Answering Who/When, What, How, Why through Constructing Data Graph, Information Graph, Knowledge Graph and Wisdom Graph

Lixu Shao*, Yucong Duan†, Xiaobing Sun‡, Honghao Gao §, Donghai Zhu¶ and Weikai Miao‖
*College of Information Science and Technology,
State Key Laboratory of Marine Resource Utilization in the South China Sea,Hainan University, China
Email: 751486692,841213174,466971642@qq.com
‡School of Information Engineering Yangzhou University, China
Email: xbsun@yzu.edu.cn
§Computing Center, Shanghai University, Shanghai, China
Email:gaohonghao@shu.edu.cn
‖Software Engineering Institute, East China Normal University, China
Email:wkmiao@sei.ecnu.edu.cn

***Abstract*—Knowledge graphs have been widely adopted, in large part owing to their schema-less nature. It enables knowledge graphs to grow seamlessly and allows for new relationships and entities as needed. Natural language questions are the most intuitive way of formulating an information need. People can formulate questions to express their information needs. Natural language questions as a query language present an ideal compromise between keyword and structured querying. Questions can be used to express complex information needs that cannot be expressed as keywords without a significant loss in structure and semantics. Knowledge graph has abundant natural semantics and can contain various and more complete information. Its expression mechanism is closer to natural language. We propose to clarify the expression of knowledge graph as a whole. We use knowledge graph to solve the Five Ws problems respectively which are guided by interrogative words such as who/when, what, how and why. We also propose to specify knowledge graph in a progressive manner as four basic forms including data graph, information graph, knowledge graph and wisdom graph.**
[1]

1. Introduction

In terms of usability, questions are a more accessible medium for expressing complex information needs. The "What, How, Who, When, Why" (Five Ws) are questions whose answers are considered to be the basis of information-gathering and problem-solving. Traditionally, users interact with search engines by providing keywords to the search engine and obtaining a list of documents that best match those keywords. The two major drawbacks of this pattern are the answer to granularity and query expression [1]. Keyword queries are highly "telegraphic" [2] which means that they have limited expressiveness with missing verbs, prepositions, clauses and phrase clues. With this limitation, users cannot express complex information needs. Knowledge graph has become a powerful tool to represent knowledge in the form of a labelled directed graph and to give semantics to textual information. A knowledge graph is a graph constructed by representing each item, entity and user as nodes, and linking those nodes that interact with each other via edges. However, there is still a lack of a standard definition of knowledge graph. In [10] the authors elaborated conceptual approaches for defining data, information, and knowledge. Data is acquired by observing the basic individual items of numbers or other information, but on their own, without context, they have no information. Information is conveyed through the context of data and data combinations, and may be suitable for analysis and interpretation. Knowledge is the general understanding and consciousness gained from the accumulated information, and the experience is adjusted so that a new background can be envisaged. Wisdom is an extrapolative and non-deterministic, non-probabilistic process. It calls upon all the previous levels of consciousness, and specifically upon special types of human programming [12]. We propose to clarify the expression of knowledge graph as a whole. We use knowledge graph to solve the Five Ws problems respectively which are instructed by interrogative words such as who/when, what, how and why. Each of them can be widely used to explore and evaluate a variety of knowledge theories and systems. We show the progressive forms of knowledge type in Table 1. Correspondingly, we propose to specify knowledge graph in a progressive manner as four basic forms including data graph, information graph, knowledge graph and wisdom graph.

In the rest of this paper, we elaborate how to use knowledge graph to solve the problem guided by interrogative words including "who & when" "what" "how" "why" in Section 2, 3, 4, and 5 respectively. The related works are

1. DOI Reference Number 10.18293 / SEKE2017-079

TABLE 1. THE PROGRESSIVE FORM OF KNOWLEDGE TYPE

(rightside: general forms)	data	information	knowledge	wisdom
Semantic load	not specified for stakeholders/machine	settled for stakeholders/machine	abstracted on known information	Known to unknown
format	discrete elements	related elements	Probabilistic or categorization	(frame or stylish expression)
knowledge answer	who/when/where	what	how	why
usage	transmission	communication	reasoning	prediction
Sub graphs	data graph	information graph	knowledge graph	wisdom pool

elaborated in Section 6. And we give our conclusions in Section 7.

