# CSCE 561
# Information Retrieval System Models

Satya Katragadda
26 August 2015

# Agenda

- Introduction to Information Retrieval
- Inverted Index
- IR System Models
- Boolean Retrieval Model

# Introduction

Information Retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers)

- "Satisfy" – what does this mean?

# Introduction: Applications of IR

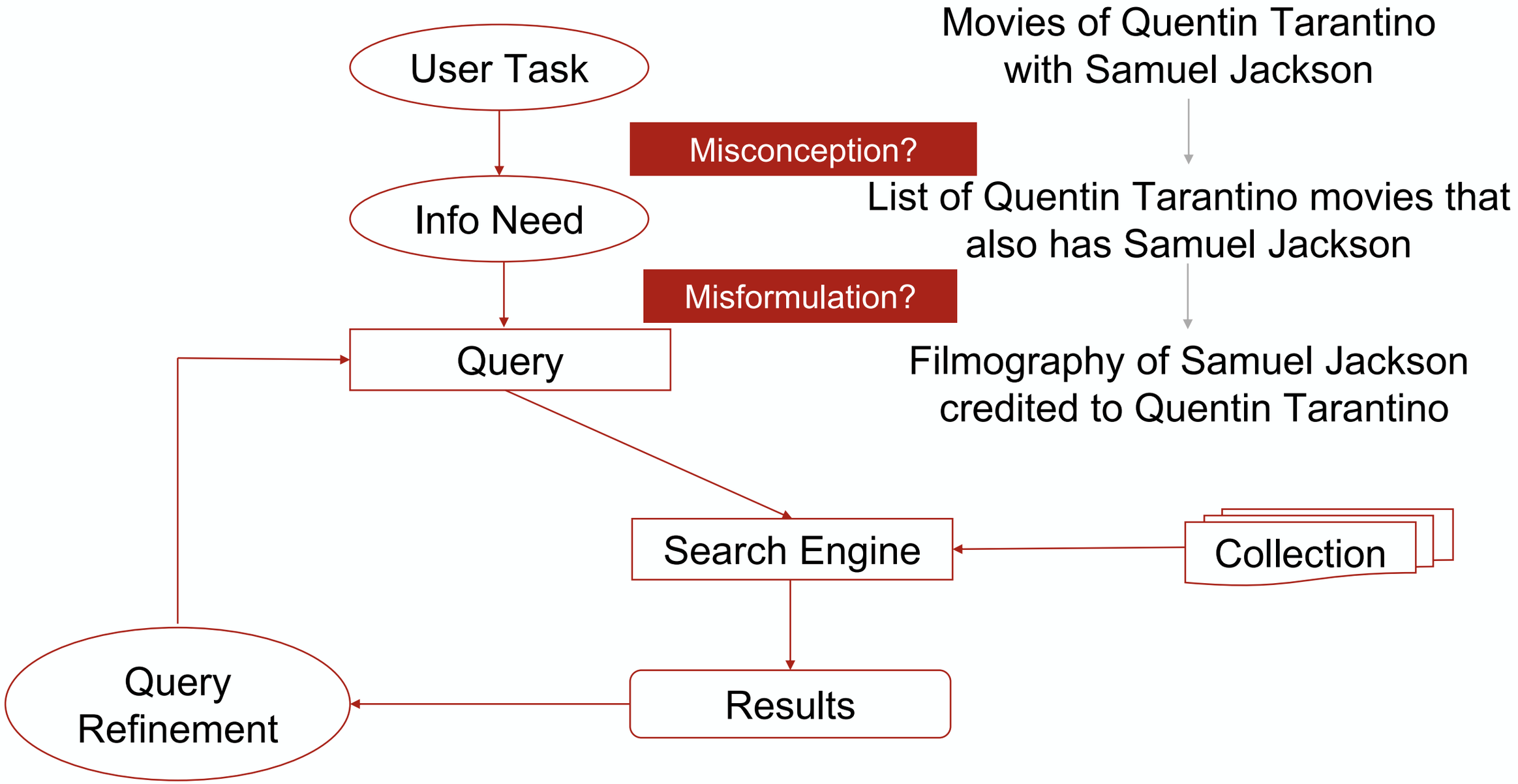
- Traditional Web Search
  - Search Engines: Google, Bing, Yahoo?
  - Digital Catalogues: IMDB, GLADYS
- Non-Traditional Searches
  - File search
  - Email search
- Other Applications
  - Text categorization
  - Text Summarization
  - Structured Document Retrieval

