# Lecture 4.
# Set Theoretic IR Models

# Alternative Set Theoretic Models

- Set-Based Model
- Extended Boolean Model
- Fuzzy Set Model

# Set-Based Model

# Set-Based Model

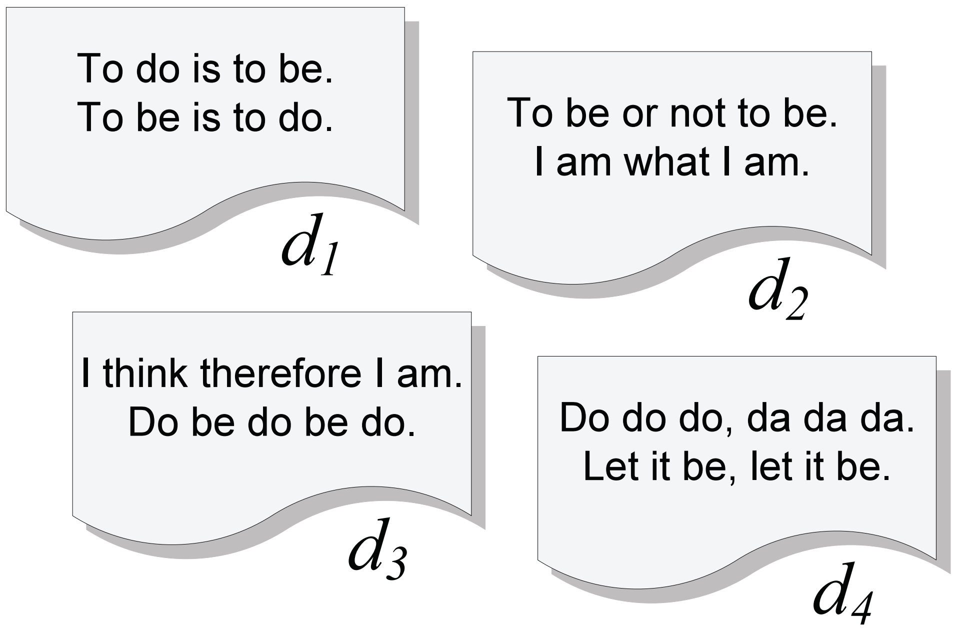
- This is a more recent approach (2005) that combines set theory with a vectorial ranking
- The fundamental idea is to use mutual dependencies among index terms to improve results
- Term dependencies are captured through **termsets**, which are sets of correlated terms
- The approach, which leads to improved results with various collections, constitutes the first IR model that effectively took advantage of term dependence with general collections

# Termsets

- **Termset** is a concept used in place of the index terms
- A termset $S_i = \{k_a, k_b, ..., k_n\}$ is a subset of the terms in the collection
- If all index terms in $S_i$ occur in a document $d_j$ then we say that the termset $S_i$ occurs in $d_j$
- There are $2^t$ termsets that might occur in the documents of a collection, where $t$ is the vocabulary size
  - However, most combinations of terms have no semantic meaning
  - Thus, the actual number of termsets in a collection is far smaller than $2^t$

# Termsets

- Let $t$ be the number of terms of the collection
- Then, the set $V_S = \{S_1, S_2, ..., S_{2^t}\}$ is the **vocabulary-set** of the collection
- To illustrate, consider the document collection below

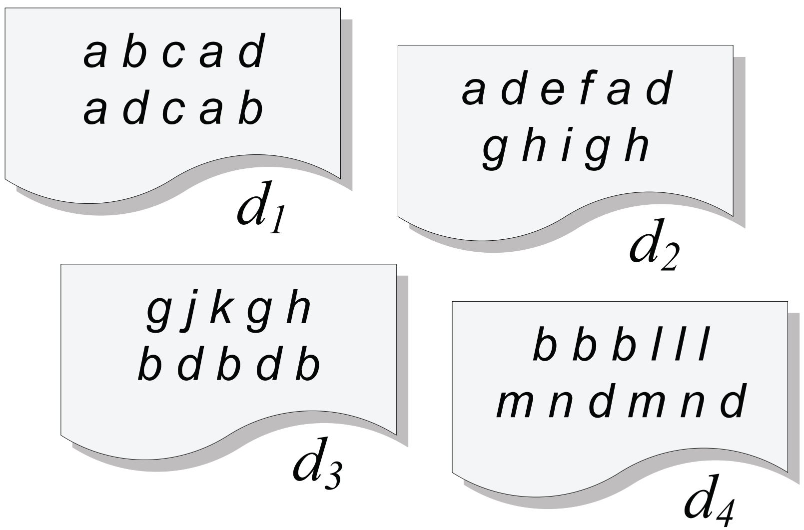


# Termsets

- To simplify notation, let us define

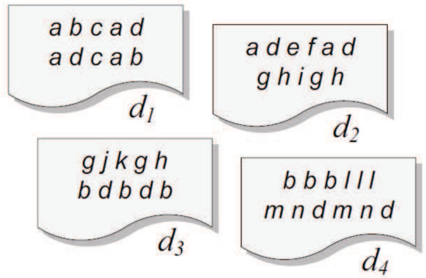
| | | | | |
|---|---|---|---|---|
| $k_a$ = to | $k_d$ = be | $k_g$ = I | $k_j$ = think | $k_m$ = let |
| $k_b$ = do | $k_e$ = or | $k_h$ = am | $k_k$ = therefore | $k_n$ = it |
| $k_c$ = is | $k_f$ = not | $k_i$ = what | $k_l$ = da | |

- Further, let the letters $a \ldots n$ refer to the index terms $k_a \ldots k_n$ , respectively



# Termsets

- Consider the query $q$ as "to do be it", i.e. $q = \{a, b, d, n\}$
- For this query, the vocabulary-set is as below




| Termset | Set of Terms | Documents |
|---|---|---|
| $S_a$ | $\{a\}$ | $\{d_1, d_2\}$ |
| $S_b$ | $\{b\}$ | $\{d_1, d_3, d_4\}$ |
| $S_d$ | $\{d\}$ | $\{d_1, d_2, d_3, d_4\}$ |
| $S_n$ | $\{n\}$ | $\{d_4\}$ |
| $S_{ab}$ | $\{a, b\}$ | $\{d_1\}$ |
| $S_{ad}$ | $\{a, d\}$ | $\{d_1, d_2\}$ |
| $S_{bd}$ | $\{b, d\}$ | $\{d_1, d_3, d_4\}$ |
| $S_{bn}$ | $\{b, n\}$ | $\{d_4\}$ |
| $S_{abd}$ | $\{a, b, d\}$ | $\{d_1\}$ |
| $S_{bdn}$ | $\{b, d, n\}$ | $\{d_4\}$ |

