Semantic Search
Focus: IR on Structured Data

8th European Summer School on Information Retrieval
Duc Thanh Tran

Institute AIFB, KIT, Germany

Tran@aifb.uni-karlsruhe.de

http://sites.google.com/site/kimducthanh

Agenda

- Why Semantic Search?
- What is Semantic Search?
- A Semantic Search direction - IR on structured data
  - Matching
  - Ranking
- Conclusions

Why Semantic Search?

Why Semantic Search?

- Solve main classes of queries,
- But long tail queries…
  - “teacher math class Goethe”
- Several **problematic cases**
  - **Ambiguous / imprecise queries**
    - “Paris Hilton”
    - “strong adventures people from Germany”
  - **Specific, complex queries** (factual, aggregated)
    - “32 year old computer scientist living in Karlsruhe”
    - “digital camera under 300 dollars produced by canon in 1992”

Many of these queries would not be asked by users, who learned over time what search technology can and can not do.

These queries require **precise understanding** of the underlying information needs and data, and **aggregating results**.

Why Semantic Search?

- Towards a Semantic Web
- Large number of Web data vocabularies published in RDFS and OWL
  - Schema.org
  - Dbpedia ontology
- Large amounts of data published in RDF / RDFa
  - Linked Data
  - Embedded metadata

Semantics captured by taxonomies, ontologies, structured metadata can help to obtain **precise understanding,** to **aggregate information** from different sources, and to **retrieve relevant results!**

Vocabularies

- DBpedia ontology

Class	Instances
Resource (overall)	1,830,000
Place	526,000
Person	416,000
Work	262,000
Species	183,000
Organisation	169,000

from : http://wiki.dbpedia.org/Ontology

Vocabularies

DBpedia [Bizer et al, JWS02]

- TradeUnion (edit)

- Person (edit)
  - Ambassador (edit)
  - Architect (edit)
  - Artist (edit)
    - Actor (edit)
      - AdultActor (edit)
      - VoiceActor (edit)
    - Comedian (edit)
    - ComicsCreator (edit)
    - MusicalArtist (edit)
    - Writer (edit)
  - Astronaut (edit)
  - Athlete (edit)

from : http://wiki.dbpedia.org/Ontology

Person (Show in class hierarchy)

Label (fr): personne
Label (pt): pessoa
Label (el): Πληροφορίες προσώπου
Label (de): Person
Label (en): person
Label (sl): Oseba
Super classes: *owl:Thing*

from : http://wiki.dbpedia.org/Ontology

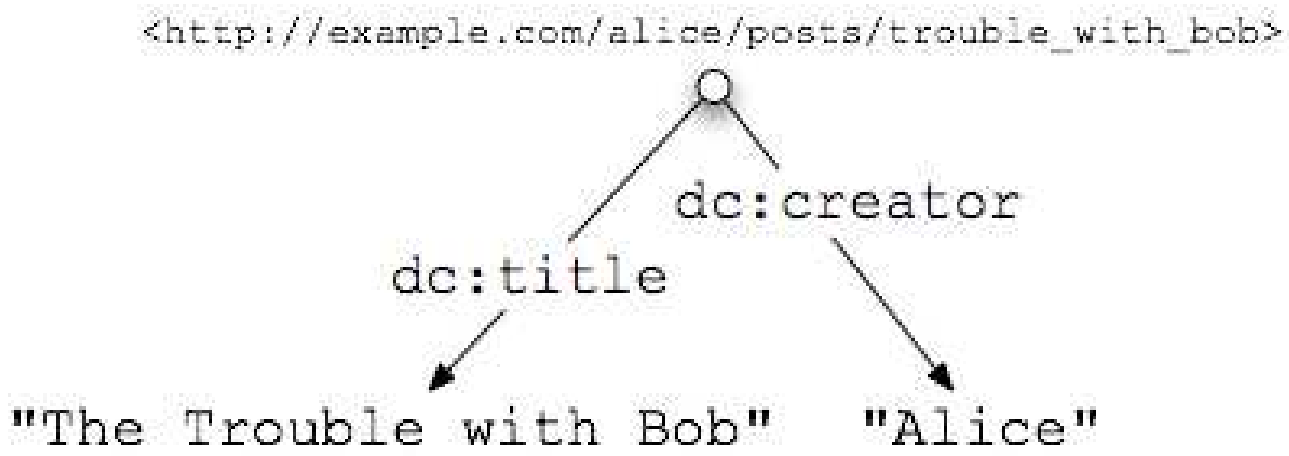
Properties on *Person*:

Name	Label	Domain	Range	Comment
abbreviation (edit)	abbreviation	*owl:Thing*	*xsd:string*	
abstract (edit)	has abstract	*owl:Thing*	*xsd:string*	
access (edit)	access	*owl:Thing*	*xsd:string*	
accessDate (edit)	access date	*owl:Thing*	*xsd:gYear*	
activeYearsEndDate (edit)	active years end date	*owl:Thing*	*xsd:date*	
activeYearsEndYear (edit)	active years end year	*owl:Thing*	*xsd:gYear*	
activeYearsStartDate (edit)	active years start date	*owl:Thing*	*xsd:date*	
activeYearsStartYear (edit)	active years start year	*owl:Thing*	*xsd:gYear*	
album (edit)	album	*owl:Thing*	Album	
alias (edit)	alias	*owl:Thing*	*xsd:string*	
almaMater (edit)	alma mater	Person	EducationalInstitution	schools that they attended
alongside (edit)	alongside	*owl:Thing*	Person	
animal (edit)	animal	*owl:Thing*	Animal	

Structured Data

Resource Description Framework (RDF)

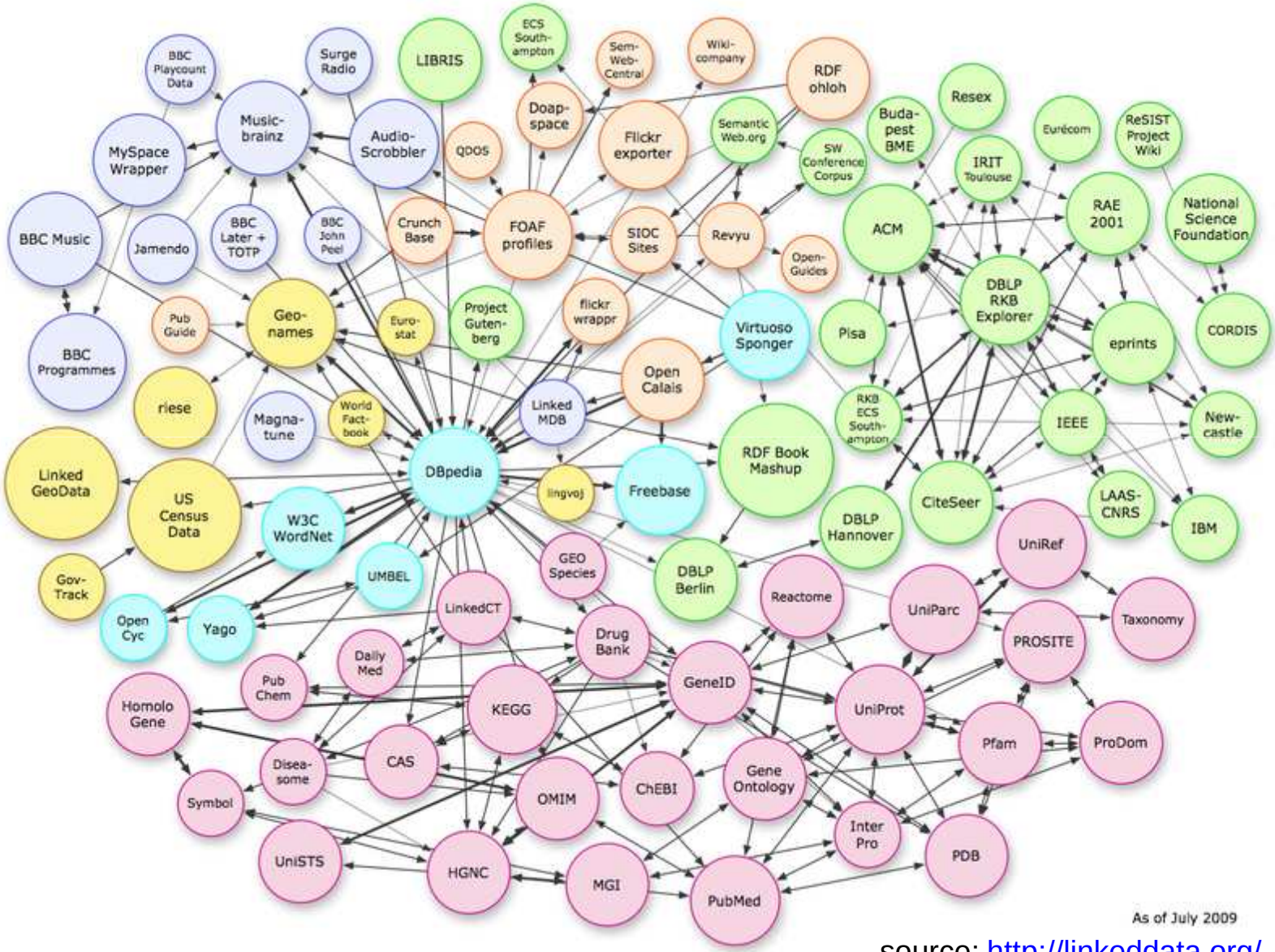
- Each resource (thing, entity) is identified by a URI
- Entity descriptions as sets of facts
  - Triples of (subject, predicate, object)
- A set of triples is published together in an RDF document (forming an RDF graph)