# Introduction: IR Assumptions

- **Collection**: A set of documents, assume its static

- **Goal**: Retrieve documents with information that is relevant to user's information need to complete a task.

# Introduction: Classic Search Model



# Introduction: Search Model

- Search for movie credits with Quentin Tarantino and Samuel Jackson that does not have Leonardo DiCaprio
- grep all movie pages with for Quentin Tarantino and Samuel Jackson, then remove the ones that contain Leonardo DiCaprio
- Problems?
  - Slow for large corpus
  - More additional Information
  - Ranked retrieval?

# Indexing

- How to store documents and terms so that we can retrieve documents
  - Efficiently
  - Effectively
  - With reasonable space requirements?

# Indexing: Term - Document Matrix

- Create a table
  - Rows: Terms
  - Columns: Document IDs
- Term-Document Incidence Matrix
- Inverted View of Collection

# Indexing: Term - Document Matrix

| | Document 1 | Document 2 | Document 3 | Document 4 |
|---|---|---|---|---|
| **Term 1** | 1 | 0 | 0 | 1 |
| **Term 2** | 0 | 1 | 1 | 1 |
| **Term 3** | 1 | 0 | 1 | 1 |
| **Term 4** | 1 | 1 | 1 | 1 |

- Rows: Vectors of documents containing term x
- Columns: Vectors of terms contained by document Y

# Indexing: Document - Term Matrix

| | Term 1 | Term 2 | Term 3 | Term 4 |
|---|---|---|---|---|
| **Document 1** | 1 | 0 | 1 | 1 |
| **Document 2** | 0 | 1 | 0 | 1 |
| **Document 3** | 0 | 1 | 1 | 1 |
| **Document 4** | 1 | 1 | 1 | 1 |

- Rows: Vectors of terms contained by document x
- Columns: Vectors of documents containing term y

# Indexing: Term – Document Matrix

- Naïve way of storage: Create and store the matrix
- Calculations
  - 500,000 Terms
  - 1,000,000 Documents
  - ½ trillion entries: most of them 0's and 1's
- Memory Impact
  - As documents/terms grows, needs constant updation
  - Can't keep up with memory

