MING: Mining Informative Entity-relationship Subgraphs

Gjergji Kasneci, Shady Elbassuoni,
Gerhard Weikum

MPI–I–2009–5-007 July 2009

Authors' Addresses

Gjergji Kasneci
Max-Planck Institute for Informatics
Campus E1.4
66123 Saarbrücken
Germany

Shady Elbassuoni
Max-Planck Institute for Informatics
Campus E1.4
66123 Saarbrücken
Germany

Gerhard Weikum
Max-Planck Institute for Informatics
Campus E1.4
66123 Saarbrücken
Germany

Keywords

Entity-relationship graphs, subgraph mining, informative subgraphs

Contents

1 Overview

"All men by nature desire knowledge."

ARISTOTLE

Many modern applications are faced with the task of knowledge discovery in large entity-relationship (ER) graphs, such as domain-specific knowledge bases or social networks. An important building block of many knowledge discovery tasks is that of finding the "closest" relationships between $k \geq 2$ given entities. We have investigated this kind of knowledge discovery task in our previous work [22]. A more general knowledge discovery scenario on ER graphs is that of mining the most "informative" subgraph for $k(\geq 2)$ given entities of interest (i.e. query entities). Intuitively, this would be the subgraph that best explains the relations between the k given query entities. This knowledge discovery scenario is more general than the one of [22] in that its focus is on whole subgraphs (and not on trees). Furthermore, in this scenario, the semantics plays a crucial role. Our notion of informativeness has a rather intuitive flavor. Hence, we are interested in measures that capture the human intuition of an informative ER subgraph. An adequate measure should favor insightful and salient relationships between the query entities. Examples for such knowledge discovery tasks are queries that aim at finding the relations between k given biomedical entities, the connections between k criminals, the most relevant data shared by k Web 2.0 users, etc.

In this work, we addresses this problem of mining most informative ER subgraphs. We define a framework for computing a new notion of informativeness of nodes. This is used for defining the informativeness of entire subgraphs. We present MING (for Mining INformative Graphs), a principled and efficient method for extracting the most informative subgraph for $k(\geq 2)$ given query entities. The viability of our approach is demonstrated through experiments on real-life datasets, with comparisons to prior work.

1.1 Motivation and Problem Statement

MOTIVATION ER graphs are abundant in the field of knowledge representation. They come in different flavors and formats (i.e. represented through relational models, XML with XLinks, or RDF triples) and cover various knowledge domains. Examples of ER graphs are GeneOntology [2] or UMLS [3] (in the biomedical domain), SUMO [28], OpenCyc [1], WordNet [4, 13], YAGO [31, 32, 33] (in the domain of general purpose knowledge bases), the ER graphs represented by IMDB (in the domain of movies and actors) and DBLP (in the domain of Computer Science publications), and many more.

Applications exploiting ER graphs are often faced with knowledge discovery tasks. Frequent scenarios here are those that aim to find the most meaningful relations between $k(\geq 2)$ entities of interest. From a graph-theoretic point of view, the goal in such scenarios would be to determine the subgraph that best explains the relations between the k entities of interest. We will interchangeably refer to these entities as *query nodes* or *query entities*. A related knowledge discovery task, namely that of finding the "closest" connections between $k(\geq 2)$ query entities, has been investigated in our previous work [22]. In contrast to [22], where the focus was on subtrees that closely interconnect the given query entities, the task considered in this work aims at finding whole subgraphs that best capture the relations between $k(\geq 2)$ query entities. Corresponding queries could ask for the relations between k given biomedical entities, the connections between k criminals, the most relevant data shared by k Web 2.0 users, etc. For large ER graphs, these queries become challenging from an algorithmic as well as from a semantics viewpoint. The answer graphs should be computed efficiently, and they should be insightful by exhibiting salient facts. This challenge calls on one hand for adequate goodness measures for capturing the semantic relatedness between the query entities, and for robust and efficient solutions on the other hand.

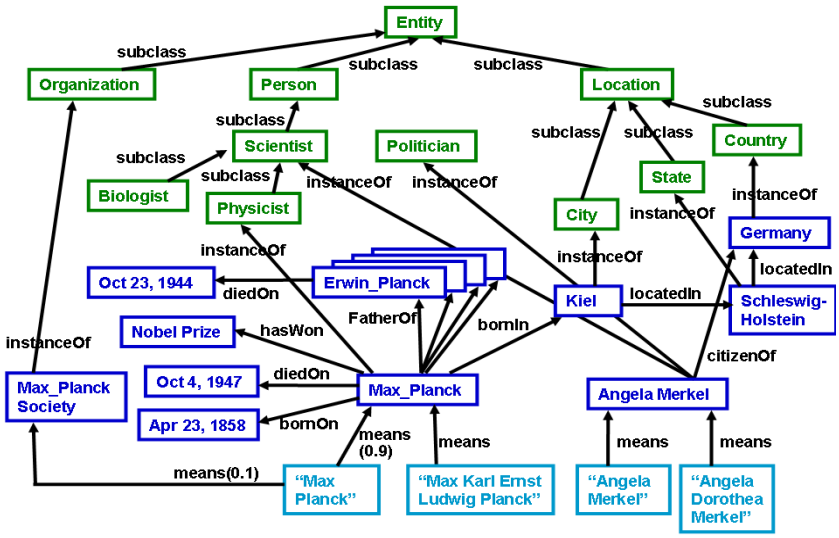

Figure 1: Sample subgraph from YAGO

PROBLEM STATEMENT Formally, the general problem that motivates this work can be stated as follows: given a set $Q = \{q_1, ..., q_k\}, k \geq 2$, of nodes of interest (i.e. query nodes) from an ER graph G and an integer $b > k$ (representing a node budget), find a connected subgraph S of G with at most b nodes that contains all query nodes and maximizes an "informativeness" function $g_{info}(S, Q)$. Intuitively, for the given node budget b, this would be the subgraph that best explains the relations between the entities represented by the query nodes, in other words, this would be the most informative subgraph. The above problem comes with two subproblems:

1. What is a good measure for representing the informativeness of relations between entities in ER graphs?

2. How to determine the most informative subgraph for the given query nodes efficiently?

Consider an ER graph about prominent persons with rich information about their careers, nationalities, interests, their birth and death dates, their prizes, etc. Note that the YAGO knowledge base is an example of an ER graph with such information about prominent persons. Figure

1 depicts an excerpt from the YAGO graph. Consider the query that asks for the relation between *Max_Planck*, *Albert_Einstein*, and *Niels_Bohr*. An informative subgraph that captures their relatedness should reveal that all three of them are physicists, scientists, Nobel Prize winners, etc, and should discourage long or obscure connections (e.g. connections through persons with same nationalities or same birth or death places as some of the query entities). Figure 2 depicts a possible answer.

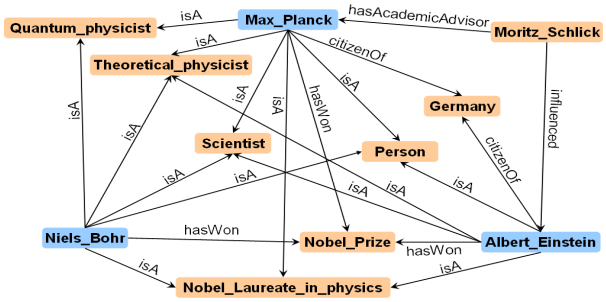

Figure 2: Answer graph returned by MING on YAGO

PROBLEMS WITH PREVIOUS APPROCAHES In previous approaches [9, 12, 14, 26, 29, 34], the notion of subgraph importance is mainly based on structural properties of the underlying graph (e.g. indegree or outdegree of a node, density or edge connectivity[1] of a subgraph, etc.). More related to our approach are techniques based on influence propagation like [12] or [34]. The approach of [12] exploits a current-flow-based algorithm and comes with an efficient two-phase solution for dealing with disk-resident graphs, but it is restricted to two query nodes. The approach of [34], CEPS, can handle more than two query nodes, and gives a random-walk-based solution for retrieving the most "central" nodes, so called *centerpieces*[2], with respect to the query nodes, but cannot be applied to disk-resident graphs in a straight-forward manner. In addition, all mentioned approaches leave aside the problem of deriving measures for capturing the semantic importance of nodes and edges in ER graphs. Other, Steiner-tree-based, approaches [5, 6, 11, 18, 19, 20, 21, 22] have addressed the problem of retrieving the top-k minimum-cost subtrees that closely interconnect the given query nodes. Their result paradigm is tree-based. Hence, these approaches are not directly applicable to our problem of retrieving well-connected, informative subgraphs. The top-k result trees can be combined into a single subgraph that interconnects the query nodes, but again, the underlying cost models are rather driven by structural properties than by the semantic importance of nodes and edges. In fact, the cost models are often modified for the sake of efficiency (see for example [18]).

In contrast, our approach gives an efficient solution for large, disk-resident ER graphs, while making the semantic aspect of entities and relationships in ER graphs a key ingredient for the measure of informativeness.

1.2 Related Work

There are various approaches which aim at identifying important subgraphs by applying structural analysis, e.g. by identifying strongly connected, dense or frequent subgraphs [14, 15, 16, 26, 27, 29], by emulating random walks, electrical circuits or other influence propagation techniques [9, 12, 23, 34], by applying graph clustering and partitioning [7, 10, 36, 37],

[1] Size of the minimum cut in a graph.

