# CS347

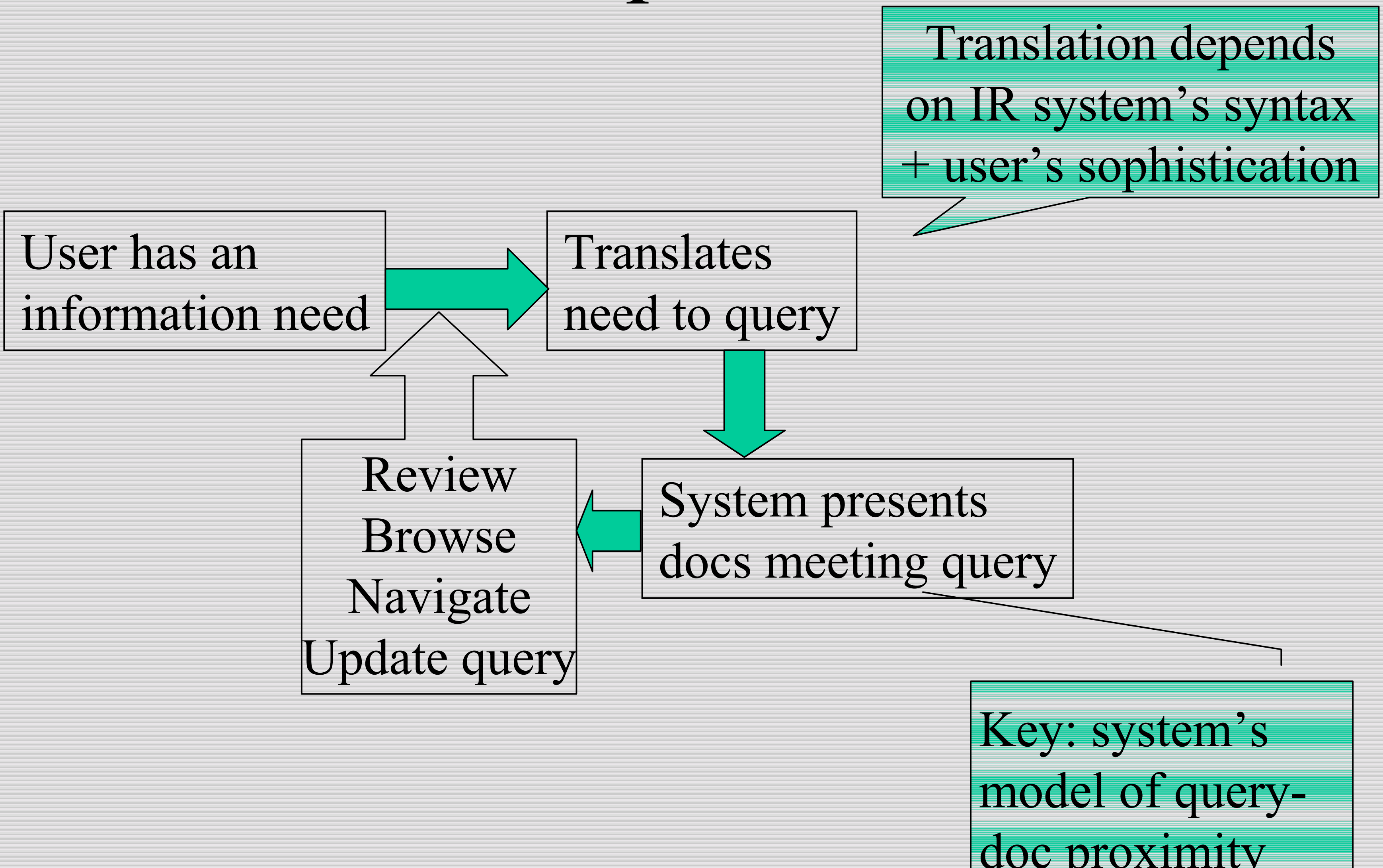
Lecture 5

April 23, 2001

# Today's topics

- Generalized query operators
  - Evidence accumulation and structured queries
- Basics of Bayesian networks
  - Bayesian nets for Text Retrieval
- Structured+unstructured queries
  - Adding database-like queries

