Empirical Software Engineering manuscript No.
(will be inserted by the editor)

The Structure and Dynamics of Knowledge Network in Domain-specific Q&A Sites: A Case Study of Stack Overflow

Deheng Ye · Zhenchang Xing · Nachiket Kapre

Received: date / Accepted: date

Abstract Programming-specific Q&A sites (e.g., Stack Overflow) are being used extensively by software developers for knowledge sharing and acquisition. Due to the cross-reference of questions and answers (referred to as *internal URL sharing* in this article [1]), knowledge is diffused in the Q&A site, forming a large knowledge network. In Stack Overflow, why do developers share URLs? How is the community feedback to the knowledge being shared? What are the unique topological and semantic properties of the resulting knowledge network in Stack Overflow? Has this knowledge network become stable? If so, how does it reach to stability? Answering these questions can help the software engineering community better understand the knowledge diffusion process in programming-specific Q&A sites like Stack Overflow, thereby enabling more effective knowledge sharing, knowledge use, and knowledge representation and search in the community. Previous work has focused on analyzing user activities in Q&A sites or mining the textual content of these sites. In this article, we present a methodology to analyze URL sharing activities in Stack Overflow. We use open coding method to analyze why users share URLs in Stack Overflow, and develop a set of quantitative analysis methods to study the structural and dynamic properties of the emergent knowledge network in Stack Overflow. We also identify system designs, community norms, and social behavior theories that help explain our empirical findings. Through this study, we obtain an in-depth understanding of the knowledge diffusion process in Stack Overflow and expose the implications of URL

Deheng Ye · Zhenchang Xing · Nachiket Kapre
School of Computer Engineering, Nanyang Technological University, Singapore
E-mail: ye0014ng@e.ntu.edu.sg

Zhenchang Xing
E-mail: zcxing@ntu.edu.sg

Nachiket Kapre
E-mail: nachiket@ntu.edu.sg

[1] Users may also reference URLs external to the Q&A site. In this paper, URL sharing refers to *internal URLs* within the Q&A site, unless otherwise stated.

sharing behavior for Q&A site design, developers who use crowdsourced knowledge in Stack Overflow, and future research on knowledge representation and search.

Keywords Mining software repositories · Crowdsourced knowledge · Domain-specific Q&A · URL sharing · Network analysis · Human factors

1 Introduction

Software engineering is a knowledge-intensive activity. Developers, whether novices or experts, frequently face different kinds of technical questions that cannot be solved using the knowledge they possess. In such situations, developers often seek help from their peers. With the advent of Web 2.0, domain-specific Q&A sites arose as the media for social information seeking for particular domains. As the most prominent Q&A site for computer programming (Mamykina et al. 2011), Stack Overflow receives hundreds of thousands of questions and answers each month from a community of millions of developers. Taken in aggregate, these questions and answers constitute a knowledge repository of developers' thoughts and needs (Barua et al. 2014).

Researchers have studied Stack Overflow to discover topics that developers discuss and the trends of those topics (Barua et al. 2014; Rosen and Shihab 2015), predict answer quality (Anderson et al. 2012) or user participation (Fugelstad et al. 2012), identity experts (Pal et al. 2012), recommend solutions to programming errors (Rahman et al. 2014) or tags to Stack Overflow questions (Wang et al. 2014), and analyze social interactions inside the cooperative community of Stack Overflow (Treude et al. 2011; Guerrouj et al. 2015), etc. In this work, we are concerned with the mechanism of knowledge diffusion in the community of Stack Overflow users. When asking a question or providing an answer in Stack Overflow, users can reference URLs of existing questions and answers. We refer to this URL reference behavior as *URL sharing*. Through URL sharing, the knowledge in existing questions and answers is diffused to a new question or answer.

We consider a question together with its entire set of answers and comments in Stack Overflow as a *knowledge unit* regarding some programming-specific issues. The behavior of URL sharing gives rise to an emergent *knowledge network*, i.e., a graph of knowledge units. Studying why users share URLs to existing knowledge units, the community feedback of shared knowledge units, the structure of the knowledge network formed and how it is formed can be a way to know more in-depth the knowledge diffusion process in Stack Overflow, thereby enabling more effective knowledge sharing in the community, improving knowledge accessibility and navigability for software developers, and inspiring future research on advanced knowledge representation and search.

