# Web-Mining Agents
## Probabilistic Information Retrieval

Prof. Dr. Ralf Möller
Universität zu Lübeck
Institut für Informationssysteme

Tanya Braun (Übungen)

# Acknowledgements

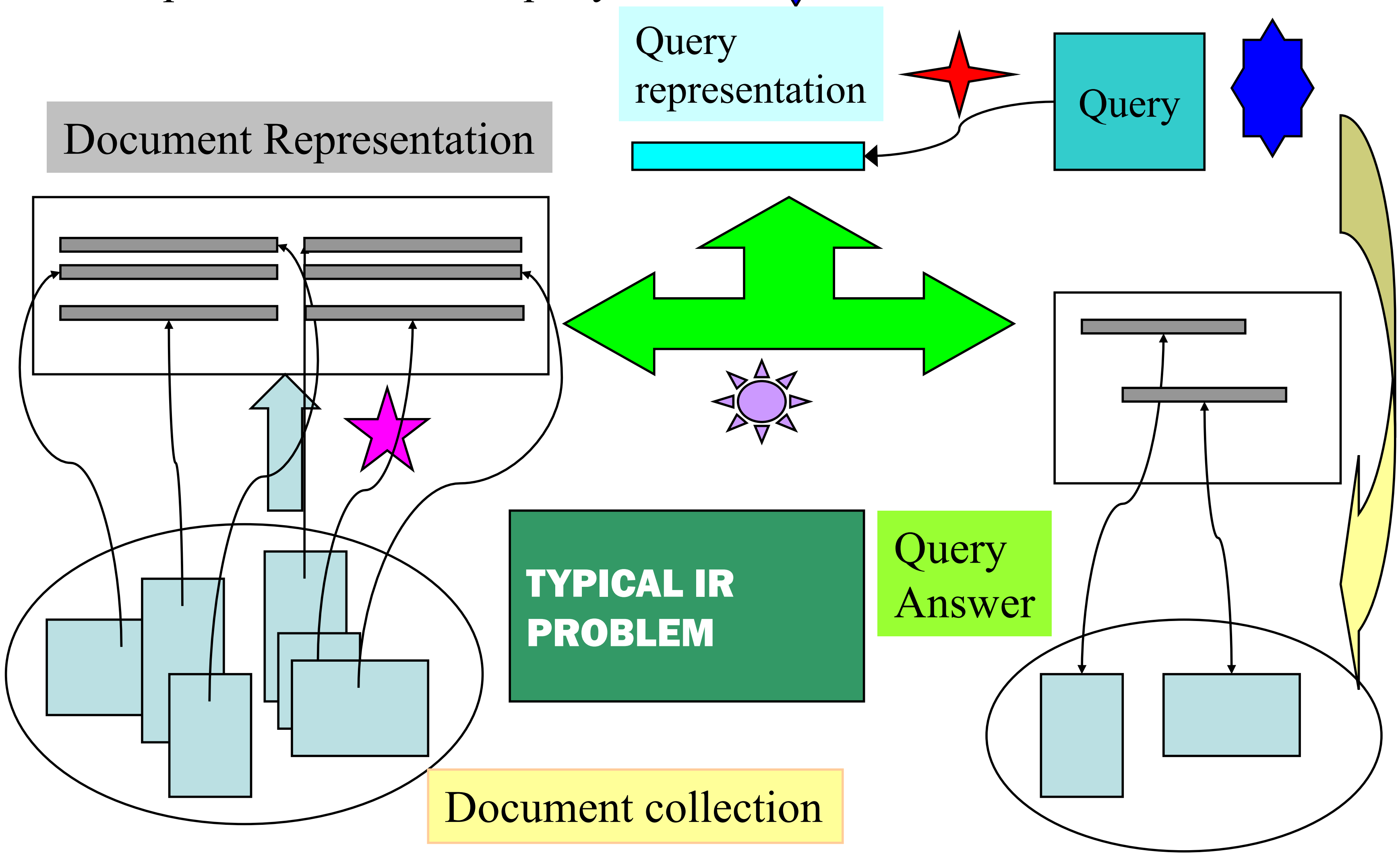
- Slides taken from:
  - Introduction to Information Retrieval
    Christopher Manning and Prabhakar Raghavan

How exact is the representation of the document ?

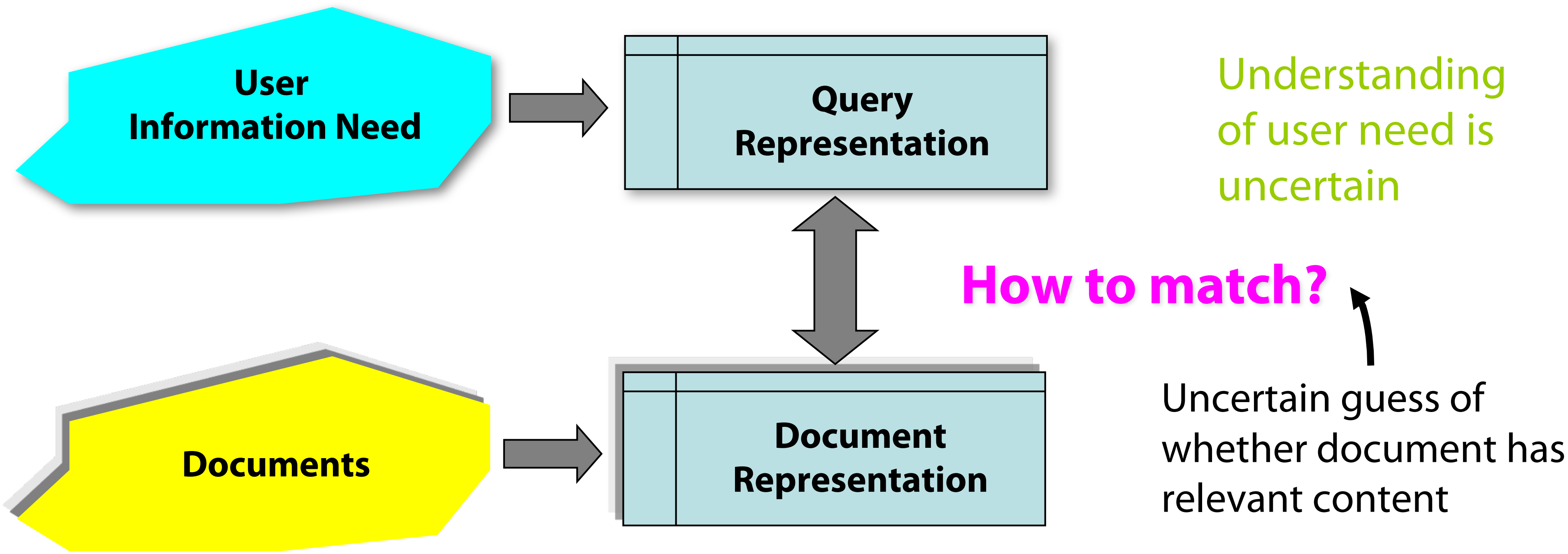
How exact is the representation of the query ?