Notice that there are 11 termsets that occur in our collection, out of the maximum of 15 termsets that can be formed with the terms in $q$

# Termsets

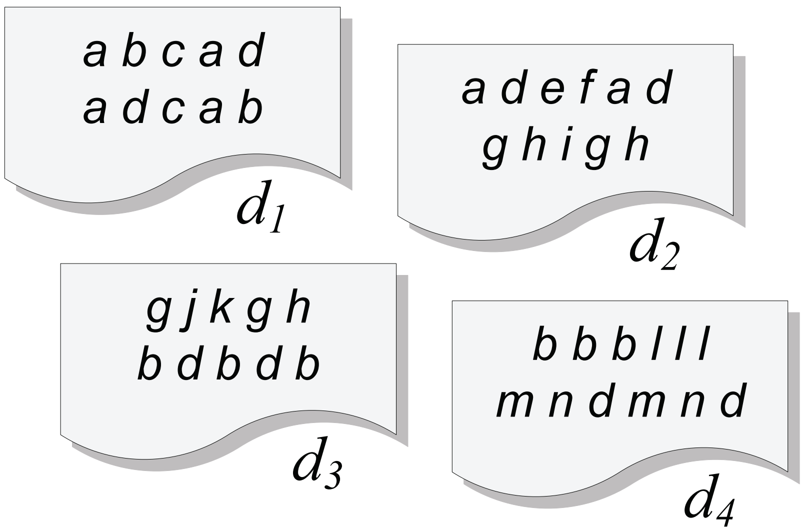
- At query processing time, only the termsets generated by the query need to be considered
- A termset composed of $n$ terms is called an $n$-termset
- Let $\mathcal{N}_i$ be the number of documents in which $S_i$ occurs
- An $n$-termset $S_i$ is said to be **frequent** if $\mathcal{N}_i$ is greater than or equal to a given threshold
  - This implies that an $n$-termset is frequent if and only if all of its $(n-1)$-termsets are also frequent
  - **Frequent termsets** can be used to reduce the number of termsets to consider with long queries

# Termsets

- Let the threshold on the frequency of termsets be 2
- To compute all frequent termsets for the query $q = \{a, b, d, n\}$ we proceed as follows

| Termset | Set of Terms | Documents |
|---|---|---|
| $S_a$ | $\{a\}$ | $\{d_1, d_2\}$ |
| $S_b$ | $\{b\}$ | $\{d_1, d_3, d_4\}$ |
| $S_d$ | $\{d\}$ | $\{d_1, d_2, d_3, d_4\}$ |
| $S_n$ | $\{n\}$ | $\{d_4\}$ |
| $S_{ab}$ | $\{a, b\}$ | $\{d_1\}$ |
| $S_{ad}$ | $\{a, d\}$ | $\{d_1, d_2\}$ |
| $S_{bd}$ | $\{b, d\}$ | $\{d_1, d_3, d_4\}$ |
| $S_{bn}$ | $\{b, n\}$ | $\{d_4\}$ |
| $S_{abd}$ | $\{a, b, d\}$ | $\{d_1\}$ |
| $S_{bdn}$ | $\{b, d, n\}$ | $\{d_4\}$ |

1. Compute the frequent 1-termsets and their inverted lists:
   - $S_a = \{d_1, d_2\}$
   - $S_b = \{d_1, d_3, d_4\}$
   - $S_d = \{d_1, d_2, d_3, d_4\}$
2. Combine the inverted lists to compute frequent 2-termsets:
   - $S_{ad} = \{d_1, d_2\}$
   - $S_{bd} = \{d_1, d_3, d_4\}$
3. Since there are no frequent 3-termsets, stop



# Termsets

- Notice that there are only 5 *frequent* termsets in our collection
- Inverted lists for frequent $n$-termsets can be computed by starting with the inverted lists of frequent 1-termsets
  - Thus, the only indice that is required are the standard inverted lists used by any IR system
- This is reasonably fast for short queries up to 4-5 terms

# Ranking Computation

- The ranking computation is based on the vector model, but adopts termsets instead of index terms
- Given a query $q$, let
  - $\{S_1, S_2, \ldots\}$ be the set of all termsets originated from $q$
  - $\mathcal{N}_i$ be the number of documents in which termset $S_i$ occurs
  - $N$ be the total number of documents in the collection
  - $\mathcal{F}_{i,j}$ be the frequency of termset $S_i$ in document $d_j$
- For each pair $[S_i, d_j]$ we compute a weight $\mathcal{W}_{i,j}$ given by

$$\mathcal{W}_{i,j} = \begin{cases} (1 + \log \mathcal{F}_{i,j}) \log(1 + \frac{N}{\mathcal{N}_i}) & \texttt{if} \;\; \mathcal{F}_{i,j} > 0 \\ 0 & \mathcal{F}_{i,j} = 0 \end{cases}$$

- We also compute a $\mathcal{W}_{i,q}$ value for each pair $[S_i, q]$

# Ranking Computation

- Consider
  - query $q = \{a, b, d, n\}$
  - document $d_1$ = ``a b c a d a d c a b''

| Termset | Set of Terms | Documents |
|---|---|---|
| $S_a$ | $\{a\}$ | $\{d_1, d_2\}$ |
| $S_b$ | $\{b\}$ | $\{d_1, d_3, d_4\}$ |
| $S_d$ | $\{d\}$ | $\{d_1, d_2, d_3, d_4\}$ |
| $S_n$ | $\{n\}$ | $\{d_4\}$ |
| $S_{ab}$ | $\{a, b\}$ | $\{d_1\}$ |
| $S_{ad}$ | $\{a, d\}$ | $\{d_1, d_2\}$ |
| $S_{bd}$ | $\{b, d\}$ | $\{d_1, d_3, d_4\}$ |
| $S_{bn}$ | $\{b, n\}$ | $\{d_4\}$ |
| $S_{abd}$ | $\{a, b, d\}$ | $\{d_1\}$ |
| $S_{bdn}$ | $\{b, d, n\}$ | $\{d_4\}$ |

| Termset | Weight | |
|---|---|---|
| $S_a$ | $\mathcal{W}_{a,1}$ | $(1 + \log 4) * \log(1 + 4/2) = 4.75$ |
| $S_b$ | $\mathcal{W}_{b,1}$ | $(1 + \log 2) * \log(1 + 4/3) = 2.44$ |
| $S_d$ | $\mathcal{W}_{d,1}$ | $(1 + \log 2) * \log(1 + 4/4) = 2.00$ |
| $S_n$ | $\mathcal{W}_{n,1}$ | $0 * \log(1 + 4/1) = 0.00$ |
| $S_{ab}$ | $\mathcal{W}_{ab,1}$ | $(1 + \log 2) * \log(1 + 4/1) = 4.64$ |
| $S_{ad}$ | $\mathcal{W}_{ad,1}$ | $(1 + \log 2) * \log(1 + 4/2) = 3.17$ |
| $S_{bd}$ | $\mathcal{W}_{bd,1}$ | $(1 + \log 2) * \log(1 + 4/3) = 2.44$ |
| $S_{bn}$ | $\mathcal{W}_{bn,1}$ | $0 * \log(1 + 4/1) = 0.00$ |
| $S_{dn}$ | $\mathcal{W}_{dn,1}$ | $0 * \log(1 + 4/1) = 0.00$ |
| $S_{abd}$ | $\mathcal{W}_{abd,1}$ | $(1 + \log 2) * \log(1 + 4/1) = 4.64$ |
| $S_{bdn}$ | $\mathcal{W}_{bdn,1}$ | $0 * \log(1 + 4/1) = 0.00$ |