# A step back


Translation depends on IR system's syntax + user's sophistication
User has an information need
Translates need to query
System presents docs meeting query
Review
Browse
Navigate
Update query
Key: system's model of query-doc proximity

# Models of query-doc proximity

- Boolean queries
  - Doc is either in or out for query
- Vector spaces
  - Doc has non-negative proximity to query
- Evidence accumulation
  - Combine score from multiple sources
- Bayesian nets and probabilistic methods
  - Infer probability that doc meets user's information need

# Evidence accumulation

- View each term in the query as providing partial evidence of match
- *tf* × *idf* + vector space retrieval is one example
  - Corpus-*dependent* (*idf* depends on corpus)
- In some situations corpus-dependent evidence is undesirable

# Corpus-independent evidence

- When is corpus-independent scoring useful?
  - When corpus statistics are hard to maintain
    - Distributed indices - more later
    - Rapidly changing corpora
  - When stable scores are desired
    - Users get used to issuing a search and seeing a doc with a score of (say) 0.9303
    - User subsequently filters by score
      - "Show me only docs with score at least 0.9"

# Corpus-independent scoring

- Document routing is a key application
  - There is a list of standing queries
    - e.g., ***bounced check*** in a bank's email customer service department
  - Each incoming doc (email) scored against all standing queries
  - Routed to destination (customer specialist) based on best-scoring standing query
- More on this with automatic classification

# Typical corpus-independent score

- Use a convex function of $tf_{ij}$
  - e.g., Score($i,j$) = $1$ - exp$(-a\times tf_{ij})$
  - $a$ is a tuning constant
  - gives a contribution of query term $i$ for doc $j$
- Given a multi-term query, compute the average contribution, over all query terms

# Bayesian Networks for Text Retrieval

- Text retrieval
  - Find the best set of documents that satisfies a user's information need
- Bayesian Network
  - Model causal relationship between events
  - Infer the belief that an event holds based on observations of other events

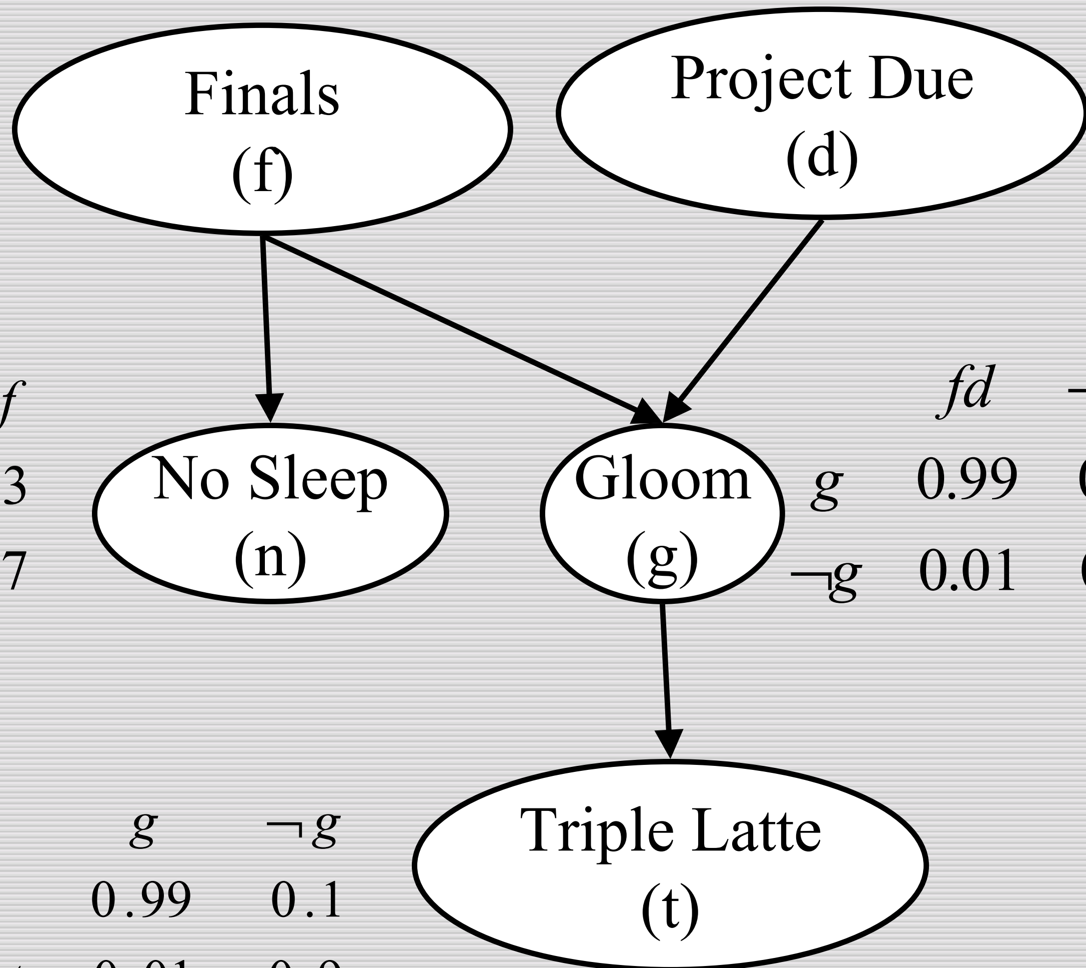
# What is a Bayesian network?