How well is query matched to data?

How relevant is the result to the query ?



# Why probabilities in IR?




In traditional IR systems, matching between each document and query is attempted in a semantically imprecise space of index terms.

Probabilities provide a principled foundation for uncertain reasoning.
*Can we use probabilities to quantify our uncertainties?*

# Probability Ranking Principle

- Collection of Documents
- User issues a query
- A set of documents needs to be returned
- **Question: In what order to present documents to user ?**
- Need a formal way to judge the "goodness" of documents w.r.t. queries.
- **Idea: Probability of relevance of the documents w.r.t. query**

Ben He, Probability Ranking Principle, Reference Work Entry, Encyclopedia of Database Systems, Ling Liu, Tamer, Öszu (Eds.), 2168-2169, Springer, **2009**.

# Probabilistic Approaches to IR

- Probability Ranking Principle (Robertson, 70ies; Maron, Kuhns, 1959)

  Robertson S.E. The probability ranking principle in IR. J. Doc., 33:294–304, **1977**.

  M. E. Maron and J. L. Kuhns. 1960. On Relevance, Probabilistic Indexing and Information Retrieval. *J. ACM* 7, 3, 216-244, **1960**.

- Information Retrieval as Probabilistic Inference (van Rijsbergen & co, since 70ies)

  van Rijsbergen C.J. Inform. Retr.. Butterworths, London, 2nd edn., **1979**.

- Probabilistic IR (Croft, Harper, 70ies)

  Croft W.B. and Harper D.J. Using probabilistic models of document retrieval without relevance information. J. Doc., 35:285–295, **1979**.

- Probabilistic Indexing (Fuhr & Co.,late 80ies-90ies)

  Norbert Fuhr. 1989. Models for retrieval with probabilistic indexing. *Inf. Process. Manage.* 25, 1, 55-72, **1989**.

# Let us recap probability theory

- Bayesian probability formulas

$$p(a \mid b)\, p(b) = p(a \cap b) = p(b \mid a)\, p(a)$$

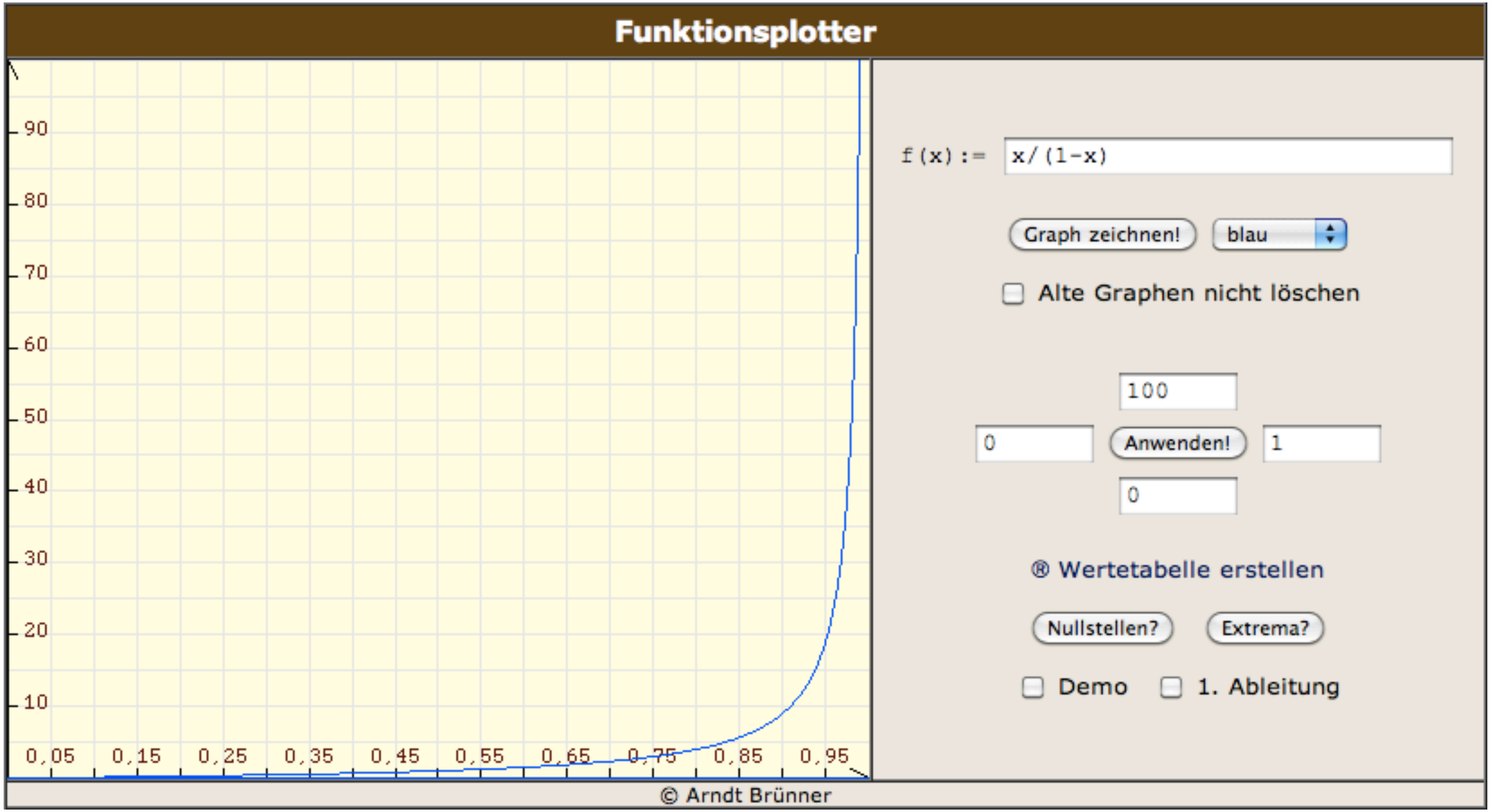
$$p(a \mid b) = \frac{p(b \mid a)\, p(a)}{p(b)}$$

$$p(\overline{a} \mid b)\, p(b) = p(b \mid \overline{a})\, p(\overline{a})$$

- Odds:

$$O(y) = \frac{p(y)}{p(\overline{y})} = \frac{p(y)}{1 - p(y)}$$

# Odds vs. Probabilities



# Probability Ranking Principle

Let x be a document in the retrieved collection.
Let R represent Relevance=true of a document w.r.t. given (fixed) query and let NR represent Relevance=false.