2. Relationship between "who/when/where", 'what", "how" and "why"

In Fig. 1 we show the relationship between "Who/When/Where", "What" "How" and "Why". We use discrete points to represent data. Data represents a fact or statement of event without relation to other things and we can use data to give answers to the questions guided by "Who/When/Where". Data requires interpretation to become information. To translate data to information, several factors must be considered. The factors involved are determined by the creator of data and the desired information. Information is data that has been given meaning by way of relational connection. This "meaning" can be useful, but does not have to be. By organizing the associated discrete data points together, we can provide answers to "what" questions. Knowledge is the appropriate collection of information and it is considered to be of great use. Knowledge is a deterministic process and can be tacit as well as explicit. When people "memorize" information, they have amassed knowledge. This knowledge has useful meaning to them, but it does not provide for, in and of itself, an integration such as would infer further knowledge. We use knowledge to answer "how" questions. The first three categories relate to the past and they deal with what has been or what is known. Only the fourth category, wisdom, deals with the future because it incorporates vision and design. With wisdom, people can create the future rather than just grasp the present and past [16]. But achieving wisdom isn't easy. People must move successively through the other three categories. By assessing and understanding the acquired knowledge, we infer what we did not know before. Wisdom can provide answers to "Why" questions.

3. Using Knowledge Graph to Answer the Question of "Who& When"

We represent entities with the logical predicate Ent(E). We represent labels with the logical predicate Lbl(L). Then an entity having labels can be defined as Ent(E) = Lbl(L). Relations are represented with the logical predicate Rel(E1,E2,R) where the relation R holds between the entities E1 and E2, for instance, R(E1,E2). We define a knowledge graph G as a set of triples of the form (s,r,t) where s,t $\in$ E and r $\in$ R. When a user presents his/her information needs

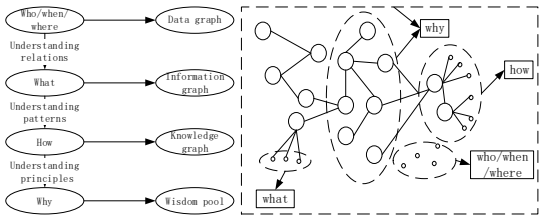

Figure 1. Relations between "Who/when/where", "What", "How" and "Why".

by asking questions, we propose to segment questions and recognize the basic entity E1, relationship type R, and type of target entity E2. The query seeks target entity E2, where R (E1, E2) holds and mentions of entities have been linked to the corresponding knowledge graph. In general, we can present a user's question to a triad form. A triple pattern query, or simply a query is a set of triple patterns Q = $q_1, ..., q_n$ and a projection set P(Q) of variables. We require the join graph of Q, where $q_i s$ are vertices and an edge exists between every pair of vertices sharing a common variable, to be a connected graph (to avoid computing Cartesian products). P (Q) is a subset of the variables in Q, defining the output structure, typically tuples of entities. We also refer to the triple pattern set Q as a query when the projection set is not relevant for the discussion. We adapt one of the figure examples from [1] to explain. When a user enters "who is the spouse of Robert Rossellini", it can be expressed as (?x, spouse, RobertRessellini). The corresponding triple pattern query is

SELECT ?x WHERE ?x spouse Robert Rossellini.

According to the knowledge graph shown in Fig. 2 we can learn that the answer to the question is P (Q) = Ingrid Bergman. The denition of an answer to a query is a natural extension of an answer to a single triple pattern.The method of dealing with "when" questions is similar to that of "who" questions.