[2] Term introduced in [34] to describe intermediate nodes that are closely connected to most of the query nodes.

by computing Steiner trees [5, 6, 11, 18, 19, 20, 21, 22], etc. To our surprise, the goodness measures for subgraphs in all these approaches are guided by two main aspects: frequency of subgraph patterns, or structural properties of subgraphs. However, for ER graphs, this is not sufficient, since (1) these graphs are usually free of redundancy, which attenuates the frequency aspect, and (2) they represent only a "biased" subset of the real world, which attenuates the structural aspect. For example, an RDF database may contain a lot of facts about a special entity X just because these facts were easy to extract. This does not mean that X is in general more important than entities for which there are fewer facts in the database. This has also been observed by Ramakrishnan et. al [30] who introduce a goodness measure that goes beyond the mere structure- or frequency-based importance. However, they too, infer this new measure directly from the ER graph. We strongly believe that a goodness measure for ER subgraphs should exploit the information redundancy of the domain from which the ER graph was derived. In the following, we discuss some related approaches by focusing on the main characteristics of their goodness measures.

STEINER TREE DETECTION In contrast to the general graph-based result paradigm of the work presented in this report, the result paradigm in this area is tree-based. The goal is to find subtrees of the underlying graph that closely interconnect the given query nodes. BANKS I [6] and BANKS II [21] use single-source-shortest-path iterators which start from the query nodes and follow the directed edges of the graph backwards (BANKS I), or backwards and forwards (BANKS II). A result tree is produced as soon as the iterators meet. The goodness measures for their result trees are based on indegrees and outdegrees of nodes as well as on edge weights. BLINKS [18] retrieves result trees efficiently by means of subgraph partitioning and indexing. It builds on the BANKS heuristics and uses a cost model that allows the combination of subresults that were computed on different partitions. Finally, the goodness measures of STAR [22] and DPBF [11] merely build on edge weights. While STAR uses a local search strategy in combination with different search space exploration heuristics, DPBF exploits a dynamic programming strategy.

COMMUNITY DETECTION In most of the community detection approaches, the goodness measures for subgraphs build on structural properties. Gibson et al. [14] address the emergence of communities in the Web graph. They exploit the HITS algorithm [24, 25] to determine the *top-k* hubs and authorities for a given topic. Usually, these hubs and authorities form a structurally dense and topic-specific core. Kumar et al. [26] exploit the hypothesized correspondence between communities and dense bipartite subgraphs to detect communities. Their algorithm is a two-step process – a careful enumeration and removal of small-sized bipartite cliques, followed by an apriori-style enumeration algorithm on the residual, hopefully smaller, graph. [15] presents a recursive shingle-based algorithm[3] which seeks clusters of similar Web pages that tend to link to the same destinations. Apart from detecting patterns of dense subgraphs, the algorithm can also recursively detect similarities between such subgraphs.

GRAPH CLUSTERING AND PARTITIONING [37] exploits edge connectivity to mine *closed subgraphs*[4] in a set of ER graphs. Efficient methods for identifying corresponding patterns are presented. SkyGraph [29] addresses the problem of discovering the most important ER subgraphs, where the importance of a graph is determined by its order (i.e. the number of nodes) and its edge connectivity. SkyGraph uses successive applications of the Min-Cut algorithm [17]

[3] Text mining method for estimating the similarity between Web pages by examining their feature overlap

[4] A graph is closed if and only if there is no supergraph that has the same support (i.e. frequency).

starting with the original graph and proceeding with all produced subgraphs. Finally, a notion of subgraph domination, introduced by the authors, leads to the most important ER subgraphs.

INFLUENCE PROPAGATION More related to our approach are techniques that build on influence propagation. For a given ER graph, HubRank [9] precomputes and indexes random walk fingerprints for a small fraction of nodes, carefully chosen using query log statistics. At query time, the nodes with indexed fingerprints are exploited to compute approximate personalized PageRank vectors for a query relevant subgraph. In [12], Faloutsos et al. present an approach that emulates electrical circuits to retrieve the subgraph that best captures the relationship between two given entity nodes. The approach proceeds by determining a well-connected candidate subgraph C that contains the two query nodes. By applying +1 voltage on one query node, the method determines (based on a current flow measure) the subgraph S of C that contains the "best" interconnections between the query nodes. The approach is generalized in [34] by a method coined CEPS, which can be applied to any number of query nodes. The problem addressed there is that of finding *centerpieces*, i.e intermediate nodes that are closely connected to most of the nodes from a node set Q of query nodes. Based on random walks with restart from each of the query nodes, the k most central nodes with respect to Q are retrieved. The method is extended to extract a connected subgraph, which, as reported, captures the intuition about important relations between the nodes of Q. However, CEPS is not applicable to disk-resident graphs in a straight-forward way.

In [30] the authors address the same problem as [12]. A current-flow-based algorithm for subgraph generation is combined with different heuristics for capturing the specificity and the selectivity of relations and entities (e.g., the entity *Theoretical_Physicist* is more specific than *Physicist*, accordingly a fact of the form (*Person*, *livesIn*, *City*) is less selective than a fact of the form (*Person*, *isMayorOf*, *City*)). However, all measures behind these heuristics are directly inferred from the graph at hand. We argue that in practice, this is not sufficient, since ER graphs represent only a "biased" subset of the real world.

1.3 Contributions and Outline

This work addresses the problem of finding subgraphs that best explain the relations between $k(\geq 2)$ query nodes in ER graph structures. We compute the most informative subgraphs in a two-phase approach, coined MING (for Mining INformative Graphs), that can efficiently deal with disk-resident ER graphs. In its first phase MING extracts a well-connected candidate subgraph that contains most of the important connections between the query nodes. In the second phase MING uses a random-walk-based learning method to determine the most informative answer graph. Our main contributions are the following:

- We give a clean notion of informativeness for nodes in ER graphs. Our informativeness measure builds on a natural extension of the random surfer model that underlies PageRank [8]. This measure is exploited to capture the informativeness of entire ER subgraphs.
- We present MING, a robust and efficient method for mining and extracting most informative subgraphs that best capture the relations between $k(\geq 2)$ query entities.
- We demonstrate the viability of our approach in an extensive evaluation on real-life datasets, based on user assessments and in comparison with state-of-the-art extraction techniques for ER graphs.

The remainder of the report is organized as follows. Section 2 introduces the notion of informativeness for ER graphs. Section 3 is dedicated to our subgraph mining and extraction algorithms. We present the experimental evaluation of our approach in Section 4, and conclude in Section 5.

2 ER-based Informativeness

As a preparation for our subgraph extraction method, in this section, we will introduce IRank, a measure for the informativeness of nodes in ER graphs. From a graph-theoretic point of view an ER graph is a labeled multigraph. In this section, for ease of presentation, we will stick to the following simpler definition of ER graphs.

DEFINITION 1: [*ER graph*]
Let Ent be a finite set of entity names and Rel be a finite set of relationship labels such that $Ent \cap Rel = \emptyset$. An entity-relationship graph *over Ent and Rel is a set of edges $G \subseteq Ent \times Rel \times Ent$.*

We denote by $Ent(G)$ the entities of G, i.e. the nodes of G, and by $Rel(G)$ the relations of G. Furthermore, we refer by facts to the edges of G. In Figure 2, the edge (*Max_Planck*, *citizenOf*, *Germany*) represents a fact about the entities *Max_Planck* and *Germany*.

Since the direction of a relationship between two entities can always be interpreted in the converse direction, we view the edges of an ER graph as bidirectional. That is, we assume that for each edge $(u, r, v) \in G$ there is an edge $(v, r^-, u) \in G$, where r^- represents the inverse relation label of r.

We believe that in order to compute the informativeness of a node in an ER graph, the link structure has to be taken into account. On the other hand, we are aware of the fact that the edges of an ER graph do not always entail a "clear" endorsement. Consequently, measures that build on the link-based endorsement hypotheses such as PageRank [8] or HITS [24, 25] are not always applicable to ER graphs in a straight forward manner. For example, Consider an RDF database about scientists that contains for each scientist only the name, the date of birth, and the profession. Suppose that the facts (*Albert_Einstein*, *instanceOf*, *Physicist*) and (*Bob_Unknown*, *instanceOf*, *Physicist*) are contained in this database. Now, consider the respective edges in the corresponding ER graph. Since the link structure of scientist nodes in this ER graph is determined by their schema, both *Albert_Einstein* and *Bob_Unknown* will have the same link structure. Consequently, in this example, they will be endorsed equally by the link structure. Furthermore, the direction of an edge in an ER graph merely corresponds to the relationship label of that edge. Analogously, the fact (*Albert_Einstein*, *instanceOf*, *Physicist*) could be represented as (*Physicist*, *hasInstance*, *Albert_Einstein*). Hence, edge directions in an ER graph do not always reflect a "clear" endorsement.

Our informativeness measure for nodes overcomes these problems by defining a special kind of endorsement that is based on co-occurrence statistics for entities and relationships. These statistics will guide a random walk process on the adjacency matrix of the ER graph. We show in the next subsection how to compute them from the domain from which the underlying ER graph was derived.

2.1 Statistics-based Edge Informativeness

For each fact represented by an edge, we compute two weights; one for each direction of the edge (note that we view edges as bidirectional). Each of these weights will represent a special kind of endorsement, derived from domain-based co-occurrence statistics for entities and relationships.

