Outline

- Exact inference by enumeration
- Exact inference by variable elimination
- Approximate inference by stochastic simulation
- Approximate inference by Markov chain Monte Carlo

Inference tasks

Simple queries: compute posterior marginal $\mathbf{P}(X_i|\mathbf{E}=\mathbf{e})$
e.g., $P(NoGas|Gauge=empty, Lights=on, Starts=false)$

Conjunctive queries: $\mathbf{P}(X_i, X_j|\mathbf{E}=\mathbf{e}) = \mathbf{P}(X_i|\mathbf{E}=\mathbf{e})\mathbf{P}(X_j|X_i, \mathbf{E}=\mathbf{e})$

Optimal decisions: decision networks include utility information;
probabilistic inference required for $P(outcome|action, evidence)$

Value of information: which evidence to seek next?

Sensitivity analysis: which probability values are most critical?

Explanation: why do I need a new starter motor?

The Normalization Shortcut

$P(B \mid j, m)$ stands for the probability distribution of B given that $J = j$ and $M = m$

By definition $P(B \mid j, m) = P(B, j, m) / P(j, m)$, so letting $\alpha = (1/P(j, m))$ lets us write:

$P(B \mid j, m) = \alpha P(B, j, m)$

Why? Because we don't have to calculate $P(j, m)$ explicitly!

$\langle P(b \mid j, m), P(\neg b \mid j, m) \rangle = \langle \alpha P(b, j, m), \alpha P(\neg b, j, m) \rangle$

By the laws of probability $P(b \mid j, m) + P(\neg b \mid j, m) = 1$, so

$\alpha P(b, j, m) + \alpha P(\neg b, j, m) = 1$

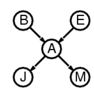
$\alpha = 1/(P(b, j, m) + P(\neg b, j, m))$

In general: α means "make distribution sum to 1"

Inference by enumeration

Slightly intelligent way to sum out variables from the joint without actually constructing its explicit representation

Simple query on the burglary network:



$\mathbf{P}(B|j, m)$
$= \mathbf{P}(B, j, m)/P(j, m)$
$= \alpha \mathbf{P}(B, j, m)$
$= \alpha \Sigma_e \Sigma_a \mathbf{P}(B, e, a, j, m)$

Rewrite full joint entries using product of CPT entries:
$\mathbf{P}(B|j, m)$
$= \alpha \Sigma_e \Sigma_a \mathbf{P}(B)P(e)\mathbf{P}(a|B, e)P(j|a)P(m|a)$
$= \alpha \mathbf{P}(B) \Sigma_e P(e) \Sigma_a \mathbf{P}(a|B, e)P(j|a)P(m|a)$

Recursive depth-first enumeration: $O(n)$ space, $O(d^n)$ time

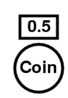
Inference by variable elimination

Variable elimination: carry out summations right-to-left, storing intermediate results (factors) to avoid recomputation

$\mathbf{P}(B|j, m)$
$= \alpha \underbrace{\mathbf{P}(B)}_{B} \Sigma_e \underbrace{P(e)}_{E} \Sigma_a \underbrace{\mathbf{P}(a|B, e)}_{A} \underbrace{P(j|a)}_{J} \underbrace{P(m|a)}_{M}$
$= \alpha \mathbf{P}(B) \Sigma_e P(e) \Sigma_a \mathbf{P}(a|B, e) P(j|a) f_M(a)$
$= \alpha \mathbf{P}(B) \Sigma_e P(e) \Sigma_a \mathbf{P}(a|B, e) f_J(a) f_M(a)$
$= \alpha \mathbf{P}(B) \Sigma_e P(e) \Sigma_a f_A(a, b, e) f_J(a) f_M(a)$
$= \alpha \mathbf{P}(B) \Sigma_e P(e) f_{\bar{A}JM}(b, e)$ (sum out A)
$= \alpha \mathbf{P}(B) f_{\bar{E}\bar{A}JM}(b)$ (sum out E)
$= \alpha f_B(b) \times f_{\bar{E}\bar{A}JM}(b)$

Inference by stochastic simulation

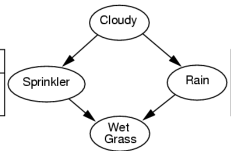
Basic idea:
1) Draw N samples from a sampling distribution S
2) Compute an approximate posterior probability $\hat{P}$
3) Show this converges to the true probability P



Outline:
– Sampling from an empty network
– Rejection sampling: reject samples disagreeing with evidence
– Likelihood weighting: use evidence to weight samples
– Markov chain Monte Carlo (MCMC): sample from a stochastic process whose stationary distribution is the true posterior

Example

P(C)
.50



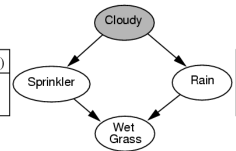
C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

Example

P(C)
.50



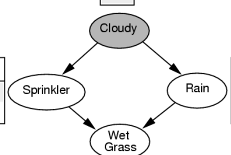
C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

Example

P(C)
.50



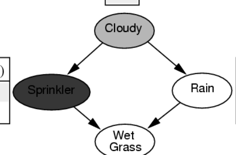
C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

Example

P(C)
.50



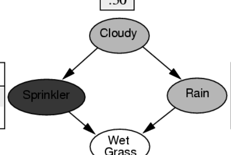
C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

Example

P(C)
.50



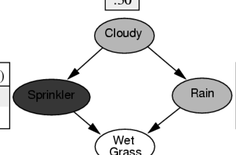
C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

Example

P(C)
.50

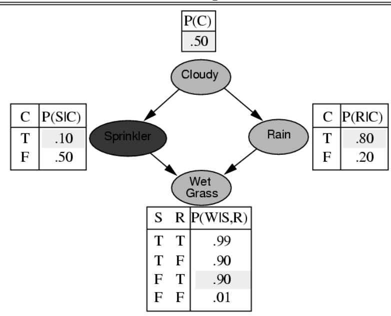


C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

Example



P(C)
.50

C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

Sampling from an empty network contd.