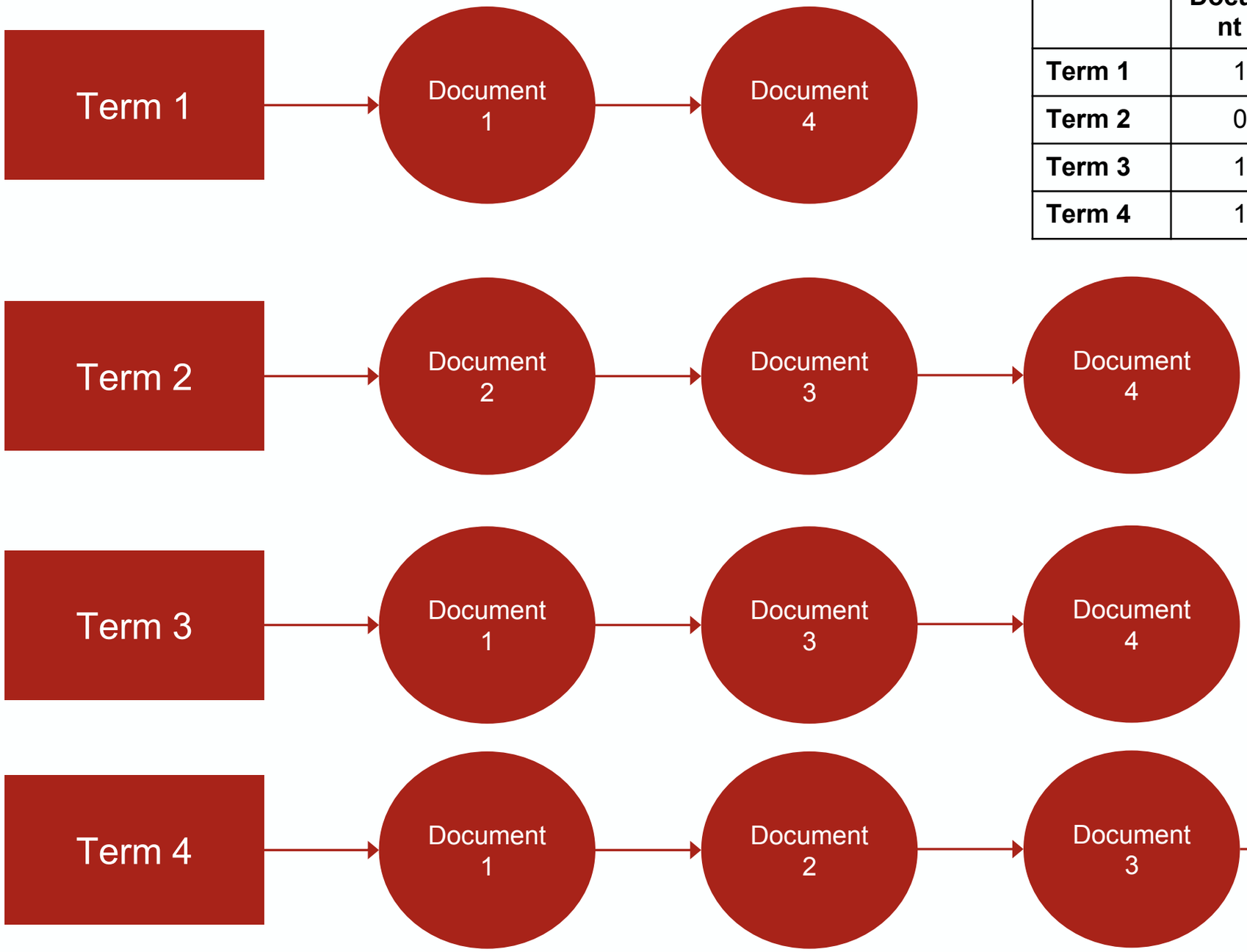
# Indexing: Term – Document Matrix

- Observation
  - Term –Document Matrix->Sparse
  - Only a small number of terms in any given document
- Assume a typical document contains 1000 terms
  - A collection of 1,000,000 documents contains 1 billion 1
  - Rest of the entries are 0’s
  - Thus, 99.8% of matrix is 0’s

# Indexing: Inverted Index

- Also called Inverted File
- Dictionary of terms
  - Vocabulary
  - Lexicon
- Each term
  - List of documents in which it appears
  - Each document is called a posting