Need to find p(R|x) - probability that a retrieved document *x* is **relevant.**

$$p(R \mid x) = \frac{p(x \mid R)\,p(R)}{p(x)}$$

$$p(NR \mid x) = \frac{p(x \mid NR)\,p(NR)}{p(x)}$$

p(*R*),p(*NR*) - prior probability of retrieving a relevant or non-relevant document, respectively

p(x|R), p(x|NR) - probability that if a relevant or non-relevant document is retrieved, it is *x*.



van Rijsbergen, Cornelis Joost. Information Retrieval, 2nd edition. Butterworths, **1979**

# Probability Ranking Principle

$$p(R \mid x) = \frac{p(x \mid R)\, p(R)}{p(x)}$$

$$p(NR \mid x) = \frac{p(x \mid NR)\, p(NR)}{p(x)}$$

Ranking Principle (Bayes' Decision Rule):
**If p(*R*|*x*) > p(*NR*|*x*) then *x* is relevant,**
**otherwise *x* is not relevant**

- Note: $p(R \mid x) + p(NR \mid x) = 1$

# Probability Ranking Principle

<u>Claim:</u> PRP minimizes the average probability of error

$$p(error \mid x) = \begin{cases} p(R \mid x) & \text{If we decide } \textbf{NR} \\ p(NR \mid x) & \text{If we decide } \textbf{R} \end{cases}$$

$$p(error) = \sum_x p(error \mid x)\, p(x)$$

p(error) is minimal when all p(error|x) are minimimal [Ripley, 1996].
Bayes' decision rule minimizes each p(error|x).

Ripley, B. D. Pattern Recognition and Neural Networks.
Cambridge University Press. **1996**

# Probability Ranking Principle

- More complex case: retrieval costs.
  - *C* - cost of retrieval of <u>relevant</u> document
  - *C′* - cost of retrieval of <u>non-relevant</u> document
  - let *d*, be a document
- Probability Ranking Principle: if

$$\boxed{C \cdot p(R \mid d) + C' \cdot (1 - p(R \mid d)) \leq C \cdot p(R \mid d') + C' \cdot (1 - p(R \mid d'))}$$

for all *d′ not yet retrieved*, then *d* **is the next document to be retrieved**

# PRP: Issues (Problems?)

- How do we compute all those probabilities?
  - Cannot compute exact probabilities, have to use estimates.
  - **Binary Independence Retrieval (BIR)**
    - See below
- Restrictive assumption
  - "Relevance" of each document is independent of relevance of other documents

# Probability Ranking Principle (PRP)

- Let us assume that:
  - *C* - cost of retrieval of relevant document
  - *C′* - cost of retrieval of non-relevant document
- Documents *d* are ranked according the to the Probability Ranking Principle when it holds that if *d* is the *next document retrieved* then

$$C \cdot p(R \mid d) + C' \cdot (1 - p(R \mid d)) \leq C \cdot p(R \mid d') + C' \cdot (1 - p(R \mid d'))$$

is true for all documents *d′ not yet retrieved*

# Relevance models

- Given: PRP to be applied
- Need to estimate probability: $P(R|q,d)$
- Binary Independence Retrieval (BIR):
  - Many documents $D$ - one query $q$
  - Estimate $P(R|q,d)$ by considering whether $d \in D$ is relevant for $q$
- Binary Independence Indexing (BII):
  - One document $d$ - many queries $Q$
  - Estimate $P(R|q,d)$ by considering whether a document $d$ is relevant for a query $q \in Q$

# Binary Independence Retrieval

- Traditionally used in conjunction with PRP
- **"Binary" = Boolean**: documents are represented as binary vectors of terms:
  - $\vec{x} = (x_1, \ldots, x_n)$
  - $x_i = 1$ iff term *i* is present in document *x*.
- **"Independence":** terms occur in documents independently
- Different documents can be modeled as same vector.

# Binary Independence Retrieval

- Queries: binary vectors of terms
- Given query **q**,
  - for each document **d** need to compute **p(Relevant=true|q,d).**
  - replace with computing **p(Relevant=true|q,x)** where **x** is vector representing **d**
- Interested only in ranking
- Will use odds:

$$O(R \mid q, \vec{x}) = \frac{p(R \mid q, \vec{x})}{p(NR \mid q, \vec{x})} = \frac{p(R \mid q)}{p(NR \mid q)} \cdot \frac{p(\vec{x} \mid R, q)}{p(\vec{x} \mid NR, q)}$$

# Binary Independence Retrieval

$$O(R \mid q, \vec{x}) = \frac{p(R \mid q, \vec{x})}{p(NR \mid q, \vec{x})} = \boxed{\frac{p(R \mid q)}{p(NR \mid q)}} \cdot \boxed{\frac{p(\vec{x} \mid R, q)}{p(\vec{x} \mid NR, q)}}$$

Constant for each query

Needs estimation

- Using **Independence** Assumption:

$$\frac{p(\vec{x} \mid R, q)}{p(\vec{x} \mid NR, q)} = \prod_{i=1}^{n} \frac{p(x_i \mid R, q)}{p(x_i \mid NR, q)}$$

- So : $O(R \mid q, d) = O(R \mid q) \cdot \prod_{i=1}^{n} \frac{p(x_i \mid R, q)}{p(x_i \mid NR, q)}$

# Binary Independence Retrieval

$$O(R \mid q, d) = O(R \mid q) \cdot \prod_{i=1}^{n} \frac{p(x_i \mid R, q)}{p(x_i \mid NR, q)}$$

- Since $x_i$ is either 0 or 1:

$$O(R \mid q, d) = O(R \mid q) \cdot \prod_{x_i = 1} \frac{p(x_i = 1 \mid R, q)}{p(x_i = 1 \mid NR, q)} \cdot \prod_{x_i = 0} \frac{p(x_i = 0 \mid R, q)}{p(x_i = 0 \mid NR, q)}$$

- Let $p_i = p(x_i = 1 \mid R, q); \quad r_i = p(x_i = 1 \mid NR, q);$

- Assume, for all terms not occuring in the query ($q_i=0$) $p_i = r_i$

Then…

# Binary Independence Retrieval

$$O(R \mid q, \vec{x}) = \boxed{O(R \mid q)} \cdot \prod_{x_i = q_i = 1} \frac{p_i}{r_i} \cdot \prod_{\substack{x_i = 0 \\ q_i = 1}} \frac{1 - p_i}{1 - r_i}$$

All matching terms

Non-matching query terms

$$= \boxed{O(R \mid q)} \cdot \prod_{x_i = q_i = 1} \frac{p_i(1 - r_i)}{r_i(1 - p_i)} \cdot \prod_{q_i = 1} \frac{1 - p_i}{1 - r_i}$$

All matching terms

All query terms

# Binary Independence Retrieval

$$O(R \mid q, \vec{x}) = \boxed{O(R \mid q)} \cdot \boxed{\prod_{x_i = q_i = 1} \frac{p_i(1-r_i)}{r_i(1-p_i)}} \cdot \boxed{\prod_{q_i = 1} \frac{1-p_i}{1-r_i}}$$

Constant for each query

Only quantity to be estimated for rankings

- Optimize Retrieval Status Value:

$$RSV = \log \prod_{x_i = q_i = 1} \frac{p_i(1-r_i)}{r_i(1-p_i)} = \sum_{x_i = q_i = 1} \log \frac{p_i(1-r_i)}{r_i(1-p_i)}$$

# Binary Independence Retrieval

- All boils down to computing RSV.

$$RSV = \log \prod_{x_i = q_i = 1} \frac{p_i(1-r_i)}{r_i(1-p_i)} = \sum_{x_i = q_i = 1} \log \frac{p_i(1-r_i)}{r_i(1-p_i)}$$

$$RSV = \sum_{x_i = q_i = 1} c_i;$$

For all query terms i:
Find docs containing term i (→ inverted index)

$$c_i = \log \frac{p_i(1-r_i)}{r_i(1-p_i)} = \log \frac{p_i}{(1-p_i)} + \log \frac{(1-r_i)}{r_i}$$

So, how do we compute $c_i$'s from our data ?