Probability that PRIORSAMPLE generates a particular event

$S_{PS}(x_1 \ldots x_n) = \prod_{i=1}^{n} P(x_i | Parents(X_i)) = P(x_1 \ldots x_n)$

i.e., the true prior probability

E.g., $S_{PS}(t, f, t, t) = 0.5 \times 0.9 \times 0.8 \times 0.9 = 0.324 = P(t, f, t, t)$

Let $N_{PS}(x_1 \ldots x_n)$ be the number of samples generated for event $x_1, \ldots, x_n$

Then we have

$$\begin{aligned}\lim_{N\to\infty} \hat{P}(x_1, \ldots, x_n) &= \lim_{N\to\infty} N_{PS}(x_1, \ldots, x_n)/N \\ &= S_{PS}(x_1, \ldots, x_n) \\ &= P(x_1 \ldots x_n)\end{aligned}$$

That is, estimates derived from PRIORSAMPLE are consistent

Shorthand: $\hat{P}(x_1, \ldots, x_n) \approx P(x_1 \ldots x_n)$

Rejection sampling

$\hat{\mathbf{P}}(X|\mathbf{e})$ estimated from samples agreeing with $\mathbf{e}$

```
function REJECTION-SAMPLING(X, e, bn, N) returns an estimate of P(X|e)
   local variables: N, a vector of counts over X, initially zero
   for j = 1 to N do
       x ← PRIOR-SAMPLE(bn)
       if x is consistent with e then
           N[x] ← N[x]+1 where x is the value of X in x
   return NORMALIZE(N[X])
```

E.g., estimate $\mathbf{P}(Rain|Sprinkler = true)$ using 100 samples
27 samples have $Sprinkler = true$
Of these, 8 have $Rain = true$ and 19 have $Rain = false$.

$\hat{\mathbf{P}}(Rain|Sprinkler = true) = \text{NORMALIZE}(\langle 8, 19\rangle) = \langle 0.296, 0.704\rangle$

Similar to a basic real-world empirical estimation procedure

Analysis of rejection sampling

$\hat{\mathbf{P}}(X|\mathbf{e}) = \alpha \mathbf{N}_{PS}(X, \mathbf{e})$ (algorithm defn.)
$= \mathbf{N}_{PS}(X, \mathbf{e})/N_{PS}(\mathbf{e})$ (normalized by $N_{PS}(\mathbf{e})$)
$\approx \mathbf{P}(X, \mathbf{e})/P(\mathbf{e})$ (property of PRIORSAMPLE)
$= \mathbf{P}(X|\mathbf{e})$ (defn. of conditional probability)

Hence rejection sampling returns consistent posterior estimates

Problem: hopelessly expensive if $P(\mathbf{e})$ is small

$P(\mathbf{e})$ drops off exponentially with number of evidence variables!

Markov Chain Monte Carlo

MCMC with Gibbs Sampling

Fix the values of observed variables

Set the values of all non-observed variables randomly

Perform a random walk through the space of complete variable assignments. On each move:

1. **Pick a variable X**
2. **Calculate Pr(X=true | all other variables)**
3. **Set X to true with that probability**

Repeat many times. Frequency with which any variable X is true is it's posterior probability.

Converges to true posterior when frequencies stop changing significantly

- **stable distribution, mixing**

Markov Blanket Sampling

How to calculate Pr(X=true | all other variables) ?

Recall: a variable is independent of all others given it's Markov Blanket

- **parents**
- **children**
- **other parents of children**

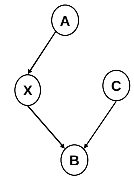
So problem becomes calculating Pr(X=true | MB(X))

- **We solve this sub-problem exactly**
- **Fortunately, it is easy to solve**

$$P(X) = \alpha P(X \mid Parents(X)) \prod_{Y \in Children(X)} P(Y \mid Parents(Y))$$

Example

$$P(X) = \alpha P(X \mid Parents(X)) \prod_{Y \in Children(X)} P(Y \mid Parents(Y))$$



$$P(X \mid A,B,C) = \frac{P(X,A,B,C)}{P(A,B,C)}$$

$$= \frac{P(A)P(X \mid A)P(C)P(B \mid X,C)}{P(A,B,C)}$$

$$= \left[\frac{P(A)P(C)}{P(A,B,C)}\right] P(X \mid A)P(B \mid X,C)$$

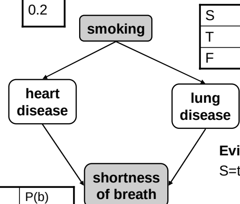
$$= \alpha P(X \mid A)P(B \mid X,C)$$

Example

P(s)
0.2

S	P(l)
T	0.8
F	0.1

S	P(s)
T	0.6
F	0.1



H	L	P(b)
T	T	0.9
T	F	0.8
F	T	0.7
F	F	0.1

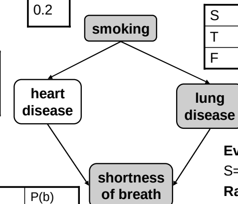
Evidence:
S=true, B=true

Example 2

P(s)
0.2

S	P(l)
T	0.8
F	0.1

S	P(h)
T	0.6
F	0.1



H	L	P(b)
T	T	0.9
T	F	0.8
F	T	0.7
F	F	0.1

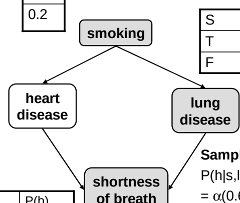
Evidence:
S=true, B=true
Randomly set H=false, L=true