# Ranking Computation

- A document $d_j$ and a query $q$ are represented as vectors in a $2^t$-dimensional space of termsets

$$\begin{aligned}\vec{d_j} &= (\mathcal{W}_{1,j}, \mathcal{W}_{2,j}, \ldots, \mathcal{W}_{2^t,j}) \\ \vec{q} &= (\mathcal{W}_{1,q}, \mathcal{W}_{2,q}, \ldots, \mathcal{W}_{2^t,q})\end{aligned}$$

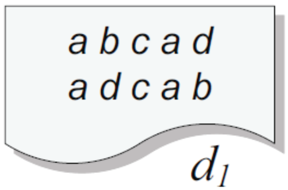
- The rank of $d_j$ to the query $q$ is computed as follows

$$sim(d_j, q) = \frac{\vec{d_j} \bullet \vec{q}}{|\vec{d_j}| \times |\vec{q}|} = \frac{\sum_{S_i} \mathcal{W}_{i,j} \times \mathcal{W}_{i,q}}{|\vec{d_j}| \times |\vec{q}|}$$

- For termsets that are not in the query $q$, $\mathcal{W}_{i,q} = 0$

# Ranking Computation

- The document norm $|\vec{d}_j|$ is hard to compute in the space of termsets
- Thus, its computation is restricted to 1-termsets
- Let again $q = \{a, b, d, n\}$ and $d_1$
- The document norm in terms of 1-termsets is given by




$$
\begin{aligned}
|\vec{d}_1| &= \sqrt{\mathcal{W}_{a,1}^2 + \mathcal{W}_{b,1}^2 + \mathcal{W}_{c,1}^2 + \mathcal{W}_{d,1}^2} \\
&= \sqrt{4.75^2 + 2.44^2 + 4.64^2 + 2.00^2} \\
&= 7.35
\end{aligned}
$$

| Termset | Weight | |
|---|---|---|
| $S_a$ | $\mathcal{W}_{a,1}$ | $(1 + \log 4) * \log(1 + 4/2) = 4.75$ |
| $S_b$ | $\mathcal{W}_{b,1}$ | $(1 + \log 2) * \log(1 + 4/3) = 2.44$ |
| $S_d$ | $\mathcal{W}_{d,1}$ | $(1 + \log 2) * \log(1 + 4/4) = 2.00$ |

# Ranking Computation

- To compute the rank of $d_1$, we need to consider the seven termsets $S_a$, $S_b$, $S_d$, $S_{ab}$, $S_{ad}$, $S_{bd}$, and $S_{abd}$
- The rank of $d_1$ is then given by

$$\begin{aligned} sim(d_1, q) &= (\mathcal{W}_{a,1} * \mathcal{W}_{a,q} + \mathcal{W}_{b,1} * \mathcal{W}_{b,q} + \mathcal{W}_{d,1} * \mathcal{W}_{d,q} + \\ &\quad \mathcal{W}_{ab,1} * \mathcal{W}_{ab,q} + \mathcal{W}_{ad,1} * \mathcal{W}_{ad,q} + \mathcal{W}_{bd,1} * \mathcal{W}_{bd,q} + \\ &\quad \mathcal{W}_{abd,1} * \mathcal{W}_{abd,q}) / |\vec{d_1}| \\ &= (4.75 * 1.58 + 2.44 * 1.22 + 2.00 * 1.00 + \\ &\quad 4.64 * 2.32 + 3.17 * 1.58 + 2.44 * 1.22 + \\ &\quad 4.64 * 2.32)/7.35 \\ &= 5.71 \end{aligned}$$

| Termset | Set of Terms | Documents |
|---|---|---|
| $\boxed{S_a}$ | $\{a\}$ | $\{d_1, d_2\}$ |
| $\boxed{S_b}$ | $\{b\}$ | $\{d_1, d_3, d_4\}$ |
| $\boxed{S_d}$ | $\{d\}$ | $\{d_1, d_2, d_3, d_4\}$ |
| $S_n$ | $\{n\}$ | $\{d_4\}$ |
| $\boxed{S_{ab}}$ | $\{a, b\}$ | $\{d_1\}$ |
| $\boxed{S_{ad}}$ | $\{a, d\}$ | $\{d_1, d_2\}$ |
| $\boxed{S_{bd}}$ | $\{b, d\}$ | $\{d_1, d_3, d_4\}$ |
| $S_{bn}$ | $\{b, n\}$ | $\{d_4\}$ |
| $\boxed{S_{abd}}$ | $\{a, b, d\}$ | $\{d_1\}$ |
| $S_{bdn}$ | $\{b, d, n\}$ | $\{d_4\}$ |

| Termset | Weight | |
|---|---|---|
| $S_a$ | $\mathcal{W}_{a,1}$ | $(1 + \log 4) * \underline{\log(1 + 4/2)} = 4.75$ |
| $S_b$ | $\mathcal{W}_{b,1}$ | $(1 + \log 2) * \underline{\log(1 + 4/3)} = 2.44$ |
| $S_d$ | $\mathcal{W}_{d,1}$ | $(1 + \log 2) * \underline{\log(1 + 4/4)} = 2.00$ |
| $S_n$ | $\mathcal{W}_{n,1}$ | $0 * \log(1 + 4/1) = 0.00$ |
| $S_{ab}$ | $\mathcal{W}_{ab,1}$ | $(1 + \log 2) * \underline{\log(1 + 4/1)} = 4.64$ |
| $S_{ad}$ | $\mathcal{W}_{ad,1}$ | $(1 + \log 2) * \underline{\log(1 + 4/2)} = 3.17$ |
| $S_{bd}$ | $\mathcal{W}_{bd,1}$ | $(1 + \log 2) * \underline{\log(1 + 4/3)} = 2.44$ |
| $S_{bn}$ | $\mathcal{W}_{bn,1}$ | $0 * \log(1 + 4/1) = 0.00$ |
| $S_{dn}$ | $\mathcal{W}_{dn,1}$ | $0 * \log(1 + 4/1) = 0.00$ |
| $S_{abd}$ | $\mathcal{W}_{abd,1}$ | $(1 + \log 2) * \underline{\log(1 + 4/1)} = 4.64$ |
| $S_{bdn}$ | $\mathcal{W}_{bdn,1}$ | $0 * \log(1 + 4/1) = 0.00$ |