DEFINITION 2: [*Fact Pattern, Match, Binding*]
Let X be a set of entity variables (placeholders for entities). A fact pattern *from an ER graph $G \subseteq Ent \times Rel \times Ent$ is a triple $(\alpha, \beta, \gamma) \in (Ent \cup X) \times Rel \times (Ent \cup X)$ in which either $\alpha \in X$ or $\gamma \in X$, such that if $\alpha \in X$ then there is an edge (α', β, γ) in G, and if $\gamma \in X$ then there is an edge (α, β, γ') in G.*

Without loss of generality, let $\alpha \in X$. The edge (α', β, γ) from G is called a match *to the fact pattern (α, β, γ), and the entity α' is called a* binding *to the variable α.*

Consider the fact pattern (*x*, *instanceOf*, *Physicist*), $x \in X$. The fact (*Max_Planck*, *instanceOf*, *Physicist*) is a match to this pattern. In general, there may be multiple matches to a fact pattern. For example, the facts (*Albert_Einstein*, *instanceOf*, *Physicist*) and (*Bob_Unknown*, *instanceOf*, *Physicist*) could be further matches to the above fact pattern. However, not all matches are equally informative. In our example, the fact (*Albert_Einstein*, *instanceOf*, *Physicist*) should have a higher informativeness than (*Bob_Unknown*, *instanceOf*, *Physicist*). More precisely, the binding *Albert_Einstein* should be more informative than *Bob_Unknown*. To capture this notion of informativeness, we introduce a probabilistic model.

Let (α, β, γ) be a fact pattern, where $\alpha \in X$. Let α' be a binding of α. We estimate the informativeness of α' given the relationship β and the entity γ as:

$$P_{info}(\alpha'|\beta,\gamma) = \frac{P(\alpha',\beta,\gamma)}{P(\beta,\gamma)} \approx \frac{W(\alpha',\beta,\gamma)}{W(\beta,\gamma)} \tag{2.1}$$

where $W(\alpha', \beta, \gamma)$ denotes the number of domain witnesses for the fact (α', β, γ), i.e. the number of occurrences of the fact (α', β, γ) in the underlying domain of the ER graph. Analogously, $W(\beta, \gamma)$ stands for the number of witnesses for the pattern $(*, \beta, \gamma)$, where the wild card '$*$' can be any entity. Note that implementation-wise it may be very difficult to identify all occurrences of a fact, especially because relationships can be expressed in elaborate ways. In this case, $P_{info}(\alpha'|\beta, \gamma)$ can be estimated in a more relaxed way as the fraction of documents that contain α' and γ out of the documents that contain γ. We give an example of such an estimation in Section 4. Finally, $P_{info}(\alpha'|\beta, \gamma)$ can be assigned as a weight to the edge $\gamma \xrightarrow{\beta} \alpha'$.

To see why this formulation captures the intuitive understanding of informativeness for facts, consider the following examples. Let p =(*Albert_Einstein*, *instanceOf*, *x*) be a fact pattern, where *x*$\in X$. Let (*Albert_Einstein*, *instanceOf*, *Physicist*) and (*Albert_Einstein*, *instanceOf*, *Philosopher*) be two respective matches. Here, the statistics-based informativeness measures how often Einstein is mentioned as a physicist as compared to how often he is mentioned as a philosopher. Assuming that the underlying ER graph represents a subset of the Web knowledge (i.e. the domain is given by the Web content), (*Albert_Einstein*, *instanceOf*, *Physicist*) is more informative than (*Albert_Einstein*, *instanceOf*, *Philosopher*), since there are more Web pages about Einstein as physicist. In this case, the statistics-based informativeness measures the degree to which Einstein is a physicist (or a philosopher, respectively).

Now consider the fact pattern p =(x, *instanceOf*, *Physicist*) and the matches (*Albert_Einstein*, *instanceOf*, *Physicist*) and (*Bob_Unknown*, *instanceOf*, *Physicist*). In this case, the statistics-based informativeness will compute how often Einstein is mentioned as a physicist as compared to how often *Bob_Unknown* is mentioned as a physicist. Since Einstein is an important individual among the physicists, (*Albert_Einstein*, *instanceOf*, *Physicist*) will have a higher informativeness than (*Bob_Unknown*, *instanceOf*, *Physicist*). Hence, in this case, informativeness measures the importance of Einstein in the world of physicists.

2.2 IRank

Our aim is an informativeness measure for nodes based on random walks on the – now weighted – ER graph. Our measure, coined *IRank* (for Informativeness Rank), is related to PageRank.

PageRank [8] computes the authority of Web pages based on the link structure of the Web. In the PageRank model a random surfer walks through a directed Web graph $G(V, E)$ with a set of nodes V and a set of edges $E \subseteq V \times V$. At any node $v \in V$, the surfer may continue the walk by following an outgoing edge from v with a probability inversely proportional to the out-degree of v. Alternatively, the surfer may decide to restart the walk by jumping to any random node with a probability inversely proportional to the number of nodes. Finally, the probability that the random surfer is at a node v is given by:

$$PR(v) = \frac{(1-q)}{|V|} + q \sum_{v' \to v} \frac{PR(v')}{O(v')} \tag{2.2}$$

where $O(v')$ stands for the number of the outgoing edges of v', and q is a damping factor that is usually set to 0.85.

The PageRank model is based on the hypothesis that every ingoing link of a Web page represents an endorsement of that Web page. However, in ER graphs, the link-based endorsement hypothesis does not always hold, and consequently methods like PageRank are not directly applicable.

Let $G \subseteq Ent \times Rel \times Ent$ be an ER graph. Let $u \in Ent$ be an entity and let $P(u)$ be the probability of encountering the entity u in the domain from which G was derived. This value can be estimated as:

$$P(u) \approx \frac{W(u)}{\sum_{v \in Ent} W(v)} \tag{2.3}$$

where again $W(u)$ denotes the number of occurrences of the entity u in the underlying domain. $P(u)$ can be viewed as an importance prior for entity nodes.

In IRank, the random surfer may decide to restart his walk from an entity $u \in Ent(G)$ with probability proportional to $P(u)$. Alternatively, the surfer may reach u from any neighboring entity v that occurs in an edge of the form $(v, r, u) \in G$ (given that the surfer is at one of these neighboring entities of u).

Let $N(u)$ denote the set of neighboring entities of u in G. The probability of reaching u via one of its neighbors would be proportional to:

$$\sum_{v \in N(u)} \sum_{\substack{r \\ (v,r,u) \in G}} P_{info}(u|r, v) \cdot IR(v) \tag{2.4}$$

where $IR(v)$ denotes the probability that the surfer is at node v, and $P_{info}(u|r, v)$ is defined as in Equation (6.1).

Finally, the accumulated informativeness at a node $u \in V$ is given by:

$$IR(u) = (1-q)P(u) + q \sum_{v \in N(u)} \sum_{\substack{r \\ (v,r,u) \in G}} P_{info}(u|r,v) \cdot IR(v) \tag{2.5}$$

For practical reasons, the outgoing edge weights (i.e. the p_{info} weights) for each entity u can be normalized by the sum of all outgoing edge weights. With this normalization step, Equation (4) represents an aperiodic and irreducible finite-state (i.e. an ergodic) Markov Chain. This guarantees the convergence and the stability of IRank. Although IRank is related to PageRank, the P_{info} values are crucial and make a big difference in the random walk process. In the next section we will see that the definition of informativeness, as given by IRank (i.e. Equation (6.5)), can be extended to capture the informativeness of subgraphs that contain $k(\geq 2)$ nodes of interest from an ER graph G.

2.3 Most Informative Subgraphs

In this section we give an overview of our approach for estimating the informativeness of connected ER subgraphs that contain $k(\geq 2)$ entities of interest. For ease of explanation, in the remainder of this work, we will use the graph-theoretic definition of connected, labeled subgraphs to refer to connected ER subgraphs.

DEFINITION 3: [*Connected ER Subgraph*]
Let $G \subseteq Ent \times Rel \times Ent$ be an ER graph. We define a connected subgraph *S of G as a graph $S = (V, E, l_V, l_E)$, where V and $E \subseteq V \times V$ are sets of nodes and edges respectively, $l_V : V \rightarrow Ent$ and $l_E : E \rightarrow 2^{Rel}$ are injective functions such that for each edge $e_{(u,v)} \in E$ with source node u and target node v and for each $r \in l_E(e_{(u,v)})$ there is an edge $(l_V(u), r, l_V(v))$ in G, and for each node $u \in V$ there is a node $v \in V$ with $(u,v) \in E$ or $(v,u) \in E$.*

In the following, for any subgraph S of an ER graph G, we will denote its node set by $V(S)$ and its edge set by $E(S)$. We say a subgraph S contains an entity $q \in Ent$ if there is a node $v \in V(S)$ with $l_V(v) = q$.

Formally, the general problem that motivates this work is the following.

DEFINITION 4: [*General Problem Definition*]
Given: *an ER graph G, a set $Q = \{q_1, ..., q_k\}, k \geq 2$ of query entities, and an integer node budget $b > k$.*
Task: *find a* connected subgraph *S of G with at most b nodes that contains all entities from Q and maximizes an informativeness function $g_{info}(S, Q)$.*

Intuitively, $g_{info}(S, Q)$ represents a local goodness function that increases in regions of G which contain facts that nicely capture the relations between the query entities, and decreases in regions whose facts do not contribute to the relatedness between the query entities. Given this purely intuitive nature of g_{info}, it is inherently hard to define corresponding functions. In fact, as we will see later, our approach aims to approximate an implicit g_{info} by exploiting Equation (6.5), in order to learn the most informative subgraph.

RECAPITULATION OF PREVIOUS APPROACHES A simpler version of the problem, namely

for two query nodes, was first introduced in [12]. The authors present an approach that emulates electrical circuits to retrieve the subgraph that best captures the relationship between two given entities. The approach proceeds by determining a well-connected candidate subgraph C of G that contains most of the important interconnections between the nodes from Q. Then a current-flow-based method determines the subgraph S of C that "best" connects the query nodes.