Example 3

P(s)
0.2

S	P(l)
T	0.8
F	0.1

S	P(h)
T	0.6
F	0.1



H	L	P(b)
T	T	0.9
T	F	0.8
F	T	0.7
F	F	0.1

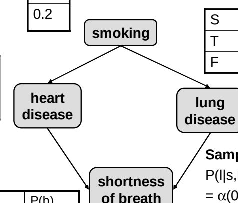
Sample H:
P(h|s,l,b)=αP(h|s)P(b|h,l)
= α(0.6)(0.9)= α 0.54
P(¬h|s,l,b)=αP(¬h|s)P(b| ¬h,l)
= α(0.4)(0.7)= α 0.28
Normalize: 0.54/(0.54+0.28)=0.66
Flip coin: H becomes true (maybe)

Example 4

P(s)
0.2

S	P(l)
T	0.8
F	0.1

S	P(h)
T	0.6
F	0.1



H	L	P(b)
T	T	0.9
T	F	0.8
F	T	0.7
F	F	0.1

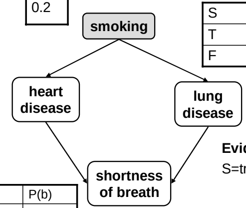
Sample L:
P(l|s,h,b)=αP(l|s)P(b|h,l)
= α(0.8)(0.9)= α 0.72
P(¬l|s,h,b)=αP(¬l|s)P(b| h, ¬ l)
= α(0.2)(0.8)= α 0.16
Normalize: 0.72/(0.72+0.16)=0.82
Flip coin: …

Example 5: Different Evidence

P(s)
0.2

S	P(s)
T	0.6
F	0.1

S	P(l)
T	0.8
F	0.1



H	L	P(b)
T	T	0.9
T	F	0.8
F	T	0.7
F	F	0.1

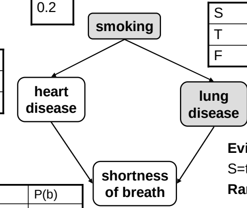
Evidence:
S=true, B=false

Example 6

P(s)
0.2

S	P(h)
T	0.6
F	0.1

S	P(l)
T	0.8
F	0.1



H	L	P(b)
T	T	0.9
T	F	0.8
F	T	0.7
F	F	0.1

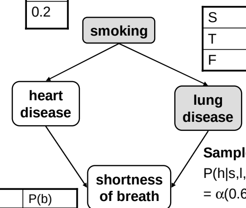
Evidence:
S=true, B=false
Randomly set H=false, L=true

Example 7

P(s)
0.2

S	P(h)
T	0.6
F	0.1

S	P(l)
T	0.8
F	0.1



H	L	P(b)
T	T	0.9
T	F	0.8
F	T	0.7
F	F	0.1

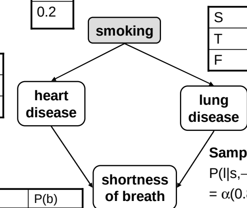
Sample H:
$P(h|s,l,\neg b)=\alpha P(h|s)P(\neg b|h,l)$
$= \alpha(0.6)(0.1)= \alpha\ 0.06$
$P(\neg h|s,l,\neg b)=\alpha P(\neg h|s)P(\neg b|\neg h,l)$
$= \alpha(0.4)(0.3)= \alpha\ 0.12$
Normalize: 0.06/(0.06+0.12)=0.33
Flip coin: H stays false (maybe)

Example 8

P(s)
0.2

S	P(h)
T	0.6
F	0.1

S	P(l)
T	0.8
F	0.1



H	L	P(b)
T	T	0.9
T	F	0.8
F	T	0.7
F	F	0.1

Sample L:
$P(l|s,\neg h,\neg b)=\alpha P(l|s)P(\neg b|\neg h,l)$
$= \alpha(0.8)(0.3)= \alpha\ 0.24$
$P(\neg l|s,\neg h,\neg b)=\alpha P(\neg l|s)P(\neg b|\neg h,\neg l)$
$= \alpha(0.2)(0.9)= \alpha\ 0.18$
Normalize: 0.24/(0.24+0.18)=0.75
Flip coin: …

Summary

Exact inference by variable elimination:
- polytime on polytrees, NP-hard on general graphs
- space = time, very sensitive to topology

Approximate inference by LW, MCMC: (and rejection sampling)
- LW does poorly when there is lots of (downstream) evidence
- LW, MCMC generally insensitive to topology
- Convergence can be very slow with probabilities close to 1 or 0
- Can handle arbitrary combinations of discrete and continuous variables

Outline

- ◇ Time and uncertainty
- ◇ Inference: filtering, prediction, smoothing
- ◇ Hidden Markov models
- ◇ Dynamic Bayesian networks
- ◇ Particle filtering

Time and uncertainty

The world changes; we need to track and predict it

Diabetes management vs vehicle diagnosis

Basic idea: copy state and evidence variables for each time step

$\mathbf{X}_t =$ set of unobservable state variables at time t
e.g., $BloodSugar_t$, $StomachContents_t$, etc.