# Binary Independence Retrieval

- Estimating RSV coefficients.
- For each term i look at the following table:

| Document | Relevant | Non-Relevant | Total |
|---|---|---|---|
| $X_i=1$ | $s$ | $n-s$ | $n$ |
| $X_i=0$ | $S-s$ | $N-n-S+s$ | $N-n$ |
| Total | $S$ | $N-S$ | $N$ |

$$p_i = p(x_i = 1 \mid R, q); \quad r_i = p(x_i = 1 \mid NR, q);$$

- Estimates: $p_i \approx \dfrac{s}{S} \qquad r_i \approx \dfrac{(n-s)}{(N-S)}$

# Binary Independence Retrieval

- Estimating RSV coefficients.
- For each term *i* look at the following table:

| Document | Relevant | Non-Relevant | Total |
|---|---|---|---|
| $X_i$=1 | $s$ | $n$-$s$ | $n$ |
| $X_i$=0 | $S$-$s$ | $N$-$n$-$S$+$s$ | $N$-$n$ |
| Total | $S$ | $N$-$S$ | $N$ |

- Estimates: $p_i \approx \frac{s}{S}$ $\quad r_i \approx \frac{(n-s)}{(N-S)}$

$$c_i \approx K(N,n,S,s) = \log \frac{s/(S-s)}{(n-s)/(N-n-S+s)}$$

# Avoid division by 0

$$c_i \approx K(N,n,S,s) = \log \frac{(s+1/2)\big/(S-s+1/2)}{(n-s+1/2)\big/(N-n-S+s+1/2)}$$

# Estimation in practice

- If non-relevant documents are approximated by the whole collection ($S=s=0$), then $r_i$ (prob. of occurrence term i in non-relevant documents for query) is $n/N$ and
  - $\log (1-r_i)/r_i = \log (N-n)/n \approx \log(1+ (N-n)/n) = \log N/n = \mathrm{IDF}$
- Idea cannot be easily extended to $p_i$
- Estimate $p_i$ (probability of occurrence of term $i$ in relevant docs):
  - From relevant documents if we know some
  - Use constant $0.5$ – then just get idf weighting of terms ($p_i$ and $1-p_i$ cancel out)
  - Determine by exploratory data analysis: $c_i = k_i + \log N/n$
- We have a nice theoretical foundation of TF.IDF (in the binary case: TF=1 or TF=0)

Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. In *Document retrieval systems*, Vol. 3. Taylor Graham Publishing, London, UK, UK 132-142. **1988**.

Greiff, Warren R., A Theory of Term Weighting Based on Exploratory Data Analysis. In: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 11-19, **1998**.

Robertson S.E., Understanding inverse document frequency: On theoretical arguments for idf. J. Doc., 60:503–520, **2004**.

# Iteratively estimating $p_i$

## Expectation Maximization:

1. Assume that $p_i$ constant over all $q_i$ in query
   - $p_i = 0.5$ (even odds) for any given doc
2. Determine guess of relevant document set from subset $V$:
   - $V$ is fixed size set of highest ranked documents on this model
3. We need to improve our guesses for $p_i$ and $r_i$, so
   - Use distribution of $q_i$ in docs in $V$. Let $V_i$ be set of documents containing $q_i$
     - $p_i = |V_i| / |V|$
   - Assume if not retrieved then not relevant
     - $r_i = (n_i - |V_i|) / (N - |V|)$
4. Go to 2. until convergence then return ranking

# Probabilistic Relevance Feedback

1. Guess a preliminary probabilistic description of *R* and use it to retrieve a first set of documents V, as above.
2. Interact with the user to refine the description: learn some definite members of R and NR
3. Reestimate $p_i$ and $r_i$ on the basis of these
   – Or can combine new information with original guess (use Bayesian prior):

$$p_i^{(2)} = \frac{|V_i| + \lambda p_i^{(1)}}{|V| + \lambda}$$

$\lambda$ is prior weight

4. Repeat, thus generating a succession of approximations to *R*.

# Binary Independence Indexing

- "Learning" from queries
  - More queries: better results

$$p(R \mid \vec{q}, \vec{x}) = \frac{p(\vec{q} \mid \vec{x}, R)\, p(R \mid \vec{x})}{p(\vec{q} \mid \vec{x})}$$

- **p(q|x,R)** - probability that if document **x** had been deemed relevant, query **q** had been asked
- The rest of the framework is similar to BIR

# Binary Independence Retrieval vs. Binary Independence Indexing

**BIR**

- Many Documents,
  One Query
- Bayesian Probability:

$$p(R \mid \vec{q}, \vec{x}) = \frac{p(\vec{x} \mid \vec{q}, R)\, p(R \mid \vec{q})}{p(\vec{x} \mid \vec{q})}$$

- Varies: document representation
- Constant: query (representation)

**BII**

- One Document,
  Many Queries
- Bayesian Probability

$$p(R \mid \vec{q}, \vec{x}) = \frac{p(\vec{q} \mid \vec{x}, R)\, p(R \mid \vec{x})}{p(\vec{q} \mid \vec{x})}$$

- Varies: query
- Constant: document

# PRP and BIR/BII: The lessons

- Getting reasonable approximations of probabilities is possible.
- Simple methods work only with restrictive assumptions:
  - **term independence**
  - **terms not in query do not affect the outcome**
  - **boolean representation of documents/queries**
  - **document relevance values are independent**
- Some of these assumptions can be removed

# Acknowledgment

- Some of the next slides are based on a presentation "An Overview of Bayesian Network-based Retrieval Models" by Juan Manuel Fernández Luna

# Bayesian Nets in IR

- Inference Network Model

  Howard Turtle, and W. Bruce Croft. Inference networks for document retrieval. In *Proc. SIGIR*, pp. 1-24. ACM Press. **1989**.

  Howard Turtle, and W. Bruce Croft. Evaluation of an inference network-based retrieval model. *TOIS* 9 (3): 187-222. **1991**.