adopted from : http://www.w3.org/TR/xhtml-rdfa-primer/

Structured Data

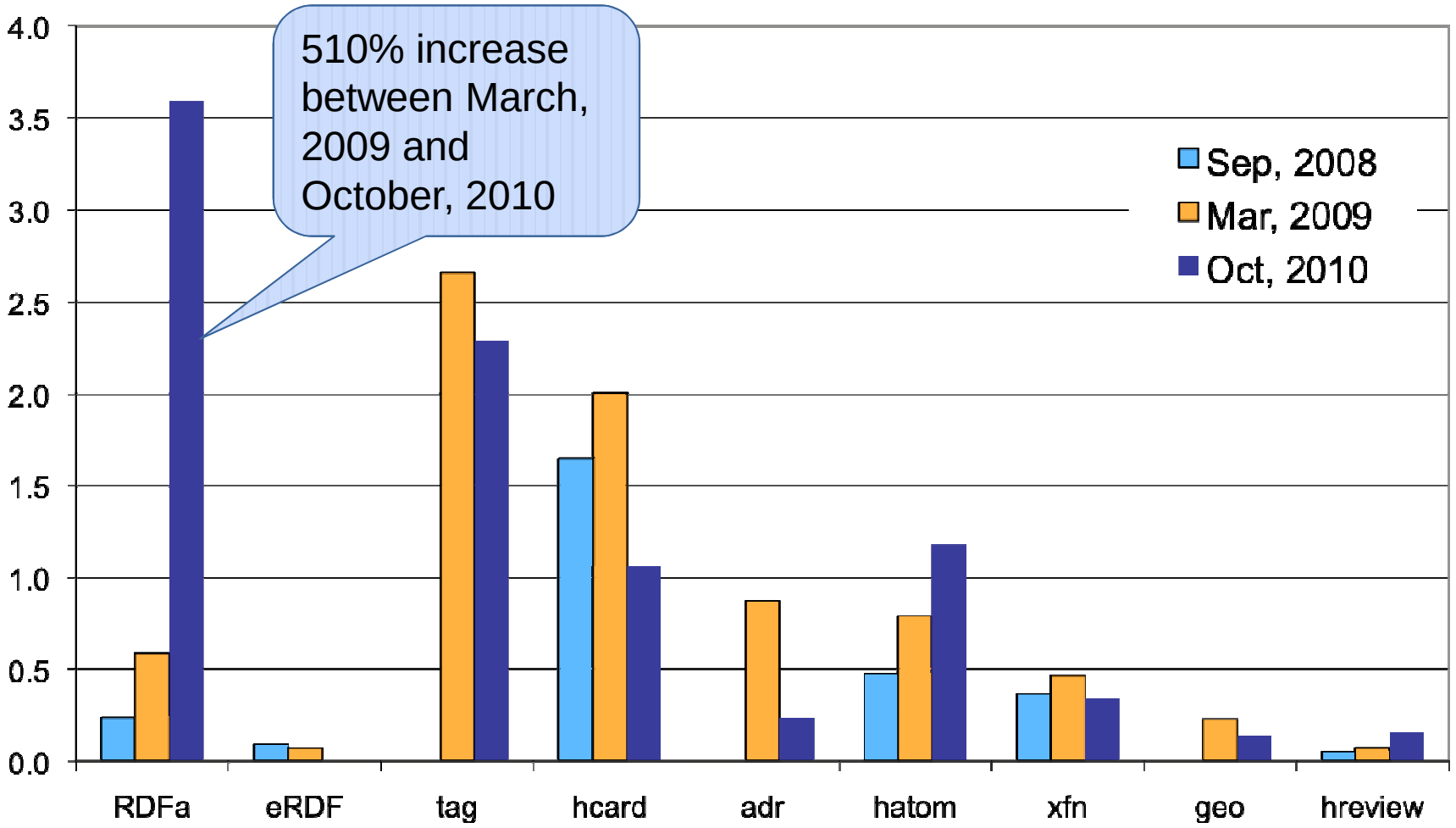
Linked Data



source: http://linkeddata.org/

Metadata

RDFa on the rise



Percentage of URLs with embedded metadata in various formats

from : http://www.slideshare.net/pmika/semtech-2011-semantic-search-tutorial

Metadata

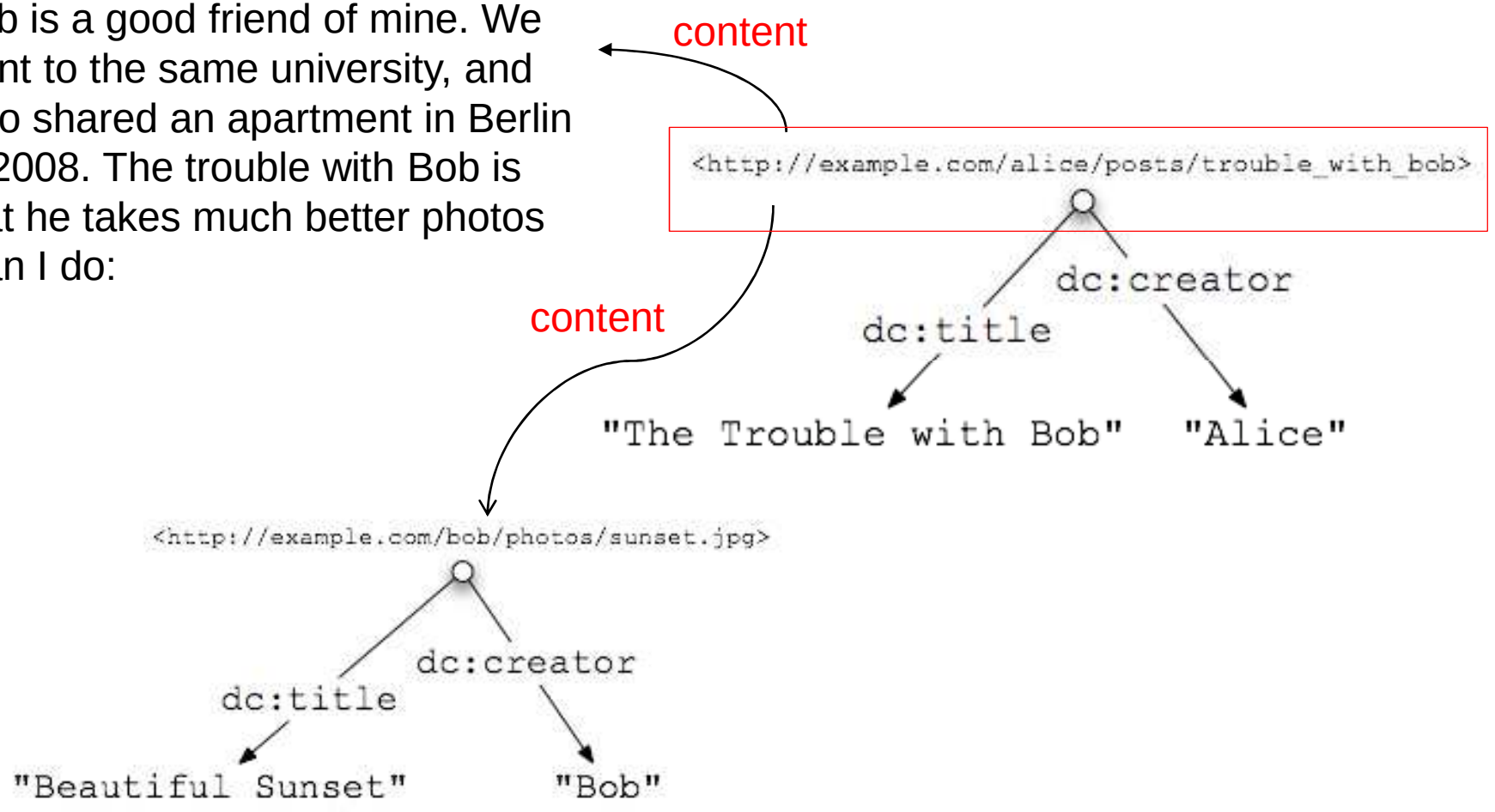
RDFa

```
…
<div about="/alice/posts/trouble_with_bob">
    <h2 property="dc:title">The trouble with Bob</h2>
    <h3 property="dc:creator">Alice</h3>

        Bob is a good friend of mine. We went to the same university, and
        also shared an apartment in Berlin in 2008. The trouble with Bob is
        that he takes much better photos than I do:

    <div about="http://example.com/bob/photos/sunset.jpg">
      <img src="http://example.com/bob/photos/sunset.jpg" />
      <span property="dc:title">Beautiful Sunset</span>
      by <span property="dc:creator">Bob</span>.
    </div>
</div>
…
```

adopted from : http://www.w3.org/TR/xhtml-rdfa-primer/

Metadata
RDFa

Bob is a good friend of mine. We went to the same university, and also shared an apartment in Berlin in 2008. The trouble with Bob is that he takes much better photos than I do:



adopted from : http://www.w3.org/TR/xhtml-rdfa-primer/

What is Semantic Search?

Structure

- Semantics
- Search tasks
  - Document, data, social media, multimedia
- Core search problems
- Semantic search exploits semantics
  - For search tasks
  - For search problems
- Many Semantic Search directions

Semantics

- Semantics is concerned with the meaning of query, data and background knowledge
- Distributional hypothesis / statistical semantics
  - “a word is characterized by the company it keeps”
  - Based on word patterns (co-occurrence frequency of the context words near a given target word)
- **Explicit semantics**
  - Various explicit representations of meaning

Explicit Semantics

- Linguistic models: relationships among **terms**
  - Taxonomies, thesauri, dictionaries of entity names
  - Term relationships: synonymous, hyponymous, broader, narrower…
  - Examples: WordNet, Roget's Thesaurus
- Conceptual models: relationships among **classes of objects**
  - Abstract and conceptual representation of data
  - Terminological part (T-Box) of ontologies, DB schema e.g. relational model
  - Concepts, RDFS classes, associations, relationships, attributes…
  - Examples: SUMO, DBpedia
- Structured data: relationships among **objects**
  - Description of concrete objects
  - Assertional part of ontologies (A-Box), DB instance
  - Tuples, instances, entities, RDF resources, foreign keys, relationships, attributes,…
  - Examples: Linked Data, metadata

Search tasks – document retrieval

- Search on textual data (documents, Web pages)
- Mainly studied in the IR community
- Data and queries
  - **Term-based representation**
- Search algorithms
  - Retrieve documents relevant for query keywords
  - Match query term against terms / content of documents
  - Leverage **statistical semantics** for dealing with ambiguity and for ranking
  - Optimized, work well for **navigational**, **topical search**
  - Less so for complex information needs
  - Web scale

Search tasks – data retrieval

- Focus on structured data and retrieve direct answers
- Data and queries
  - **Structured models**
- Search algorithms
  - Retrieve direct answers that match structured queries
  - **Structure matching**: term / content based relevance less the focus, but structure filtering based on joins
  - Use **relational semantics** in structured data
  - Optimized for **complex structured information needs** / queries, less so for text-based relevance
  - More complex processing → efficiency, scalability

Search tasks

Addressing complex information needs

Combination of data and document retrieval

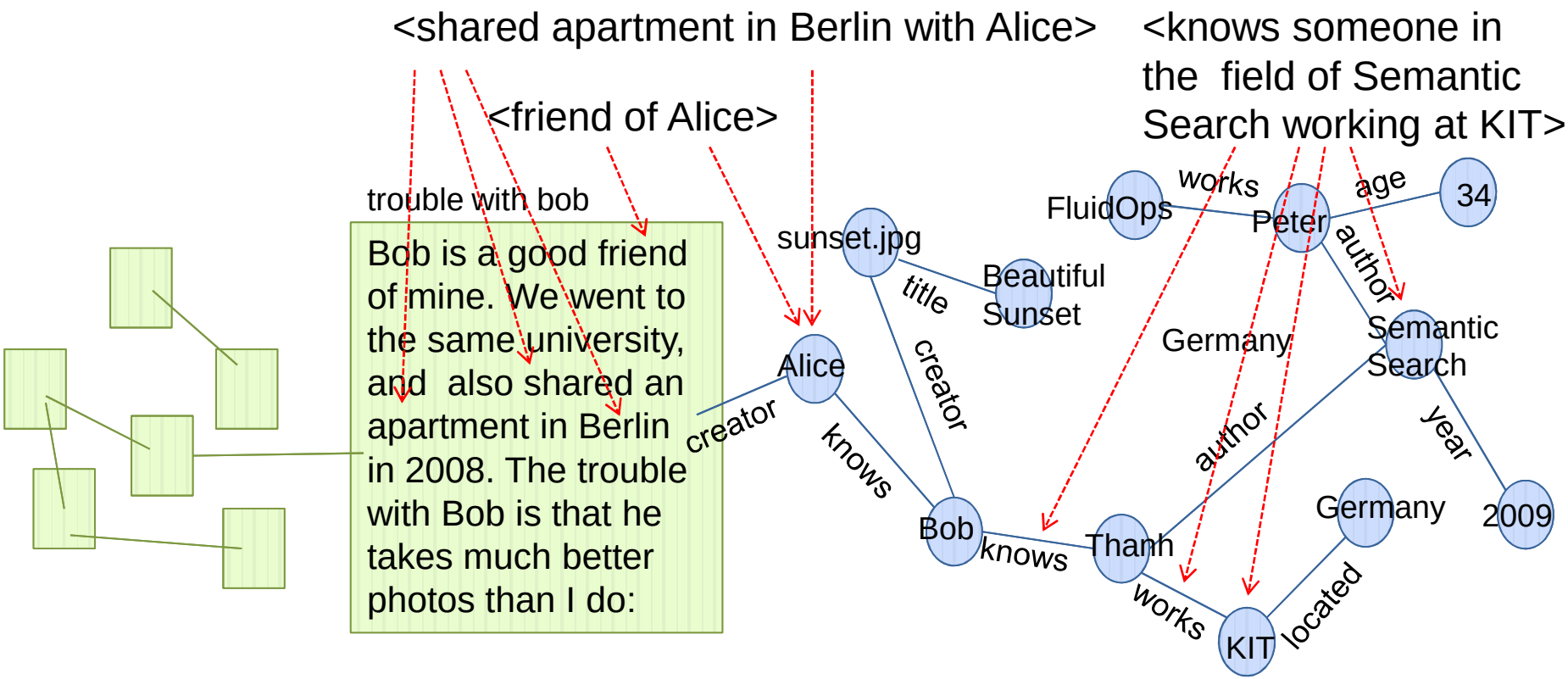
- **Movies** directed by Stephen Spielberg where synopsis mentions dinosaurs.
- **Publications** authored by 32 year old computer scientist living in Karlsruhe, which mention Semantic Search
- **Information** about a friend of Alice, who shared an apartment with her in Berlin and knows someone in the field of Semantic Search working at KIT

Structured data with textual attribute values (content, description)

Documents with metadata

Search tasks
e.g. combination of data and document retrieval

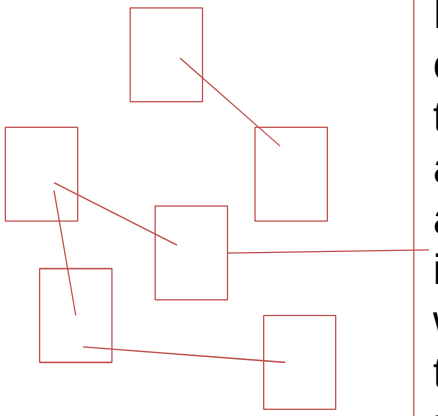
- *"Information about a friend of Alice, who shared an apartment with her in Berlin and knows someone in the field of Semantic Search working at KIT".*