# Closed Termsets

- The concept of frequent termsets allows simplifying the ranking computation
- Yet, there are many frequent termsets in a large collection
  - The number of termsets to consider might be prohibitively high with large queries
- To resolve this problem, we can further restrict the ranking computation to a smaller number of termsets
- This can be accomplished by observing some properties of termsets such as the notion of **closure**

# Closed Termsets

- The **closure of a termset** $S_i$ is the set of all frequent termsets that co-occur with $S_i$ in the same set of docs
- Given the closure of $S_i$, the largest termset in it is called a **closed termset** and is referred to as $\Phi_i$
- We formalize, as follows
  - Let $D_i \subseteq C$ be the subset of all documents in which termset $S_i$ occurs and is frequent
  - Let $S(D_i)$ be a set composed of the frequent termsets that occur in all documents in $D_i$ and only in those

# Closed Termsets

- Then, the closed termset $S_{\Phi_i}$ satisfies the following property

$$\nexists S_j \in S(D_i) \mid S_{\Phi_i} \subset S_j$$

- Frequent and closed termsets for our example collection, considering a minimum threshold equal to 2

| frequency($S_i$) | frequent termset | closed termset |
|---|---|---|
| 4 | d | d |
| 3 | b, bd | bd |
| 2 | a, ad | ad |
| 2 | g, h, gh, ghd | ghd |

| Termset | Set of Terms | Documents |
|---|---|---|
| $S_a$ | $\{a\}$ | $\boxed{\{d_1, d_2\}}$ |
| $S_b$ | $\{b\}$ | $\boxed{\{d_1, d_3, d_4\}}$ |
| $S_d$ | $\{d\}$ | $\boxed{\{d_1, d_2, d_3, d_4\}}$ |
| $S_n$ | $\{n\}$ | $\{d_4\}$ |
| $S_{ab}$ | $\{a, b\}$ | $\{d_1\}$ |
| $S_{ad}$ | $\{a, d\}$ | $\boxed{\{d_1, d_2\}}$ |
| $S_{bd}$ | $\{b, d\}$ | $\boxed{\{d_1, d_3, d_4\}}$ |
| $S_{bn}$ | $\{b, n\}$ | $\{d_4\}$ |
| $S_{abd}$ | $\{a, b, d\}$ | $\{d_1\}$ |
| $S_{bdn}$ | $\{b, d, n\}$ | $\{d_4\}$ |

# Closed Termsets

- Closed termsets encapsulate smaller termsets occurring in the same set of documents
- The ranking $sim(d_1, q)$ of document $d_1$ with regard to query $q$ is computed as follows:
  - $d_1 =$ ''a b c a d a d c a b''
  - $q = \{a, b, d, n\}$
  - minimum frequency threshold = 2

1

| Termset | Set of Terms | Documents |
|---|---|---|
| $\boxed{S_a}$ | $\{a\}$ | $\{d_1, d_2\}$ |
| $\boxed{S_b}$ | $\{b\}$ | $\{d_1, d_3, d_4\}$ |
| $\boxed{S_d}$ | $\{d\}$ | $\{d_1, d_2, d_3, d_4\}$ |
| $S_n$ | $\{n\}$ | $\{d_4\}$ |
| $\boxed{S_{ab}}$ | $\{a, b\}$ | $\{d_1\}$ |
| $\boxed{S_{ad}}$ | $\{a, d\}$ | $\{d_1, d_2\}$ |
| $\boxed{S_{bd}}$ | $\{b, d\}$ | $\{d_1, d_3, d_4\}$ |
| $S_{bn}$ | $\{b, n\}$ | $\{d_4\}$ |
| $\boxed{S_{abd}}$ | $\{a, b, d\}$ | $\{d_1\}$ |
| $S_{bdn}$ | $\{b, d, n\}$ | $\{d_4\}$ |

d
b, bd
a, ad

d
bd
ad

delete

$$sim(d_1, q) = (\mathcal{W}_{d,1} * \mathcal{W}_{d,q} + \mathcal{W}_{ab,1} * \mathcal{W}_{ab,q} + \mathcal{W}_{ad,1} * \mathcal{W}_{ad,q} + \mathcal{W}_{bd,1} * \mathcal{W}_{bd,q} + \mathcal{W}_{abd,1} * \mathcal{W}_{abd,q}) / |\vec{d_1}|$$

delete

$$= (2.00 * 1.00 + 4.64 * 2.32 + 3.17 * 1.58 + 2.44 * 1.22 + 4.64 * 2.32)/7.35$$

$$= 4.28$$

| Termset | Weight | |
|---|---|---|
| $S_a$ | $\mathcal{W}_{a,1}$ | $(1 + \log 4) * \log(1 + 4/2) = 4.75$ |
| $S_b$ | $\mathcal{W}_{b,1}$ | $(1 + \log 2) * \log(1 + 4/3) = 2.44$ |
| $S_d$ | $\mathcal{W}_{d,1}$ | $(1 + \log 2) * \underline{\log(1 + 4/4)} = 2.00$ |
| $S_n$ | $\mathcal{W}_{n,1}$ | $0 * \log(1 + 4/1) = 0.00$ |
| $S_{ab}$ | $\mathcal{W}_{ab,1}$ | $(1 + \log 2) * \underline{\log(1 + 4/1)} = 4.64$ |
| $S_{ad}$ | $\mathcal{W}_{ad,1}$ | $(1 + \log 2) * \underline{\log(1 + 4/2)} = 3.17$ |
| $S_{bd}$ | $\mathcal{W}_{bd,1}$ | $(1 + \log 2) * \underline{\log(1 + 4/3)} = 2.44$ |
| $S_{bn}$ | $\mathcal{W}_{bn,1}$ | $0 * \log(1 + 4/1) = 0.00$ |
| $S_{dn}$ | $\mathcal{W}_{dn,1}$ | $0 * \log(1 + 4/1) = 0.00$ |
| $S_{abd}$ | $\mathcal{W}_{abd,1}$ | $(1 + \log 2) * \underline{\log(1 + 4/1)} = 4.64$ |
| $S_{bdn}$ | $\mathcal{W}_{bdn,1}$ | $0 * \log(1 + 4/1) = 0.00$ |

$$sim(d_1, q) = (\mathcal{W}_{a,1} * \mathcal{W}_{a,q} + \mathcal{W}_{b,1} * \mathcal{W}_{b,q} + \mathcal{W}_{d,1} * \mathcal{W}_{d,q} + \mathcal{W}_{ab,1} * \mathcal{W}_{ab,q} + \mathcal{W}_{ad,1} * \mathcal{W}_{ad,q} + \mathcal{W}_{bd,1} * \mathcal{W}_{bd,q} + \mathcal{W}_{abd,1} * \mathcal{W}_{abd,q}) / |\vec{d_1}|$$