- Belief Network Model

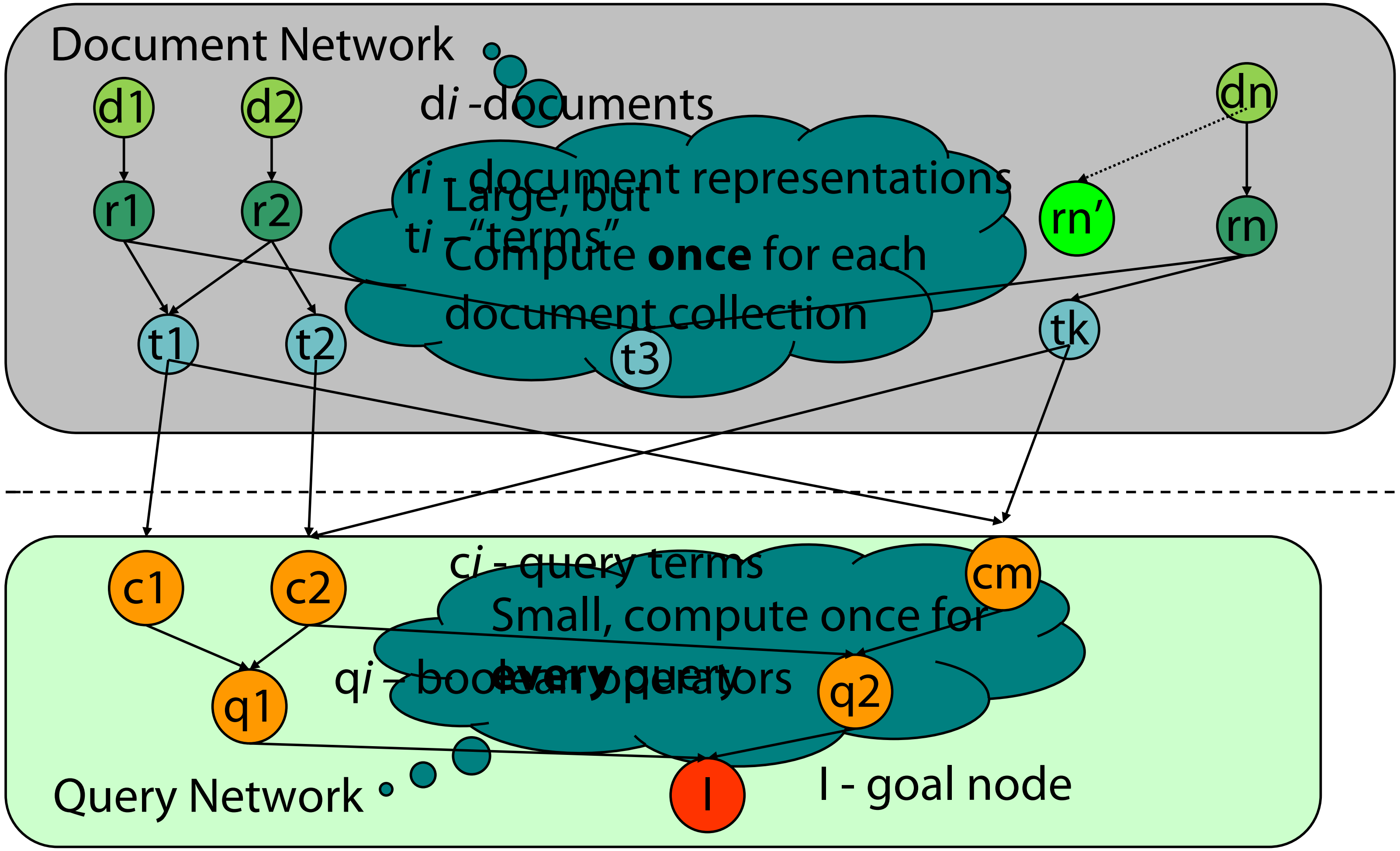
  Berthier A. N. Ribeiro and Richard Muntz. A belief network model for IR. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval* (SIGIR '96). ACM, New York, NY, USA, 253-260. **1996**.

- Bayesian Network Retrieval Model

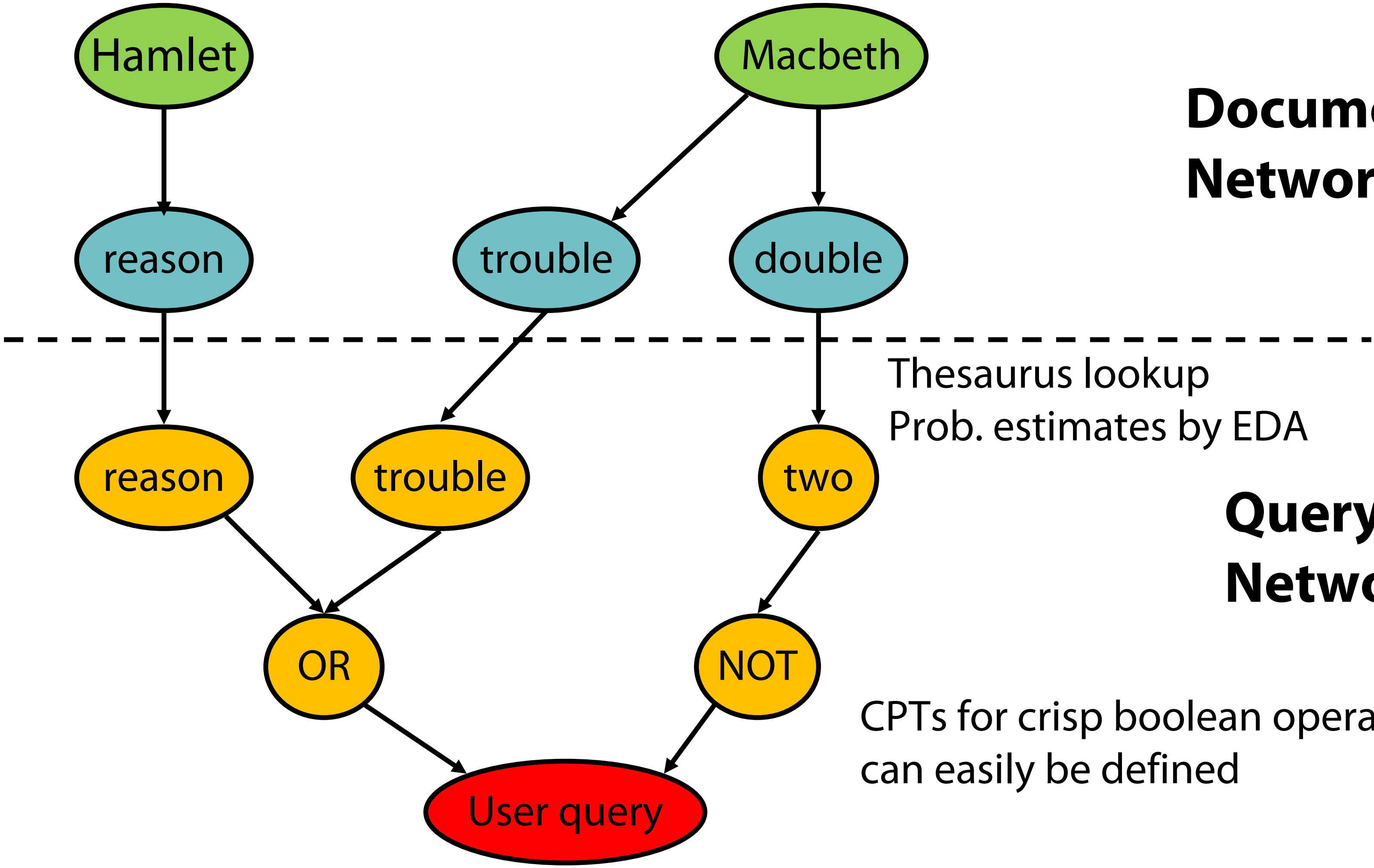
  Luis M.de Campos, Juan M.Fernández-Luna, Juan F.Huete, Clustering terms in the Bayesian network retrieval model: a new approach with two term-layers, Applied Soft Computing, Volume 4, Issue 2, Pages 149-158, May **2004**.