Core search problems
Term ambiguity

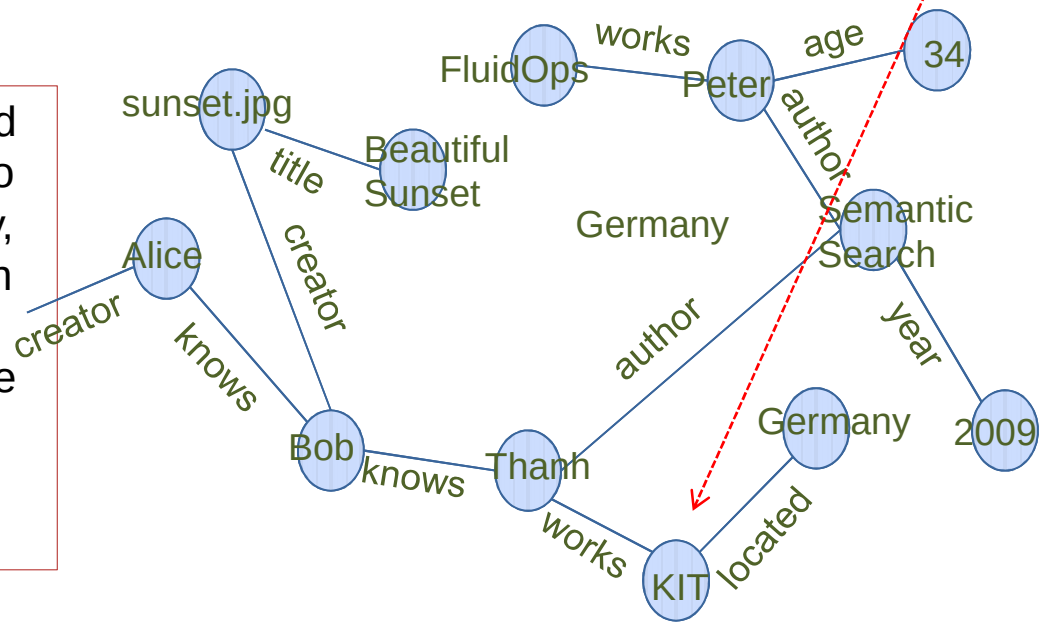
apartment shared Berlin Alice

knows someone works at KIT



trouble with bob

Bob is a good friend of mine. We went to the same university, and also shared an apartment in Berlin in 2008. The trouble with Bob is that he takes much better photos than I do:

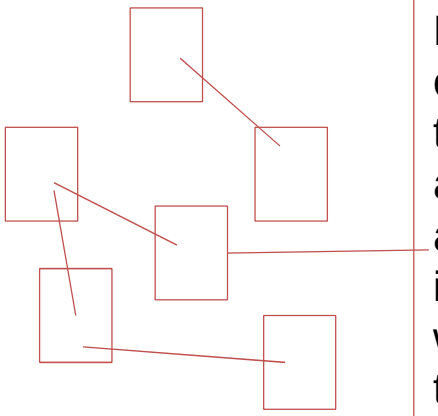


Syntax / Semantic

Is “BerllinNN” same as “Berlin”? What is meant by “KIT”?

Core search problems
Structure ambiguity

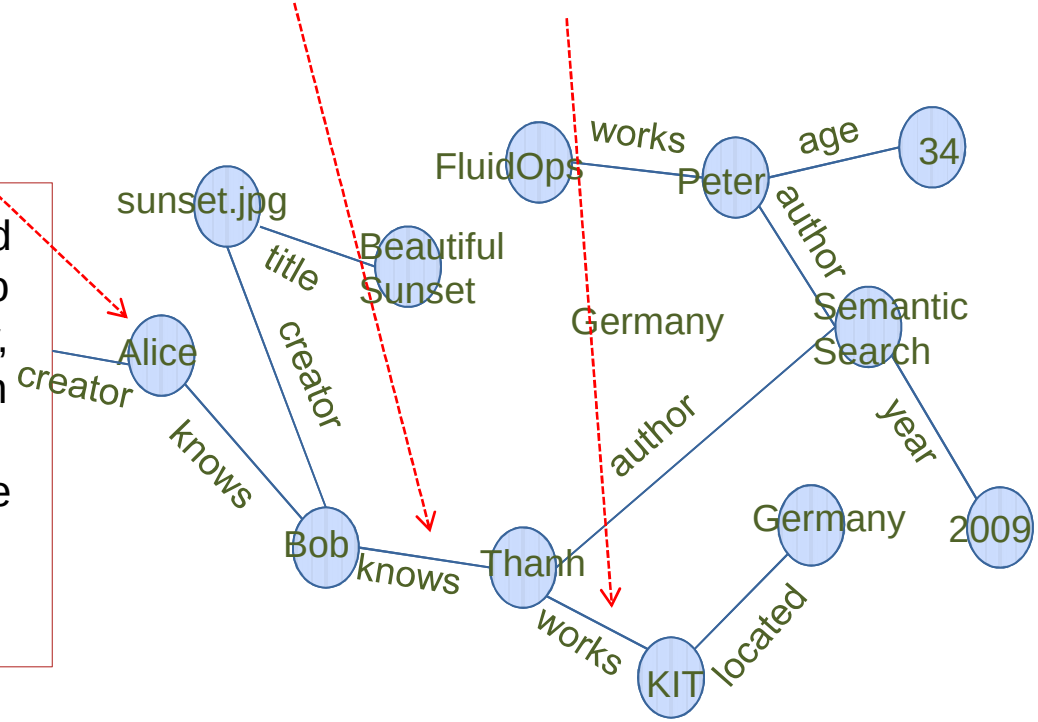
apartment shared Berlin Alice



trouble with bob

Bob is a good friend of mine. We went to the same university, and also shared an apartment in Berlin in 2008. The trouble with Bob is that he takes much better photos than I do:

knows someone works at KIT



Explicit semantics in structured data reduces structure ambiguity

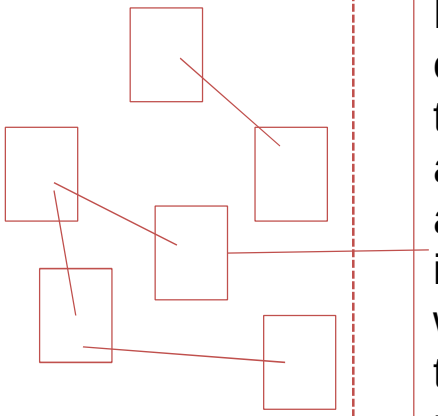
What is the connection between “Berlin” and “Alice”?

What is the relationship between “someone” and KIT?

Core search problems
Content ambiguity

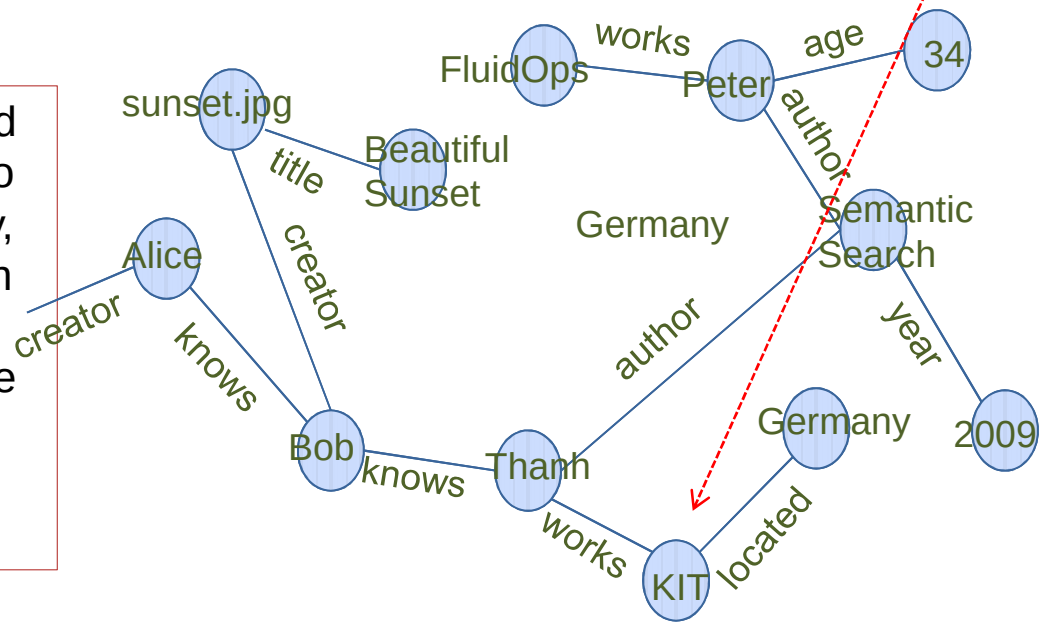
apartment shared Berlin Alice

knows someone works at KIT



trouble with bob

Bob is a good friend of mine. We went to the same university, and also shared an apartment in Berlin in 2008. The trouble with Bob is that he takes much better photos than I do:



Understanding of **term** and **structure** in content helps!

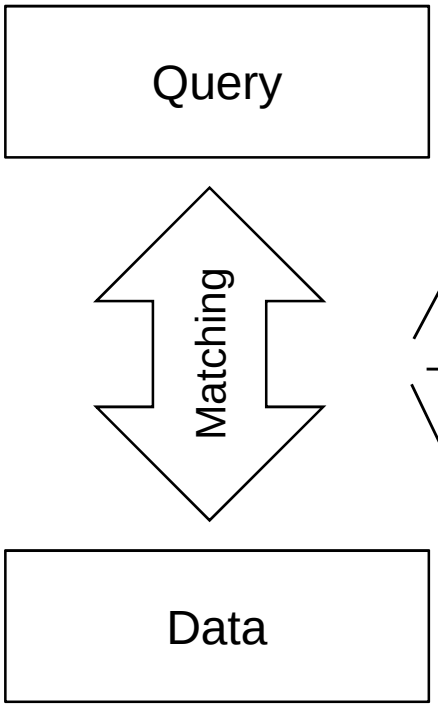
Is the document about Berlin (as a city)?
Is the element's content / label about KIT?

Is the graph about "apartment shared Berlin Alice knows someone works at KIT"?

Core search problems

Dealing with ambiguities: matching and ranking

2 scenarios:
ambiguity (IR) vs.
no ambiguity (DB)



- Exact
  - Complete
  - Sound
- Approximate
  - Not complete
  - Not sound
  - Both the above
- Ranked
  - **Matching** + **ranking**
  - Top-k

Ranked: **ambiguities** in query and data representation → results cannot be guaranteed to **exactly match** the query (i.e. multiple interpretations lead to multiple non-equivalent matches).

Ambiguities at level of **elements** (term, content) and **relationships between elements** (structure)

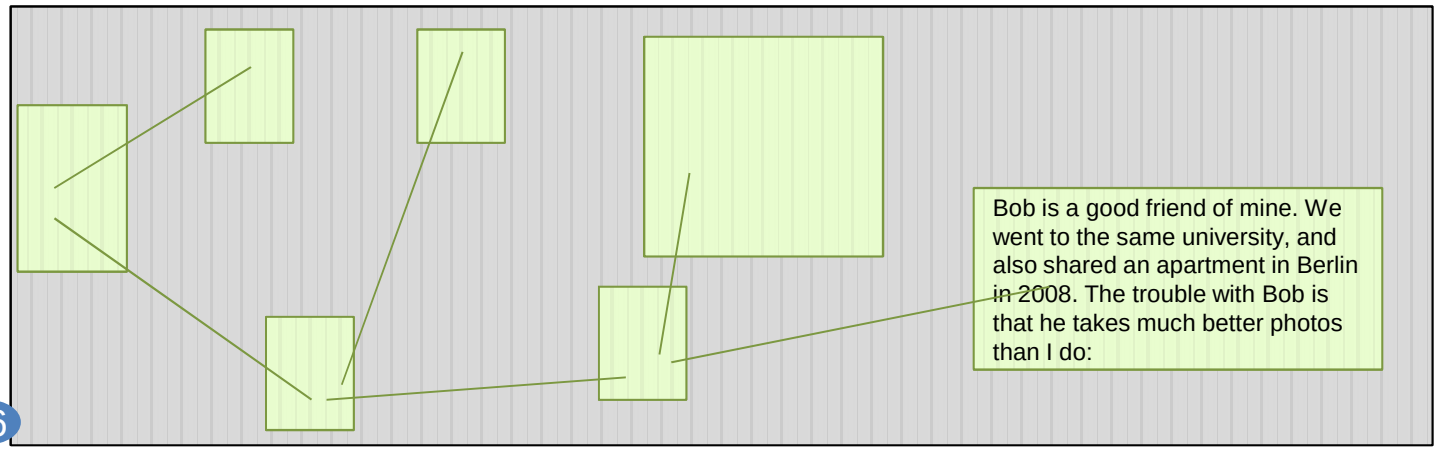
Matching mainly focuses on **efficiency of computing matches** whereas **ranking** deals with **degree of matching** (relevance)!

Search

- *“Information about a friend of Alice, who shared an apartment with her in Berlin and knows someone in the field of Semantic Search working at KIT”.*

- **Term**: which Berlin?
- **Content**: which documents are about Berlin?