# Closed Termsets

- Thus, if we restrict the ranking computation to closed termsets, we can expect a reduction in query time
- Smaller the number of closed termsets, sharper is the reduction in query processing time

# Extended Boolean Model

# Extended Boolean Model

- In the Boolean model, no **ranking** of the answer set is generated
- One alternative is to extend the Boolean model with the notions of **partial matching** and **term weighting**
- This strategy allows one to combine characteristics of the Vector model with properties of Boolean algebra

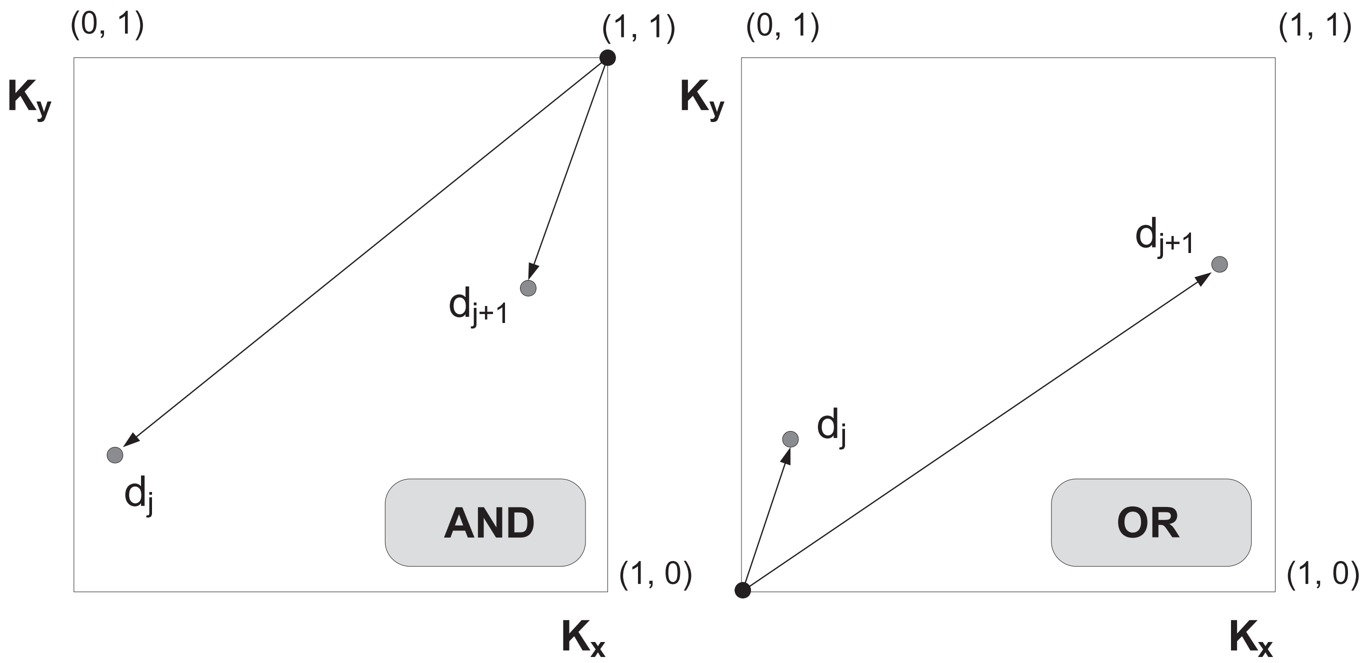
# The Idea

- Consider a conjunctive Boolean query given by $q = k_x \wedge k_y$
- For the boolean model, a doc that contains a single term of $q$ is as irrelevant as a doc that contains none
- However, this **binary decision** criteria frequently is not in accordance with common sense
- An analogous reasoning applies when one considers purely disjunctive queries

A document containing K x and K y is as relevant as a document containing only one.

# The Idea

- When only two terms $x$ and $y$ are considered, we can plot queries and docs in a two-dimensional space




- A document $d_j$ is positioned in this space through the adoption of weights $w_{x,j}$ and $w_{y,j}$

# The Idea

- These weights can be computed as normalized tf-idf factors as follows

$$w_{x,j} = \frac{f_{x,j}}{max_x \;\; f_{x,j}} \times \frac{idf_x}{max_i \;\; idf_i}$$

- where
  - $f_{x,j}$ is the frequency of term $k_x$ in document $d_j$
  - $idf_i$ is the inverse document frequency of term $k_i$, as before
- To simplify notation, let
  - $w_{x,j} = x$ and $w_{y,j} = y$
  - $\vec{d_j} = (w_{x,j}, w_{y,j})$ as the point $d_j = (x, y)$

