells:

\[ P(S,M | G) = P(S | G) \cdot P(M | G) \]

<table>
<thead>
<tr>
<th>S</th>
<th>M</th>
<th>G=no</th>
<th>G=yes</th>
</tr>
</thead>
<tbody>
<tr>
<td>no</td>
<td>empty</td>
<td>a00·b00</td>
<td>a01·b01</td>
</tr>
<tr>
<td>no</td>
<td>full</td>
<td>a00·b10</td>
<td>a01·b11</td>
</tr>
<tr>
<td>yes</td>
<td>empty</td>
<td>a10·b00</td>
<td>a11·b01</td>
</tr>
<tr>
<td>yes</td>
<td>full</td>
<td>a10·b10</td>
<td>a11·b11</td>
</tr>
</tbody>
</table>

\[
\begin{array}{c|c|c}
S & G=no & G=yes \\
\hline
\text{no} & a00 & a01 \\
\text{yes} & a10 & a11 \\
\end{array}
\]

\[
\begin{array}{c|c|c}
M & G=no & G=yes \\
\hline
\text{empty} & b00 & b01 \\
\text{full}  & b10 & b11 \\
\end{array}
\]
Properties of Conformal Products

- Commutative
- Associative
- Work on normalized or unnormalized tables
- Work on joint or conditional tables
Conditional Independence Allows Us to Simplify the Joint Distribution

\[ P(G, M, S) = P(M, S | G) \cdot P(G) \quad [\text{chain rule}] \]

\[ = P(M | G) \cdot P(S | G) \cdot P(G) \quad [\text{CI}] \]
Bayesian Networks

- One node for each random variable
- Each node stores a probability distribution $P(\text{node} \mid \text{parents(\text{node})})$
- Only direct dependencies are shown
- Must not contain directed cycles
- Joint distribution is conformal product of node distributions:
  $P(\text{G}, \text{M}, \text{S}) = P(\text{G}) \cdot P(\text{M} \mid \text{G}) \cdot P(\text{S} \mid \text{G})$
Efficient Inference in Bayesian Networks

Inference Problem 1: Conditional Probability Query

- Observe the values of one or more variables: A=a, B=b
- Infer the conditional probability distribution of one or more query variables X, Y: \( P(X,Y | A=a, B=b) \)
Inference Problem 2: Most Probable
Explanation

– Given observed values for some variables
  A=a, B=b

– Find the most probable configuration of the remaining variables
Both of these problems can be solved reasonably efficiently

- Requires time exponential in the size of the “induced tree width”
- Networks with many undirected cycles have high induced tree width
- Networks with no undirected cycles have a tree width equal to the size of the biggest single node
Computational Complexity

Scaling behavior of an algorithm
- Input size: number of bits required to describe the input to the algorithm
- Running time: time required to compute the answer
Example: Insertion Sort

- **Input:** array of N numbers (each 32 bits long)
- **Output:** same array sorted in ascending order
- **Algorithm:** repeatedly select the smallest item and swap it into the next empty cell in the array

Smallest so far: 12

```
12  2  38  21  16  19
```
Example: Insertion Sort

smallest so far: 2

12
2
38
21
16
19
Example: Insertion Sort

smallest so far: 2
Example: Insertion Sort

smallest so far: 2
Example: Insertion Sort

smallest so far: 2

<table>
<thead>
<tr>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
</tr>
<tr>
<td>38</td>
</tr>
<tr>
<td>21</td>
</tr>
<tr>
<td>16</td>
</tr>
<tr>
<td>19</td>
</tr>
</tbody>
</table>
Example: Insertion Sort

smallest so far: 2

12
2
38
21
16
19
Now Swap

smallest so far:  2
Now Swap

smallest so far: 2
Find the Smallest Remaining Item

2
12
38
21
16
19

smallest so far: 12
Find the Smallest Remaining Item

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>12</td>
<td>38</td>
<td>21</td>
<td>16</td>
</tr>
<tr>
<td>19</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

smallest so far: 12
Find the Smallest Remaining Item

smallest so far: 12

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>12</td>
<td>38</td>
<td>21</td>
<td>16</td>
</tr>
<tr>
<td>19</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Find the Smallest Remaining Item

smallest so far: 12
Find the Smallest Remaining Item

smallest so far: 12
Swap (trivial)

smallest so far: 38

2
12
38
21
16
19
Find third smallest

smallest so far: 21
Find third smallest

smallest so far: 16
Find third smallest

smallest so far: 16
smallest so far: 21
The number of steps is proportional to $N^2$