- Is a directed acyclic graph
- Nodes
  - – Events or Variables
    - Assume values.
    - For our purposes, all Boolean
- Links
  - – model dependencies between nodes

# Toy Example

| | |
|---|---|
| $f$ | 0.3 |
| $\neg f$ | 0.7 |

| | |
|---|---|
| $d$ | 0.4 |
| $\neg d$ | 0.6 |




| | $f$ | $\neg f$ |
|---|---|---|
| $n$ | 0.9 | 0.3 |
| $\neg n$ | 0.1 | 0.7 |

| | $fd$ | $\neg fd$ | $f\neg d$ | $\neg f\neg d$ |
|---|---|---|---|---|
| $g$ | 0.99 | 0.9 | 0.8 | 0.3 |
| $\neg g$ | 0.01 | 0.1 | 0.2 | 0.7 |

| | $g$ | $\neg g$ |
|---|---|---|
| $t$ | 0.99 | 0.1 |
| $\neg t$ | 0.01 | 0.9 |

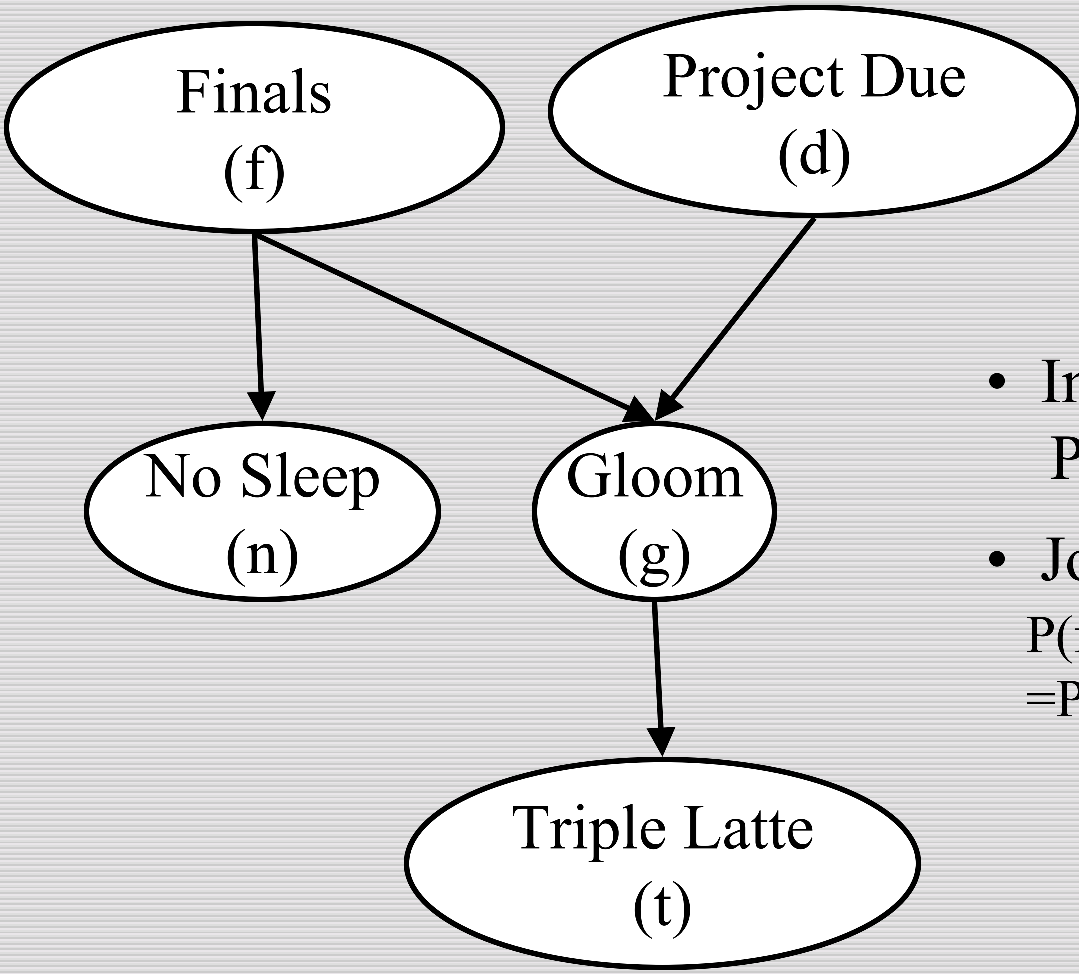
# Links as dependencies

- Link Matrix
  - Attached to each node
    - Give influences of parents on that node.
  - Nodes with no parent get a “prior probability”
    - e.g., *f, d.*
  - interior node : conditional probability of all combinations of values of its parents