Distributional semantics / statistical reasoning over topic models, language models

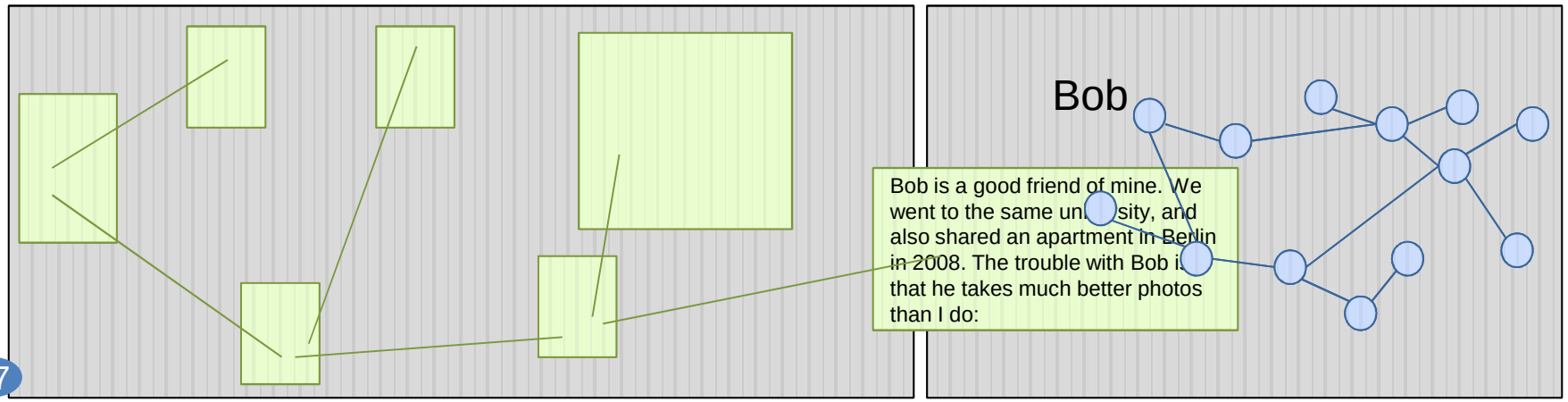


Semantic Search

- *“Information about a friend of Alice, who shared an apartment with her in Berlin and knows someone in the field of Semantic Search working at KIT”.*

  - **Structure**: friend of, knows, shares

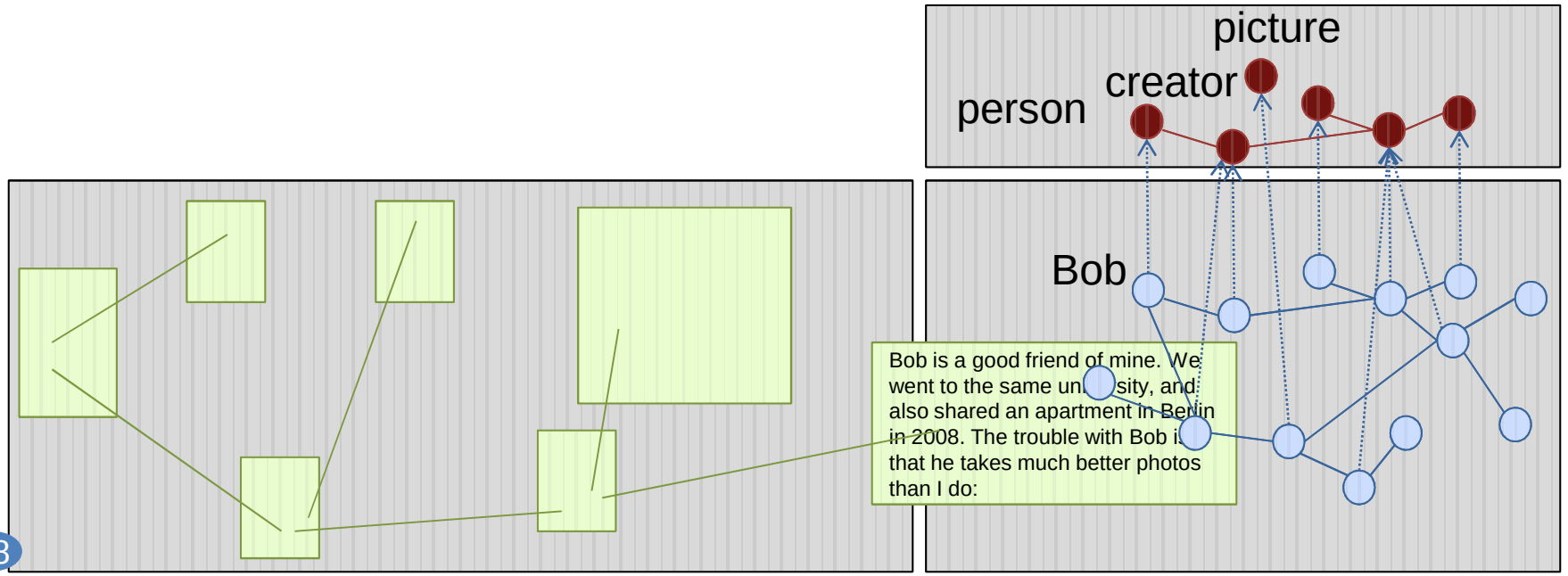
Relational semantics of **structured data** in various datasets



Semantic Search

- *“Information about a friend of Alice, who shared an apartment with her in Berlin and knows someone in the field of Semantic Search working at KIT”.*
- **Term**: creator is a subclass of person

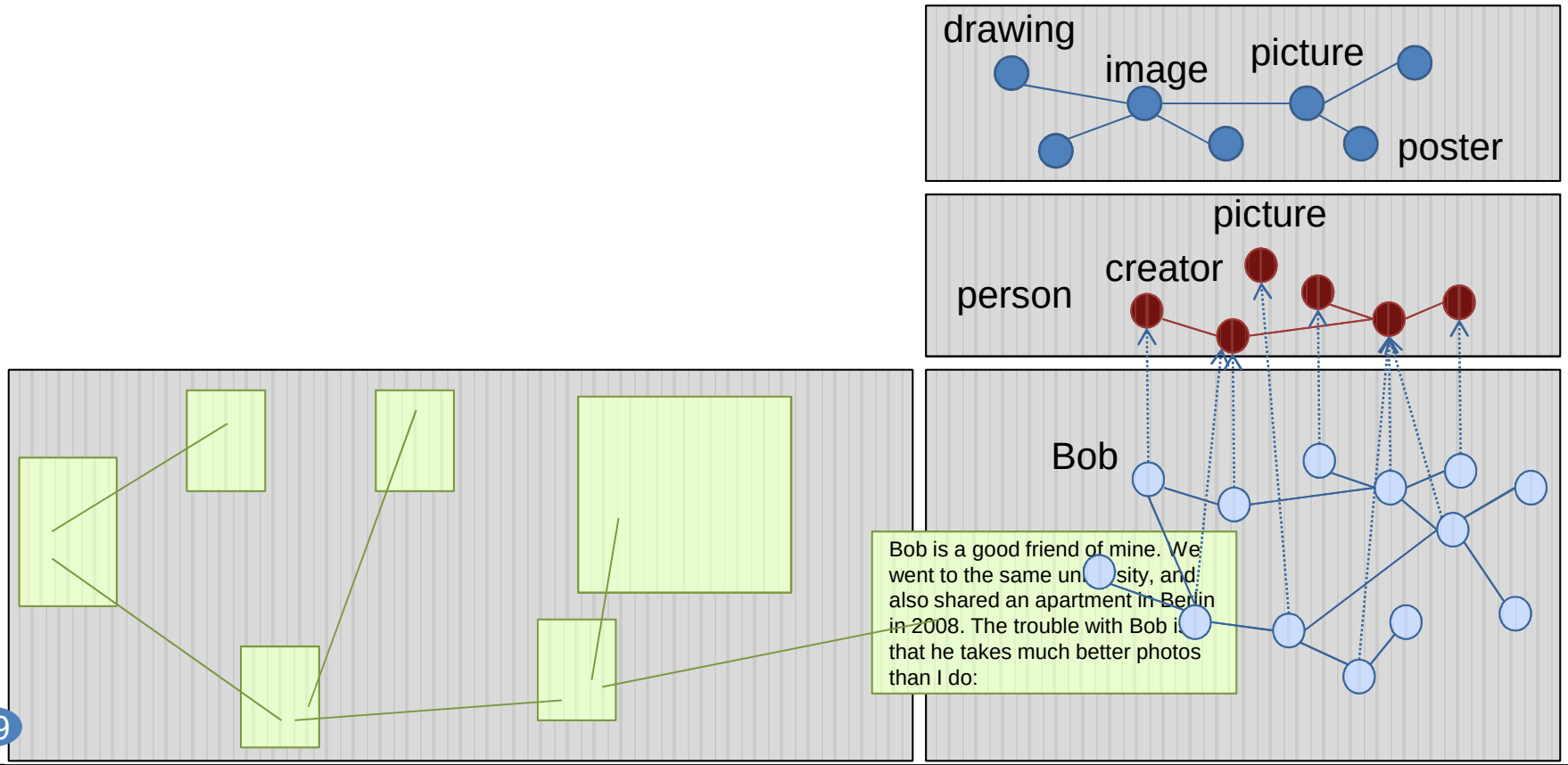
Semantics captured in **conceptual models**, e.g. class subsumption, instance classification (logic-based reasoning)



Semantic Search

- *“Information about a friend of Alice, who shared an apartment with her in Berlin and knows someone in the field of Semantic Search working at KIT”.*
- **Term**: image is synonym of picture

Semantics captured in **linguistic models**, e.g. reasoning over term relationships



Semantic Search

- A retrieval paradigm that exploits **semantics** of
  - data, query, background knowledge [Tran et al, JWS11]

  to interpret and incorporate the intent of query and the meaning of data into the search algorithms (more generally: search process)
- Different directions, employing various models of semantics of
  - Terms (linguistic models)
  - Concepts (conceptual models)

  to deal with **ambiguous queries**
  - Relational information (structured data) in
  - Different datasets

  to produce complex structured, aggregated results to answer **complex information needs**
- Orthogonal to retrieval tasks / specific type of approaches
  - Document retrieval
  - Data retrieval
  - Multimedia retrieval
  - Social media retrieval

Semantic Search

????

	Semantic Search	Information Retrieval (IR)	Data Retrieval (DB)	Multimedia Retrieval
Keyword query	**X**	**X**	x	
Structured query	x		**X**	
Textual data	**X**	**X**	x	
Structured data	**X**	x	**X**	
Conceptual model	**X**		**X**	
Linguistic model	**X**	x		
Term matching	**X**	**X**	x	
Structure matching	**X**		**X**	
Content matching	**X**	**X**		
Ranking	**X**	**X**	x	

Focus of the following technical parts

IR on structured data

- Motivation
  - IR is user-centric!
  - Text-based querying paradigms more intuitive for end-users!
  - Keyword search widely adopted!
- Focus
  - **Keyword query on structured data**, i.e. “a direction” of semantic search, which employs semantics of
    - Relational information (structured data) in
    - Different datasets

    to produce complex structured, aggregated results to answer complex information needs
- Similar, complementary to DB keyword search tutorial, emphasizes [Chen et al, SIGMOD09]
  - The role of textual data: data graphs with textual content nodes
  - The role of semantics
  - **Ranking**

Matching

Structure

- **Keyword search: keywords over data graphs**
  - Term matching
  - Content matching
  - Structure matching
- Schema-based keyword search
- Schema-agnostic keyword search
  - Online search algorithms
  - Index-based approaches

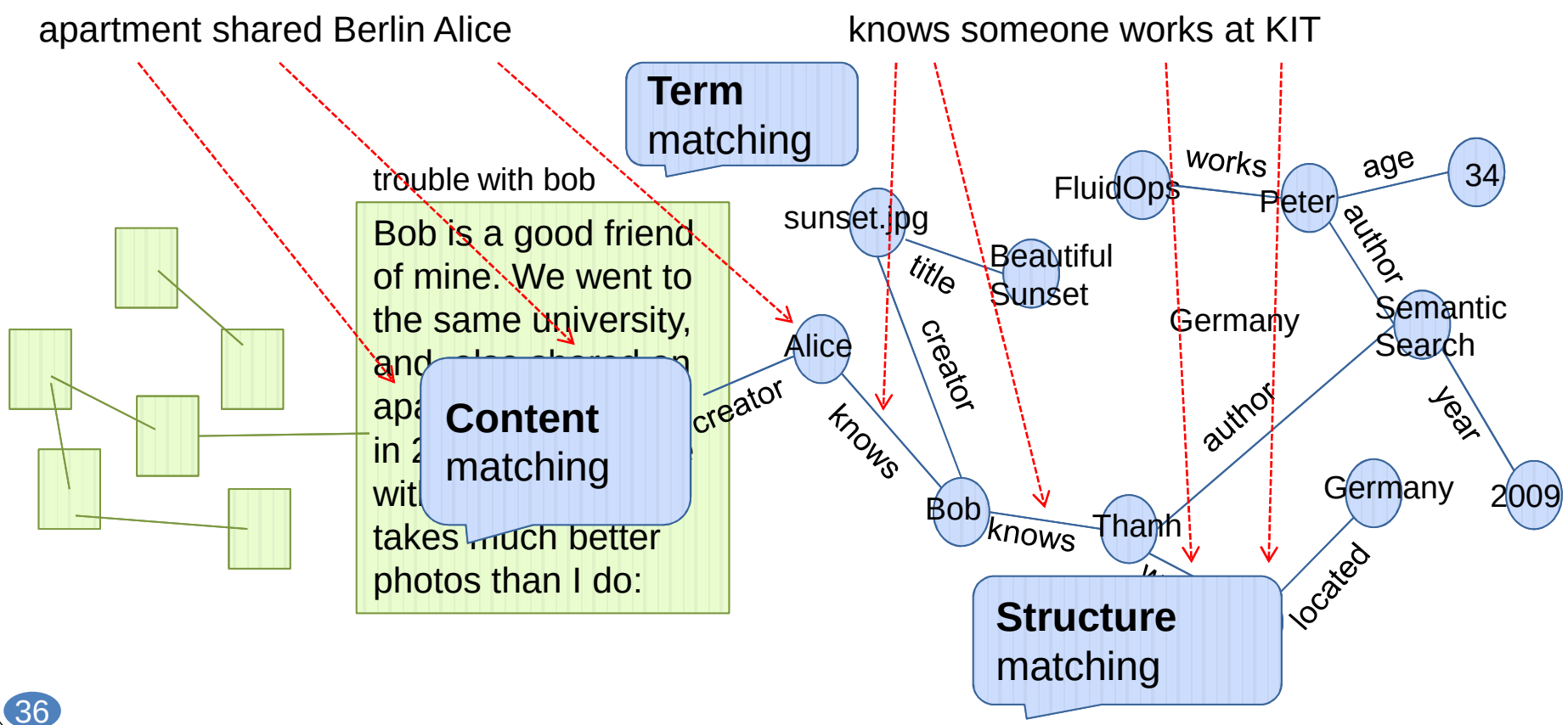
Keyword search approaches

- Finding “substructures” matching keyword nodes
- Different **result semantics** for different types of data
  - Textual data (Web pages connected via hyperlink)
  - DB (tuple connected via foreign keys)
  - XML (elements/attributes via parent-child edges)
- Commonly used results: **Steiner tree** / subgraph
  - Connect keyword matching elements
  - Contain one keyword matching element for every query keyword
  - Minimal substructures: closely connected keyword nodes
- Query is ambiguous, lacks explicit structure constraints
  - NP-hard, thus **efficiency of matching is a problem**
  - Large amounts on candidate matches, thus **ranking** is a problem