- 6 steps required to find smallest number
- 5 steps required to find second smallest
- 4 steps to find third smallest
- 3 steps to find fourth smallest
- 2 steps to find fifth smallest
- 1 step to find sixth smallest

$$1 + 2 + \cdots + N = N^2 \frac{(N + 1)}{2} = O(N^2)$$
Big Oh notation

A sequence of numbers $x_1, x_2, \ldots, x_n, \ldots$ grows as $O(n^2)$ if there exists a time $T_0$ and a constant $k$ such all $x_t \leq k*n^2$ for $t > T_0$
“Fast” and “Slow” algorithms

Fast algorithms in computer science
- grow as $O(n)$, $O(n \log n)$
- “heap sort” $O(N \log N)$

Practical, but inefficient
- grow as $O(n^2)$, $O(n^3)$
- matrix multiplication $O(n^3)$
- Bayesian networks with bounded induced tree width

Impractical
- grow as $2^{O(n)}$
- arbitrary Bayesian networks
Problem Complexity

- Computational Complexity Theory
  - can prove that certain problems are inherently uncomputable
  - can prove that other problems are likely to be intractable (i.e., likely to require exponential time)
    - NP-Complete problems
  - can prove that other problems have efficient algorithms (i.e., likely to require polynomial time)
    - Polynomial time problems
Influence Diagrams
(AKA Decision Diagrams)

Three kinds of nodes

- random nodes (Bayesian network nodes)
  - ovals

- decision nodes: represent a decision that must be made
  - squares

- utility node: specify the utility of the different outcomes
  - hexagons or diamonds
Should I Fill the Tank?

- **Decision variable:** FillTank
  - at the time we make the FillTank decision, we will know the value of Meter1 and Starts1
- **Utility node:** Utility
  - the utility depends on the cost of filling the tank and on whether the car starts
Demonstration
Solving Decision Problems

For each combination of parent node values, we must compute the expected utility of the possible decisions.

Example: FillTank

- parents: meter1, starts1
- \( P(\text{starts2| meter1=false, starts1=false, FillTank=false}) = [0.0918, 0.9082] \)
- expected utility is \( 0.0918 \times 50 + 0.9082 \times 0 = 4.60 \)

Requires probabilistic inference to compute \( P(\text{starts2| meter1=false, starts1=false, FillTank=false}) \)
Example: Meter1=false; Starts1=false

P(starts2|fill, m1=false, s1=false)

<table>
<thead>
<tr>
<th></th>
<th>starts2=false</th>
<th>starts2=true</th>
</tr>
</thead>
<tbody>
<tr>
<td>fill=false</td>
<td>0.908</td>
<td>0.092</td>
</tr>
<tr>
<td>fill=true</td>
<td>0.208</td>
<td>0.792</td>
</tr>
</tbody>
</table>

U(fill | starts2, m1=false, s1=false)

<table>
<thead>
<tr>
<th></th>
<th>starts2=false</th>
<th>starts2=true</th>
</tr>
</thead>
<tbody>
<tr>
<td>fill=false</td>
<td>0</td>
<td>50</td>
</tr>
<tr>
<td>fill=true</td>
<td>-30</td>
<td>20</td>
</tr>
</tbody>
</table>

E[ U(fill | m1=false, s1=false)]

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>fill=false</td>
<td>4.60</td>
</tr>
<tr>
<td>fill=true</td>
<td>9.60</td>
</tr>
</tbody>
</table>
Formulating Decision Problems

- Decide network structure
  - Identify the decision variables
  - Identify the utility variable
  - Determine what random variables should be influenced by decisions and should be input to the utility variable
  - Determine what random variables will be known when the decision is made
  - For each random variable, consider what parents it might have (recursively)
Formulation (2)

- Choose values of the decision variable
  - What are the possible decision alternatives (very important)

- Elicit the utility table
  - What is the value of each possible outcome
  - This can be very difficult. There is a methodology based on identifying “gambles” to which you are indifferent

- Elicit the probability tables
  - Fit probability tables from data
  - Get probability estimates from users
Example: Arrow-Fisher Development Scenario

Decision variables:
- $d_1$: amount of land developed in first period (real number in $[0,1]$)
- $d_2$: amount of land developed in second period (real number in $[0,1 - d_1]$)

Utility variable:
- $U$: total benefit: sum of $U_1$ and $U_2$
Analysis

- Arrow & Fisher analyze the problem without defining a specific distribution for Result 1 or Result 2 or a specific utility function U2.

- Challenge: draw the decision tree for the Arrow & Fisher analysis.
Decision Tree

- **d1 = 0**
  - result1
  - **d2 = 0**
    - result2
    - u2
    - u2
    - u2
  - **u2**
    - u2
    - u2
    - u2
  - **u2**
    - u2
    - u2
    - u2
- **d1 > 0**
  - result1
  - **d2 = 1 - d1**
    - result2
    - u2
    - u2
    - u2
  - **u2**
    - u2
    - u2
    - u2
  - **u2**
    - u2
    - u2
    - u2

- **d2**