CEPS [34] allows any number of query nodes, and addresses the problem of finding *center-pieces*, i.e intermediate nodes that are closely connected to most of the query nodes. Random surfers exercising random walks with restart from each query node help determining a subgraph S of G that captures the main relations between the query nodes. While [12] can efficiently deal with disk-resident graphs, CEPS is not directly applicable to them.

OUR APPROACH AT A GLANCE Following the strategy of [12], our approach, too, proceeds by generating a well-connected candidate subgraph C that contains all the query entities and most of the important interconnections between them. The focus in this generation phase is on recall rather than on precision; that is, during this generation phase, most of the spurious regions of the graph G are removed.

The next phase aims at mining the most informative connected subgraph S in the generated candidate graph C that contains all entities from Q. Deviating from the general problem statement of Definition 4, we view the mining task of this phase as a classification task. That is, we aim to classify the nodes of C into *informative* and *uninformative* nodes, with respect to the query entities. Such a learning approach has two main advantages: (1) it avoids the explicit definition of an informativeness function g_{info}, and (2) it avoids crude and non-transparent thresholding on edge and node scores in the extraction phase.

The main steps of our mining approach are the following:

1. As a first step, we apply the STAR algorithm from [22] to find a minimum-cost tree T in the generated candidate graph C that contains all entities from Q. In this step, the cost function for any subtree T of C that contains all query nodes is given by $\sum_{e \in E(T)} d(e)$, where $d(e)$ can be any distance function that is inversely proportional to the connection strength between the two end nodes of e. Apart from being very efficient, STAR comes with a nice approximation guarantee, and experiments on real-life data sets have shown that the trees it returns are minimal in the majority of the cases. Note that the tree T determined in this step already represents a "close" relation between the entities in Q.

2. In a second step, all nodes of T are labeled *informative*. All the nodes on the rim of the candidate graph C, that do not contribute to any path that interconnects query nodes are labeled *uninformative*. The main assumption in this step is that T already captures some relatedness between the query entities.

3. Finally, for each unlabeled node $v \in V(C)$ and for each label $l \in \mathcal{L} = \{$*informative, uninformative*$\}$ we estimate the probability $P_l(v)$ that a random walker who behaves according to Equation (4) starts at any node labeled l and ends up at any node labeled l by passing through v. The label of v is then determined by:

$$lab(v) = \arg\max_{l \in \mathcal{L}} P_l(v)$$

where $lab(v)$ denotes the label of v.

Finally, we extract a connected subgraph S of C that contains all the query entities and at most b nodes, and maximizes $\sum_{v \in V(S)} P_l(v)$, for $l =$*informative*.

In the following, we discuss the details of our approach.

3 The MING Approach

Our approach, MING, consists of two steps. Given an ER graph G and k query entities, in the first step, MING generates a well-connected candidate subgraph C that contains all the query entities and most of the important interconnections between them. The second step consists in learning the most informative connected subgraph S of C that interconnects the query entities.

3.1 Phase I: Candidate Subgraph Generation

Our generation algorithm for the candidate subgraph is related to the one presented in [12]. It proceeds by applying a series of expansions starting at each node representing a query entity $q_i \in Q$. The expansion process continues until a stopping condition is fulfilled. The stopping condition guarantees that the candidate subgraph is well-connected and that most of the important interconnections between the query nodes are retrieved. In contrast to the extraction algorithms from [12] and [34], which use a best-first expansion strategy (i.e., in each expansion step, the most promising node is expanded), we exploit a *balanced expansion heuristic*. That is, in each step we expand the most promising node from the expansion set with the lowest cardinality. As shown in [18], this heuristic performs very well in practice and has satisfactory bounds on the worst case performance. Furthermore, the expansions are guided by co-occurrence statistics for entity pairs derived from the underlying domain. That is, the edge weights for the expansion are given by P_{info}, see Equation (6.1), Section 2.

ALGORITHM 1: `candidateGeneration` (Q, G)
Input: ER graph G,
set Q of query entities
Output: well-connected subgraph C that contains all entities from Q
1 `Set` $Pe(q_i) = \{q_i\}$ `//for all` $q_i \in Q$
2 `Set` $Ex(q_i) = \{\}$ `//for all` $q_i \in Q$
3 `WHILE not stoppingCondition DO`
4 $q = \arg\min_{q_j \in Q} |Ex(q_j)|$
5 $v = \arg\max_{v \in Pe(q)} \sum_{u \in Ex(q)} P_{info}(u|v) + P_{info}(v|u)$
6 `expand` (v)
7 $Pe(q) = Pe(q) \backslash \{v\}$
8 $Ex(q) = Ex(q) \cup \{v\}$
9 $Pe(q) = Pe(q) \cup \{u | u \in N(v), u \notin Ex(q) \cup Pe(q)\}$
10 `END WHILE`
11 `RETURN connected subgraph` C `composed of the` $Ex(q_i)$

Let $D(q_i)$ be the set of nodes discovered through a series of expansions starting at q_i. Let $Ex(q_i)$ and $Pe(q_i)$ be the sets of *already expanded* and *pending* (i.e. not yet expanded) nodes within $D(q_i)$, respectively. As before, for an entity node v we denote by $N(v)$ the set of neighboring entity nodes of v in the ER graph G.

A high-level overview of our candidate generation method is given by Algorithm 1. Line 4 of Algorithm 1 shows that in each expansion step we choose the query node $q \in Q$ such that $|Ex(q)|$ has the lowest cardinality among all expanded node sets. This is part of the balanced expansion strategy. In line 5, we expand the pending node $v \in Pe(q)$ that is "best connected" to nodes from $Ex(q)$, with respect to our P_{info} measure, where $P_{info}(u|v)$ is defined as:

$$P_{info}(u|v) := \sum_{\substack{r \\ (u,r,v) \in G}} P_{info}(u|r, v) \tag{3.1}$$

and $P_{info}(u|r, v)$ is defined as in Equation (6.1). $P_{info}(v|u)$ is defined analogously.

Then, we move the expanded node v from $Pe(q)$ to $Ex(q)$ and update $Pe(q)$ with the neighbors of v that have not yet been seen (lines 7-9). In analogy to the algorithm in [12], the stopping condition puts limits on the number of nodes in the intersection $\bigcap_i Ex(q_i)$ of the expanded sets. Algorithm 1 generates a candidate subgraph in $O(|Q||Ent(G)|^2)$ steps. Note that the subgraph extracted in this phase contains only a few thousands of nodes and edges and can be easily processed in main memory.

3.2 Phase II: Learning Informative ER Subgraphs

Given the candidate subgraph C, we run the STAR algorithm [22] to determine a subtree T of C that closely interconnects all entities from Q. Assuming that T already captures some relatedness between the query entities, each node $v \in V(T)$ is viewed as informative, hence these nodes are labeled *informative*. Nodes on the rim of C that do not contribute to any connection between query entities (these are nodes of degree one that do not represent entities from Q, see example in Figure 3) are viewed as uninformative, consequently they are labeled *uninfomative*.

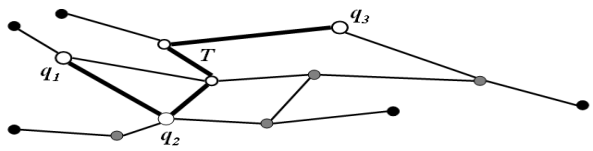

Figure 3: Sample candidate subgraph C with query nodes q_1, q_2, q_3.

In Figure 3, the bold edges represent the edges of T. The black nodes are labeled *uninformative*, the white nodes are labeled *informative*.

Let $\mathcal{L} = \{$*informative*, *uninformative*$\}$ be the set of possible class labels. Let $L \subseteq V(C)$ be the set of labeled nodes of C. For any node $v \in L$, let $lab(v)$ denote the label of v. In addition, for any node $v \in V(C) \setminus L$ and for any $l \in \mathcal{L}$, let $P_l(v)$ denote the probability of a random walker starting at any l-labeled node and ending up at any l-labeled node by passing through v. In this setting, our informative subgraph mining problem can be cast into a learning problem.

DEFINITION 5: [*Informative Subgraph Learning*]
Given: *the connected candidate subgraph C that contains all query nodes $q_1, ..., q_k \in Q, k \geq 2$, and an integer node budget $b >= |V(T)|$.*
Tasks:

1. *determine for each node $v \in V(C)$ a label $lab(v)$ such that*

$$lab(v) = \arg\max_{l \in \mathcal{L}} P_l(v) \tag{3.2}$$

2. *for $l =$ informative, extract a* connected subgraph *S of C with at most b nodes that contains all nodes of T and maximizes $\sum_{v \in V(S)} P_l(v)$.*

Apart from guaranteeing the connectedness of the final subgraph S, we will see in section 4 that T also helps constructing result graphs in which all query nodes are similarly well interconnected.

CLASSIFICATION ALGORITHM The intuition behind our classification method is the following. Consider all paths in C that connect any two nodes labeled l and cross an unlabeled node v. The higher the number of such paths, the higher the probability is that v is also labeled l. On the other hand, the longer these paths are, the smaller the probability is that v is labeled l. In order to estimate $P_l(v)$, we need methods that capture and reward robust structural connectivity and discourage long and loose connections.

We estimate the probability $P_l(v)$ (i.e. that v is labeled l) by the probability of a random walker starting at any node labeled l, passing through v, and ending up at any node labeled l. In fact, we view the probability $P_l(v)$ as composed of two probabilities $P_l^1(v)$ and $P_l^2(v)$, where $P_l^1(v)$ denotes the probability of reaching v when starting a random walk from any node labeled l, and $P_l^2(v)$ stands for the probability of reaching any node labeled l by a random walk that starts at v. Figure 4 depicts this composition. It is straightforward to see that $P_l(v) = P_l^1(v) \cdot P_l^2(v)$.