Keyword search on hybrid data graphs

Example information need

*"Information about a friend of **Alice**, who **shared an apartment with her in Berlin** and knows someone working at KIT."*



Term matching

- Distance-based (**syntax**)
  - Levenshtein distance (edit distance)
  - Hamming distance
  - Jaro-Winkler distance
- Dictionary-based (**semantics**)
  - Taxonomy
  - Dictionary of similar words
  - Translation memory
  - Ontologies

Term matching via **reasoning over term relationships**, e.g. image is synonym of picture

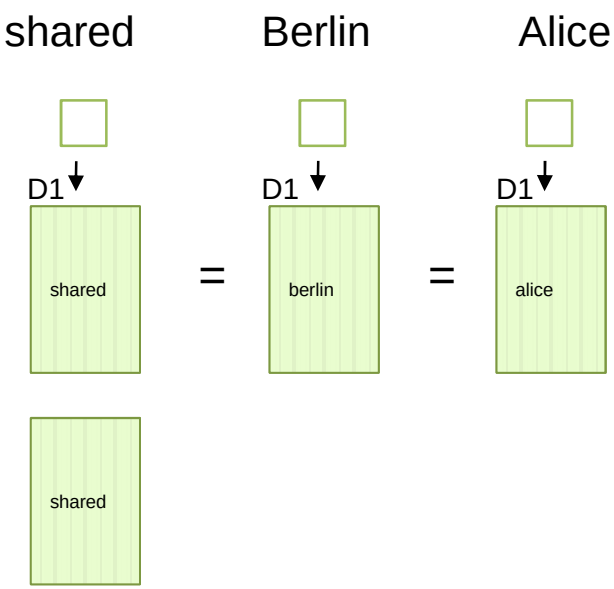
Term matching via **reasoning over concepts**, e.g. creator is a subclass of person

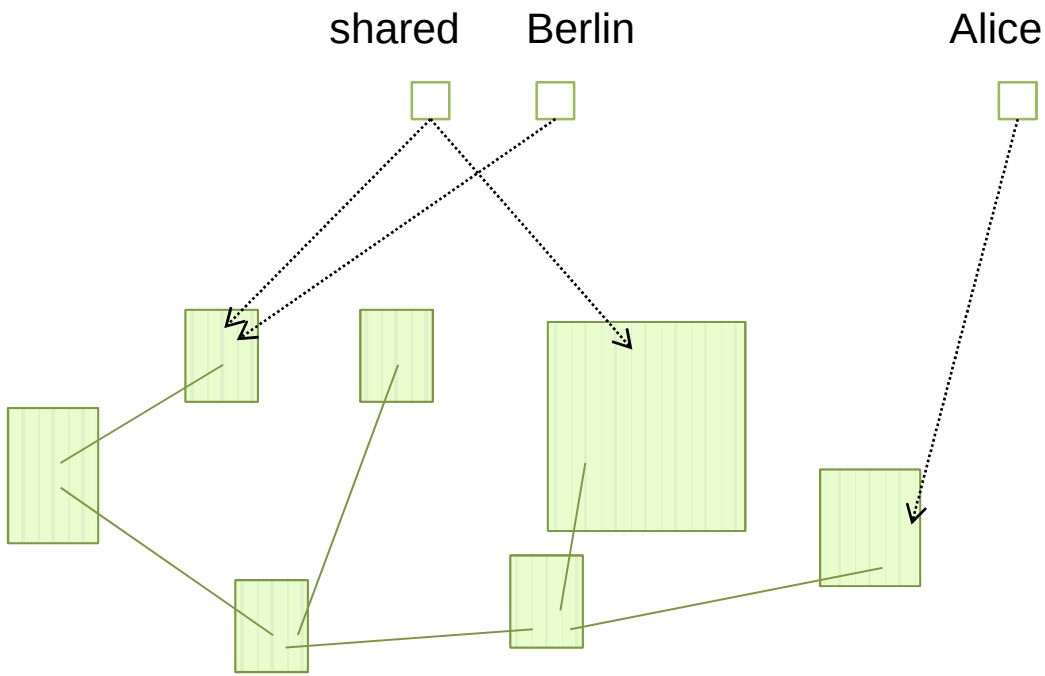
Content matching

- **Retrieve** partial matches
  - Inverted list (inverted index)

    $k_i \rightarrow \{< d_1, pos, score, ...>,$

    $< d_2, pos, score, ...>, ...\}$
- **Combine** partial matches**:** union or join





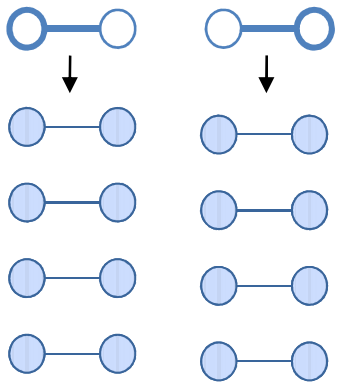
Structure matching

Structure not explicitly given in query → exploration / other kinds of join

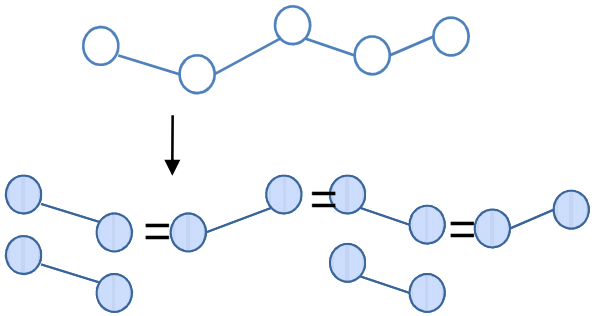
- Retrieve structured data given **patterns**
  - Index on tables
  - Multiple “redundant” indexes to cover different access patterns
- Combine: union or join
  - Blocking, e.g. linear merge join (required sorted input)
  - Non-blocking, e.g. symmetric hash-join
  - Materialized join indexes

Per1 ns:works **?v** **?v** ns:name “KIT”

SP-index PO-index

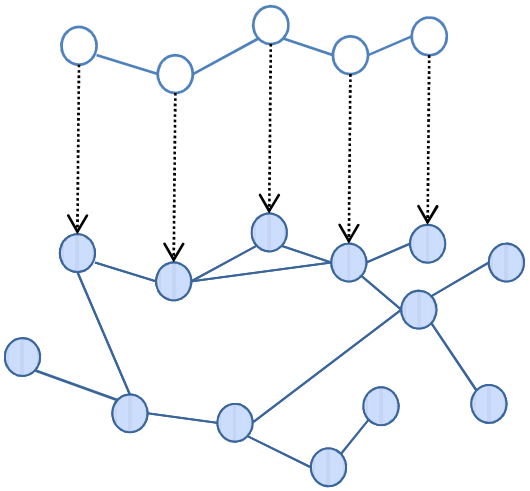


Per1 ns:works Ins1 Ins1 ns:name KIT



Per1 ns:works **Ins1 Ins1** ns:name KIT

?x ns:knows ?y. ?x ns:knows ?z.
?z ns: works ?v. ?v ns:name “KIT”



Structure

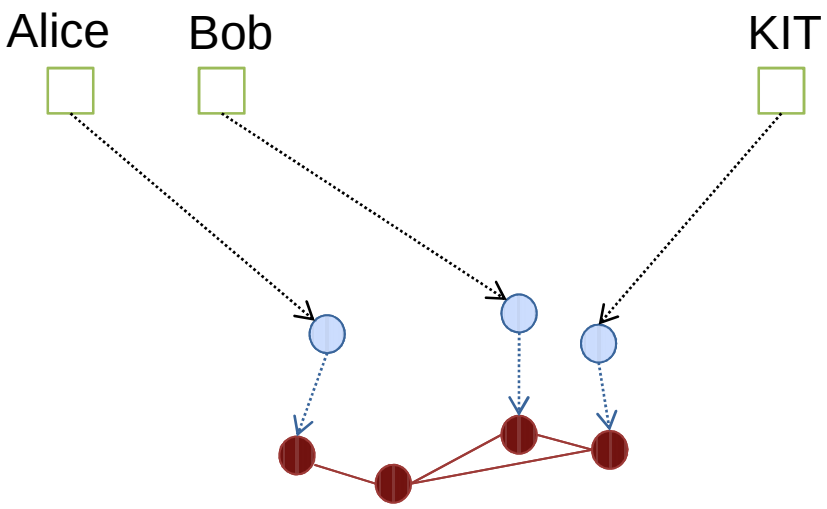
- Keyword search: keywords over data graphs
  - Term matching
  - Content matching
  - Structure matching
- **Schema-based keyword search**
- Schema-agnostic keyword search
  - Online search algorithms
  - Index-based approaches

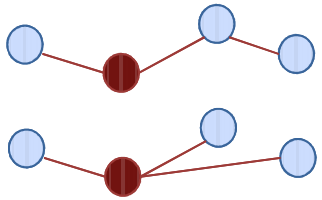
Matching in keyword search – schema-based

- Operate on schema graph [Hristidis et al, VLDB02] [Agrawal et al, ICDE02]
- Query interpretation [Tran et al, ICDE09]
  - Compute queries instead of results
  - Query presentation
  - Query processing by DB engine
- Leverage the power of underlying DB query engine

Linguistic semantics for term matching, **conceptual** and **relational** semantics for interpreting structure matches.

[Qin et al, SIGMOD09]





Result 1

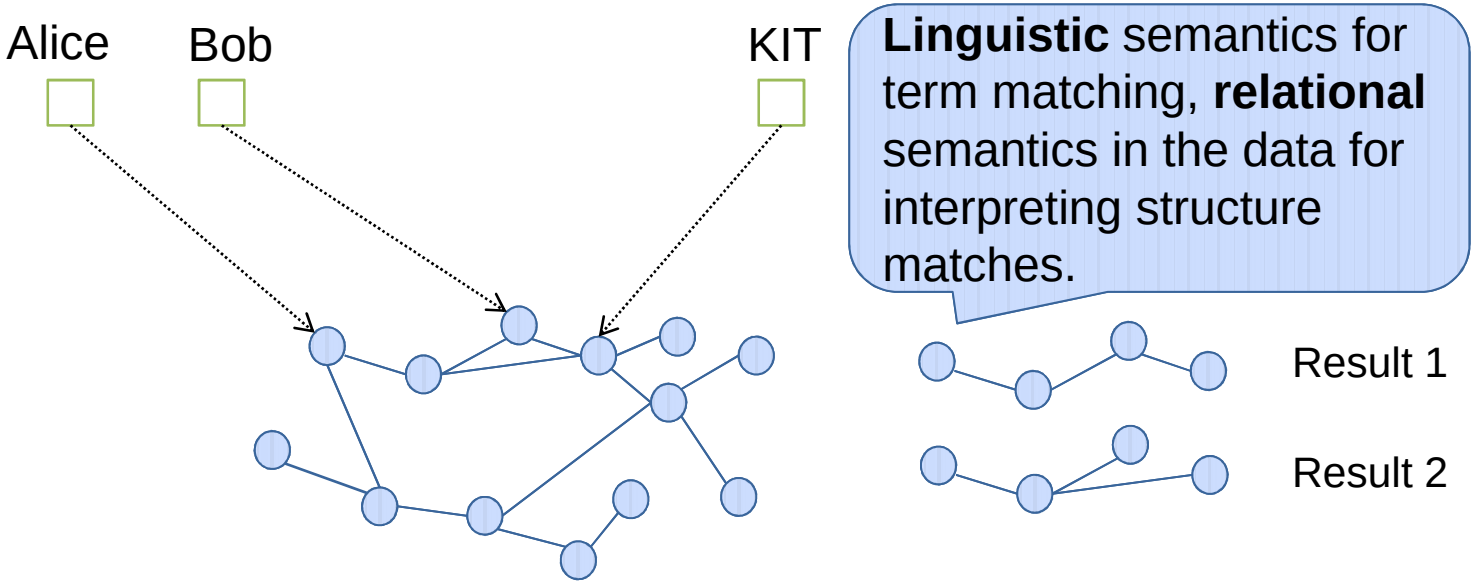
Result 2

Structure

- Keyword search: keywords over data graphs
  - Term matching
  - Content matching
  - Structure matching
- Schema-based keyword search
- **Schema-agnostic keyword search**
  - Online search algorithms
  - Index-based approaches