$\mathbf{E}_t =$ set of observable evidence variables at time t
e.g., $MeasuredBloodSugar_t$, $PulseRate_t$, $FoodEaten_t$

This assumes **discrete time**; step size depends on problem

Notation: $\mathbf{X}_{a:b} = \mathbf{X}_a, \mathbf{X}_{a+1}, \ldots, \mathbf{X}_{b-1}, \mathbf{X}_b$

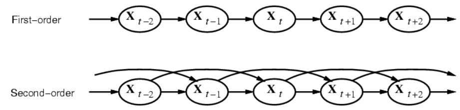
Chapter 15 3 31

Markov processes (Markov chains)

Construct a Bayes net from these variables: parents?

Markov assumption: $\mathbf{X}_t$ depends on **bounded** subset of $\mathbf{X}_{0:t-1}$

First-order Markov process: $\mathbf{P}(\mathbf{X}_t|\mathbf{X}_{0:t-1}) = \mathbf{P}(\mathbf{X}_t|\mathbf{X}_{t-1})$
Second-order Markov process: $\mathbf{P}(\mathbf{X}_t|\mathbf{X}_{0:t-1}) = \mathbf{P}(\mathbf{X}_t|\mathbf{X}_{t-2}, \mathbf{X}_{t-1})$

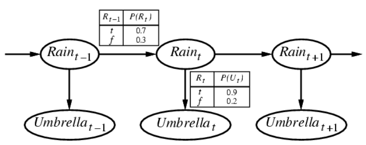


Sensor Markov assumption: $\mathbf{P}(\mathbf{E}_t|\mathbf{X}_{0:t}, \mathbf{E}_{0:t-1}) = \mathbf{P}(\mathbf{E}_t|\mathbf{X}_t)$

Stationary process: transition model $\mathbf{P}(\mathbf{X}_t|\mathbf{X}_{t-1})$ and sensor model $\mathbf{P}(\mathbf{E}_t|\mathbf{X}_t)$ fixed for all t

Chapter 15 4 32

Example



First-order Markov assumption not exactly true in real world!

Possible fixes:
1. **Increase order** of Markov process
2. **Augment state**, e.g., add $Temp_t$, $Pressure_t$

Example: robot motion.
Augment position and velocity with $Battery_t$

Chapter 15 5 33

Inference tasks

Filtering: $\mathbf{P}(\mathbf{X}_t|\mathbf{e}_{1:t})$
belief state—input to the decision process of a rational agent

Prediction: $\mathbf{P}(\mathbf{X}_{t+k}|\mathbf{e}_{1:t})$ for $k > 0$
evaluation of possible action sequences;
like filtering without the evidence

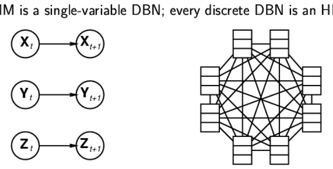
Smoothing: $\mathbf{P}(\mathbf{X}_k|\mathbf{e}_{1:t})$ for $0 \leq k < t$
better estimate of past states, essential for learning

Most likely explanation: $\arg\max_{\mathbf{x}_{1:t}} P(\mathbf{x}_{1:t}|\mathbf{e}_{1:t})$
speech recognition, decoding with a noisy channel

Chapter 15 6 34

DBNs vs. HMMs

Every HMM is a single-variable DBN; every discrete DBN is an HMM

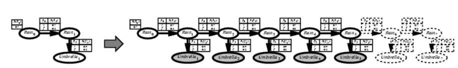


Sparse dependencies $\Rightarrow$ exponentially fewer parameters;
e.g., 20 state variables, three parents each
DBN has $20 \times 2^3 = 160$ parameters, HMM has $2^{20} \times 2^{20} \approx 10^{12}$

Chapter 15 33 35

Exact inference in DBNs

Naive method: unroll the network and run any exact algorithm



Problem: inference cost for each update grows with t

Rollup filtering: add slice $t + 1$, "sum out" slice t using variable elimination

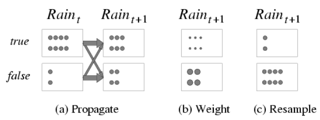
Largest factor is $O(d^{n+1})$, update cost $O(d^{n+2})$
(cf. HMM update cost $O(d^{2n})$)

Chapter 15 35 36

Particle filtering

Basic idea: ensure that the population of samples ("particles") tracks the high-likelihood regions of the state-space

Replicate particles proportional to likelihood for $\mathbf{e}_t$



(a) Propagate (b) Weight (c) Resample

Widely used for tracking nonlinear systems, esp. in vision

Also used for simultaneous localization and mapping in mobile robots
10^5-dimensional state space

Chapter 15 37 37

Particle filtering contd.

Assume consistent at time t: $N(\mathbf{x}_t|\mathbf{e}_{1:t})/N = P(\mathbf{x}_t|\mathbf{e}_{1:t})$

Propagate forward: populations of $\mathbf{x}_{t+1}$ are

$$N(\mathbf{x}_{t+1}|\mathbf{e}_{1:t}) = \Sigma_{\mathbf{x}_t} P(\mathbf{x}_{t+1}|\mathbf{x}_t)N(\mathbf{x}_t|\mathbf{e}_{1:t})$$

Weight samples by their likelihood for $\mathbf{e}_{t+1}$:

$$W(\mathbf{x}_{t+1}|\mathbf{e}_{1:t+1}) = P(\mathbf{e}_{t+1}|\mathbf{x}_{t+1})N(\mathbf{x}_{t+1}|\mathbf{e}_{1:t})$$