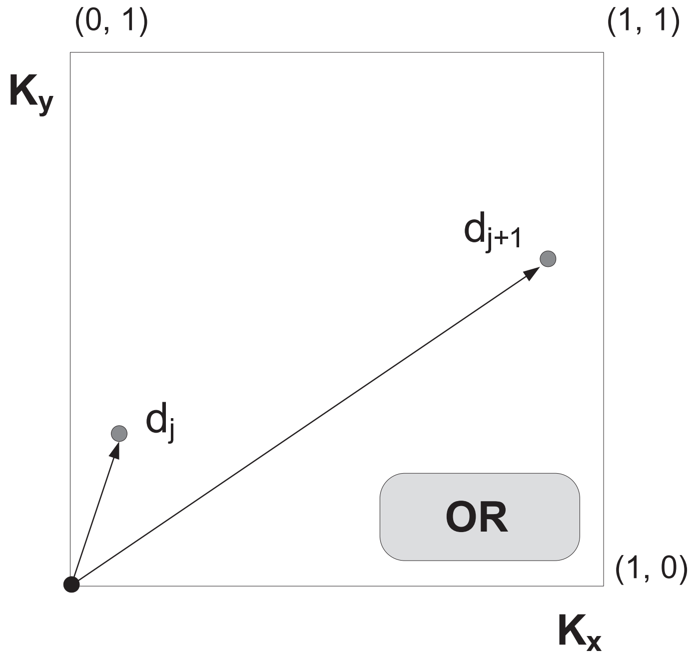
# The Idea

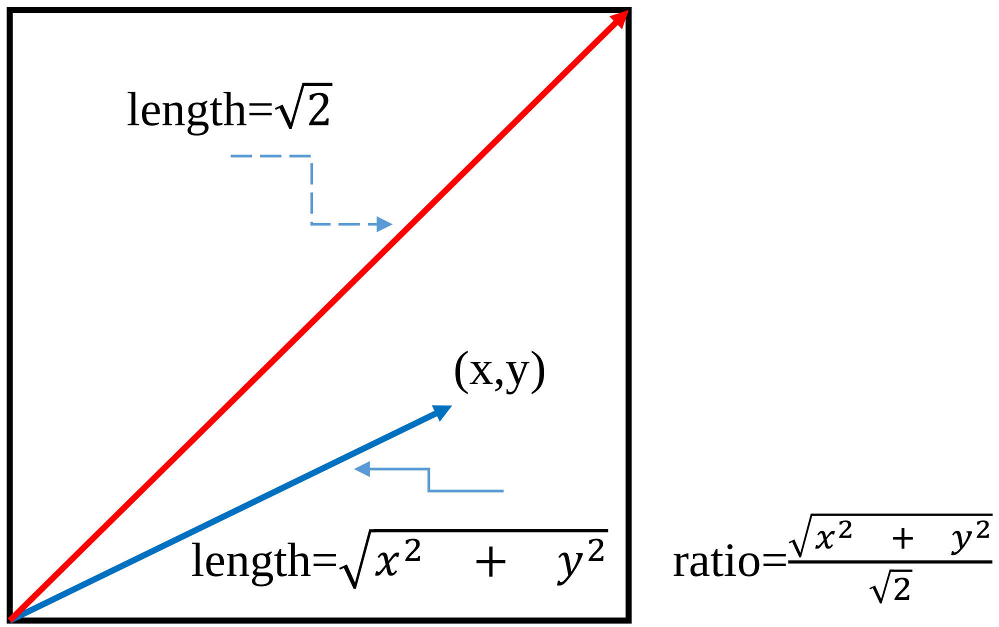
- For a disjunctive query $q_{or} = k_x \vee k_y$, the point $(0, 0)$ is the least interesting one

越遠離(0,0)越好

- This suggests taking the distance from $(0, 0)$ as a measure of similarity

$$sim(q_{or}, d) = \sqrt{\frac{x^2 + y^2}{2}}$$



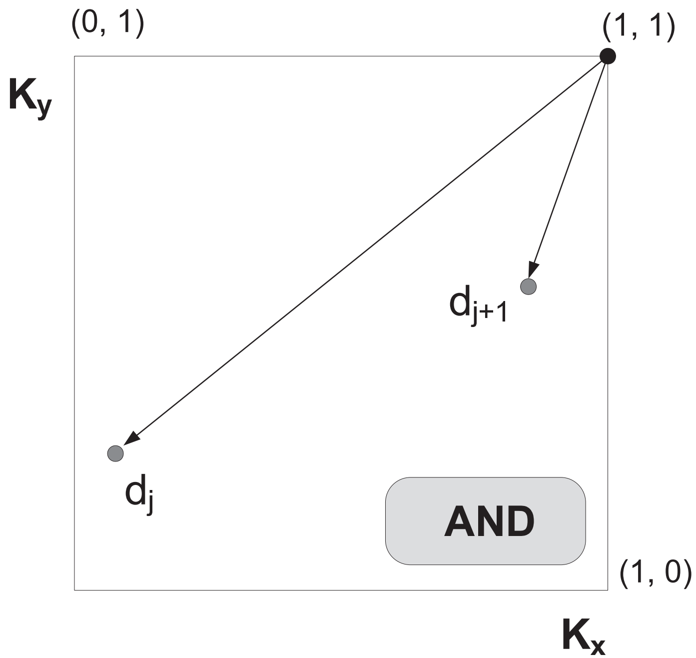


length=$\sqrt{2}$
(x,y)
length=$\sqrt{x^2 + y^2}$
ratio=$\frac{\sqrt{x^2 + y^2}}{\sqrt{2}}$

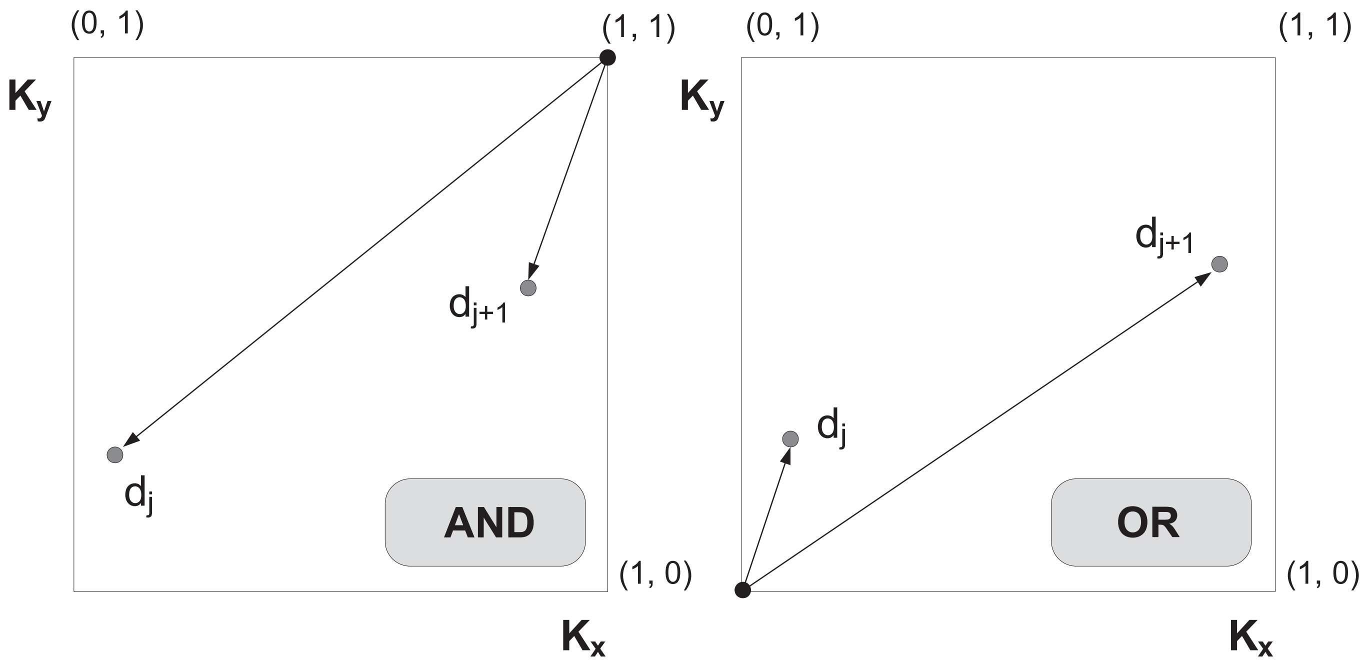
# The Idea

- For a conjunctive query $q_{and} = k_x \wedge k_y$, the point $(1, 1)$ is the most interesting one 越近(1,1)越好
- This suggests taking the complement of the distance from the point $(1, 1)$ as a measure of similarity