Matching in keyword search – schema-agnostic

- Operate on data graph
  - No schema needed
  - Flexibly support different types of data e.g. hybrid data graphs
  - Native tailored optimization
- **Online in-memory graph search** [Kacholia et al, VLDB05]
- **Using materialized indexes** [He et al, SIGMOD07] [Li et al, SIGMOD08] [Tran et al, CIKM11]



Structure

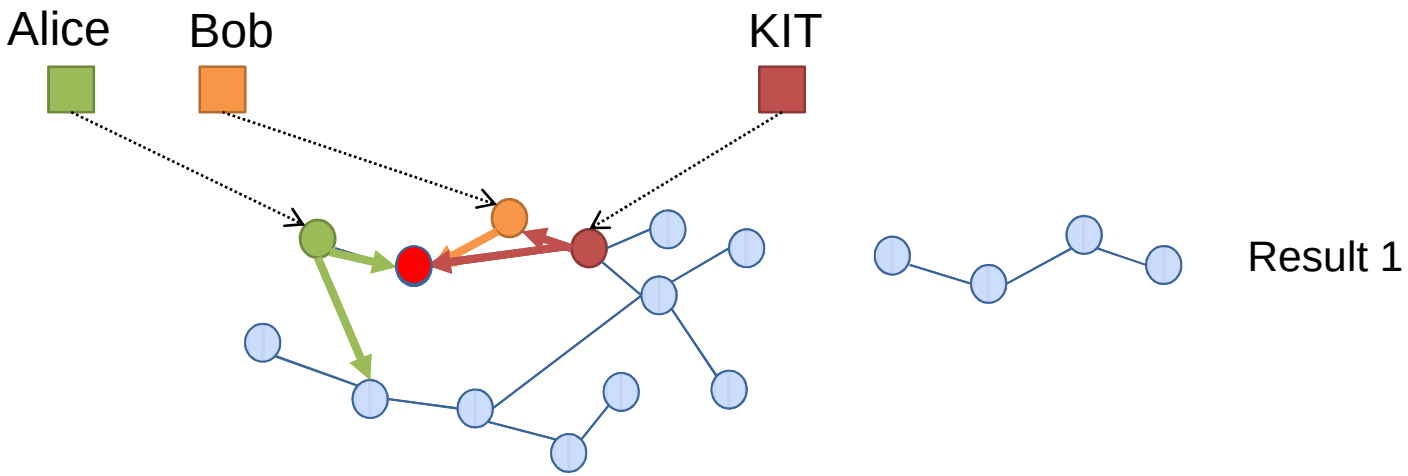
- Keyword search: keywords over data graphs
  - Term matching
  - Content matching
  - Structure matching
- Schema-based keyword search
- Schema-agnostic keyword search
  - **Online search algorithms**
  - Index-based approaches

Online search – top-k exploration

- Compute Steiner tree with distinct roots
- Backward expansion strategy
- Run Dijkstra's single-source-shortest-path algorithms
  - Explore shortest keyword-root paths
  - To find root (an answer)
  - Until k answers are found
  - Approximate: no top-k guarantee, i.e. further answers found later from other expansion paths may have higher score

[Bhalotia et al, ICDE02]

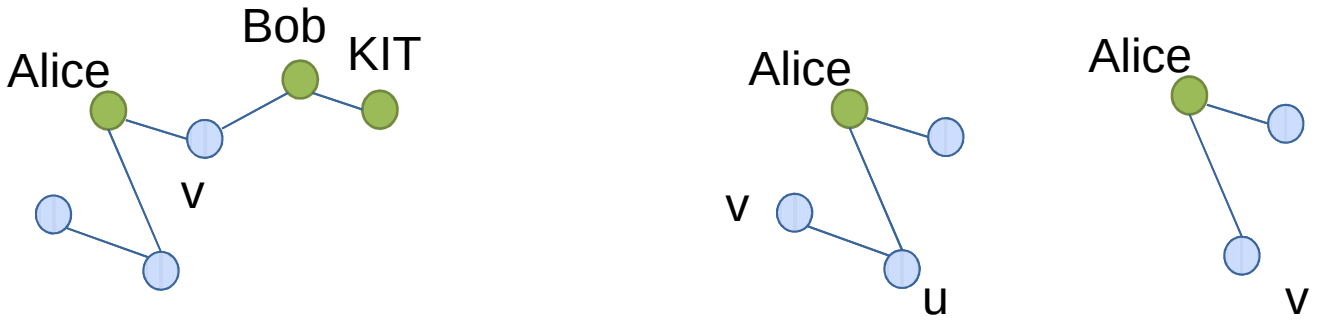
- Complete top-k: terminate safely when lower bound of top-k candidate is higher than upper bound of what can be achieved with remaining inputs [Tran et al, ICDE09]



Online search – dynamic programming

[Ding et al, ICDE07]

- Search problem has **optimal substructure**, i.e. optimal solutions constructed from optimal solutions of subproblems
  - T(v,p,h): rooted at node v ∈ V, height ≤ h, containing a set of keywords p, minimum cost
- Result computation formulated as a recursive series of simpler calculations
  - Tree grow: $\mathrm{T}(v, p, h) = \min_{u \in N(v)} \{(v,u) \oplus \mathrm{T}(u, p, h-1)\}$
  - Tree merge: $\mathrm{T}(v, p_1 \cup p_2, h) = \min_{p_1 \cap p_1 \neq \varnothing} \{\mathrm{T}(v, p_1, h) \oplus \mathrm{T}(v, p_2, h)\}$



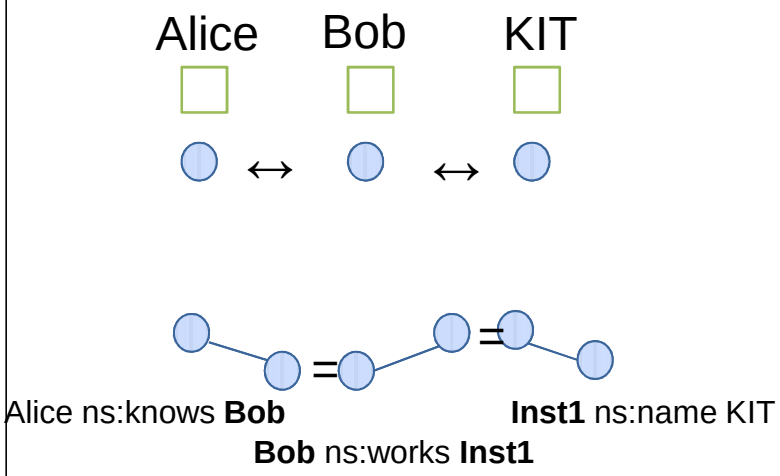
Structure

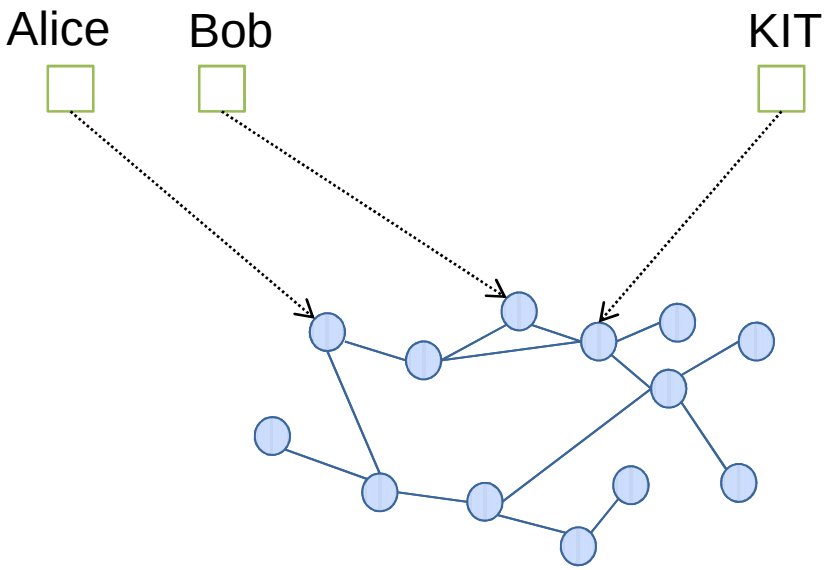
- Taxonomy of matching approaches
- Keyword search: keywords over data graphs
  - Term matching
  - Content matching
  - Structure matching
- Schema-based keyword search
- Schema-agnostic keyword search
  - Online search algorithms
  - **Index-based approaches**

Index-based

- Retrieve keyword elements
  - Using inverted index

    ki → {< n1, score, ...>, < n1, score, ...>,…}
- Retrieve parts of results using materialized index (paths up to graphs)
- Combine via path “join” or graph pruning





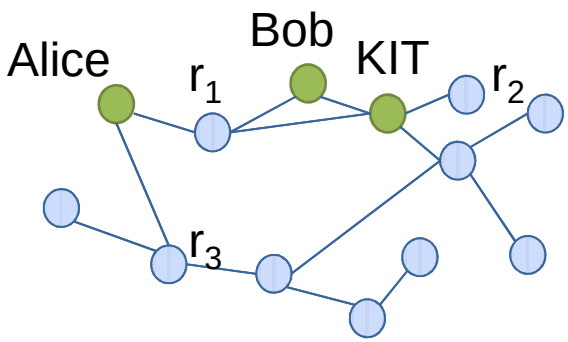
Index-based – path index

Path index (retrieval) + selection top-k (combine)

[He et al, SIGMOD07]

- Results: Steiner trees with distinct root semantics
- Index shortest node-node path (distance) for all possible pairs of nodes
- Results computed via selection top-k (TA algorithm)
  - Each candidate r_i is an object with |q| attributes, i.e. shortest distance (minimal cost) from r_i to all keyword nodes $k_1, k_2, \ldots, k_{|q|}$
  - Score is aggregation of attribute score (cost)

Root	Alice	Bob	KIT
$r_1 = 3$	1	1	1
$r_2 = 6$	3	2	1
$r_3 = 7$	1	3	3



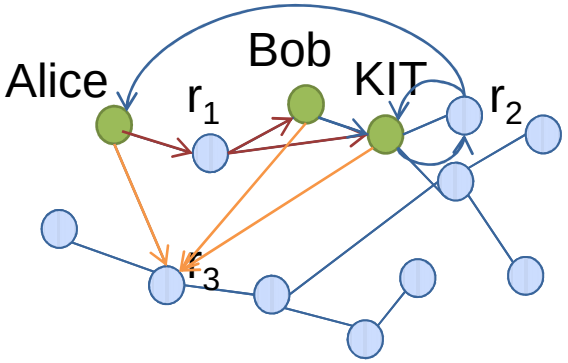
Index-based

Selection top-k (combine)

- TA algorithm
  - |q| inputs, **sorted** according to cost (using index)
  - While |results| < k
    - In round-robin fashion, select keyword node k_i
    - Using node-to-node index (keyword-to-root), retrieve root r_i
    - Retrieving other attribute nodes of r_i via root-to-keyword lookup
    - Add to candidate list

Alice	Bob	KIT
1 → r_1	1 → r_1	1 → r_1
1→ r_3	2 → r_2	1 → r_2
3 → r_2	3 → r_3	3 → r_3