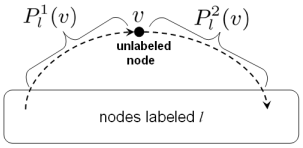

Figure 4: Probability P_l composed of the probabilities P_l^1 and P_l^2.

In a random walk process such as the one represented by IRank (see Equation (6.5)), long paths are not punished. In other words, the length of paths does not have any influence on the random walker. This is different in a random walk with restarts process. There, nodes that are far away from the starting nodes will be visited less frequently, because of the restart probability. Hence, in order to estimate the probability P_l^1 we modify IRank to a Random Walk with Restarts (RWR) process. The idea behind this RWR is that it starts at any l-labeled node v and visits its neighbors with a probability that is proportional to the corresponding edge weights (as edge weights on C we consider the P_{info} values derived from Equation (6.1)). With a probability $(1-q)$ the random walker can return to any node v' labeled l, and with a probability q the walker follows the edges, where q is a damping factor.

As reported in [35] and [34], RWRs unveil very nice properties when it comes to capturing the structural connectivity between nodes. They overcome several limitations of traditional graph distance measures such as maximum flow, shortest paths, etc. In particular, long connectivity paths are discouraged in a natural way by the restart probability.

For each node v, let

$$pr_l(v) = \begin{cases} \frac{1}{\#\{v \in L; lab(v)=l\}}, & lab(v) = l \\ 0, & otherwise \end{cases}$$

and let $R_l := [pr_l(v)]_{v \in V(C)}$ be the vector representing the restart probabilities. Let $P := [P_l^1(v)]_{v \in V(C)}$ denote the *steady-state probability* vector of an RWR starting at nodes labeled l. The RWR is formally described by

$$P = q\tilde{A}_W P + (1-q)R_l \tag{3.3}$$

where $\tilde{A}_W$ is the normalized, weighted adjacency matrix of the ER graph. Note that $\tilde{A}_W$ contains the normalized P_{info} values derived from the underlying domain. The vector P can be computed by iterating the following equation until convergence.

$$R^{i+1} = q\tilde{A}_W R^i + (1-q)R_l \tag{3.4}$$

where R^0 is set to R_l. By applying this method once for each $l \in \mathcal{L}$, we can estimate for each unlabeled node v the probability $P_l^1(v)$.

In order to compute P_l^2 for an unlabeled node v, we could use the same RWR technique. More precisely, we could run an RWR for every unlabeled node v and compute $P_l^2(v)$ as $P_l^2(v) = \sum_{u:lab(u)=l} P_v(u)$, where $P_v(u)$ would denote the stationary probability of u as determined by the RWR starting at v. However, there might be several hundreds of unlabeled nodes in C, and running an RWR for each of the unlabeled nodes is highly inefficient in practice. Hence, we estimate the P_l^2 in a more relaxed but more efficient way.

Let u be an unlabeled node in C. The probability of having been at node u one step before reaching any node v labeled l is given by:

$$P(u,1) = \sum_{\substack{v:lab(v)=l \\ v \in N(u)}} P_{info}(v|u) \tag{3.5}$$

where $N(u)$ denotes the set of neighboring nodes of u in C, and $P_{info}(v|u)$ for the edge $e = (u, v) \in E(C)$ is defined analogously to Equation (6.6) as:

$$P_{info}(v|u) := \sum_{r \in l_E(e)} P_{info}(l_V(v)|r, l_V(u))$$

Now, one can recursively define the probability that u is reached $s > 1$ steps before any node labeled l as:

$$P(u, s) = \sum_{v \in V(C) \setminus L} P_{info}(v|u) \cdot P(v, s-1) \tag{3.6}$$

Intuitively, s represents the depth of the recursion.

As shown in Algorithm 2, the above recursion can be computed in an iterative manner in time $O(|E(C)|)$.

ALGORITHM 2: `p2lEstimation(`C`)`
Input: ER subgraph C,
Output: estimated value of $P_l^2(v)$ for all $v \in V(C)$,

```
1  X := {v|lab(v) = l}
2  FOR EACH v ∈ X
3    P_l^2(v) = 1/|X|
4  END FOR
5  Y := ∅; U := V(C) \ L
6  WHILE U is not empty DO
7    FOR EACH (u,v) ∈ E(C) such that u ∈ U, v ∈ X
8      compute P_l^2(u) = Σ_{v:lab(v)=l} P_info(v|u) P_l^2(v)
9      insert u into Y
10   END FOR
11   U := U \ Y
12   X := Y; Y := ∅
13 END WHILE
```

In lines 1 - 4 of Algorithm 2, all nodes in X (which are exactly the nodes labeled l) are assigned the same P_l^2 value $\frac{1}{|X|}$. The set U (line 5) contains in each iteration (lines 6 - 13) all unlabeled nodes that have no P_l^2 value. In each iteration, we exclude from U (line 11) all nodes for which a P_l^2 value was determined during the iteration (represented by the set Y, line 5). At the end of each iteration the set X is set to Y. In lines 7 - 10, for each edge (u, v) with $u \in U$ and $v \in X$ we compute $P_l^2(u)$ (line 8). The algorithm terminates when the set U is empty.

At this point, each node v of C has for each $l \in \mathcal{L}$ a probability $P_l(v) = P_l^1(v) \cdot P_l^2(v)$. The label of each node in C can now be easily determined by Equation (5). Finally, the connected subgraph S we were looking for consists exactly of those nodes v for which $lab(v) =$ *informative*. In case $|V(S)| > b$, we successively remove from S the node v that does not belong to T and has minimal $P_l(v)$, for $l =$ *informative*. By the construction of our mining method, it is easy to see that S fulfills the desired properties of Definition 5.

Although the special problem addressed in this section comes with two classes of nodes (i.e, *informative* and *uninformative* nodes), our learning method can easily be generalized to more than two classes. One would have to compute the $P_l(v)$ probabilities as described above for each class label l. This way the subgraph that best represents a certain class of nodes could be retrieved.

4 Experimental Evaluation

For the evaluation of MING we focused on two aspects: (1) extraction efficiency, and (2) quality of the mined subgraphs. In this section, we will present performance results of MING in comparison with the state-of-the-art approaches FSD (for Fast Subgraph Discovery) [12] and CEPS [34].

COMPETITORS In its first phase, FSD efficiently extracts a well-connected candidate subgraph C that contains the query nodes. The candidate generation algorithm applies a series of expansions starting from the query nodes. The expansions follow a best-first strategy and stop when a stopping condition is fulfilled. In a second phase, a final answer graph S is mined from C. This is done by means of a current-flow-based algorithm. In contrast, the more recent approach, CEPS, extracts the most important subgraph S (that captures the main relations between the query nodes) directly from G by determining the most central nodes of G with respect to the query nodes. This is done by applying an RWR from each query node. For each node, the stationary probabilities from each RWR are multiplied to a final node score. The top-k nodes with highest scores constitute the central nodes (i.e. the *centerpieces*). To extract the final subgraph S from G, the authors propose an extraction algorithm that generalizes the candidate generation algorithm of FSD for more than two query nodes. While the first phase of MING pursues the same goal as the first phase of FSD, the second phase of MING is rather related to CEPS. All methods are implemented in Java.

EVALUATION ASPECTS As for the efficiency aspect, we have evaluated the performance of MING on the task of extracting the candidate subgraph C. Therefor, we have compared the running times of our candidate subgraph generation method (represented by Algorithm 1) with the running times of a generalized FSD that works for more than two query nodes. In a second set of experiments, we have evaluated the running time of MING on the task of determining the most informative subgraph S from C. Here, we have compared the mining efficiency of MING with the mining efficiency of CEPS (for same candidate subgraphs C). All efficiency experiments were performed on a 2 GHz Pentium machine with 2 GB of main memory and an Oracle Database (version 9.1) as the underlying persistent storage.

As for the quality aspect, we have conducted an extensive user evaluation to asses the informativeness (i.e. the intuitive understanding of relatedness between given query entities) of result graphs returned by MING and CEPS.

DATA SETS As data sets we have used YAGO [31, 32, 33] and DBLP. The ER graph given by the latest version of YAGO contains more than 2 million nodes (i.e. entities) and 20 million edges (i.e. facts). YAGO combines facts extracted from Wikipedia with facts from Word-

Net [13]. It supports more than 100 interesting relationship labels (e.g., *fatherOf, hasWonPrize, hasAcademicAdvisor, graduatedFrom, bornIn, bornOnDate, marriedTo, actedIn,* etc.), and knows the majority of the entities known to Wikipedia.

From the latest XML version of DBLP we extracted an ER graph consisting of 2 million nodes and 9 million edges. Apart from being sparser than the YAGO graph, the DBLP graph is also in terms of entities and relationship labels much less diverse. Its edges are assigned relationship labels that describe important information about publications and authors (such as, *hasAuthor, appearedIn, publishedInYear, coAuthorOf* and *hasPublicationType*).

Both ER graphs (i.e. the YAGO graph and the DBLP graph) are stored in a relational database with the simple schema

$$EDGE(E_1, relation, E_2, PinfoE_1E_2, PinfoE_2E_1),$$

where E_1, E_2 are entity names and $PinfoE_iE_j$ is a score approximating the value $P_{info}(E_j|relation, E_i)$ as given by Equation (1). For YAGO these scores were estimated by means of co-occurrence statistics for entities. These statistics were directly derived from the Wikipedia corpus. More precisely, $P_{info}(E_j|relation, E_i)$ was estimated as

$$P_{info}(E_j|relation, E_i) \approx \frac{W(E_i, E_j)}{W(E_i)} \tag{4.1}$$

where $W(E_i, E_j)$ denotes the number of Wikipedia pages in which E_i and E_j co-occur, and $W(E_i)$ denotes the number of pages in which E_i occurs. Despite the vague nature of this estimation, it turns out that it captures the intuition described in Section 2.1 in the overwhelming majority of the cases. For DBLP on the other hand, it is very difficult to find an adequate domain from which co-occurrence statistics can be derived. Hence, for the DBLP facts we assume uniform P_{info} values.