Some subsequent slides are based on a presentation "An Overview of Bayesian Network-based Retrieval Models" by Juan Manuel Fernández Luna

# Inference Network Model



# Inference Network Model: *“reason trouble –two”*




**Document Network**

Thesaurus lookup
Prob. estimates by EDA

**Query Network**

CPTs for crisp boolean operators can easily be defined

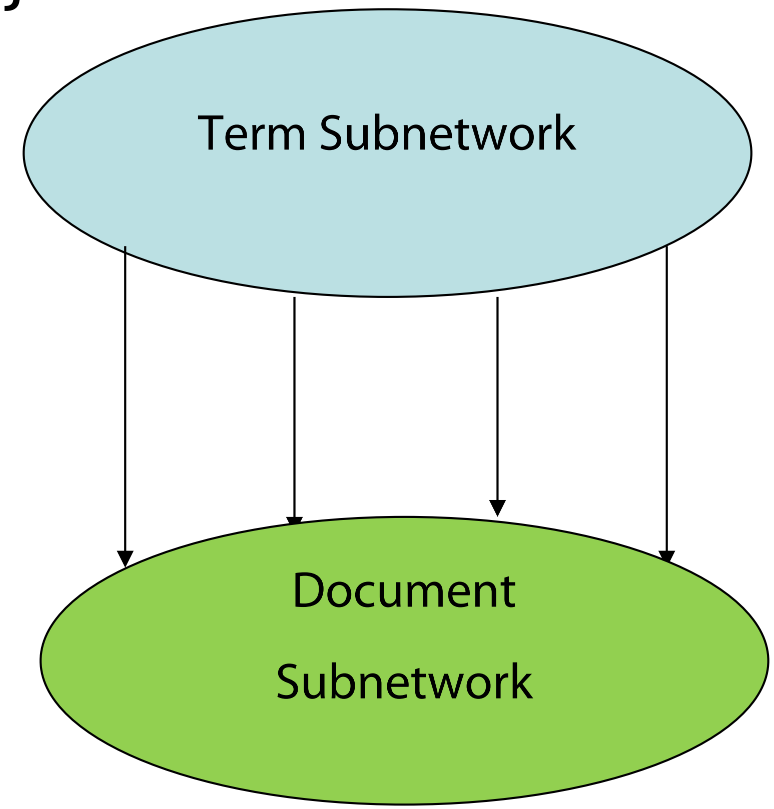
# Inference Network Model [89, 91]

- Construct document network (once !)
- For each query
  - Construct query network (on the fly !)
  - Attach it to document network
  - Find doc subset $\mathbf{d_i}$**s** maximizing **P(I | $\mathbf{d_i}$s)** (best subset)
  - Retrieve these $\mathbf{d_i}$**s** as the answer to query.
- But:
  - Powerset of docs defines huge search space
  - Exact answers for queries **P(I | $\mathbf{d_i}$s)** rather "expensive"
    - BN structure has loops (no polytree)

# Bayesian Network Retrieval Model [96]

Mentioned or
not mentioned

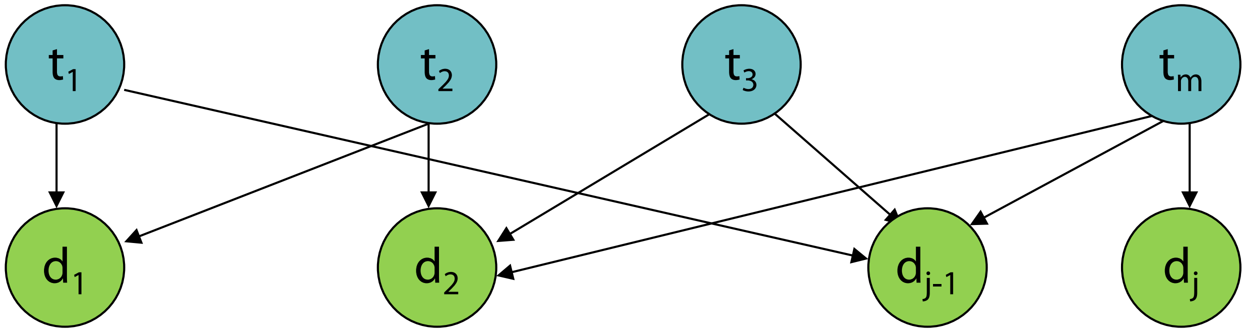
$T_i \in \{\neg t_i, t_i\}$




$D_j \in \{\neg d_j, d_j\}$

Relevant or
not relevant

# Belief Network Model [1996]




Probability Distributions:

- Term nodes: $p(t_j)=1/m$, $p(\neg t_j)=1-p(t_j)$
- Document nodes: $p(D_j \mid Pa(D_j))$, $\forall D_j$

$$p(d_j \mid Q) = \alpha \sum_{\tau} p(d_j \mid \tau)\, p(Q \mid \tau)\, p(\tau)$$

But... If a document has been indexed by, say, 30 "most important" terms, we need to estimate (and store) $2^{30}$ probabilities.

# Bayesian Network Retrieval Model

## Probability functions

$$\boxed{p(D_j \mid pa(D_j)) = \sum_{\substack{T_i \in D_j \\ t_j \in pa(D_j)}} w_{ij}}$$

$$where\ 0 \le w_{ij}\ and \sum_{T_i \in D_j} w_{ij} \le 1$$

$pa(D_j)$ being a configuration of the parents of $D_j$.