    - e.g., *n,g,t.*

# Independence Assumption

- Variables not connected by a link: no direct conditioning.
- Joint probability - obtained from link matrices.
- See examples on next slide.

# Independence Assumption




- Independence assumption: P(t|g f)=P(t|g)
- Joint probability

P(f d n g t)
=P(f) P(d) P(n|f) P(g|f d) P(t|g)

# Chained inference

- Evidence - a node takes on some value
- Inference
  - Compute *belief* (probabilities) of other nodes
    - conditioned on the known evidence
  - Two kinds of inference: *Diagnostic* and *Predictive*
- Computational complexity
  - General network: NP-hard

  $\Rightarrow$ polytree networks - tractable.

# Key inference tool

- Bayes' theorem: for any two events *a,c*

$$P(a \cap c) = P(a \mid c)P(c) = P(c \mid a)P(a)$$

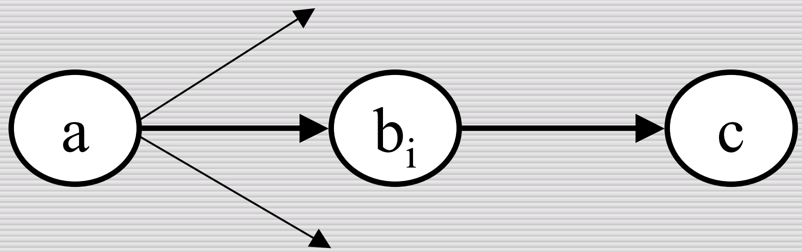
Implies, for instance:

$$P(a \mid c) = \frac{P(c \mid a)P(a)}{P(c)}$$

# Diagnostic Inference

- Propagate beliefs through parents of a node
- Inference rule

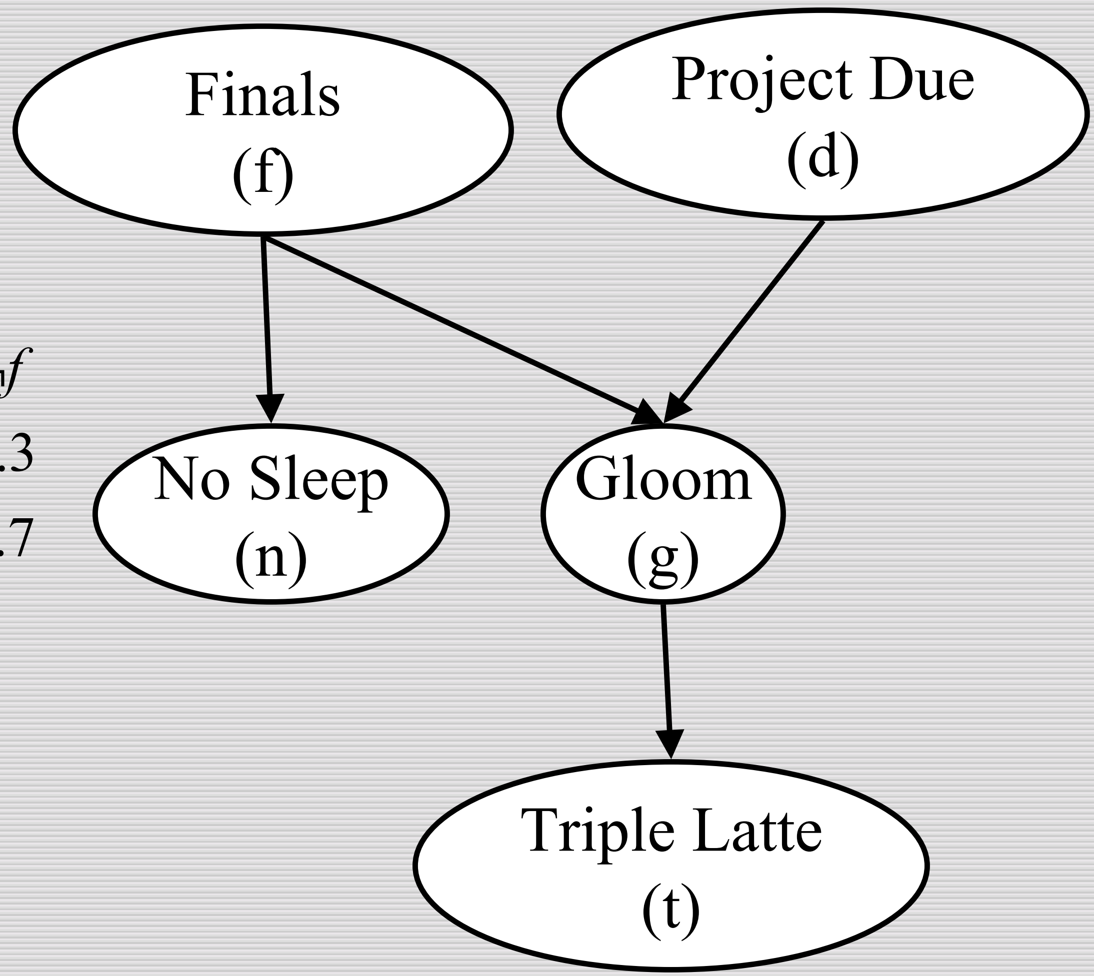
$$P(a \mid c) = \frac{P(a)\sum_{b_i} P(c \mid b_i) P(b_i \mid a)}{P(c)}$$



# Diagnostic inference

| | |
|---|---|
| $f$ | 0.3 |
| $\neg f$ | 0.7 |

| | $f$ | $\neg f$ |
|---|---|---|
| $n$ | 0.9 | 0.3 |
| $\neg n$ | 0.1 | 0.7 |




Evidence: $n$=true

Belief: $P(f|n)=?$

# Diagnostic inference

Inference Rule

$$P(f \mid n) = \frac{P(f)P(n \mid f)}{P(n)} = \frac{0.27}{P(n)}$$

$$P(\neg f \mid n) = \frac{P(\neg f)P(n \mid \neg f)}{P(n)} = \frac{0.21}{P(n)}$$