# Indexing: Inverted Index

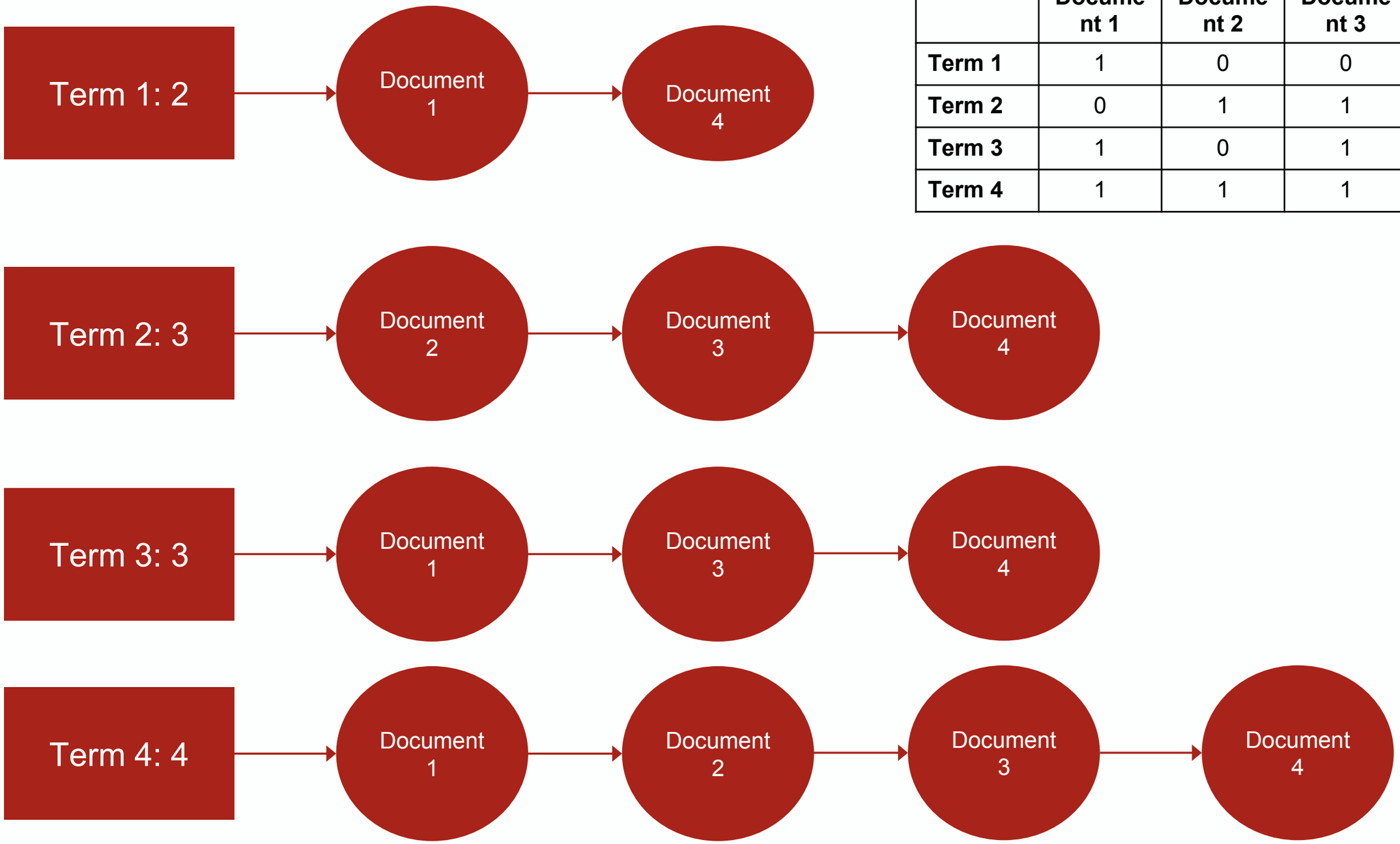



| | Document 1 | Document 2 | Document 3 | Document 4 |
|---|---|---|---|---|
| **Term 1** | 1 | 0 | 0 | 1 |
| **Term 2** | 0 | 1 | 1 | 1 |
| **Term 3** | 1 | 0 | 1 | 1 |
| **Term 4** | 1 | 1 | 1 | 1 |

# Indexing: Inverted Index

- Dictionary: Sorted alphabetically
- Each posting: sorted by ID
- Storage
  - Dictionary kept in memory
  - Postings can be stored in memory or disk

# Indexing: Inverted Index, changed




| | Document 1 | Document 2 | Document 3 | Document 4 |
|---|---|---|---|---|
| **Term 1** | 1 | 0 | 0 | 1 |
| **Term 2** | 0 | 1 | 1 | 1 |
| **Term 3** | 1 | 0 | 1 | 1 |
| **Term 4** | 1 | 1 | 1 | 1 |

# IR System Models

- S=(D, Q, T, V, F)
  - D: Documents Representation Space
  - Q: Query Representation Space
  - T: The set of Index terms (Indexing Vocabulary)
  - F: D X Q → V
  - V: The set of Retrieval Status Value

# IR System Models

- S = (D, Q, T, V, F)
- If the retrieval status values are unique, then the ordering is linear
- If the retrieval status values are non-unique, it is a weak ordering
  - Select 10 relevant results?