4. Using Knowledge Graph to Answer the Question of "What"

4.1. Inferring the property of an entity according to rules

Compared with existing modelling tools such as UML Class Diagrams, knowledge graph has abundant natural

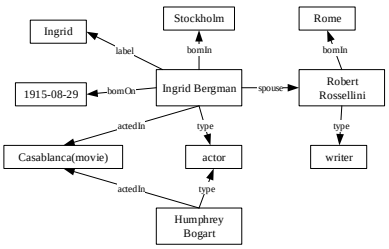

Figure 2. An example knowledge graph of people (only a fragment is shown).

semantics, and its expression mechanism is closer to natural language. Knowledge graph can contain various and more complete information. According to algorithm 1, we can generate rules from a large number of training data sets. When a user searches for an entity, if collection of the entity′s (relation, label) value pair matches to a certain rule, then we can know which class the entity belong to. For example, we give a set of rules about verifying vertebrate categories as shown below:

r1: (is a, flying animal) ∧ (has, feather) ∧ (is a, constant temperature animal) → bird.

r2: (is a, aquatic animal) ∧ (has, squama) ∧ (breath with, gills) → fish.

r3: (is a, variable temperature animal) ∧ (has, squama) ∧ (breath with, lung) → reptile.

r4: (is a, viviparous animal) ∧ (is a, constant temple animal) → mammal.

r5: (is a, variable temperature animal) ∧ (is a, semi-aquatic animal) ∧ (breath with, lung) → amphibian.

Algorithm 1 Extracting rules according to the data set

Require: Training record E;
Ensure: collection of relation-label value pair $A(R_i, L_i)$, an ordered set of classes $Y_0\{y_1, y_2, , y_k\}$, initial list of rules R={}
For every class y ∈ $Y_0 - y_k$ **do**
while the termination condition is not satisfied **do**
1. generate a rule r ← (A, y);
2. remove the training records covered by r from E;
3. append r to the end of the rule list R: R ← R ∪ r
End while
End For

According to the above rules, we give partial knowledge graph about the classification of vertebrates as Fig. 3 shows. Entity swallow and its labels can be represented as Ent(swallow) = Lbl(flying animal), Lbl(feather). In Fig. 3, Ent(swallow) and Ent (bird) have two matching characteristics, so we can recognize the swallow as a bird at the probability of two-thirds. The probability, denoted as P can be computed according to (1):

$$P = \frac{|Ent(E1) \cap Ent(E2)|}{|Ent(E1)|} \quad (1)$$

where $|Ent(E1) \cap Ent(E2)|$ indicates the number of common labels of entity E1 and E2, $|Ent(E1)|$ indicates the number of labels of entity E1.

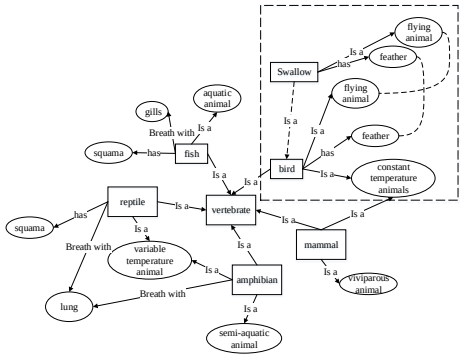

Figure 3. Partial knowledge graph of vetebrates.

4.2. Establishing New Association through Knowledge Reasoning

In knowledge graph we can establish a new association between more entities through knowledge reasoning, so as to expand relations between entities and increase the edge density of the knowledge graph. Reasoning often requires the support of relevant rules, such as "spouse" and "male" to reason out "husband", from the date of birth and the current time to reason out of age and so on. These rules can be constructed manually by people, but often time-consuming and laborious, and it is difficult to exhaust all the reasoning rules in complex relationships. At present, it mainly relies on the co-occurrence of relations, and uses association mining techniques to automatically find reasoning rules. The classical method of using relational rule to achieve relational extraction is Path Ranking Algorithm, which uses each different relational path as a one-dimensional feature. By constructing a large number of relational paths in knowledge graph, we are able to obtain the eigenvector of relation classification and a relational classifier to extract the relationship. For example, there is doubt that whether it is correct to establish a relationship profession between entity Charlotte Bronte and entity writer in Fig. 4. We can calculate the rate of correctness of a relationship, denoted as $Cr(E1, R, E2)$, according to (2):

$$Cr(E1, R, E2) = \sum_{\pi \in Q} P(E1 \rightarrow E2, \pi)\theta(\pi) \quad (2)$$

where P(E1, E2) indicates one path between E1 and E2, Q indicates all paths starting from E1 and ending with E2, $\theta(\pi)$ represents the weight obtained by training.