Resample to obtain populations proportional to W:

$$\begin{aligned} N(\mathbf{x}_{t+1}|\mathbf{e}_{1:t+1})/N &= \alpha W(\mathbf{x}_{t+1}|\mathbf{e}_{1:t+1}) = \alpha P(\mathbf{e}_{t+1}|\mathbf{x}_{t+1})N(\mathbf{x}_{t+1}|\mathbf{e}_{1:t}) \\ &= \alpha P(\mathbf{e}_{t+1}|\mathbf{x}_{t+1})\Sigma_{\mathbf{x}_t} P(\mathbf{x}_{t+1}|\mathbf{x}_t)N(\mathbf{x}_t|\mathbf{e}_{1:t}) \\ &= \alpha' P(\mathbf{e}_{t+1}|\mathbf{x}_{t+1})\Sigma_{\mathbf{x}_t} P(\mathbf{x}_{t+1}|\mathbf{x}_t)P(\mathbf{x}_t|\mathbf{e}_{1:t}) \\ &= P(\mathbf{x}_{t+1}|\mathbf{e}_{1:t+1}) \end{aligned}$$

Chapter 15 38 38

The Location Stack: Design and Sensor-Fusion for Location-Aware Ubicomp

Jeffrey Hightower

39

39

A survey & taxonomy of location technologies


Ad hoc signal strength · Physical contact · GPS · DC magnetic pulses · Cellular E-911
Infrared proximity · Ultrasonic time of flight · Laser range-finding · Stereo vision

40

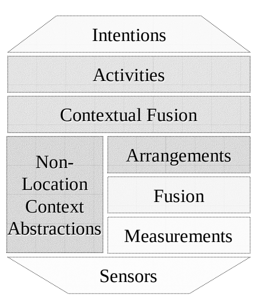
[Hightower and Borriello, *IEEE Computer*, Aug 2001]

40

The Location Stack

5 Principles

1. *There are fundamental measurement techniques.*
2. *There are standard ways to combine measurements.*
3. *There are standard object relationship queries.*
4. *Applications are concerned with activities.*
5. *Uncertainty is important.*



41

[Hightower, Brumitt, and Borriello, *WMCSA*, Jan 2002]

41

Principle 4: *Applications are concerned with activities*.

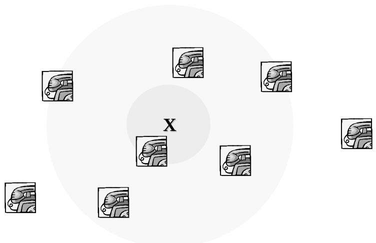
- Dinner is in progress.
- A presentation is going on in Mueller 153.
- Jeff is walking through his house listening to The Beatles.
- Jane is dispensing ethylene-glycol into beaker #45039.
- Elvis has left the building.

42

42

Principle 5: *Uncertainty is important.*

Example: routing phone calls to nearest handset

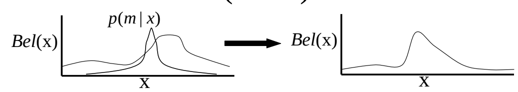


43

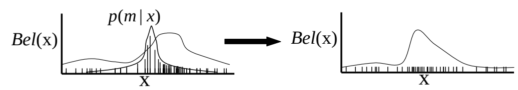
[Hightower and Borriello, *Ubicomp LMUC Workshop*, Sep 2001]

43

Fusion using Monte Carlo localization (MCL)



$$Bel(x_t) = p(x_t \mid m_t ... m_0)$$
$$Bel(x_t) = \eta p(m_t \mid x_t) \int p(x_t \mid x_{t-1}) Bel(x_{t-1}) dx_{t-1}$$



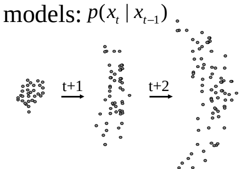
44

44

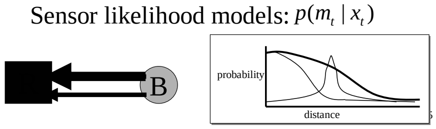
MCL details

Motion models: $p(x_t \mid x_{t-1})$

Stochastically shift all particles



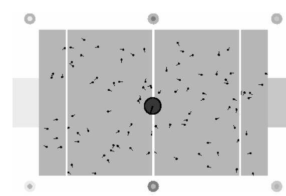
Sensor likelihood models: $p(m_t \mid x_t)$



45

2D MCL Example: Robocup

- 1 Object
- 2 types of Measurements
  - Vision marker distance
  - Odometry
- Red dot is most likely state.
  - (x,y,orientation)



46

[Fox et al., *Sequential Monte Carlo Methods in Practice*, 2000]

46

Adaptive MCL

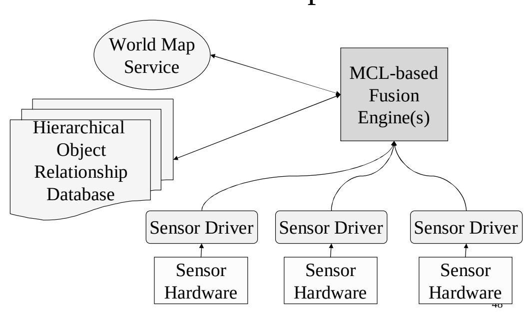
- Performance improvement: adjust sample count to best represent the posterior.
  1. Assume we know the true *Bel(x)* represented as a multinomial distribution.
  2. Determine number of samples such that with probability *(1-p)*, the Kullback-Leibler distance between the true posterior and the particle filter representation is less than ε

47

[Fox, *NIPS*, 2002]