Normalize

$$P(f \mid n) + P(\neg f \mid n) = 1$$

$$\Rightarrow \quad P(n) = 0.48$$

Beliefs

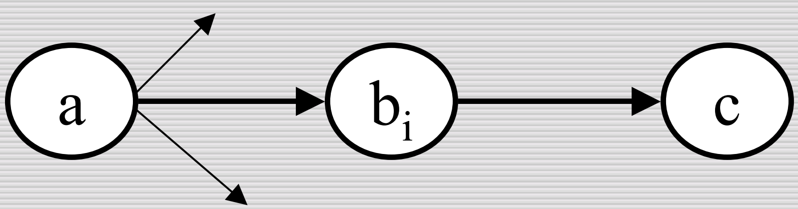
$$P(f \mid n) = 0.56$$

$$P(\neg f \mid n) = 0.44$$

# Predictive Inference

- Compute belief of child nodes of evidence
- Inference rule

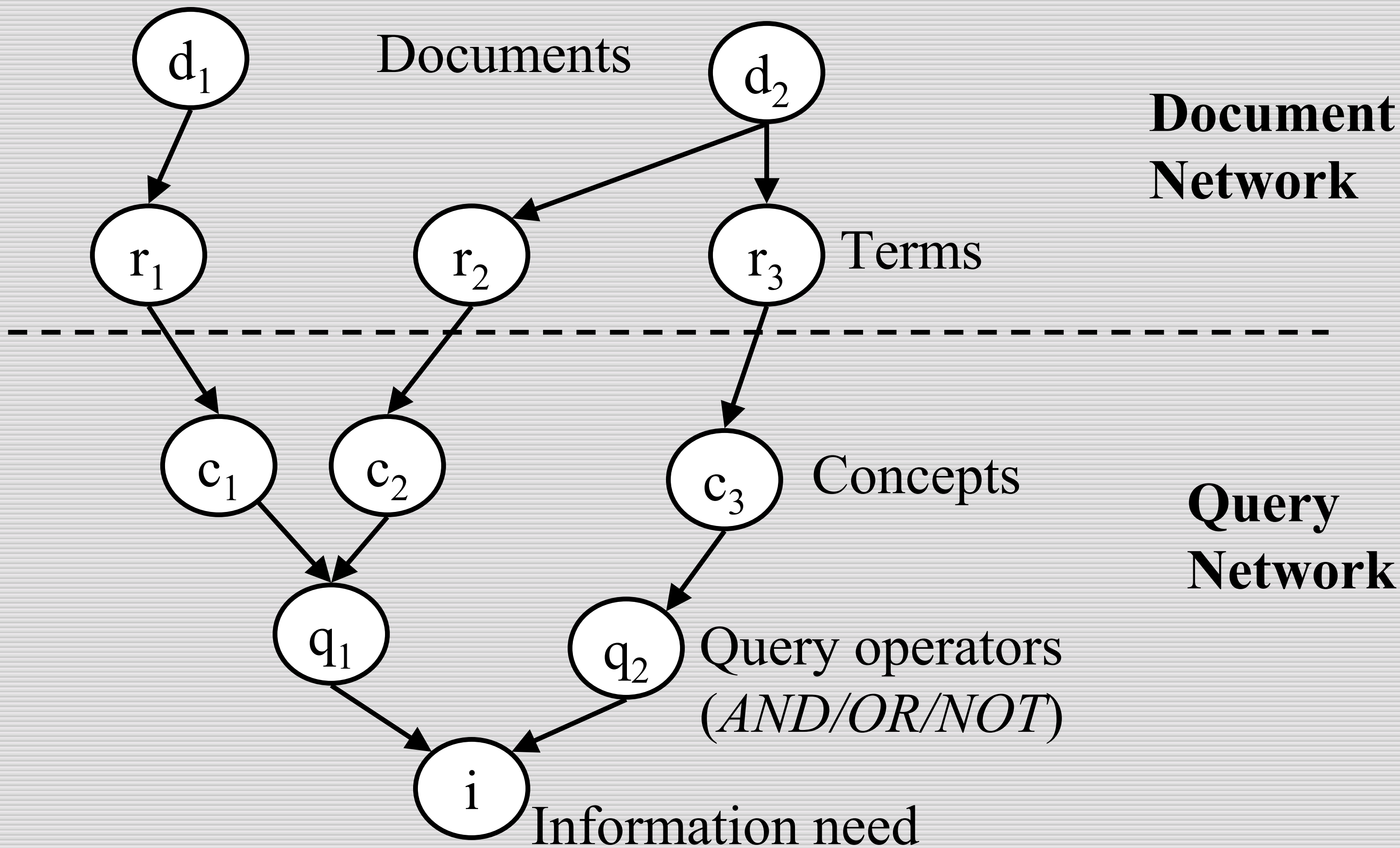
$$P(c \mid a) = \sum_b P(c \mid b_i) P(b_i \mid a)$$