4.3. Semantic links and active push of information

Knowledge graph as an important support for semantic query includes a large number of named entities and semantic relations. Knowledge graph can provide an open knowledge access interface and to a certain extent it reflects the real world of inter-entity relations. The graph structure of knowledge graph is not restricted by form. Knowledge graph

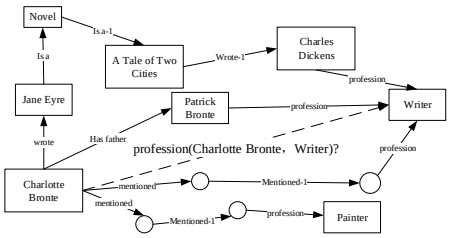

Figure 4. Partial knowledge graph of people.

can express abundant natural semantics and can supplement related information among terms. The graph-based nature of knowledge graph makes possible a linkage to other graphs thus resulting in an easy integrating of multiple kinds of information and an enhancement in integrity of information. By exploring the graph, new connections and commonalities between items and users can be discovered and exploited. For example, in Fig. 5, when a user enters the question "What happened to Air France on September 15?", we can find the corresponding entity nodes that are "Air France" and "9.15" in the knowledge graph. Then we learn that there is a staff strike event occurred in Air France on September 15. Through entity-relation links and relevant information pushing, we can get an extended knowledge graph and show the user more detail information such as flight numbers and follow-up effects of the incident.

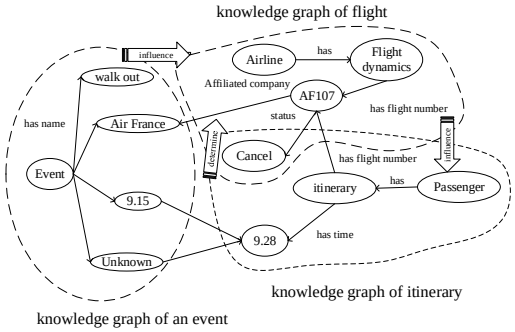

Figure 5. A combined knowledge graph.

5. Using Knowledge Graph to Answer the Question of "How"

Knowledge graph can also show a process of dealing with an event when a user enters a question "how to do...". Path queries on a knowledge graph can be used to answer compositional questions such as What languages are spoken by people living in Lisbon?. However, knowledge graphs often have missing facts (edges) which disrupt path queries. Recent models for knowledge base completion impute missing facts by embedding knowledge graphs in vector spaces. We show that these models can be recursively applied to answer path queries, but that they suffer from cascading errors. This motivates a new "compositional" training objective, which dramatically improves all models ability to answer path queries, in some cases more than doubling accuracy. Let E be a set of entities and R be a set of binary relations. A knowledge graph G is defined as a set of triples of the form (s,r,t) where s,t $\in$ E and r $\in$R. A path query q consists of an initial anchor entity, s, followed by a sequence of relations to be traversed, p = (r1,...,rk). The answer or denotation of the query, [q], is the set of all entities that can be reached from s by traversing p. Formally this can be defined recursively:

$$[s] \stackrel{\text{def}}{=} \{s\}, [q/r] \stackrel{\text{def}}{=} \{t : \exists s \in [q], (s, r, t) \in G\}$$

We define the set of candidate answers to a query C(q) as the set of all entities that "type match", namely those that participate in the final relation of q at least once. And let N(q) be the incorrect answers:

$$C(s/r1/.../rk) \stackrel{\text{def}}{=} \{t|\exists e, (e, r_k, t) \in G\}, N(q) \stackrel{\text{def}}{=} C(q)\backslash[q].$$

For example, when a user enters a "how to recruit", we can find other entities that have an "include" relationship with the entity "recruitment process" based on the keyword recruit in Fig. 6. We can then query other entities related to the previous entity follow the path if necessary and continue to perform these two steps until all relevant entities are found.