47

Location Stack Implementation



48

48

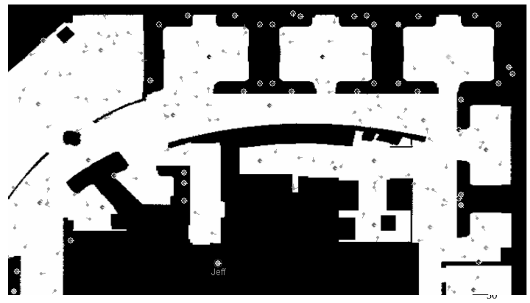
Location Stack Supported Technologies

1. VersusTech commercial infrared badge proximity system
2. RF Proximity using the Berkeley motes
3. SICK LMS-200 180º infrared laser range finders
4. MIT Cricket ultrasound range beacons
5. Indoor harmonic radar, *in progress*
6. 802.11b WiFi triangulation system, *in progress*
7. Cellular telephone E-OTD, *planned*

49

49

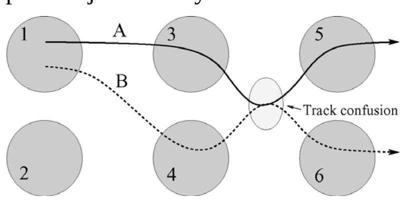
The Location Stack in action



50

Person Tracking with Anonymous and Id-Sensors: Motivation

- Accurate anonymous sensors exist
- Id-sensors are less accurate but provide explicit object identity information.



51

51

Person Tracking with Anonymous and Id-Sensors: Concept

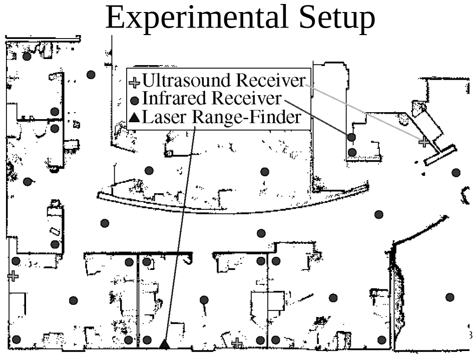
- Use Rao-Blackwellised particle filters to efficiently estimate locations
  1. Each particle is an association history between Kalman filter object tracks and observations.
  2. Due to initial id uncertainty, starts by tracking using only anonymous sensors and estimating object id's with sufficient statistics.
  3. Once id estimates are certain enough, sample id them using a fully Rao-Blackwellised particle filter over both object tracks and id assignments.

52

52

[Fox, Hightower, and Schulz., *Submitted to IJCAI*, 2003]

Experimental Setup



53

Experimental Setup

54

54

Person Tracking with Anonymous and Id-Sensors: Result

- Our 2 phase Rao-Blackwellised particle filter algorithm is quite effective.

55

Conclusion

Relying on a single location technology to support all UbiComp applications is inappropriate. Instead, the ***Location Stack*** *provides:*

1. The ability to fuse measurements from many technologies including both anonymous and id-sensors while preserving sensor uncertainty models.
2. Design abstractions enabling system evolution as new sensor technologies are created.
3. A common vocabulary to partition the work and research problems appropriately.

56

56

Natural Language Processing

CSE 592 Applications of AI
Winter 2003

Information Retrieval
Speech Recognition
Syntactic Parsing
Semantic Interpretation

57

Example Applications

- Spelling and grammar checkers
- Finding information on the WWW
- Spoken language control systems: banking, shopping
- Classification systems for messages, articles
- Machine translation tools

58

The Dream

59

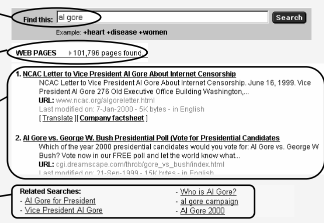
Information Retrieval

(Thanks to Adam Carlson)

60

Motivation and Outline

- Background
  - Definitions
- The Problem
  - 100,000+ pages
- The Solution
  - Ranking docs
  - Vector space
  - Probabilistic approaches
- Extensions
  - Relevance feedback, clustering, query expansion, etc.



61

What is Information Retrieval

- Given a large repository of documents, how do I get at the ones that I want
  - Examples: Lexus/Nexus, Medical reports, AltaVista
- Different from databases
  - Unstructured (or semi-structured) data
  - Information is (typically) text
  - Requests are (typically) word-based

62

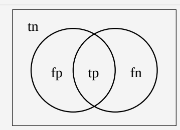
Information Retrieval Task

- Start with a set of documents
- User specifies *information need*
  - Keyword query, Boolean expression, high-level description
- System returns a list of documents
  - Ordered according to relevance
- Known as the *ad-hoc retrieval problem*

63

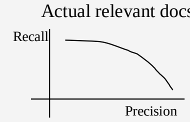
Measuring Performance

- Precision $\frac{tp}{tp + fp}$
  - Proportion of selected items that are correct
- Recall $\frac{tp}{tp + fn}$
  - Proportion of target items that were selected
- Precision-Recall curve
  - Shows tradeoff



System returned these

Actual relevant docs



64

Basic IR System

- Use word overlap to determine relevance
  - Word overlap alone is inaccurate
- Rank documents by similarity to query
- Computed using *Vector Space Model*

65

Vector Space Model

- Represent documents as a matrix
  - Words are rows
  - Documents are columns
  - Cell i,j contains the number of times word i appears in document j
  - Similarity between two documents is the cosine of the angle between the vectors representing those words