1) $r_1 = 1+1+1=3$
2) $r_2 = 2+1+3=6$
3) $r_1 = 3+1+3=7$

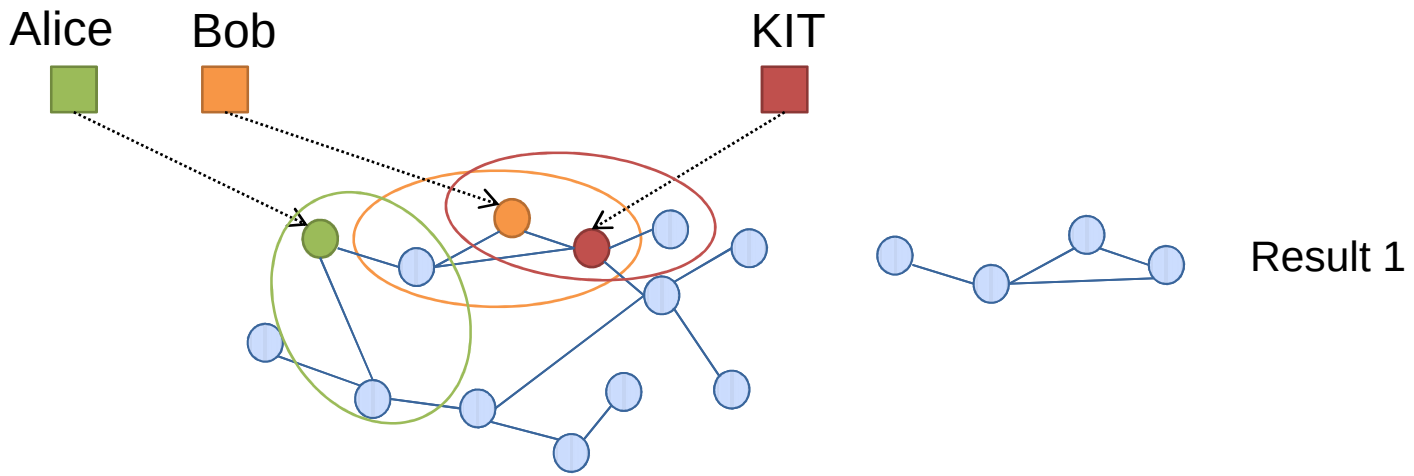


Index-based

Graph index (retrieval) + graph pruning (combine)

[Li et al, SIGMOD08]

- Index r-radius graphs
  - Subgraphs with radius r
  - Maximal → pruning redundant overlapping graphs
  - k_i → G_{ki}
- Compute **r-Radius Steiner graphs**
  - Retrieved G_{ki} for every k_i in q
  - Union G_{ki} , i.e. the set of r-radius graphs that contain all or a portion of the keywords in q
  - Pruning: extract non-Steiner nodes, i.e. those that do not participate in paths connecting keyword elements



Index-based

2-hop cover graph index (retrieval) [Ladwig et al, CIKM11]

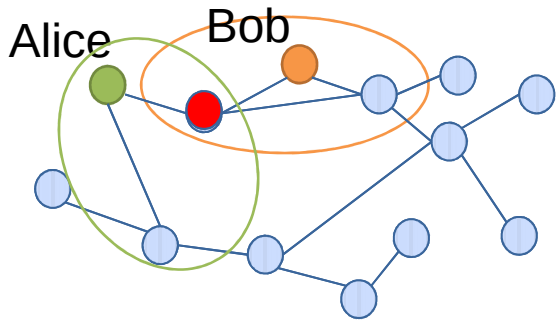
- Use d-length 2-hop cover for graph indexing, i.e. a set of neighbourhood labels NB_n:
  - If there is a path of length d or less between u and v then

$$NB_u \cap NB_v \neq empty$$

  - All paths of length d or less between u and v are (w is the **hop node**)

$$\langle u,\ldots,w,\ldots,v\rangle, w \in NB_u \cap NB_v$$

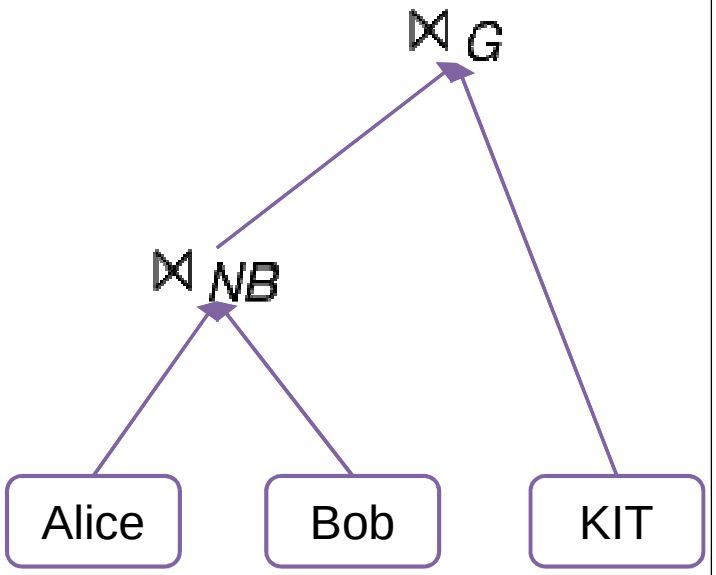
- Trivially, set of d-length neighbourhoods is a d-length 2-hop cover, fined grained pruning a path level reduces that size!



Index-based

Join top-k (combine)

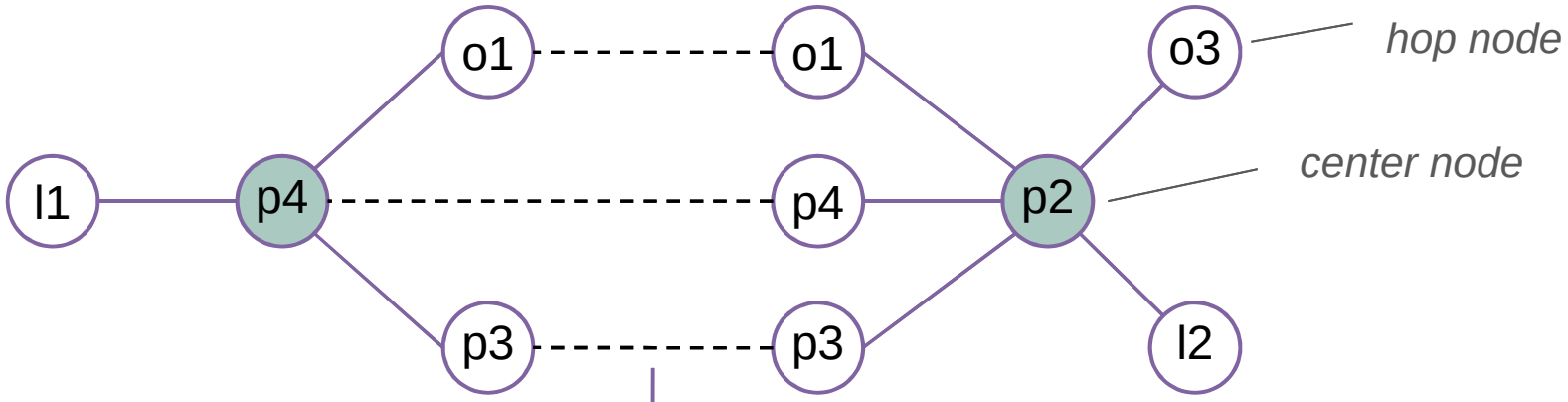
- Result: subgraphs with query-specific **online** scores
- Use neighborhoods of cover to find paths between every pair of keyword elements and join them until they are all connected
- Process
  - Data access to retrieve **keyword neighborhoods**
  - **Neighborhood join** to obtain a **keyword graph**
  - **Graph joins** to combine keyword neighborhood with a keyword graph
  - Join = RankJoin (i.e. use existing join top-k techniques)



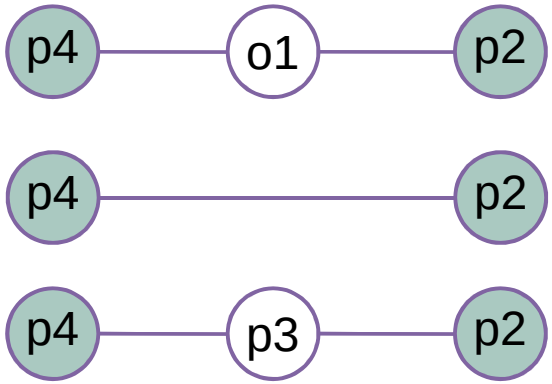
Index-based

Join top-k (combine) – neighborhood join

- Join two keyword neighborhoods
- Two path entries are joined when same hop node



Result: keyword graphs

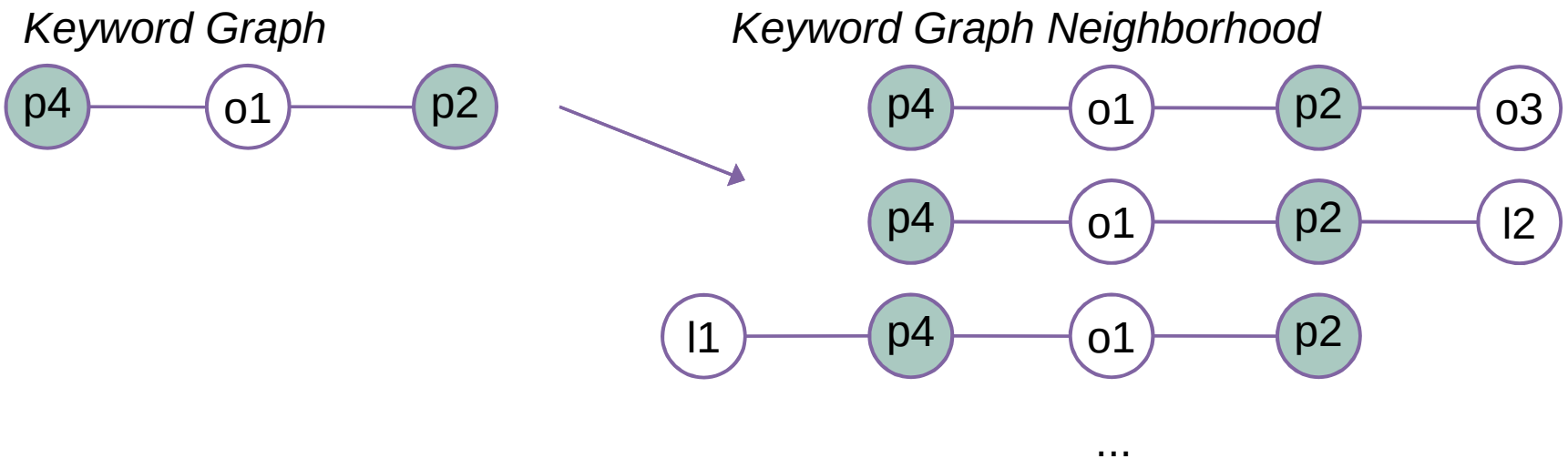


all paths of length *d* between *p4* and *p2* through *o1*

Index-based

Join top-k (combine) – graph join

- Expand keyword graphs to keyword graph neighborhoods



- Graph Join: joins keyword graph neighborhood with keyword neighborhood

Taxonomy of matching approaches

Conceptual semantics

- Schema-based vs. schema-agnostic
- Online search
  - Complete top-k
  - Approximate top-k
  - Backward expansion, bidirectional search, undirected subgraph exploration, dynamic programming

Relational semantics in the data

- Indexing for retrieval + join for combine
  - Path retrieval, then path join
  - Graph retrieval, then graph pruning
  - Graph retrieval, then neighborhood / graph join (neighborhood indexed as a set of paths)

Ranking

Structure

- **Ranking paradigms**
  - Explicit model of relevance
  - No notion of relevance
- Features
  - Content-based
  - Structure-based
  - Structured-content-based

Relational semantics

Ranking paradigms

- No explicit notion of relevance: similarity between the query and the document model
  - Vector space model (cosine similarity)

$$Sim(q,d) = Cos((w_{1,d},\ldots,w_{t,d}),(w_{1,q},\ldots,w_{k,q}))$$

  - Language models (KL divergence)

$$Sim(q,d) = -KL(\theta_q \,\|\theta_d) = -\sum_{t\in V} P(t\mid\theta_q)\log\left(\frac{P(t\mid\theta_q)}{P(t\mid\theta_d)}\right.$$

- Explicit relevance model
  - Foundation: **probability ranking principle**
  - Ranking results by the posterior probability (odds) of being observed in the relevant class:

$$P(D\mid R) = \prod_{t\in D} P(t\mid R)\prod_{t\notin D}(1-P(t\mid N))$$

Features

- Features are orthogonal to retrieval models
  - Weights for query / document vectors?
  - Language models for document / queries?
  - Relevance models?
  - What to use for learning to rank?

Features
Dealing with ambiguities

- Content features
  - **Co-occurrences**
    - Terms K that often co-occur form a contextual interpretation, i.e. topics (cluster hypothesis, distributional semantics)
    - “Berlin” and “apartment” → geographic context → Berlin
  - **Frequencies**: *d* more likely to be “about” a query term *k* when *d* more often, mentions *k* (probabilistic IR)
- Structure features
  - **Structured-content**-based: consider relevance at fine-grained level of attributes
  - **Link**-based popularity
  - **Proximity**-based
- **Semantics** captured in conceptual and linguistic models?
  - Only exploited for matching to generate candidates so far

Term ambiguity

Content ambiguity

Structure ambiguity

Relational semantics

Content-based features – frequency

- Document statistics, e.g.
  - Term frequency
  - Document length
- Collection statistics, e.g.
  - Inverse document frequency

$$w_{t,d} = \frac{tf}{|d|} * idf$$

- Background language models

$$P(t \mid \theta_d) = \lambda \frac{tf}{|d|} + (1 - \lambda) P(t \mid C)$$

- An object is more likely about “Berlin”?
  - When it contains a relatively high number of mentions of the term “Berlin”
  - When number of mentions of term in the overall collection is relatively low

Structure-based features – links

How to incorporate it into a content-based retrieval model?