4.1 Efficiency

The runtime of FSD and MING is clearly dominated by the candidate subgraph extraction task. For our comparison we used two query sets, one for DBLP and one for YAGO. Each set contained 30 randomly generated queries, where each query consisted of 3 entities. For both query sets, the average runtime of FSD was compared with the average runtime of our candidate extraction method. Both methods were evaluated for each query, based on the same stopping condition (see Algorithm 1). Additionally, both methods were treated uniformly as far as the overhead for database calls is concerned. The results are presented in Figure 5. The candidate generation method of MING clearly outperforms FSD's generation method. The better runtimes of the methods on the YAGO graph can be explained through the denser structure of YAGO.

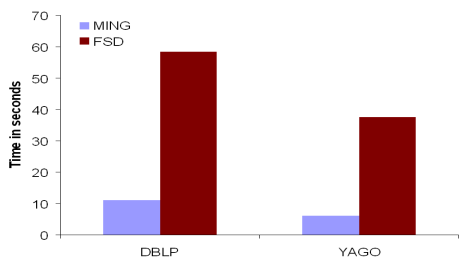

Figure 5: Avg. extraction runtimes for FSD and MING.

In a second experiment, we evaluated the performance of CEPS and MING on the task of mining the final answer graph from a given candidate subgraph. For each of the graphs (YAGO

and DBLP), we randomly generated query sets of queries with 3,4,5, and 6 query nodes. Each set contained 15 queries, resulting in 60 queries per graph. For each candidate subgraph C generated by MING for each query, we measured the average time needed by MING and CEPS to mine the final subgraph S. The results are presented in Figures 6 and 7. The good runtime of CEPS for three query nodes reflects the fact that MING uses a more intricate mining technique. MING applies the STAR algorithm and two RWRs on the candidate subgraph (one from the informative nodes and one from the uninformative nodes). Although CEPS runs one RWR per query node, in the case of three query nodes the running times are comparable. However, as the number of query nodes increases, MING clearly outperforms CEPS. The worse runtimes of both methods on the DBLP graph can be explained by the fact that the subgraphs extracted from DBLP are of a higher order than the subgraphs extracted from YAGO. This leads to higher runtimes for the RWR computations.

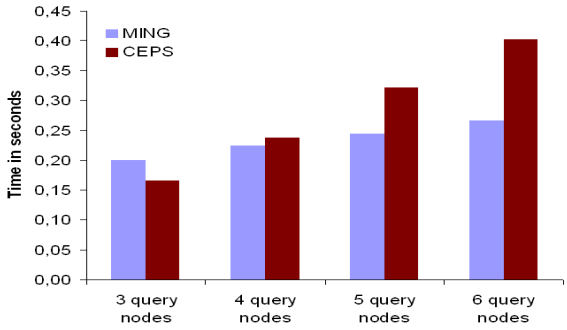

Figure 6: Avg. mining times for CEPS and MING on subgraphs from DBLP.

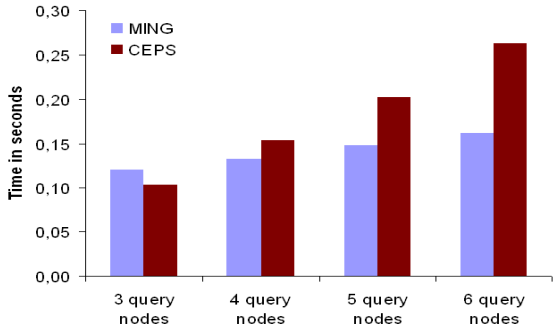

Figure 7: Avg. mining times for CEPS and MING on subgraphs from YAGO.

4.2 Quality

In order to evaluate the quality of returned subgraphs we conducted a user evaluation. The result graphs of MING and CEPS were shown to human judges who had to decide which of the subgraphs better captured the intuition of relatedness for given query entities.

QUERIES In general, it is quite difficult for users to decide whether an ER graph that interconnects a given set of query entities is informative. The reason for this is threefold: (1) informativeness is an intuitive and also subjective notion, (2) a user's intuition has to be supported by the data in the underlying ER graph, and (3) a user needs to have very broad knowledge to assess the informativeness of a result graph for any set of given query nodes (especially when the query nodes represent rather obscure entities). Therefore, for this evaluation, we generated queries in which the query nodes represented famous individuals. Thanks to Wikipedia, YAGO is very rich in terms of famous individuals and contains plenty of interesting facts about them.

In order to generate our queries, we extracted from the Wikipedia lists, a list of famous physicists, a list of famous philosophers, and a list of famous actors. From each of these lists we randomly generated 20 queries, each of them consisting of 2 to 3 query entities, resulting in a set of 60 queries in total. The queries are presented in the appendix.

COMPARISON As ER graph for the user evaluation we chose the YAGO graph. The diversity of YAGO makes it simpler for users to assess whether a result graph captures the intuitive notion of informativeness or not. For each of the 60 queries above, we presented the results produced by CEPS and MING to human judges on a graph-visualization Web interface, without telling them which method produced which graph. Note that none of the judges was familiar with the project. In the visualization interface, we used the same visualization features for both methods. For visualization purposes, the result graphs of CEPS and MING were pruned, whenever they had more than 15 nodes. By restricting the result graphs to such a small number of nodes, both methods were challenged to maintain only the most important of informative nodes in the result graphs. CEPS comes with its own pruning parameter (i.e. visualization parameter). For each query the users were given the possibility to decide which of the presented subgraphs was more informative. In addition, both result graphs could be marked *informative* (or *uninformative*), if the user perceived them as equally informative (or uninformative). The results are presented in Table 1.

	MING	**CEPS**
# times preferred over competitor	182	4
# times marked *informative*	185	7
# times marked *uninformative*	25	103
# times both marked *informative*	3	
# times both marked *uninformative*	21	

Table 1: Results of the user evaluation

RESULTS There were 210 assessments in total, corresponding to more than 3 assessments per query. The result graphs produced by MING were marked 185 times as informative, and out of these, 182 times, they were perceived more informative than the results produced by CEPS. On the other hand, the MING results were marked 25 times as uninformative, and out of these, only 4 times, they were perceived to be less informative than the results produced by CEPS. The results of both methods were perceived in 3 cases as equally informative, and in 21 cases equally uninformative.

The fundamental factor for the qualitative superiority of MING is its subgraph learning method. It learns informative and structurally robust paths between the nodes of an initial tree T that closely interconnects the query nodes. For this, it exploits random walks with restarts guided by co-occurrence statistics derived from the underlying domain. To illustrate the main difference between CEPS and MING, we depict in Figure 8 the answers produced by MING and CEPS for the query that asks for the relations between the *Jessica Lange*, *Robert Redford*, and *Sally Field*. The result graphs were both restricted to 8 nodes. Note that restricting the result graphs to such a small number of nodes, forces both methods to maintain only the most important nodes in their results (i.e. the nodes with the highest scores). The result graph of MING (left graph in Figure 8) has identified the path that connects Sally Field and Jessica Lange through the Academy Award as informative. Furthermore, it has also identified the path

that connects Sally Field and Robert Redford through the node labeled "Californian_actor" as informative. These are both findings that are missed by CEPS.

As observed in our experiments, one of the shortcomings of CEPS is that the quality of its result graphs degrades if some of the query nodes occur in dense regions of the underlying ER graph. In this case, the result graphs become skewed towards the denser regions, especially when the number of result nodes is restricted to a small number. The node representing Robert Redford occurs in a dense region of the YAGO graph, reflecting the fact that Robert Redford has acted in several movies that were produced or directed by him. Consequently, a considerable amount of the RWR starting from this node is absorbed by this region. This leads to a skewed result graph that overemphasizes facts on individual query entities and misses salient relations between the entities. MING, on the other hand, avoids skewed result graphs by running an RWR from the nodes of the tree returned by the STAR algorithm. In our example, the node labeled "American_film_actor" is part of this tree, and contributes equally to the informativeness of nodes in the neighborhood as the query nodes. This way, MING manages to capture the informative relations that Robert Redford and Sally Field are from California, that Jessica Lange and Sally Field are both Academy Award winners, and that all three actors are alive.

These results fortify our assumption that MING indeed captures the intuitive notion of informativeness that we have described in this paper in most of the cases.

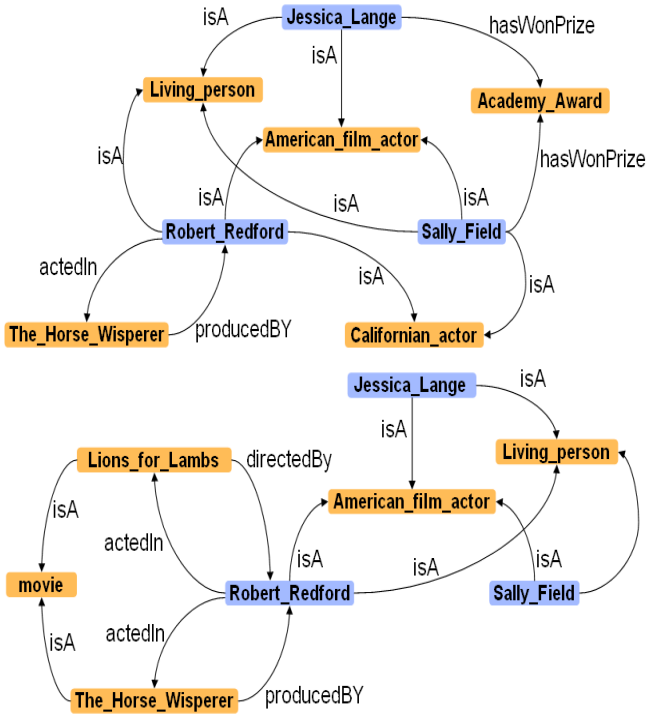

Figure 8: Answer graphs produced by MING (above) and CEPS (below).