66

Vector Space Example

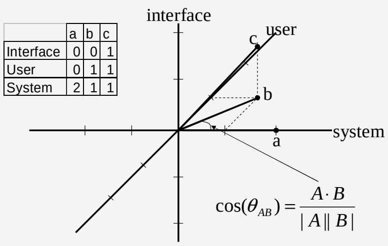
a: System and human system engineering testing of EPS
b: A survey of user opinion of computer system response time
c: The EPS user interface management system
d: Human machine interface for ABC computer applications
e: Relation of user perceived response time to error measurement
f: The generation of random, binary, ordered trees
g: The intersection graph of paths in trees
h: Graph minors IV: Widths of trees and well-quasi-ordering
i: Graph minors: A survey

	a	b	c	d	e	f	g	h	I
Interface	0	0	1	0	0	0	0	0	0
User	0	1	1	0	1	0	0	0	0
System	2	1	1	0	0	0	0	0	0
Human	1	0	0	1	0	0	0	0	0
Computer	0	1	0	1	0	0	0	0	0
Response	0	1	0	0	1	0	0	0	0
Time	0	1	0	0	1	0	0	0	0
EPS	1	0	1	0	0	0	0	0	0
Survey	0	1	0	0	0	0	0	0	1
Trees	0	0	0	0	0	1	1	1	0
Graph	0	0	0	0	0	0	1	1	1
Minors	0	0	0	0	0	0	0	1	1

67

Vector Space Example cont.

	a	b	c
Interface	0	0	1
User	0	1	1
System	2	1	1



$$\cos(\theta_{AB}) = \frac{A \cdot B}{|A||B|}$$

68

Similarity in Vector Space

$$A \cdot B = A_1 B_1 + A_2 B_2 + \ldots + A_n B_n$$

Measures word overlap

$$\cos(\theta_{AB}) = \frac{A \cdot B}{|A||B|}$$

Other metrics exist

Normalizes for different length vectors

$$|A| = \sqrt{\sum_{i=1}^{n} A_i^2}$$

69

Answering a Query Using Vector Space

- Represent query as vector
- Compute distances to all documents
- Rank according to distance
- Example
  - "computer system"

	Query	a	b	c	d	e	f	g	h	I
Interface	**0**	0	0	1	0	0	0	0	0	0
User	**0**	0	1	1	0	1	0	0	0	0
System	**1**	2	1	1	0	0	0	0	0	0
Human	**0**	1	0	0	1	0	0	0	0	0
Computer	**1**	0	1	0	1	0	0	0	0	0
Response	**0**	0	1	0	0	1	0	0	0	0
Time	**0**	0	1	0	0	1	0	0	0	0
EPS	**0**	1	0	1	0	0	0	0	0	0
Survey	**0**	0	1	0	0	0	0	0	0	1
Trees	**0**	0	0	0	0	0	1	1	1	0
Graph	**0**	0	0	0	0	0	0	1	1	1
Minors	**0**	0	0	0	0	0	0	0	1	1

70

Common Improvements

- The vector space model
  - Doesn't handle morphology (eat, eats, eating)
  - Favors common terms
- Possible fixes
  - Stemming
    - Convert each word to a common root form
  - Stop lists
  - Term weighting

71

Handling Common Terms

- Stop list
  - List of words to ignore
    - "a", "and", "but", "to", etc.
- Term weighting
  - Words which appear everywhere aren't very good discriminators – give higher weight to rare words

72

tf * idf

$$w_{ik} = tf_{ik} * \log(N / n_k)$$

T_k = term k in document D_i
tf_{ik} = frequency of term T_k in document D_i
idf_k = inverse document frequency of term T_k in C
N = total number of documents in the collection C
n_k = the number of documents in C that contain T_k

$$idf_k = \log\left(\frac{N}{n_k}\right)$$

73

Inverse Document Frequency

- IDF provides high values for rare words and low values for common words

For a collection of 10000 documents

$$\log\left(\frac{10000}{10000}\right) = 0$$

$$\log\left(\frac{10000}{5000}\right) = 0.301$$

$$\log\left(\frac{10000}{20}\right) = 2.698$$

$$\log\left(\frac{10000}{1}\right) = 4$$

74

Probabilistic IR

- Vector space model robust in practice
- Mathematically *ad-hoc*
  - How to generalize to more complex queries?
    `(intel or microsoft) and (not stock)`
- Alternative approach: model problem as finding documents with highest probability of being relevant to the query
  - Requires making some simplifying assumptions about underlying probability distributions
  - In certain cases can be shown to yield same results as vector space model

75

Probability Ranking Principle

- For a given query Q, find the documents D that maximize the odds that the document is relevant (R):

$$\frac{P(r \mid D,Q)}{P(\neg r \mid D,Q)} = P(Q \mid D,r) \times \frac{P(r \mid D)}{P(\neg r \mid D)}$$

76

Probability Ranking Principle

- For a given query Q, find the documents D that maximize the odds that the document is relevant (R):

$$\frac{P(r \mid D,Q)}{P(\neg r \mid D,Q)} = P(Q \mid D,r) \times \frac{P(r \mid D)}{P(\neg r \mid D)}$$

Probability of document relevance to *any* query – *i.e.*, the inherent quality of the document

77

Probability Ranking Principle

- For a given query Q, find the documents D that maximize the odds that the document is relevant (R):

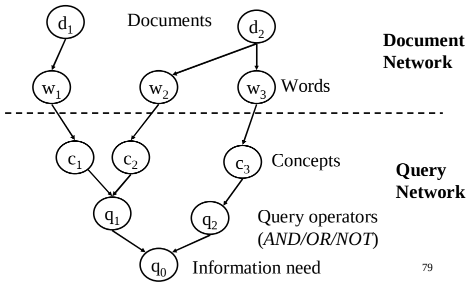
$$\frac{P(r \mid D,Q)}{P(\neg r \mid D,Q)} = P(Q \mid D,r) \times \frac{P(r \mid D)}{P(\neg r \mid D)}$$

Probability that if document is indeed relevant, then the query is in fact Q

But where do we get that number?