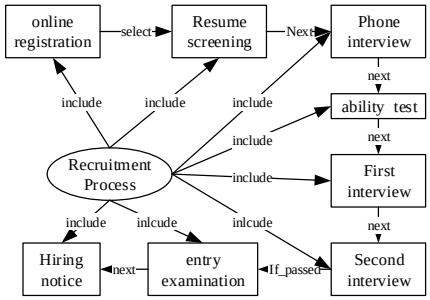

Figure 6. Partial knowledge graph of recruitment process.

6. Using Knowledge Graph to Answer the Question of "Why"

When people want to know the cause of something, she/he usually raise questions related to why, and the reason for one thing includes both direct and indirect causes. Five Whys is an iterative interrogative technique used to explore the cause-and-effect relationships underlying a particular problem [15]. The primary goal of the technique is to determine the root cause of a defect or problem by repeating the question "Why?" Each answer forms the basis of the next question. We give a description in Algorithm 2. The "Why" in the name derives from an anecdotal observation on the number of iterations needed to resolve the problem. We give an example in Fig. 7 by asking why the vehicle cannot starting. Through the first inquiry we know the reason is the battery is dead. Then we continue to ask questions

"Why the battery is dead?" Through five iterations, we can know that the root cause is that the vehicle was not maintained according to the recommended service schedule. The questioning for this example could be taken further to a sixth, seventh, or higher level, but five iterations of asking why is generally sufficient to get to a root cause. The key is to encourage the trouble-shooter to avoid assumptions and logic traps and instead trace the chain of causality in direct increments from the effect through any layers of abstraction to a root cause that still has some connection to the original problem. Note that, in this example, the fifth why suggests a broken process or an alterable behaviour, which is indicative of reaching the root-cause level. It is interesting to note that the last answer points to a process. This is one of the most important aspects in the Five Whys approach - the real root cause should point toward a process that is not working well or does not exist.

Algorithm 2 Iterative interrogative algorithm

Input: Question Q {Why Entity E_0 ...?};
Output: A directed acyclic graph G
while E_i is the root cause of E_0 **do**
1. Find the entity E_i that matches the triple $(E_0, causedBy, E_i)$;
2. i++;
3. Add node E_i and relationship $causedBy$ to G.
End while

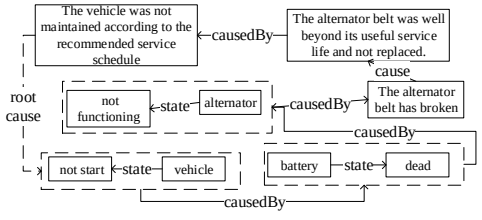

Figure 7. An example knowledge graph about causes of not starting vehicle.

In addition to inquiry the causes of the occurrence of an event, "why" can also be used to question the causality. A cause and its effect can be of different kinds of entity. For example, when a user inquiries about why smoke damage people's lungs, smoke is the cause and damage to lungs is the relevant effect. We show the detail reason through knowledge graph by finding the middle nodes. In Fig.8 we can find that there are five paths between entity smoke and entity lung. Each path indicates a reason for that smoking will damage people's lungs. In algorithm 3 we elaborate how to find all the complete paths between two entities in order to find the detailed cause of the causal relationship between them.

Algorithm 3 Finding all the complete paths between two entities

Input: Entity E1 and Entity E2;
Output: Knowledge graph G with all paths between E1 and E2
while E_2 is visited **do**
1. Find the first neighbour E_i of E1 and mark it as visited;
2. E1=E_i;
End while
3. Back to the last node;
4. Repeat step 1 and step 2.