5 Conclusion

The method presented in this work, MING, is a significant step forward in analyzing large entity-relationship graphs and extracting informative subgraphs that connect a given set of entities. Our experimental studies have shown that MING is not only much more efficient than prior approaches to this problem, but also produces outputs that are considered more informative by end-users.

Our work contributes to more advanced forms of knowledge discovery on graph-structured data. We are witnessing a strong momentum in knowledge-sharing communities, knowledge-base development, social networks and interoperability across different networks, integration of different kinds of biological networks, and other exciting trends towards a richer knowledge society. Thus, we believe that our work fills an important need and will find applications in a wide variety of fields.

6 Appendix

6.1 MING Queries for the User Evaluation

In order to generate queries for the user evaluation, we extracted from the Wikipedia lists, a list of famous physicists, a list of famous philosophers, and a list of famous actors. From each of these lists we randomly generated 20 queries, each of them consisting of 2 to 3 query entities, resulting in a set of 60 queries in total. The queries are presented in the following:

QUERIES

1. Paul_Dirac — Enrico_Fermi — Max_Born
2. Max_Planck — James_Clerk_Maxwell — Niels_Bohr
3. Richard_Feynman — Michael_Faraday — Ernest_Rutherford
4. Louis_de_Broglie — Max_Born — Michael_Faraday
5. Niels_Bohr — Ernest_Rutherford — Max_Born
6. Isaac_Newton — James_Clerk_Maxwell — Werner_Heisenberg
7. James_Clerk_Maxwell — Niels_Bohr — Stephen_Hawking
8. Werner_Heisenberg — Enrico_Fermi — Paul_Dirac
9. Max_Planck — Werner_Heisenberg — Enrico_Fermi
10. Niels_Bohr — Michael_Faraday — Max_Born
11. Edwin_Hubble — Albert_Einstein
12. Stephen_Hawking — Johannes_Kepler
13. Werner_Heisenberg — Nicolaus_Copernicus
14. Ernest_Rutherford — Blaise_Pascal
15. Hideki_Yukawa — Max_Planck
16. James_Clerk_Maxwell — Hideki_Yukawa
17. Albert_Einstein — Wolfgang_Pauli

18. Ernest_Rutherford — Johannes_Kepler
19. Ludwig_Boltzmann — Richard_Feynman
20. Isaac_Newton — Edmond_Halley
21. Val_Kilmer — Kristin_Davis — Josh_Hartnett
22. Pam_Grier — Matt_Damon — Sharon_Stone
23. Tom_Sizemore — Al_Pacino — Jennifer_Garner
24. Harrison_Ford — Robert_Redford — Sally_Field
25. Sandra_Bullock — Jennifer_Aniston — Kevin_Spacey
26. Michael_Douglas — Billy_Bob_Thornton — Kim_Delaney
27. Sigourney_Weaver — Winona_Ryder — Michael_Keaton
28. Sarah_Michelle_Gellar — Salma_Hayek — Viggo_Mortensen
29. Gina_Gershon — Michael_Douglas — Brittany_Murphy
30. Jennifer_Garner — Sally_Field — Jake_Gyllenhaal
31. Jeanne_Tripplehorn — Jennifer_Aniston — Diane_Lane
32. Clint_Eastwood — Helen_Hunt — Edie_Falco
33. Liv_Tyler — Dennis_Quaid — Teri_Hatcher
34. Demi_Moore — Ashton_Kutcher — Bruce_Willis
35. Jessica_Alba — Leonardo_DiCaprio — Billy_Crystal
36. Maria_Bello — Michael_Douglas — Uma_Thurman
37. George_Clooney — Liam_Neeson — Jake_Gyllenhaal
38. Uma_Thurman — Jake_Gyllenhaal — Jennifer_Garner
39. Kevin_Spacey — Halle_Berry — Julia_Roberts
40. Jodie_Foster — Teri_Hatcher — Christina_Ricci
41. Max_Weber — Georg_Wilhelm_Friedrich_Hegel — Ernst_Mach
42. Rudolf_Carnap — Thomas_Abbt — Max_Horkheimer
43. Johann_Gottfried_Herder — Plato — Gottfried_Leibniz
44. Arthur_Schopenhauer — Moritz_Schlick — Ludwig_Wittgenstein
45. Plato — Friedrich_Nietzsche — Bertrand_Russell
46. Ernst_Mach — Edmund_Husserl — Adam_Smith

47. Plato — Blaise_Pascal — Gottlob_Frege
48. Max_Horkheimer — Arthur_Schopenhauer — Heinrich_Hertz
49. Adam_Smith — Johann_Gottlieb_Fichte — Karl_Wilhelm_Friedrich_Schlegel
50. Max_Horkheimer — Blaise_Pascal — Bernard_Bolzano
51. Karl_Marx — Jean-Paul_Sartre — Ludwig_Wittgenstein
52. Bertrand_Russell — Albert_Einstein
53. Georg_Wilhelm_Friedrich_Hegel — Heinrich_Hertz
54. Arthur_Schopenhauer — Karl_Marx
55. Adam_Smith — Georg_Wilhelm_Friedrich_Hegel
56. Albert_Einstein — Edmund_Husserl
57. Johann_Augustus_Eberhard — Friedrich_Nietzsche
58. Gottlob_Frege — Bernard_Bolzano
59. Karl_Wilhelm_Friedrich_Schlegel — Karl_Marx
60. Albert_Einstein — Friedrich_Nietzsche

Bibliography

[1] Cycorp: Overview of operncyc. http://www.cyc.com/cyc/opencyc. Accessed 01-June-2009.

[2] The gene ontology. http://www.geneontology.org/. Accessed 01-June-2009.

[3] Unified medical language system. http://www.nlm.nih.gov/research/umls/. Accessed 01-June-2009.

[4] Wordnet: a lexical database for the english language. http://wordnet.princeton.edu/. Accessed 01-June-2009.

[5] S. Agrawal, S. Chaudhuri, and G. Das. DBXplorer: A system for keyword-based search over relational databases. In *Proc. of ICDE*, 2002.

[6] G. Bhalotia, A. Hulgeri, C. Nakhe, S. Chakrabarti, and S. Sudarshan. Keyword searching and browsing in databases using BANKS. In *Proc. of ICDE*, 2002.

[7] U. Brandes, M. Gaertler, and D. Wagner. Experiments on graph clustering algorithms. In *European Symposium on Algorithms*. Springer-Verlag, 2003.

[8] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *Computer Networks and ISDN Systems*, 1998.

[9] S. Chakrabarti. Dynamic personalized pagerank in entity-relation graphs. In *WWW*, 2007.

[10] I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. In *KDD*, 2003.

[11] B. Ding, J. Yu, S. Wang, L. Qing, X. Zhang, and X. Lin. Finding top-k min-cost connected trees in databases. In *Proc. of ICDE*, 2007.

[12] C. Faloutsos, K. S. McCurley, and A. Tomkins. Fast discovery of connection subgraphs. In *KDD*, 2004.

[13] C. Fellbaum, editor. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[14] D. Gibson, J. M. Kleinberg, and P. Raghavan. Inferring web communities from link topology. *HYPERTEXT '98*, 1998.

[15] D. Gibson, R. Kumar, and A. Tomkins. Discovering large dense subgraphs in massive graphs. In *VLDB*. VLDB Endowment, 2005.

[16] M. Girvan and M. E. Newman. Community structure in social and biological networks. *Proc. Natl. Acad. Sci. USA*, 99(12), 2002.

[17] E. Hartuv and R. Shamir. A clustering algorithm based on graph connectivity. In *Information Processing Letters*, 2000.

[18] H. He, H. Wang, J. Yang, and P. Yu. BLINKS: Ranked keyword searches on graphs. In *Proc. of SIGMOD*, 2007.

[19] V. Hristidis, L. Gravano, and Y. Papakonstantinou. Efficient ir-style keyword search over relational databases. In *Proc. of VLDB*, 2003.

[20] V. Hristidis and Y. Papakonstantinou. DISCOVER: Keyword search in relational databases. In *Proc. of VLDB*, 2002.

[21] V. Kacholia, S. Pandit, S. Chakrabarti, S. Sudarshan, R. Desai, and H. Karambelkar. Bidirectional expansion for keyword search on graph databases. In *Proc. of VLDB*, 2005.

[22] G. Kasneci, M. Ramanath, M. Sozio, F. M. Suchanek, and G. Weikum. STAR: Steiner-Tree Approximation in Relationship Graphs. In *ICDE 2009*. IEEE, 2009.

[23] D. Kempe, J. M. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *KDD*, 2003.

[24] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5), 1999.

[25] J. M. Kleinberg. Hubs, authorities, and communities. *ACM Comput. Surv.*, 31(4), 1999.

[26] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling the web for emerging cyber-communities. In *Computer Networks*, 1999.

[27] M. Kuramochi and G. Karypis. Frequent subgraph discovery. *ICDM*, 2001.

[28] I. Niles and A. Pease. Towards a standard upper ontology. In *FOIS 2001*. ACM Press, 2001.