# IRS Models: Subject Catalog Model

- S=(D, Q, T, V, F)
  - T = set of subject headings
  - Q = T
  - $D = 2^T$
  - V = {0,1}
  - $F_q(d)$, where q ϵ Q, d ϵ D
    - 1, if q ϵ d
    - 0, other wise

# IRS Models: Coordination level System

- S=(D, Q, T, V, F)
  - Q = $2^T$
  - D = $2^T$
  - V = {0,1}
  - $F_q(d)$, where q ϵ Q, d ϵ D
    - 1, if q ⊆ d
    - 0, other wise
  - $F'_q(d)$, where q ϵ Q, d ϵ D
    - 1, if $|q \cap d| > k$
    - 0, other wise

# IRS Models: Boolean System

- S=(D, Q, T, V, F)
  - D = $2^T$
  - Q = E (Expression)
  - V = {0,1}
  - $F_q(d)$, where q ϵ Q, d ϵ D
    - 1, if q evaluates to True w.r.t. Document
    - 0, other wise

# Boolean System: Expression

- Let $t \in T$
  - Then $t \in E$
- If $e \in E$
  - Then $\neg e \in E$
- If $e_1, e_2 \in E$
  - $e_1 \vee e_2 \in E$
  - $e_1 \wedge e_2 \in E$
  - Nothing else is an element of E.

# Boolean System: Document Representation

- Set of Documents ID's
  - $D = \{d_\alpha\}$, $\alpha = 1,2,\ldots,p$
- Set of all term ID's
  - $T = \{t_i\}$, $i = 1,2,\ldots,n$