There have been several studies on hyperlink networks such as Wikipedia (Preusse et al. 2013) and the general Web (Broder et al. 2000). However, the knowledge network formed by URL sharing activities in domain-specific Q&A sites is a rich, and so far untapped source of information. Due to the domain-specific nature, the knowledge network formed by URL sharing in Stack Overflow is much smaller than other types of hyperlink network, such as Wikipedia and the general Web. Furthermore, the way

that the knowledge network is formed in Stack Overflow is very different from that of other community-curated knowledge networks such as Wikipedia. Although both are formed through decentralized and collaborative efforts of a community of users, the structure of the Wikipedia knowledge network reflects the underlying structure of relevant knowledge. In contrast, individual knowledge units in Stack Overflow discuss ad-hoc questions posted by the developers. Knowledge units could be opportunistically referenced in other knowledge units based on the information value and relevance of the discussions and the community awareness of the presence of these discussions.

Considering the size of the knowledge network and the ad-hoc and opportunistic nature of URL sharing activities, would *stable* structure emerge in the knowledge network in Stack Overflow? By stable, we mean that the community has subconsciously developed implicit consensus about the relevance and relationships among knowledge units, and newly added relationships mostly reinforce already-present relationships. In this work, we propose a semi-automatic methodology (a mixture of qualitative and quantitative methods) for analyzing the knowledge network formed by URL sharing activities in Stack Overflow. Using the Stack Overflow data dump released on Mar. 16th, 2015, we aim to answer a series of research questions qualitatively and quantitatively: *why* Stack Overflow users share URLs, what *topological and semantic correlations* between knowledge units URL sharing activities create, whether the knowledge network formed has become *stable* (i.e., *scale-free* in terms of degree distributions), *how* the knowledge network has evolved over time, and what *factors* and *theories* can help explain the structure and dynamics of the knowledge network in Stack Overflow.

Our analysis allows us to make a number of interesting observations, including: Developers share URLs for various purposes, connecting different pieces of domain knowledge into a complex network of highly-recognized knowledge units; The knowledge network can be well partitioned into knowledge clusters that represent correlated topics; Although the size of the knowledge network in Stack Overflow keeps growing with the rapid growth of users and the fast development of software technologies, the knowledge units that developers care the most about are stable, which form the structural skeleton of the knowledge network; The knowledge network structure becomes stable quickly in Stack Overflow, but still, there are constant small structural changes with the burst of popularity of new software technologies or some newly recognized high-value knowledge units.

Our research methodology and empirical results can be useful to many parties. Q&A site designers can use our methodology to study the knowledge diffusion process in other domain-specific Q&A sites. They can also use our results to determine how to support knowledge sharing more effectively in Q&A process. The knowledge network can be exploited to better support developers to search and navigate the knowledge repository in Stack Overflow. Further, our results can be extended to develop entity-centric knowledge representation, making searching semantic-aware and more intuitive for users.

Our contributions in this paper are three-fold:

- This work is the first systematic study towards understanding the structure and dynamics of the knowledge network formed by URL sharing in programming-specific Q&A sites.
- We develop a semi-automatic methodology for studying the structure and dynamics of the knowledge network in programming-specific Q&A sites.
- We expose the implications of the knowledge network in Stack Overflow for Q&A site design, knowledge accessibility, and future research on knowledge representation and search.

2 Related Work

Our work is related to several research lines, including knowledge management in domain-specific Q&A sites like Stack Overflow, network analysis in social media, social behavior modeling, and empirical software engineering. Here we review some representative work in each of these research aspects.

Knowledge Management in Stack Overflow. Stack Overflow has become a programming knowledge base for software developers (Parnin et al. 2012). A community-curated list of publications using Stack Overflow data can be found at `http://meta.stackexchange.com/q/134495`.