# Bayesian Network Retrieval Model

Retrieval:

1. Instantiate $T_Q \in Q$ to $T_Q = true$ (evidence)
2. Run a propagation algorithm in the network.
3. Rank the documents according $p(d_j \mid Q), \forall D_j$

**Note:**

Graph is not a polytree

$\Downarrow$

Only an approximation of probability values computed
Ranking order only approximated

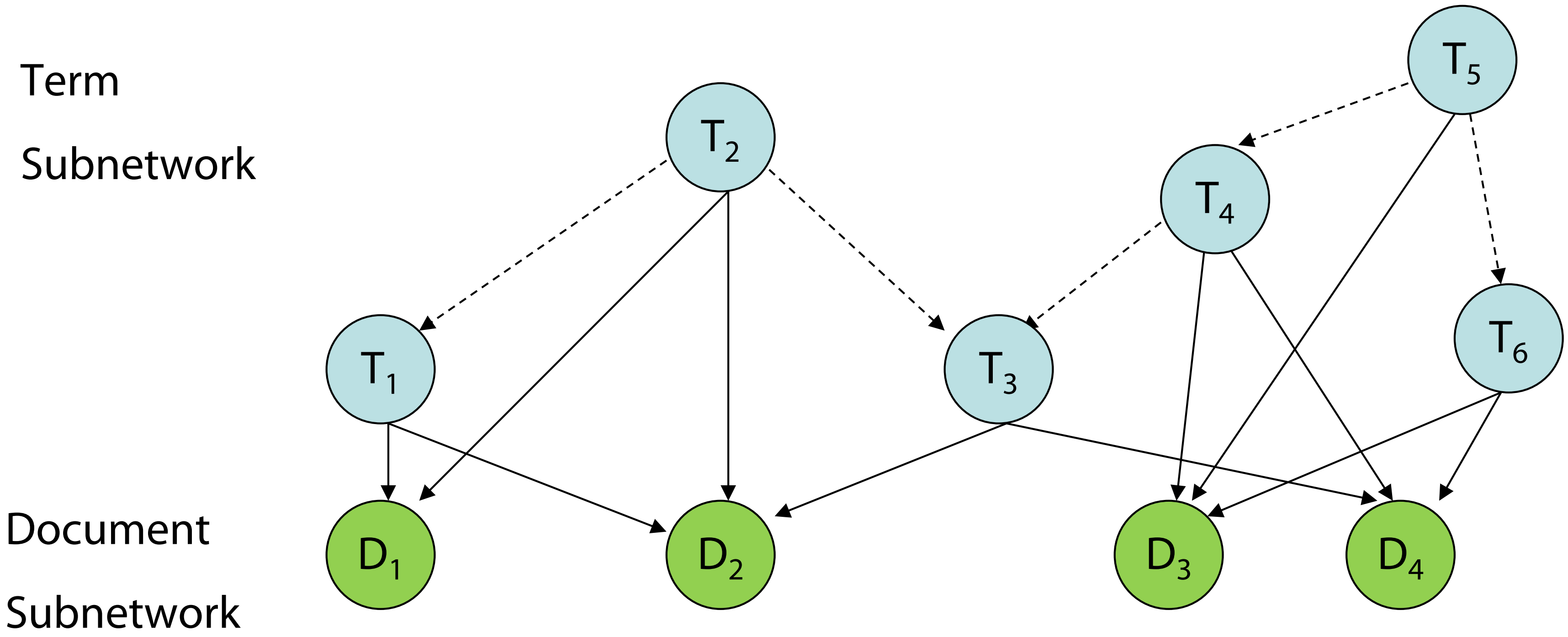
# Bayesian Network Retrieval Model

Removing the term independency restriction:

- We are interested in representing the main relationships among terms in the collection.

**Term subnetwork ≡ Polytree**

# Bayesian Network Retrieval Model



# Bayesian Network Retrieval Model

Conditional Distributions
(term nodes with parents):

(based on Jaccard´s coefficient)

$$p(\overline{t_i} \mid pa(T_i)) = \frac{n(<\overline{t_i}, pa(T_i)>)}{n(<\overline{t_i}>) + n(pa(T_i)) - n(<\overline{t_i}, pa(T_i)>)}$$

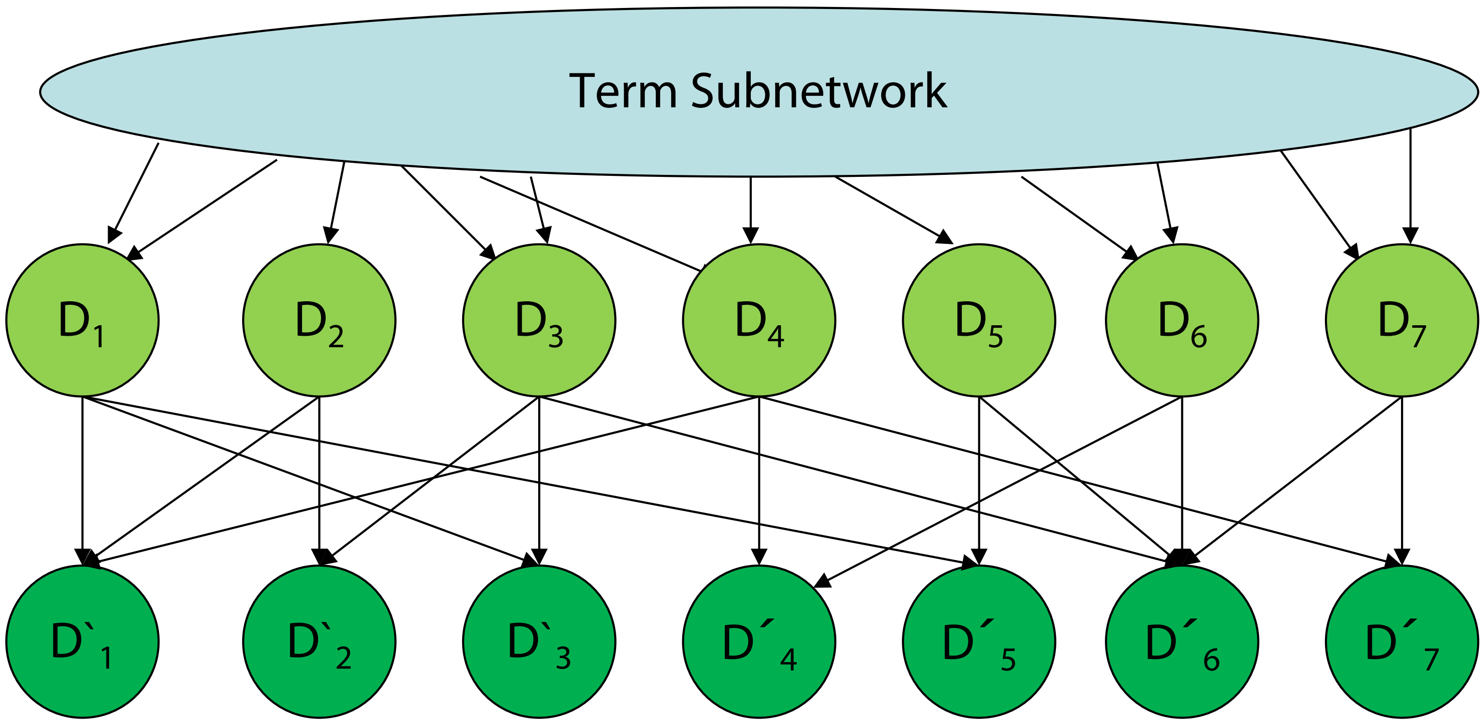
$$p(t_i \mid pa(T_i)) = 1 - p(\overline{t_i} \mid pa(T_i))$$

# PRP and Recommendations

Those who retrieved $d_i$ were also interested in $d_j$

Compute $p(d_j|d_i)$

Use noisy-or CPTs



# Bayesian Network Retrieval Model

Retrieval?