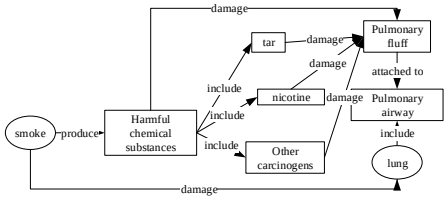

Figure 8. Causal relationship between smoke and lungs.

We expand Fig. 6 to illustrate details of the recruitment in a more detailed diagram. In Fig. 9, we can easily find the time and place of the recruitment as well as the interviewers who are responsible for the interview based on some discrete data points. These discrete data points can be the answers to "when", "where" and "who" respectively. Linking discrete data points through relevant connections can reflect some of the basic situation of the recruitment. Contents in the right green box show the recruitment process which is the answers to "how" questions. And according to contents in the purple box we can understand the reasons for holding the recruitment. Certainly, there are some other reasons not shown in the figure which yet can be inferred from the existing known knowledge. For example, reason for selecting Alice and Bob as the interviewers may be that Alice and Bob are very experienced.

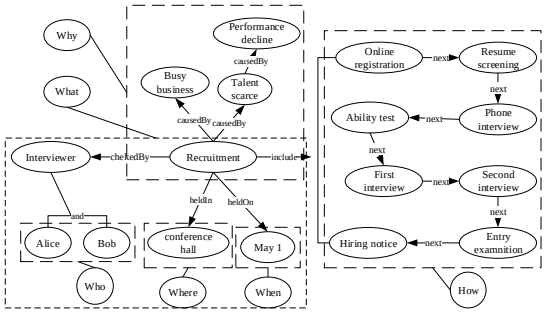

Figure 9. An example of reflecting Five Ws on a knowledge graph.

7. Related works

For users with complex information needs, they can express their needs by proposing natural language questions. These questions are subsequently interpreted with respect to the specific knowledge graph at hand by mapping them to a triple pattern query, which can be issued to the knowledge graph, returning the desired answers to the user [3, 4, 5, 8]. This new setting of large knowledge graphs presents an opportunity to tackle the question answering problem using new approaches. By working against a knowledge graph, crisp entities can be returned as answers. By exploiting the structure provided by the knowledge graph and extracting relationship between entities, one can also answer complex questions that require multiple joins, corresponding to paths

in the knowledge graph [6, 13, 14]. In these cases, a knowledge graph can be used to return answers that are proper tuples of entities, rather than singletons. Moreover, users will be able to ask for how an answer was derived by looking at the query that produced it. In [7] the authors presented a semantic parsing framework for question answering using a knowledge base and defined a query graph as the meaning representation that can be directly mapped to a logical form. Semantic parsing is reduced to query graph generation, formulated as a staged search problem. In [9] the authors presented a natural-language question-answering system that gives access to the accumulated knowledge of one of the largest community projects via an automatically acquired structured knowledge base. Inheriting concepts from ECML, at the highest abstraction level, in [11] the authors elaborated a contract in the eSourcing ontology and eSML answering three conceptual questions including the "who, where, and what" question.

8. Conclusion

The availability of large knowledge graphs presents a challenge and an opportunity that are, in some sense, duals of each other. The challenge lies in how to make an abundance of knowledge easily accessible to humans. Only a fraction of potential consumers of this knowledge will be versed in formulating triple pattern queries that express their information needs. Even for such people, the sheer size of data and the lack of a schema mean that formulating a triple pattern query can be a challenging task, requiring multiple rounds of tedious query reformulation. A solution to the above problem is to allow users to query knowledge graphs by proposing natural language questions. We proposed in this work to introduce knowledge graph to solve questions guided by the interrogative words including who/when, what, how and why(Five Ws). We elaborate three algorithms in this dissertation to facilitate effective querying of knowledge graphs and acquisition of knowledge. The contributions of this work are in the areas of question answering over knowledge graphs, relaxed knowledge graph querying combining structured and unstructured data, and open information extraction. These works initially validate that we can use natural language to express knowledge graph and then evaluate a large number of knowledge theories and systems. On the other hand, data, information, knowledge, and wisdom forms progressive layers of expression. We can dig information from data, obtain knowledge from information and acquire wisdom from knowledge. Our work lays the foundation of investigation from data to wisdom. In the next stage, we will deal with the 5W problems for the same background and different background at data, information, knowledge and wisdom level respectively. In addition, we will deal with the ambiguity of mapping problem to the query which is both structural and informative.