Some researchers investigate topics and their trends in Stack Overflow (Allamanis and Sutton 2013; Barua et al. 2014; Bajaj et al. 2014; Rosen and Shihab 2015) using topic models. For example, Barua et al. (2014) study the general topics and their relationships and trends using LDA (Blei et al. 2003), while Rosen et al. (2015) focus on what mobile developers on Stack Overflow talk about. Some analyze the contents and social connections in Stack Overflow to perform typical data mining tasks, such as prediction on user participation (Fugelstad et al. 2012), expert identification (Pal et al. 2012)), tag recommendation (Wang et al. 2014), recommending solutions to programming errors (Rahman et al. 2014)), and how user activities correlate Q&A site properties (Anderson et al. 2012). Subramanian et al. link code snippets on Stack Overflow to API documentation (Subramanian et al. 2014). Others study developers' behavior and social dynamics inside the cooperative community of Stack Overflow (Vasilescu et al. 2014; Treude et al. 2011; Mamykina et al. 2011; Guerrouj et al. 2015; Wang et al. 2013; Sunshine et al. 2015; Squire 2015). For example, Guerrouj et al. (2015) examine the influence of App churn on Stack Overflow discussions of the corresponding App. Vasilescu et al. (2014) investigate R programmers' transition from traditional mail lists to Stack Exchange. Squire (2015) measures the utility of Stack Overflow for developer support. Mamykina et al. (2011) find that productive competition, community credibility and designers' involvement with regular users are important factors for Stack Overflow's success. There is also research investigating sentiments in Stack Overflow discussions (Novielli et al. 2015).

Compared with existing work, our study focuses on a unique perspective of Q&A behavior, i.e., URL sharing in Q&A sites. We reveal what kind of knowledge network URL sharing behavior brings to us and how Q&A site designers, software developers and software engineering researchers can benefit from an in-depth understanding of

the formation and characteristics of the programming-specific knowledge network in Stack Overflow. The recent work by Gómez et al. (2013) is the most similar work to ours. It also studies URL sharing in Stack Overflow, but it examines only some basic statistics of URL sharing activities, similar to the first two sub-questions (types of URLs shared and the size of the knowledge network) of our basic analysis in Section 4. Our work is the first attempt to systematically study the structure and dynamics of the knowledge network in Stack Overflow.

Analysis of Emergent Networks. The prosperity of social computing has incubated a number of emergent networks, such as user-user network in Q&A sites (Zhang et al. 2007) and in Twitter (Java et al. 2007), tag-tag network in social bookmarking systems (Halpin et al. 2007) and in Q&A sites (Wang et al. 2014), and entity-entity network extracted from Wikipedia (Cucerzan 2007). Different from these networks, we study domain-specific knowledge network formed by ad-hoc Q&A activities in Stack Overflow. Our study reveals that the structural properties of Stack Overflow's knowledge network are very different from those of other types of networks. For example, the indegree distribution of Stack Overflow's knowledge network follows a strict power law (Clauset et al. 2009), while its outdegree distribution does not. In contrast, both indegree and outdegree distributions of Twitter's user-user network follow a power law (Java et al. 2007). Broder et al. (2000) show that both indegree and outdegree distributions of the general Web follow a power law, while Meusel et al. (2014) analyze a more recent crawling data of the general Web and show that neither indegree distribution nor outdegree distribution follow a strict power law. The unique properties of Stack Overflow's knowledge network bring in the need for new knowledge search algorithms because network structure affects the performance of search algorithms as pointed out by Zhang et al. (2007).

Explanatory Models of Social Behaviors. Researchers develop semi-analytical models to understand the stability of stochastic process. In social computing, these models have been widely applied to explain social tagging behavior and the generation of folksonomies in social bookmarking systems, such as the Polya Urn model adapted in (Golder and Huberman 2006), improved Yule-Simon model (Cattuto et al. 2007), semantic imitation model (Fu et al. 2010) and a generative tagging model (Halpin et al. 2007). In contrast, we propose an explanatory model based on information value (Halpin et al. 2007) and preferential attachment (Barabási and Albert 1999) theories to explain how and why the structure of Stack Overflow's knowledge network emerges from its users' URL sharing behavior.