- PageRank
  - Link analysis algorithm
  - Measuring relative importance of nodes
  - Link counts as a vote of support
  - The PageRank of a node recursively depends on the number and PageRank of all nodes that link to it (incoming links)
- ObjectRank [Hristidis et al, TDS08]
  - Types and semantics of links vary in structured data
  - Authority transfer schema graph specifies connection strengths
  - Recursively compute authority transfer data graph

- An object (about “Berlin”) is more important?
  - When a relatively large number of objects are linked to it

Structure-based features – proximity

adopted from: [Chen et al, SIGMOD09]

- EASE, XRANK, BLINKS, etc.
- EASE [Li et al, SIGMOD08]
  - Proximity between a pair of keywords

How to incorporate it into a content-based retrieval model?

$$sim(k_i, k_j) = \frac{1}{|C_{k_i} \cup C_{k_j}|} \cdot \sum_{\substack{n_i \in C_{k_i} \\ n_j \in C_{k_j}}} \sum_{n_i \leftrightsquigarrow n_j} \frac{1}{(|n_i \leftrightsquigarrow n_j| + 1)^2}$$

  - Overall score of a JRT is aggregation on the score of keyword pairs
- XRANK [Guo et al, SIGMOD03]
  - Ranking of XML documents / elements
  - Proximity of n is defined based on w, the smallest text window in n that contains all search keywords

$$p(n, k1, k2, \ldots, k_l) = |w|$$

- A structured result (e.g. Steiner tree) is more relevant?
  - When it is more compact s.t. elements are closely related

Structured-content-based model

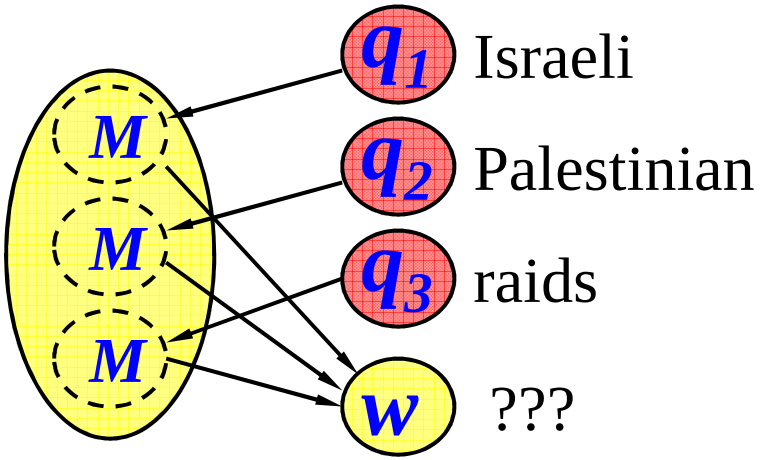
- Consider structure of objects during content-based modeling, i.e., to obtain structured content-based model
  - Content-based model for structured objects, structured documents, database tuples…

$$P(t \mid \theta_d) = \sum_{f \in F_d} \alpha_f P(t \mid \theta_f)$$

- An object is more likely about “Berlin”?
  - When its (important) **fields** / attributes contain a relatively high number of mentions of the term “Berlin”

Structured-content-based model

Relevance model [Lavrenko et al, SIGIR01]



sample probabilities

P(w|Q)	w
.077	palestinian
.055	israel
.034	jerusalem
.033	protest
.027	raid
.011	clash
.010	bank
.010	west
.010	troop
…	

$$P(w \mid R) \approx P(w \mid q_1 \ldots q_k) = \frac{P(w, q_1 \ldots q_k)}{P(q_1 \ldots q_k)}$$

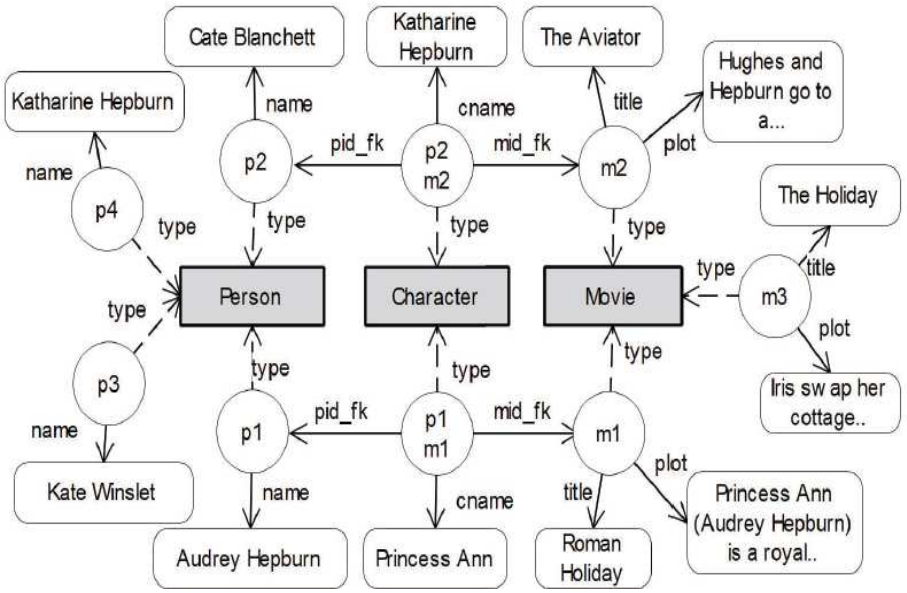
$$P(w, q_1 \ldots q_k) = \sum_{M \in UM} P(M) P(w \mid M) \prod_{i=1}^{k} P(q_i \mid M)$$

Structured-content-based model

Edge-specific relevance model [Bicer et al, CIKM11]

- Given a query Q={q1,…,qn}, a set of resources (FR) are retrieved
  - E.g. Q={Hepburn, Holiday}, FR = {m1, p1, p4,m2, p2m2,m3}
- Based on FR results, an edge specific RM_{FR} is constructed for each unique edge *e*:

$$RM^{e}_{F_R}(v) = \frac{\sum_{r \in F_R} P_{r \xrightarrow{e} a}(v \mid a) \prod_{i=1}^{m} P_{r \xrightarrow{*} a}(q_i \mid a)}{\sum_{r' \in F_R} \prod_{i=1}^{m} P_{r' \xrightarrow{*} a}(q_i \mid a)}$$



words	RM_q^{name}	$RM_q^{character}$	RM_q^{title}	RM_q^{plot}
hepburn	0.30	0.17	0.02	0.11
holiday	0.02	0.03	0.5	0.1
audrey	0.14	0.02	0.01	0.06
katharine	0.12	0.12	0.02	0.03
princess	0.01	0.01	0.02	0.1
roman	0.01	0.01	0.26	0.03
...	...	...	...	...

Structured-content-based model

Edge-specific resource model

- Edge-specific resource model:

$$RM_r^e(v) = (1-\lambda_r)P_{r\xrightarrow{e}a}(v \mid a) + \lambda_r P_{r\xrightarrow{*}a}(v \mid a)$$

  - Smoothing with model for the entire resource
- Use RM for query expansion: the score of a resource calculated based on cross-entropy of edge-specific RM_{FR} and edge-specific RM_r:

$$Score(r) = \sum_{e\in E}\alpha_e \sum_{v\in V} RM_{F_R}^e(v)\ \log RM_r^e(v)$$

  - Alpha allows to control the importance of edges

Ranking Steiner tree / join result tuples (JRT)

- Ranking aggregated JRTs:
  - The cross entropy between the edge-specific RM_{FR} (query model) and geometric mean of **combined** edge-specific RM_{JRT}:

$$RM^e_{JRT}(v) = \sqrt[m]{RM^e_{r_1}(v) \ldots RM^e_{r_m}(v)}.$$

$$Score(JRT) = \sum_{e \in E} \alpha_e \sum_v RM^e_{F_R}(v) \log \sqrt[m]{RM^e_{r_1}(v) \ldots RM^e_{r_m}(v)}$$

- The proposed ranking function is monotonic with respect to the individual resource scores (a necessary property for using top-k keyword search algorithms)

Taxonomy of ranking approaches

- Explicitly vs. non-explicitly relevance-based
- Different approaches for model construction
  - Content-based ranking
  - **Structure-based ranking**
  - **Structured-content-based ranking**

Relational semantics in the data

Conclusions

Conclusions

- Semantic search is about using semantics in **structured data**, **conceptual** and **linguistic models**
  - Interpreting term and content**,** inferring relationships
  - Deal with ambiguities at term, content and structure level
  - Mainly used during matching to generate candidates
  - Semantics in structured data can improve ranking
- Keyword search on structure data as semantic search
  - Support **complex information needs** (long tail)
  - Exploit relational semantics to interpret keywords
  - Complexity requires specialized indexes and efficient exploration algorithms

…Selected challenges

- Conceptual, structured data model of text
  - Large-scale **knowledge extraction** / **linking**
  - New models for interacting with and maintaining hybrid content
- **Hybrid content management**
  - Indexing hybrid content
  - Processing hybrid queries
  - **Ranking hybrid results** (facts combined with text)
- **Querying paradigm** for complex retrieval tasks
  - Keywords?
  - Keywords + facets?
- Rich retrieval process: from querying to browsing to intuitive **presentation**, supporting complex **analysis** of data / results

References

- Agrawal, S., Chaudhuri, S., and Das, G. (2002). DBXplorer: A system for keyword-based search over relational databases. In ICDE, pages 5-16.
- Amer-Yahia, S. and Shanmugasundaram, J. (2005). XML full-text search: Challenges and opportunities. In VLDB, page 1368.
- Bao, Z., Ling, T. W., Chen, B., and Lu, J. (2009). Effective xml keyword search with relevance oriented ranking. In ICDE, pages 517-528.
- Bhalotia, G., Nakhe, C., Hulgeri, A., Chakrabarti, S., and Sudarshan, S. (2002). Keyword Searching and Browsing in Databases using BANKS. In ICDE, pages 431-440.
- Bicer, V., Tran, T. (2011): Ranking Support for Keyword Search on Structured Data using Relevance Models. In CIKM.
- Bizer, G., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S. (2009): DBpedia - A crystallization point for the Web of Data. J. Web Sem. (WS) 7(3):154-165
- Chen, Y., Wang, W., Liu, Z., Lin, X.(2009): Keyword search on structured and semi-structured data. SIGMOD 2009:1005-1010
- Ding, B., Yu, J. X., Wang, S., Qin, L., Zhang, X., and Lin, X. (2007). Finding top-k min-cost connected trees in databases. In ICDE, pages 836-845.
- Golenberg, K., Kimelfeld, B., and Sagiv, Y. (2008). Keyword proximity search in complex data graphs. In SIGMOD, pages 927-940.
- Guo, L., Shao, F., Botev, C., and Shanmugasundaram, J. (2003). XRANK: Ranked keyword search over XML documents. In SIGMOD.
- He, H., Wang, H., Yang, J., and Yu, P. S. (2007). BLINKS: Ranked keyword searches on graphs. In SIGMOD, pages 305-316.
- Hristidis, V., Hwang, H., and Papakonstantinou, Y. (2008). Authority-based keyword search in databases. ACM Trans. Database Syst., 33(1):1-40

- Hristidis, V. and Papakonstantinou, Y. (2002). Discover: Keyword search in relational databases. In VLDB.
- Kacholia, V., Pandit, S., Chakrabarti, S., Sudarshan, S., Desai, R., and Karambelkar, H. (2005). Bidirectional expansion for keyword search on graph databases. In VLDB, pages 505-516.
- Kimelfeld, B. and Sagiv, Y. (2006). Finding and approximating top-k answers in keyword proximity search. In PODS, pages 173-182.
- Ladwig, G., Tran, T. (2011): Index Structures and Top-k Join Algorithms for Native Keyword Search Databases. In CIKM.
- Lavrenko, V. Croft, W.B. (2001): Relevance-Based Language Models. In SIGIR, pages 120-127.
- Li, G., Ooi, B. C., Feng, J., Wang, J., and Zhou, L. (2008). EASE: an effective 3-in-1 keyword search method for unstructured, semi-structured and structured data. In SIGMOD.
- Liu, F., Yu, C., Meng, W., and Chowdhury, A. (2006). Effective keyword search in relational databases. In SIGMOD, pages 563-574.
- Luo, Y., Lin, X., Wang, W., and Zhou, X. (2007). SPARK: Top-k keyword query in relational databases. In SIGMOD, pages 115-126.
- Qin, L., Yu J. X., Chang, L. (2009) Keyword search in databases: the power of RDBMS. In SIGMOD, pages 681-694.
- Tran, T., Herzig, D., Ladwig, G. (2011): SemSearchPro: Using Semantics throughout the Search Process. In Journal of Web Semantics, 2011.
- Tran, T., Wang, H., Rudolph, S., Cimiano, P. (2009): Top-k Exploration of Query Graph Candidates for Efficient Keyword Search on RDF. In ICDE.
- Vagelis Hristidis, L. G. and Papakonstantinou, Y. (2003). Efficient ir-style keyword search over relational databases. In VLDB.

Thanks!

Tran Duc Thanh
ducthanh.tran@kit.edu
http://sites.google.com/site/kimducthanh/