# Model for Text Retrieval

- Goal
  - Given a user's information need (evidence), find probability a doc satisfies need
- Retrieval model
  - Model docs in a *document network*
  - Model information need in a *query network*

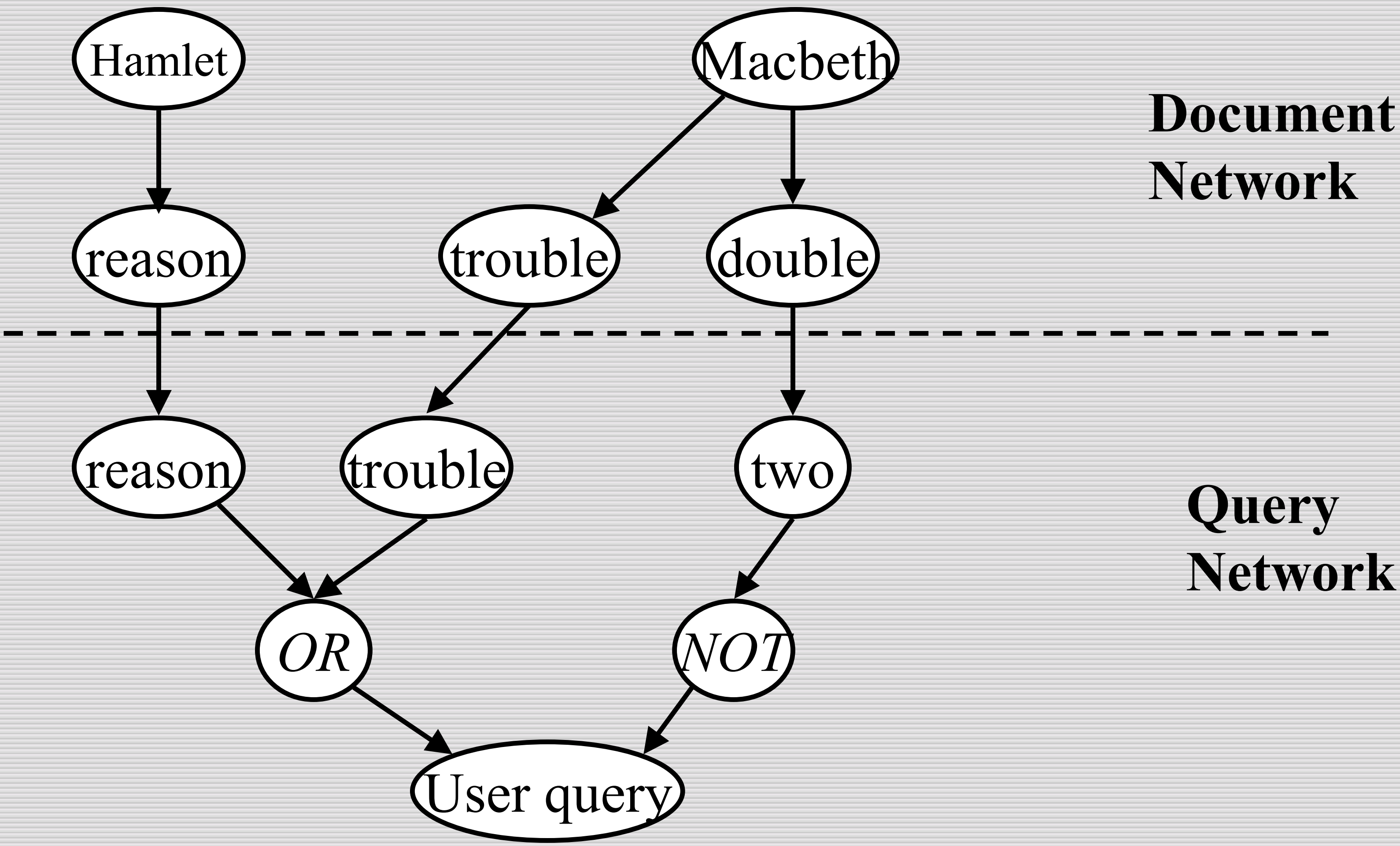
# Bayesian nets for text retrieval



# Link matrices and probabilities

- Prior doc probability $P(d) = 1/n$
- $P(r|d)$
  - within-document term frequency
  - $tf \times idf$ - based
- $P(c|r)$
  - 1-to-1
  - thesaurus
- $P(q|c)$: canonical forms of query operators