Empirical Rules in Software Engineering. Finally, our finding regarding the stability of the programming-specific knowledge network in Stack Overflow, in terms of the knowledge units developers care the most about and the reinforcement of existing relationships among knowledge units, is an enrichment to the discovered power laws in the empirical software engineering context, for example, the "small world" effect in reverse engineering (Hassan and Holt 2004), 20% code having 80% errors (Pressman 2010), and power laws in modules comprising software systems (Louridas et al. 2008).

3 Research Methodology

In this section, we define the knowledge network, describe the dataset we used and our research methods.

3.1 Definition of Knowledge Network

A *knowledge network* is a directed graph $KN(V, E)$. The node set V contains *knowledge units*. A knowledge unit consists of a question and its entire set of answers and other related information in the Q&A site. Formally, a knowledge unit is represented as a 3-tuple $\langle q, T, A\rangle$, where q is a question, T is the set of tags t_m ($1 \leq m \leq 5$ in Stack Overflow) of the question q, and A is the set of answers a_n ($n \geq 0$) to the question q. Each question or answer is referred to as a post in Stack Overflow. A post is represented as a 4-tuple $\langle id, body, comments, attributes\rangle$, where *id* is the post unique identifier, *body* is the content of the post, *comments* is the set of comments c_i ($i \geq 0$) associated with the post, *attributes* is the set of metrics of the post (e.g., votes, viewcount, favorites).

The edge set E contains the *associations* from one knowledge unit to another. Let ku_s and ku_t be the two knowledge units. There is a directed association from ku_s to ku_t denoted as $ku_s \rightarrow ku_t$, if and only if the posts of ku_s reference at least one post of ku_t. ku_s and ku_t are referred to as *source unit* and *target unit* respectively. We refer to the number of associations pointing to a knowledge unit *ku* as the *indegree* of *ku*, denoted as *indegree*(*ku*), and the number of associations leaving from the *ku* as the *outdegree* of *ku*, denoted as *outdegree*(*ku*). Indegree represents the degree of attention that a knowledge unit attracts in the community, and outdegree represents the amount of help that a knowledge unit seeks from others. If *indegree*(*ku*)+*outdegree*(*ku*)=0, *ku* is an isolated knowledge unit, i.e., *ku* is not associated with any other knowledge units.

Figure 1 gives an example of knowledge unit, association, and knowledge network. The left-hand question (ID=4547310) and all its answers comprise knowledge unit 1 (ku_1), and the right-hand question (ID=53513) and all its answers comprise knowledge unit 2 (ku_2). ku_1 discusses whether there is an *isempty()* method for iterating a stack in Python. ku_2 discusses how to check if a list is empty in Python. Clearly, these two knowledge units are correlated. In fact, in Answer 2 of ku_1, the answerer references the question of ku_2, which creates the association $ku_1 \rightarrow ku_2$. ku_1 and ku_2, together with their association, form a (smallest) knowledge network.

3.2 Dataset

We use the latest Stack Overflow data dump that contains the data for the period of Jul. 31st, 2008 to Mar. 16th, 2015. To build the knowledge network as defined above, we scan the *posts* and *comments* tables of the data dump to extract knowledge units, and parse the *post_links* table to determine associations between knowledge units. Although the focus in this paper is the internal URLs of Stack Overflow (i.e., the URLs

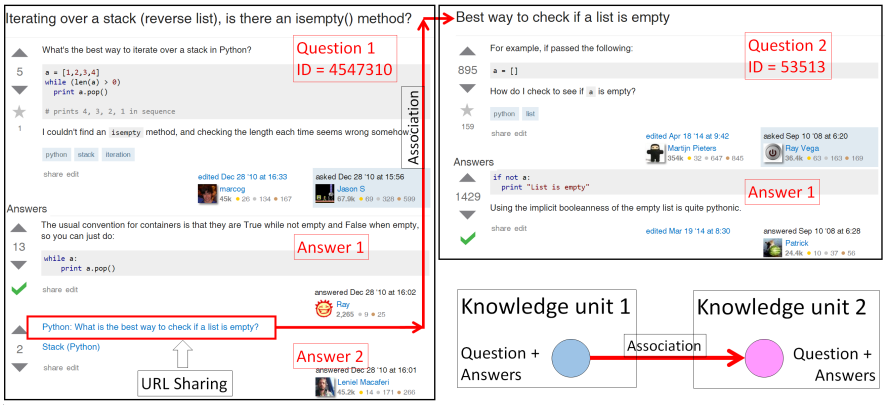

Fig. 1 An Example of the Knowledge Network

with domain name *stackoverflow.com*), we also extract URLs external to *stackoverflow.com* from the body and comments of the posts. This allows us to compare the sharing activities of internal URLs of Stack Overflow with that of external URLs.