# Boolean System: Document Representation

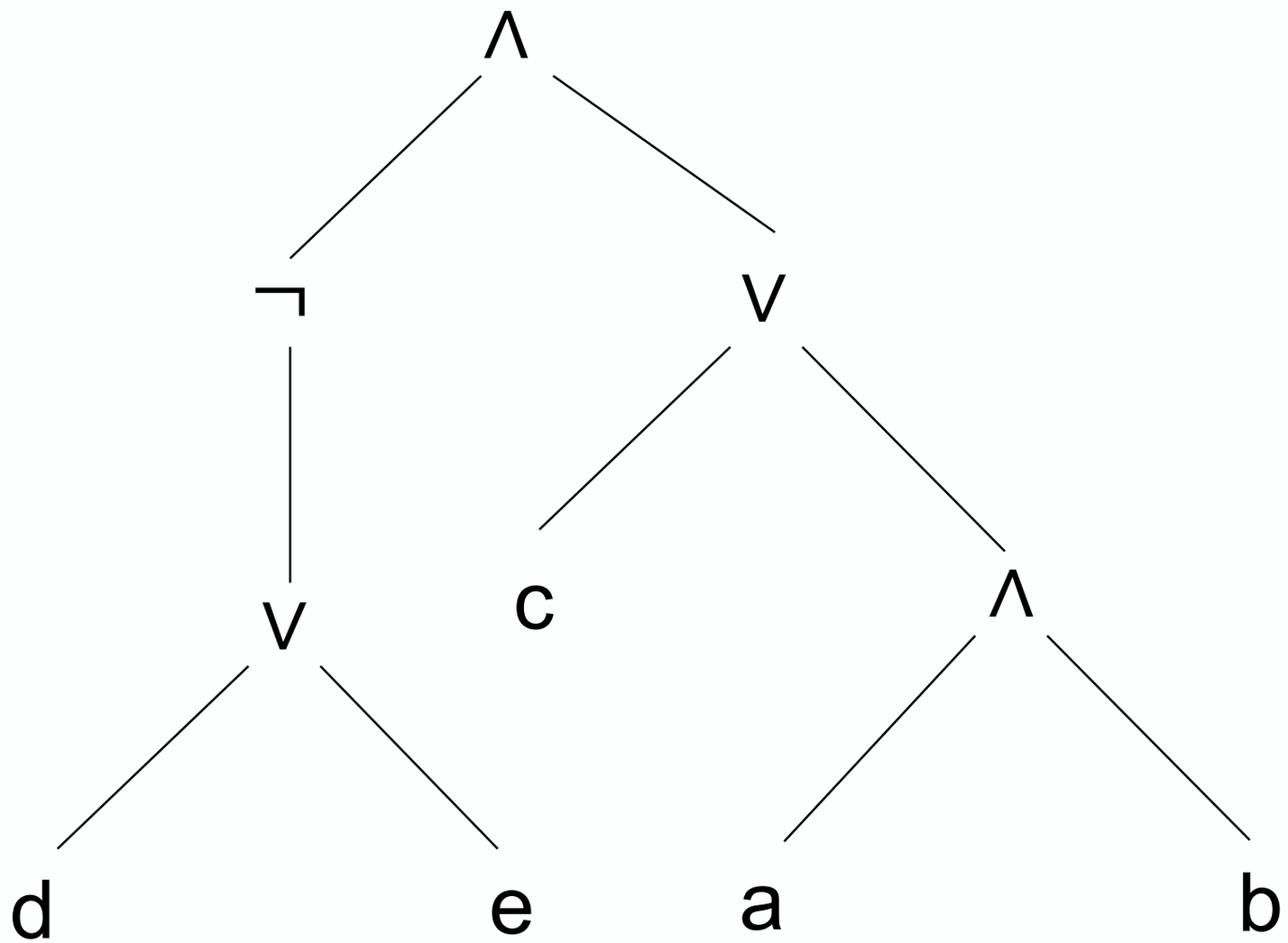
- Relation
  - $D = \{ < d_\alpha, t_i, \mu_D(d_\alpha, t_i) >\}$
  - $\mu_D = D \times T \rightarrow \{0,1\}$
  - $\mu_D(d_\alpha, t_i)$
    - 1, if $d_\alpha$ contains $t_i$
    - 0, otherwise
- $D_{t_i} = \{d_\alpha \in D \mid \mu_D(d_\alpha, t_i) = 1\}$
- $d_\alpha \equiv D_d = \{t_i \in T \mid \mu_D(d_\alpha, t_i) = 1\}$

# Boolean System: Retrieval Function

- $RSV \equiv F$
- $RSV_t(d_\alpha) = \mu_D(d_\alpha, t)$
- $RSV_{\neg e}(d_\alpha) = 1 - RSV_e(d_\alpha)$
- $RSV_{e_1 \vee e_2}(d_\alpha) = RSV_{e_1}(d_\alpha) \vee RSV_{e_2}(d_\alpha)$
- $RSV_{e_1 \wedge e_2}(d_\alpha) = RSV_{e_1}(d_\alpha) \wedge RSV_{e_2}(d_\alpha)$