[29] A. Papadopoulos, A. Lyritsis, and Y. Manolopoulos. Skygraph: an algorithm for important subgraph discovery in relational graphs. *Data Mining and Knowledge Discovery*, 17(1), 2008.

[30] C. Ramakrishnan, W. H. Milnor, M. Perry, and A. P. Sheth. Discovering informative connection subgraphs in multi-relational graphs. *SIGKDD Explor. Newsl.*, 7(2), 2005.

[31] F. Suchanek. *Automated Construction and Growth of a Large Ontology*. PhD thesis, Universität des Saarlandes, 2008.

[32] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A Core of Semantic Knowledge. In *Proc. of WWW*, 2007.

[33] F. M. Suchanek, G. Kasneci, and G. Weikum. YAGO: A Large Ontology from Wikipedia and WordNet. *Elsevier Journal of Web Semantics*, 2008.

[34] H. Tong and C. Faloutsos. Center-piece subgraphs: problem definition and fast solutions. In *KDD*. ACM Press, 2006.

[35] H. Tong, C. Faloutsos, and J.-Y. Pan. Fast random walk with restart and its applications. In *ICDM*. IEEE Computer Society, 2006.

[36] S. M. v. van Dongen. *Graph Clustering by Flow Simulation*. PhD thesis, University of Utrecht, 2000.

[37] X. Yan, J. X. Zhou, and J. Han. Mining closed relational graphs with connectivity constraints. In *KDD*, New York, NY, USA, 2005. ACM Press.

MPI-I-2009-5-007	G. Kasneci, S. Elbassuoni, G. Weikum	MINERS: Mining Informative Entity Relationship Subgraphs
MPI-I-2009-RG1-002	P. Wischnewski, C. Weidenbach	Contextual rewriting
MPI-I-2009-5-006	S. Bedathur, K. Berberich, J. Dittrich, N. Mamoulis, G. Weikum	Scalable phrase mining for ad-hoc text analytics
MPI-I-2009-5-004	N. Preda, F.M. Suchanek, G. Kasneci, T. Neumann, G. Weikum	Coupling knowledge bases and web services for active knowledge
MPI-I-2009-5-003	T. Neumann, G. Weikum	The RDF-3X engine for scalable management of RDF data
MPI-I-2008-RG1-001	A. Fietzke, C. Weidenbach	Labelled splitting
MPI-I-2008-5-004	F. Suchanek, M. Sozio, G. Weikum	SOFI: a self-organizing framework for information extraction
MPI-I-2008-5-003	F.M. Suchanek, G. de Melo, A. Pease	Integrating Yago into the suggested upper merged ontology
MPI-I-2008-5-002	T. Neumann, G. Moerkotte	Single phase construction of optimal DAG-structured QEPs
MPI-I-2008-5-001	F. Suchanek, G. Kasneci, M. Ramanath, M. Sozio, G. Weikum	STAR: Steiner tree approximation in relationship-graphs
MPI-I-2008-4-003	T. Schultz, H. Theisel, H. Seidel	Crease surfaces: from theory to extraction and application to diffusion tensor MRI
MPI-I-2008-4-002	W. Saleem, D. Wang, A. Belyaev, H. Seidel	Estimating complexity of 3D shapes using view similarity
MPI-I-2008-1-001	D. Ajwani, I. Malinger, U. Meyer, S. Toledo	Characterizing the performance of Flash memory storage devices and its impact on algorithm design
MPI-I-2007-RG1-002	T. Hillenbrand, C. Weidenbach	Superposition for finite domains
MPI-I-2007-5-003	F.M. Suchanek, G. Kasneci, G. Weikum	Yago : a large ontology from Wikipedia and WordNet

MPI-I-2007-5-002	K. Berberich, S. Bedathur, T. Neumann, G. Weikum	A time machine for text search
MPI-I-2007-5-001	G. Kasneci, F.M. Suchanek, G. Ifrim, M. Ramanath, G. Weikum	NAGA: searching and ranking knowledge
MPI-I-2007-4-008	J. Gall, T. Brox, B. Rosenhahn, H. Seidel	Global stochastic optimization for robust and accurate human motion capture
MPI-I-2007-4-007	R. Herzog, V. Havran, K. Myszkowski, H. Seidel	Global illumination using photon ray splatting
MPI-I-2007-4-006	C. Dyken, G. Ziegler, C. Theobalt, H. Seidel	GPU marching cubes on shader model 3.0 and 4.0
MPI-I-2007-4-005	T. Schultz, J. Weickert, H. Seidel	A higher-order structure tensor
MPI-I-2007-4-004	C. Stoll, E. de Aguiar, C. Theobalt, H. Seidel	A volumetric approach to interactive shape editing
MPI-I-2007-4-003	R. Bargmann, V. Blanz, H. Seidel	A nonlinear viseme model for triphone-based speech synthesis
MPI-I-2007-4-002	T. Langer, H. Seidel	Construction of smooth maps with mean value coordinates
MPI-I-2007-4-001	J. Gall, B. Rosenhahn, H. Seidel	Clustered stochastic optimization for object recognition and pose estimation
MPI-I-2007-2-001	A. Podelski, S. Wagner	A method and a tool for automatic veriication of region stability for hybrid systems
MPI-I-2007-1-003	A. Gidenstam, M. Papatriantafilou	LFthreads: a lock-free thread library
MPI-I-2007-1-002	E. Althaus, S. Canzar	A Lagrangian relaxation approach for the multiple sequence alignment problem
MPI-I-2007-1-001	E. Berberich, L. Kettner	Linear-time reordering in a sweep-line algorithm for algebraic curves intersecting in a common point
MPI-I-2006-5-006	G. Kasnec, F.M. Suchanek, G. Weikum	Yago - a core of semantic knowledge
MPI-I-2006-5-005	R. Angelova, S. Siersdorfer	A neighborhood-based approach for clustering of linked document collections
MPI-I-2006-5-004	F. Suchanek, G. Ifrim, G. Weikum	Combining linguistic and statistical analysis to extract relations from web documents
MPI-I-2006-5-003	V. Scholz, M. Magnor	Garment texture editing in monocular video sequences based on color-coded printing patterns

MPI-I-2006-5-002	H. Bast, D. Majumdar, R. Schenkel, M. Theobald, G. Weikum	IO-Top-k: index-access optimized top-k query processing
MPI-I-2006-5-001	M. Bender, S. Michel, G. Weikum, P. Triantafilou	Overlap-aware global df estimation in distributed information retrieval systems
MPI-I-2006-4-010	A. Belyaev, T. Langer, H. Seidel	Mean value coordinates for arbitrary spherical polygons and polyhedra in $\mathbb{R}^3$
MPI-I-2006-4-009	J. Gall, J. Potthoff, B. Rosenhahn, C. Schnoerr, H. Seidel	Interacting and annealing particle filters: mathematics and a recipe for applications
MPI-I-2006-4-008	I. Albrecht, M. Kipp, M. Neff, H. Seidel	Gesture modeling and animation by imitation
MPI-I-2006-4-007	O. Schall, A. Belyaev, H. Seidel	Feature-preserving non-local denoising of static and time-varying range data
MPI-I-2006-4-006	C. Theobalt, N. Ahmed, H. Lensch, M. Magnor, H. Seidel	Enhanced dynamic reflectometry for relightable free-viewpoint video
MPI-I-2006-4-005	A. Belyaev, H. Seidel, S. Yoshizawa	Skeleton-driven laplacian mesh deformations
MPI-I-2006-4-004	V. Havran, R. Herzog, H. Seidel	On fast construction of spatial hierarchies for ray tracing
MPI-I-2006-4-003	E. de Aguiar, R. Zayer, C. Theobalt, M. Magnor, H. Seidel	A framework for natural animation of digitized models
MPI-I-2006-4-002	G. Ziegler, A. Tevs, C. Theobalt, H. Seidel	GPU point list generation through histogram pyramids
MPI-I-2006-4-001	A. Efremov, R. Mantiuk, K. Myszkowski, H. Seidel	Design and evaluation of backward compatible high dynamic range video compression
MPI-I-2006-2-001	T. Wies, V. Kuncak, K. Zee, A. Podelski, M. Rinard	On verifying complex properties using symbolic shape analysis
MPI-I-2006-1-007	H. Bast, I. Weber, C.W. Mortensen	Output-sensitive autocompletion search
MPI-I-2006-1-006	M. Kerber	Division-free computation of subresultants using bezout matrices
MPI-I-2006-1-005	A. Eigenwillig, L. Kettner, N. Wolpert	Snap rounding of Bézier curves
MPI-I-2006-1-004	S. Funke, S. Laue, R. Naujoks, L. Zvi	Power assignment problems in wireless communication

MPI-I-2005-5-002	S. Siersdorfer, G. Weikum	Automated retraining methods for document classification and their parameter tuning
MPI-I-2005-4-006	C. Fuchs, M. Goesele, T. Chen, H. Seidel	An emperical model for heterogeneous translucent objects
MPI-I-2005-4-005	G. Krawczyk, M. Goesele, H. Seidel	Photometric calibration of high dynamic range cameras
MPI-I-2005-4-004	C. Theobalt, N. Ahmed, E. De Aguiar, G. Ziegler, H. Lensch, M.A. Magnor, H. Seidel	Joint motion and reflectance capture for creating relightable 3D videos
MPI-I-2005-4-003	T. Langer, A.G. Belyaev, H. Seidel	Analysis and design of discrete normals and curvatures
MPI-I-2005-4-002	O. Schall, A. Belyaev, H. Seidel	Sparse meshing of uncertain and noisy surface scattered data