1. Compute $p(d_j|Q)$, $\forall D_j$

   (1st document layer)

2. Compute $p(d'_j|Q)$, $\forall D'_j$

   (2nd document layer)

$$\boxed{p(d'_j \mid Q) = \frac{1}{S_j} \sum_{D_i \in Pa(D'_j)} p(d_j \mid d_i)\, p(d_i \mid Q)}$$

Where $S_j$ is a normalising constant

# Language Models

- A new approach to probabilistic IR, derived from work in automatic speech recognition, OCR and MT
- Language models attempt to statistically model the use of language in a collection to estimate the probability that a query was *generated* from a particular document
- The assumption is, roughly, that if the query could have come from the document, then that document is likely to be relevant

Acknowledgment: Slides taken from a presentation on "Principles of Information Retrieval" by Ray Larson

Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval* (SIGIR '98). ACM, New York, NY, USA, 275-281. **1998**.

# Ponte and Croft LM

- For the original Ponte and Croft Language Models the goal is to estimate:

$$p(Q \mid M_d)$$

- That is, the probability of a **query** given the **language model of document** *d*. One approach would be to use:

$$p_{ml}(t \mid M_d) = \frac{tf_{(t,d)}}{dl_d}$$

- Maximum likelihood estimate of the probability of term *t* in document *d*, where $tf_{(t,d)}$ is the term count in doc *d* and $dl_d$ is the total number of tokens in document *d*

# Ponte and Croft LM

- The ranking formula then coult be:
  - For each document d in the collection…
- There are problems with this (*not least of which is that it is zero for any document that doesn't contain all query terms*)
- A better estimator might be the mean probability of *t* in documents containing it ($df_t$ is the document frequency of *t*)

$$p(Q \mid M_d) = \prod_{t \in Q} p_{ml}(t \mid M_d)$$

$$p_{avg}(t) = \frac{\left(\sum_{d_{(t \in d)}} p_{ml}(t \mid M_d)\right)}{df_t}$$

# Ponte and Croft LM

- There are still problems with this estimator, in that it treats each document with *t* as if it came from the SAME language model
- The final form with a "risk adjustment" is as follows…

# Ponte and Croft LM

- Let,

$$\hat{p}(t \mid M_d) = \begin{cases} p_{ml}(t,d)^{(1.0-\hat{R}_{t,d})} \times p_{avg}(t)^{\hat{R}_{t,d}} & \text{if } tf_{(t,d)} > 0 \\ \dfrac{cf_t}{cs} & \text{otherwise} \end{cases}$$

- Where

$$\hat{R}_{t,d} = \left( \frac{1.0}{(1.0 + \bar{f}_t)} \right) \times \left( \frac{\bar{f}_t}{(1.0 + \bar{f}_t)} \right)^{tf_{t,d}}$$

- I.e. the geometric distribution, $f_t$ is the mean term freq in the doc and $cf_t$ is the raw term count of $t$ in the collection and $cs$ is the collection size (in term tokens)

- Then,

$$\hat{p}(Q \mid M_d) = \prod_{t \in Q} \hat{p}(t \mid M_d) \times \prod_{t \notin Q} 1.0 - \hat{p}(t \mid M_d)$$

# Ponte and Croft LM

- When compared to a fairly standard tfidf retrieval on the TREC collection this basic Language model provided significantly better performance (5% more relevant documents were retrieved overall, with about a 20% increase in mean average precision
- Additional improvement was provided by *smoothing* the estimates for low-frequency terms

# Lavrenko and Croft LM

- Notion of relevance lacking
- → Lavrenko and Croft
  - Reclaim ideas of the probability of relevance from earlier probabilistic models and includes them into the language modeling framework with its effective estimation techniques

Victor Lavrenko and W. Bruce Croft. Relevance based language models. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval* (SIGIR '01). ACM, New York, NY, USA, 120-127. **2001**.

# BIR vs. Ponte and Croft

- The basic form of the older probabilistic model (Binary independence model) is

$$P(D \mid R) = \prod_{w \in D} P(w \mid R) \times \prod_{w \notin D} (1.0 - P(w \mid N))$$

- While the Ponte and Croft Language Model is very similar

$$P(Q \mid M_d) = \prod_{t \in Q} P(t \mid M_d) \times \prod_{t \notin Q} (1.0 - P(t \mid M_d))$$

# Lavrenko and Croft LM

- Similarity in FORM obvious, what distinguishes the two is how the individual word (term) probabilities are estimated
- Basically they estimate the probability of observing a word in the relevant set using the probability of co-occurrence between the words and the query adjusted by collection level information

$$P(t_1,\ldots,t_n \mid M_d) = \prod_{i=1}^{n} \big((1-\lambda)P(t_i \mid C) + \lambda P(t_i \mid D)\big)$$

- Where λ is a parameter derived from a test collection
- Lurking danger of overtraining (word like "the", "of", "and" or misspellings): focus on modeling terms distinguishing the model from the general model of a collection [Zaragoza et al. 03]



Zaragoza, H., Hiemstra, D., Tipping, M., & Robertson, S. E. *Bayesian Extension to the Language Model for Ad Hoc Information Retrieval*. 4-9. **2003**

# Good and Bad News

- Standard Vector Space Model
  - Empirical for the most part; success measured by results
  - Few properties provable
- Probabilistic Models
  - Advantages
    - Based on a firm theoretical foundation
    - But: construction of the BN is engineering work
    - Theoretically justified optimal ranking scheme
  - Disadvantages
    - Binary word-in-doc weights (not using term frequencies)
    - Often: Independence of terms (can be alleviated)
    - Amount of computation required is high
    - Has never worked convincingly better in practice

# Main difference

- Vector space approaches can benefit from dimension reduction (and need it indeed)
- Actually, dimension reduction is indeed required for relevance feedback in vector space models
- Dimension reduction: Compute "topics"
- Can we exploit topics in probability-based retrieval models?