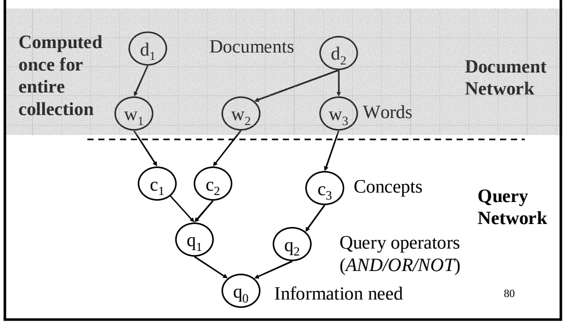
78

Bayesian nets for text retrieval



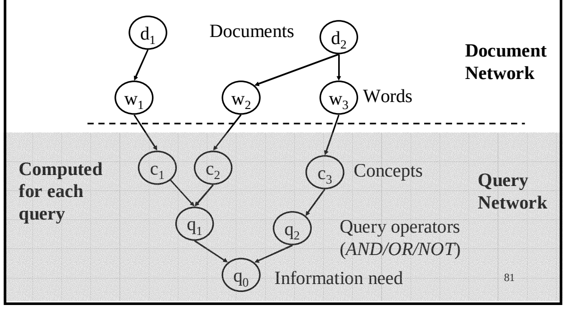
79

Bayesian nets for text retrieval



80

Bayesian nets for text retrieval



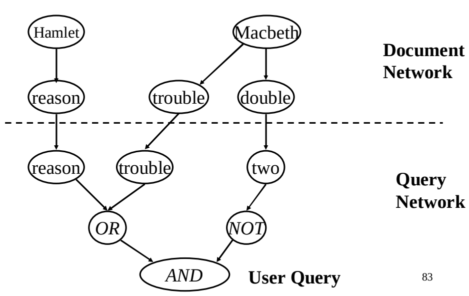
81

Conditional Probability Tables

- *P(d)* = prior probability document *d* is relevant
  - Uniform model: *P(d)* = 1 / Number docs
  - In general, document quality *P(r | d)*
- *P(w | d)* = probability that a random word from document *d* is *w*
  - Term frequency
- *P(c | w)* = probability that a given document word *w* has same meaning as a query word *c*
  - Thesarus
- $P(q \mid c_1, c_2, \ldots)$ = canonical form of operators AND, OR, NOT, *etc.*

82

Example



83

Details

- Set head q_0 of user query to "true"
- Compute posterior probability $P(D \mid q_0)$
- "User information need" doesn't have to be a query - can be a user profile, *e.g.*, other documents user has read
- Instead of just words, can include phrases, inter-document links
- Link matrices can be modified over time.
  - User feedback
  - The promise of "personalization"

84

Extensions

- Meet demands of web-based systems
- Modified ranking functions for the web
- Relevance feedback
- Query expansion
- Document clustering
- Latent Semantic Indexing
- Other IR tasks

85

IR on the Web

- Query AltaVista with "Java"
  - Almost 10^7 pages found
- Avoiding latency
  - User wants (initial) results **fast**
- Solution
  - Rank documents using word-overlap
  - Use special data structure - *inverted index*

86

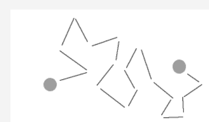
Improved Ranking on the Web

- Not just arbitrary documents
- Can use HTML tags and other properties
  - Query term in <TITLE></TITLE>
  - Query term in <IMG>, <HREF>, *etc*. tag
  - Check date of document (prefer recent docs)
  - PageRank (Google)

87

PageRank

- Idea: Good pages link to other good pages
  - Round 1: count in-links *Problems?*
  - Round 2: sum weighted in-links
  - Round 3: and again, and again…
- Implementation: Repeated random walk on snapshot of the web
  - weight ≈ frequency visited



Relevance Feedback

- System returns initial set of documents
- User identifies relevant documents
- System refines query to get documents more like those identified by user
  - Add words common to relevant docs
  - Reposition query vector closer to relevant docs
- Lather, rinse, repeat…

89

Query Expansion

- Given query, add words to improve recall
  - Workaround for synonym problem
- Example
  - boat → boat OR ship
- Can involve user feedback or not
- Can use thesaurus or other online source
  - WordNet

90

Document Clustering

- Group similar documents
  - Similar means “close in vector space”
- If a document is relevant, return whole cluster
- Can be combined with relevance feedback
- GROUPER

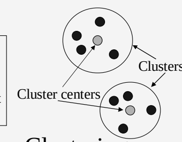
  http://www.cs.washington.edu/research/clustering

91

Clustering Algorithms

- K-means

```
Initialize k cluster centers
Loop
   Assign all document to closest center
   Move cluster centers to better fit assignment
Until little movement
```



- Hierarchical Agglomerative Clustering

```
Initialize each document to a singleton cluster
Loop
   Merge two closest clusters
Until k clusters exist
```

Many ways to measure distance between clusters

92

Latent Semantic Indexing

- Creates modified vector space
- Captures transitive co-occurrence information
  - If docs A & B don’t share any words, with each other, but both share lots of words with doc C, then A & B will be considered similar
- Simulates query expansion and document clustering (sort of)

93

Variations on a Theme

- Text Categorization
  - Assign each document to a *category*
  - Example: automatically put web pages in Yahoo hierarchy
- Routing & Filtering
  - Match documents with users
  - Example: news service that allows subscribers to specify “send news about high-tech mergers”

94