$$sim(q_{and}, d) \quad = \quad 1 - \sqrt{\frac{(1-x)^2 + (1-y)^2}{2}}$$



# The Idea




$$sim(q_{or}, d) = \sqrt{\frac{x^2 + y^2}{2}}$$

$$sim(q_{and}, d) = 1 - \sqrt{\frac{(1-x)^2 + (1-y)^2}{2}}$$

# Generalizing the Idea

- We can extend the previous model to consider Euclidean distances in a t-dimensional space
- This can be done using *p-norms* which extend the notion of distance to include p-distances, where $1 \leq p \leq \infty$
- A generalized conjunctive query is given by
  - $q_{and} = k_1 \ \wedge^p \ k_2 \ \wedge^p \ \ldots \wedge^p k_m$
- A generalized disjunctive query is given by
  - $q_{or} = k_1 \ \vee^p \ k_2 \ \vee^p \ \ldots \vee^p k_m$

# Generalizing the Idea

- The query-document similarities are now given by

$$sim(q_{or}, d_j) = \left(\frac{x_1^p + x_2^p + \ldots + x_m^p}{m}\right)^{\frac{1}{p}}$$

$$sim(q_{and}, d_j) = 1 - \left(\frac{(1-x_1)^p + (1-x_2)^p + \ldots + (1-x_m)^p}{m}\right)^{\frac{1}{p}}$$

  where each $x_i$ stands for a weight $w_{i,d}$

- If $p = 1$ then (vector-like)
  - $sim(q_{or}, d_j) = sim(q_{and}, d_j) = \frac{x_1 + \ldots + x_m}{m}$

- If $p = \infty$ then (Fuzzy like)
  - $sim(q_{or}, d_j) = max(x_i)$
  - $sim(q_{and}, d_j) = min(x_i)$

# Properties

- By varying $p$, we can make the model behave as a vector, as a fuzzy, or as an intermediary model
- The processing of more general queries is done by grouping the operators in a predefined order
- For instance, consider the query $q = (k_1 \wedge^p k_2) \vee^p k_3$
  - $k_1$ and $k_2$ are to be used as in a vectorial retrieval while the presence of $k_3$ is required
- The similarity $sim(q, d_j)$ is computed as

$$sim(q,d) = \left( \frac{\left(1 - \left(\frac{(1-x_1)^p + (1-x_2)^p}{2}\right)^{\frac{1}{p}}\right)^p + x_3^p}{2} \right)^{\frac{1}{p}}$$

# Conclusions

- Model is quite powerful
- Properties are interesting and might be useful
- Computation is somewhat complex
- However, distributivity operation does not hold for ranking computation:
  - $q_1 = (k_1 \ \vee \ k_2) \ \wedge \ k_3$
  - $q_2 = (k_1 \ \wedge \ k_3) \ \vee \ (k_2 \ \wedge \ k_3)$
  - $sim(q_1, d_j) \neq sim(q_2, d_j)$

# Fuzzy Set Model

# Fuzzy Set Model

- Matching of a document to a query terms is approximate or vague
- This **vagueness** can be modeled using a fuzzy framework, as follows:
  - each query term defines a **fuzzy** set
  - each doc has a **degree of membership** in this set
- This interpretation provides the foundation for many IR models based on fuzzy theory
- In here, we discuss the model proposed by Ogawa, Morita, and Kobayashi

# Fuzzy Set Theory

- Fuzzy set theory deals with the representation of classes whose boundaries are not well defined
- Key idea is to introduce the notion of a **degree of membership** associated with the elements of the class
- This degree of membership varies from 0 to 1 and allows modelling the notion of **marginal** membership
- Thus, membership is now a **gradual** notion, contrary to the crispy notion enforced by classic Boolean logic

# Fuzzy Set Theory

- A fuzzy subset $A$ of a universe of discourse $U$ is characterized by a membership function

$$\mu_A : U \rightarrow [0, 1]$$

- This function associates with each element $u$ of $U$ a number $\mu_A(u)$ in the interval $[0, 1]$
- The three most commonly used operations on fuzzy sets are:
  - the complement of a fuzzy set
  - the union of two or more fuzzy sets
  - the intersection of two or more fuzzy sets

# Fuzzy Set Theory

- Let,
  - $U$ be the universe of discourse
  - $A$ and $B$ be two fuzzy subsets of $U$
  - $\overline{A}$ be the complement of $A$ relative to $U$
  - $u$ be an element of $U$
- Then,

$$
\begin{aligned}
\mu_{\overline{A}}(u) &= 1 - \mu_A(u) \\
\mu_{A \cup B}(u) &= max(\mu_A(u), \mu_B(u)) \\
\mu_{A \cap B}(u) &= min(\mu_A(u), \mu_B(u))
\end{aligned}
$$

# Fuzzy Information Retrieval

- Fuzzy sets are modeled based on a thesaurus, which defines term relationships
- A thesaurus can be constructed by defining a term-term correlation matrix $C$
- Each element of $C$ defines a normalized correlation factor $c_{i,\ell}$ between two terms $k_i$ and $k_\ell$

$$c_{i,l} = \frac{n_{i,l}}{n_i + n_l - n_{i,l}}$$

  where

  - $n_i$: number of docs which contain $k_i$
  - $n_l$: number of docs which contain $k_l$
  - $n_{i,l}$: number of docs which contain both $k_i$ and $k_l$

# Fuzzy Information Retrieval

- We can use the term correlation matrix $C$ to associate a fuzzy set with each index term $k_i$
- In this fuzzy set, a document $d_j$ has a degree of membership $\mu_{i,j}$ given by

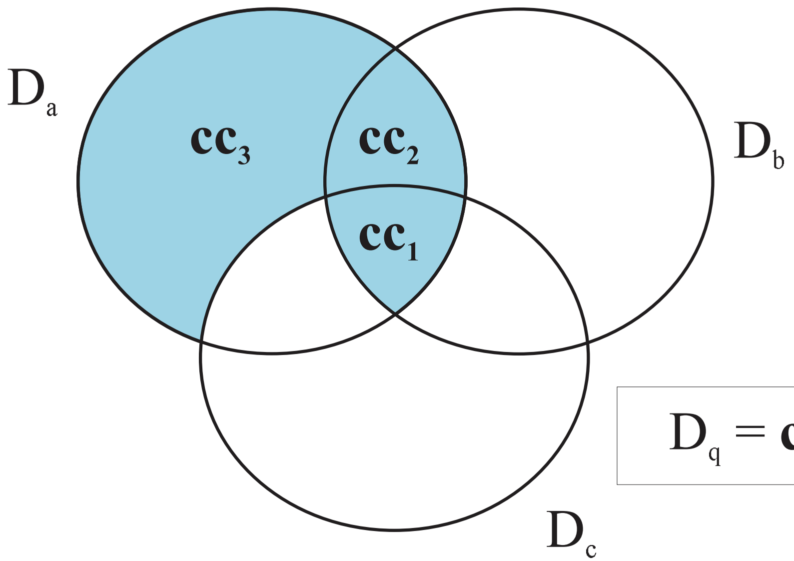
$$\mu_{i,j} = 1 - \prod_{k_l \in d_j} (1 - c_{i,l})$$