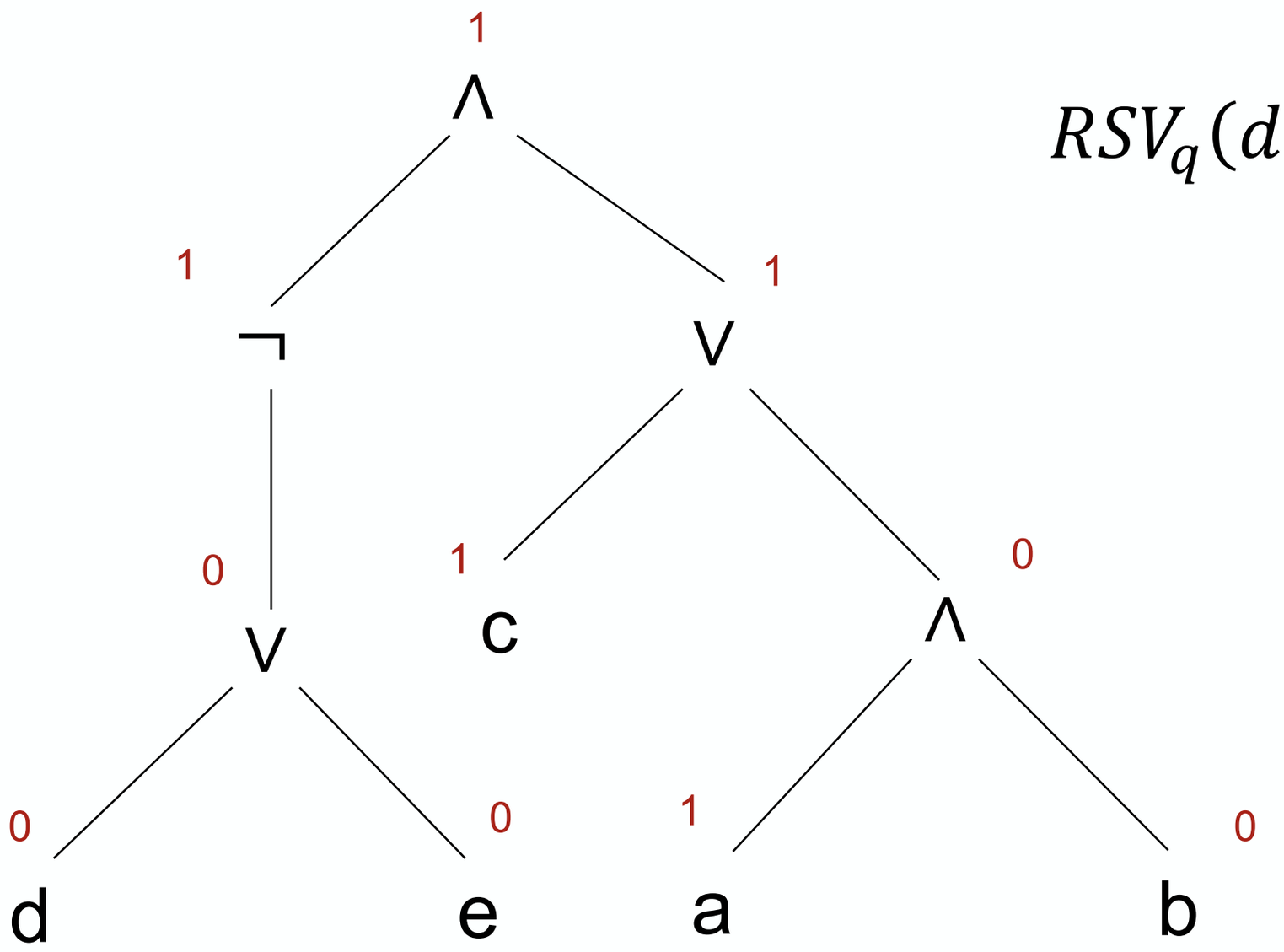
# Boolean System: Example

$$q = \neg(d \vee e) \wedge (c \vee (a \wedge b))$$



# Boolean System: Example

$$q = \neg(d \vee e) \wedge \big(c \vee (a \wedge b)\big), D_{d_\alpha} = \{a, c\}$$




$$RSV_q(d_\alpha) = 1$$

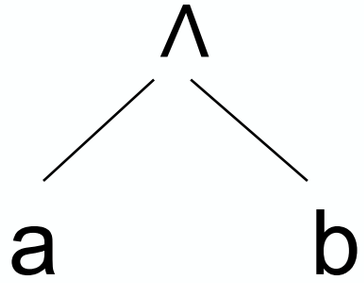
# Processing Boolean Query: Method 1

- $D_{t_1 \vee t_2} = \{d_\alpha \in D \mid \mu_D(d_\alpha, t_1) \vee \mu_D(d_\alpha, t_2) = 1\}$
- $D_{t_1 \wedge t_2} = \{d_\alpha \in D \mid \mu_D(d_\alpha, t_1) \wedge \mu_D(d_\alpha, t_2) = 1\}$