Acknowledgments

This paper is supported by NSFC under Grant (No.61363007, No.61502294, No.61662021, No. 61661019), NSF of Hainan No.ZDYF2017128, Natural Science Foundation of Shanghai under Grant No.15ZR1415200 and STCSM project No.14YF1404300.

References

[1] M. Yahya, *Question answering and query processing for extended knowledge graphs*, 3rd ed. Clerk Maxwell, A Treatise on Electricity and Magnetism, vol. 2. Oxford: Clarendon, 1892, pp.6873.

[2] M. Joshi, U. Sawant and S. Chakrabarti, *Knowledge Graph and Corpus Driven Segmentation and Answer Inference for Telegraphic Entity-seeking Queries*. Conference on Empirical Methods in Natural Language Processing. 2014:1104-1114.

[3] K. Guu and J. Miller and P. Liang, *Traversing Knowledge Graphs in Vector Space*. Computer Science, 2015.

[4] Y. Lin and Z. Liu and H. Luan, et al., *Modeling Relation Paths for Representation Learning of Knowledge Bases*. Computer Science, 2015.

[5] J. Carroll and I. Dickinson and C. Dollin, et al., *Jena: implementing the semantic web recommendations*. International World Wide Web Conference on Alternate Track Papers & Posters. ACM, 2004:74-83.

[6] P. Minervini and N. Fanizzi and C. D'Amato, et al., *Scalable Learning of Entity and Predicate Embeddings for Knowledge Graph Completion*. IEEE, International Conference on Machine Learning and Applications. IEEE, 2015:162-167.

[7] W. T. Yih and M. W. Chang and X. He, et al., *Semantic Parsing via Staged Query Graph Generation: Question Answering with Knowledge Base*. Meeting of the Association for Computational Linguistics and the, International Joint Conference on Natural Language Processing. 2015:1321-1331.

[8] A. Bordes and S. Chopra and J. Weston, *Question Answering with Subgraph Embeddings*. Computer Science, 2014.

[9] P. Adolphs and M. Theobald and U. Schfer, et al., *YAGO-QA: Answering Questions by Structured Knowledge Queries*. IEEE Fifth International Conference on Semantic Computing. IEEE Computer Society, 2011:158-161.

[10] Chaim Zins, *Conceptual approaches for defining data, information, and knowledge*. Journal of the American society for information science and technology, 2007, 58(4):479-493.

[11] A. Norta and L. Ma and Y. Duan, et al., *eContractual choreography-language properties towards cross-organizational business collaboration*. Journal of Internet Services and Applications, 2015, 6(1): 8.

[12] T. Aven, *A conceptual framework for linking risk and the elements of the datainformationknowledgewisdom (DIKW) hierarchy*. Reliability Engineering & System Safety, 2013, 111:3036.

[13] N. Lao and W. W. Cohen, *Relational retrieval using a combination of path-constrained random walks*. Machine Learning, 2010, 81(1):53-67.

[14] K. Xu, S. Reddy, Y. Feng, et al., *Question Answering on Freebase via Relation Extraction and Textual Evidence*. 2016.

[15] O. Serrat, *The five whys technique*. Asian Development Bank, 2009.

[16] G. Bellinger and D. Castro, *Data, information, knowledge, and wisdom*. European Congress of Rheumatology Eular. BMJ Publishing Group, 2004:276-276.