# Example


Hamlet
Macbeth
Document Network
reason
trouble
double
reason
trouble
two
Query Network
OR
NOT
User query

# Extensions

- Prior probs don't have to be *1/n*.
- "User information need" doesn't have to be a query - can be words typed, in docs read, any combination …
- Link matrices can be modified over time.
  - User feedback.
- The promise of "personalization"

# Computational details

- Document network built at indexing time
- Query network built/scored at query time
- Representation:
  – Link matrices from docs to any single term are like the postings entry for that term.

# Exercise

- Consider ranking docs for a 1-term query. What is the difference between
  - A cosine-based vector-space ranking where each doc has $tf \times idf$ components, normalized;
  - A Bayesian net in which the link matrices on the docs-to-term links are normalized $tf \times idf$?

# Semi-structured search

- Structured search - search by restricting on attribute values, as in databases.
- Unstructured search - search in unstructured files, as in text.
- Semi-structured search: combine both.

# Terminology

- Each document has
  - structured *fields* (aka *attributes*, *columns*)
  - free-form text
- Each field assumes one of several possible *values*
  - e.g., language (French, Japanese, etc.); price (for products); date; …
- Fields can be *ordered* (price, speed), or *unordered* (language, color).

# Queries

- A query is any combination of
  - text query
  - field query
- A field query specifies one or more values for one or more fields
  - for numerical values, ranges possible
    - e.g., *price < 5000.*

# Example

- Find all docs in corpus with
  - *Price < 10000*
  - *Year > 1996*
  - *Model = Toyota*, and
  - text matches (***excellent*** *OR* ***good*** *NEAR* ***condition***).
- Don't want to hit underlying database.
  - Demo.

# Indexing: structured portion

- For each fields, order docs by values for that field
  - e.g., sorted by authors' names, language …
- Maintain range indices (in memory) for each value of each attribute
  - like a postings entry
  - counts are like *freq* in postings.

# Query processing

- Given value for each field, determine counts of matching docs
- Process query using optimization heuristics
  - Lightest axis first
- Merge with text search postings.

# Numerical attributes

- Expensive to maintain a separate postings for each value of a numerical attribute
  - e.g., *price*
- Bucket into numerical ranges, maintain postings for each bucket
- At the user interface, present only bucket boundaries
  - e.g., if index buckets price into steps of $5000, present only these buckets to user

# General ranges

- If the UI allows the user to specify an arbitrary numerical range
  - in the used-car section of cars.com: *price, year*
  - e.g., *price* between 1234 and 5678.
- Need to walk through the postings entry for (say) the bucket 0-5000, until 1234 reached
- At most two postings entries need a walk-through

# Resources

- MIR 2.6, 2.8.
- R.M. Tong, L.A. Appelbaum, V.N. Askman, J.F. Cunningham. Conceptual Information Retrieval using RUBRIC. Proc. ACM SIGIR 247-253, (1987).

# Bayesian Resources


E. Charniak. Bayesian nets without tears. *AI Magazine* 12(4): 50-63 (1991).

http://www.aaai.org/Library/Magazine/Vol12/12-04/vol12-04.html

H.R. Turtle and W.B. Croft. Inference Networks for Document Retrieval. *Proc. ACM SIGIR*: 1-24 (1990).

D. Heckerman. A Tutorial on Learning with Bayesian Networks. Microsoft Technical Report MSR-TR-95-06 http://www.research.microsoft.com/~heckerman/

F. Crestani, M. Lalmas, C. J. van Rijsbergen, I. Campbell. Is This Document Relevant? ... Probably: A Survey of Probabilistic Models in Information Retrieval. ACM Computing Surveys 30(4): 528-552 (1998).

http://www.acm.org/pubs/citations/journals/surveys/1998-30-4/p528-crestani/