- The above expression computes an algebraic sum over all terms in $d_j$
- A document $d_j$ belongs to the fuzzy set associated with $k_i$, if its own terms are associated with $k_i$

# Fuzzy Information Retrieval

- If $d_j$ contains a term $k_l$ which is closely related to $k_i$, we have
  - $c_{i,l} \sim 1$
  - $\mu_{i,j} \sim 1$
  - and $k_i$ is a good fuzzy index for $d_j$

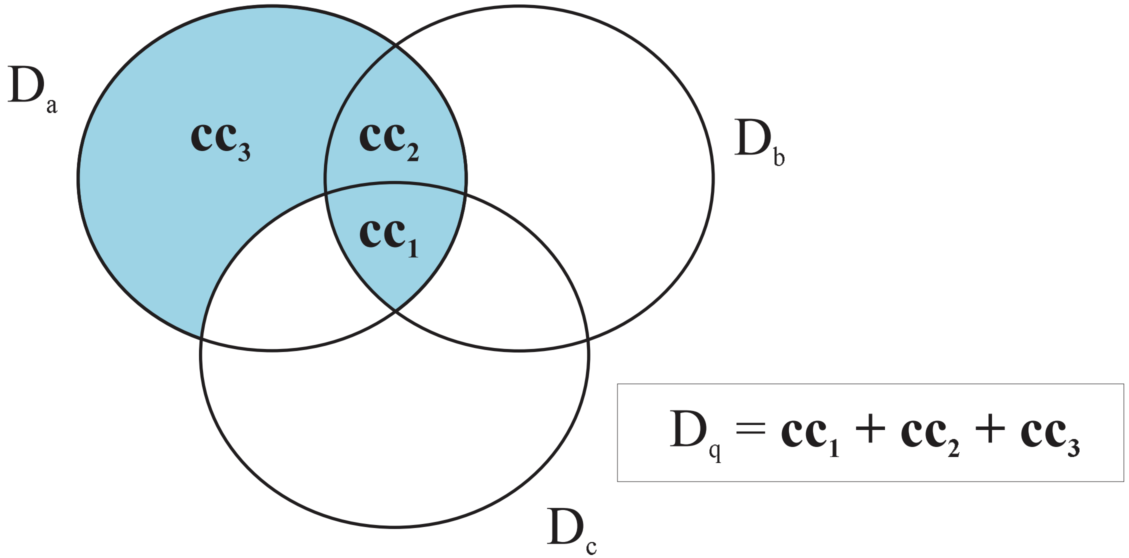
# Fuzzy IR: An Example




$$D_q = cc_1 + cc_2 + cc_3$$

- Consider the query $q = k_a \wedge (k_b \vee \neg k_c)$
- The disjunctive normal form of $q$ is composed of 3 conjunctive components (cc), as follows:
  $\vec{q}_{dnf} = (1,1,1) + (1,1,0) + (1,0,0) = cc_1 + cc_2 + cc_3$
- Let $D_a$, $D_b$ and $D_c$ be the fuzzy sets associated with the terms $k_a$, $k_b$ and $k_c$, respectively

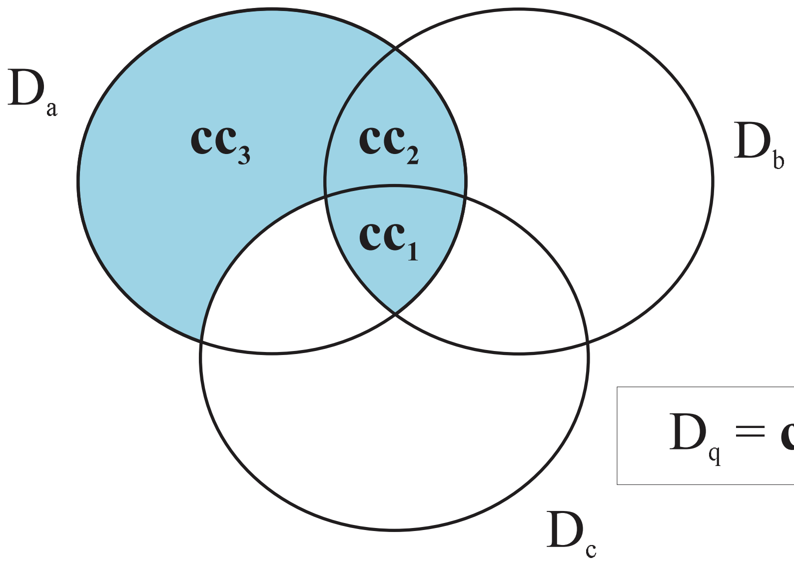
# Fuzzy IR: An Example




- Let $\mu_{a,j}$, $\mu_{b,j}$, and $\mu_{c,j}$ be the degrees of memberships of document $d_j$ in the fuzzy sets $D_a$, $D_b$, and $D_c$. Then,

$$
\begin{aligned}
cc_1 &= \mu_{a,j}\mu_{b,j}\mu_{c,j} \\
cc_2 &= \mu_{a,j}\mu_{b,j}(1 - \mu_{c,j}) \\
cc_3 &= \mu_{a,j}(1 - \mu_{b,j})(1 - \mu_{c,j})
\end{aligned}
$$

# Fuzzy IR: An Example




$D_q = \mathbf{cc_1} + \mathbf{cc_2} + \mathbf{cc_3}$

$$
\begin{aligned}
\mu_{q,j} &= \mu_{cc_1+cc_2+cc_3,j} \\
&= 1 - \prod_{i=1}^{3}(1 - \mu_{cc_i,j}) \\
&= 1 - (1 - \mu_{a,j}\mu_{b,j}\mu_{c,j}) \times \\
&\quad (1 - \mu_{a,j}\mu_{b,j}(1 - \mu_{c,j})) \times (1 - \mu_{a,j}(1 - \mu_{b,j})(1 - \mu_{c,j}))
\end{aligned}
$$

# Conclusions

- Fuzzy IR models have been discussed mainly in the literature associated with fuzzy theory
- They provide an interesting framework which naturally embodies the notion of term dependencies
- Experiments with standard test collections are not available