- $D_t$ = set of documents containing term t
- T = {a, b, c, d, e}

# Processing Boolean Query: Method 1

- Input
  - $a \wedge b$




- Output
  - $D_a \cap D_b$

# Processing Boolean Query: Method 1

| Output | Query |
|---|---|
| $D_t$ | t |
| $D_{e1}$ | e1 |
| $D_{e2}$ | e2 |
| $D_{e_1 \vee e_2}$ | $e_1 \vee e_2$ |
| $D_{e_1 \wedge e_2}$ | $e_1 \wedge e_2$ |
| $D \backslash D_e$ | $\neg e$ |

# Processing Boolean Query: Method 2

- AND queries $e_1 \wedge e_2$
  - Construct a merged list M for $D_{e_1}$ and $D_{e_2}$
  - Transfer all duplicated records $O_d$ on merge list to output

- OR queries $e_1 \vee e_2$
  - Construct a merged list M for $D_{e_1}$ and $D_{e_2}$
  - Transfer all unique records $O_u$ on merge list to output

# Processing Boolean Query: Method 2

- NOT queries $e_1 \wedge \neg e_2$
  - Construct a merged list M for $D_{e_1}$ and $D_{e_2}$
  - Remove all the items appearing only once on this list → First_List
  - Create a merge list composed of $D_{e_1}$ and First_List → Second_List
  - Remove items appearing more than once from Second_List
  - Transfer the remaining items (those that are alone) to output – $O_a$
  - $e_1 \setminus (e_1 \wedge e_2)$

# Boolean System: Method 2, Example

- $q = \left((t_1 \vee\ t_2) \wedge \neg t_3\right)$
- $D_{t_1}: \{1,3\}, D_{t_2}: \{1,2\}, D_{t_3}: \{2,3,4\}$
- $(t_1 \vee\ t_2)$
- M($D_{t_1}$, $D_{t_2}$): { 1, 1, 2, 3}
- O($t_1 \vee\ t_2$): {1, 2, 3} – unique
- M: Merge Operation, O: Output Selection

# Boolean System: Method 2, Example

- $((t_1 \vee t_2) \wedge \neg t_3)$
- O$(t_1 \vee t_2)$: {1, 2, 3} $\rightarrow D_{t_{1\vee}t_2}$
- $D_{t_3}$: { 2, 3, 4}
- M$(D_{t_{1\vee}t_2}, D_{t_3})$: { 1, 2, 2, 3, 3, 4}
- O$((t_1 \vee t_2) \wedge t_3)$: {2, 3} – duplicate
- M$(D_{t_{1\vee}t_2}, D_{(t_1\vee t_2)\wedge t_3})$: {1, 2, 2, 3, 3}
- O$\big((t_1 \vee t_2) \wedge \neg t_3\big)$: {1} - alone

# Boolean System: Variations

- Extended Boolean
  - Has standard operations
    - AND, OR and NOT
  - Plus
    - Term Proximity
      - Within X words, sentences, paragraphs
    - Wildcard Matching
- Fuzzy
  - Allow for range
  - Function F no longer restricted to {0,1}

# Questions?

# References


- Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze, Introduction to Information Retrieval, Chapter 1, 2008.

- Abraham Bookstein and William Cooper, “A General Mathematical Model for Information Retrieval Systems”, The Library Quarterly, Vol 26, no. 2, pp 153 - 67.

- Ryan Benton’s Lecture Notes: http://www.cacs.louisiana.edu/~cmps561/notes/CMPS561Fall10-BooleanIR-fa2011.pdf