3.3 Overview of Research Methods and Results

In Stack Overflow, knowledge units can have different relationships, for example, knowledge unit *A* is *similar to B*, *A* is *different from C*, *A* can *help solve* the question of *D*, and so on. These relationships are an integral part of the crowdsourced developer knowledge. We hypothesize that Stack Overflow users can recognize the underlying relationships among knowledge units and make them explicit by referencing URLs of relevant knowledge units when asking questions and providing answers. We also hypothesize that, as time goes by, such URL sharing behavior will create a stable knowledge network that captures the important relationships among knowledge units. We validate our hypotheses through answering a series of research questions as raised in our Introduction Section. We also integrate our quantitative methods as a web service at `http://knowledge-so.appspot.com`.

RQ1. *What are the basic characteristics of URL sharing activities in Stack Overflow?*

In Section 4, we show that sharing of Stack Overflow internal URLs occurs much more frequently than sharing of external URLs. A large knowledge network with 2,378,327 non-isolated knowledge units and 2,271,053 associations has been formed in Stack Overflow. Importantly, the community feedback of the non-isolated knowledge units is statistically significantly higher than that of the isolated knowledge units.

RQ2. *Why do developers share URLs extensively in Stack Overflow?*

In Section 5, we use open coding method to identify, name and categorize the underlying purposes of URL sharing when developers ask questions and provide answers. We further explain our open coding results using the community norms of

Stack Overflow.

RQ3. *How does developers' URL sharing behavior affect the semantic correlations of different knowledge units?* To be specific, do knowledge units correlate with other knowledge units with similar (or dissimilar) degrees or topics, which is known as assortative (or disassortative) mixing [2]?

In Section 6, we build the correlation profile (Maslov et al. 2004) for the knowledge network. We demonstrate that knowledge units in Stack Overflow exhibit neither assortative nor disassortative mixing (Newman 2002) in terms of degree correlations. We analyze the knowledge network reachable from a specific knowledge unit (results of the knowledge unit (question ID=53513) reported in this paper). We show that the knowledge units exhibit high modularity [3] and assortative mixing of semantic topics.

RQ4. *Is the knowledge network formed in Stack Overflow scale-free?*

In Section 7, we perform strict power law detection (Clauset et al. 2009), rather than simple linear fitting, on the indegree and outdegree distributions of Stack Overflow's knowledge units, respectively. We show that the indegree distribution of knowledge units follows power law strictly, while outdegree does not (although visually it can be fitted using a straight line). This suggests that the indegree distribution of the knowledge network has become scale-free.

RQ5. *How does the indegree of the knowledge network evolve into its current structure?*

In Section 8, we use Jensen-Shannon divergence (JSD) (Lin 1991) to study how a sequence of time-ordered degree distributions of knowledge units converge to the current distribution. We show that the JSD between consecutive indegree distributions of top N (results of N = 10, 50, 100 reported in this paper) knowledge units with highest indegrees converge very fast while it takes much longer time for the JSD to converge with respect to the final distribution. After the convergence to the stable distribution, there can still be small but constant changes in the distribution over time.

RQ6. *What factors can help explain the structure and dynamics of Stack Overflow's knowledge network?*

In Section 9, we propose an explanatory generative model based on information value (Halpin et al. 2007) and preferential attachment (Barabási and Albert 1999) theories, to explain and simulate how the knowledge network has been formed in Stack Overflow.

Finally and importantly, we recall the implications of each section in this paper and summarize and discuss these implications in detail in Section 10.

[2] In network science, assortative mixing is a bias in favor of connections between nodes with similar characteristics; disassortative mixing is a bias in favor of connections between dissimilarly characterized nodes. (Newman 2002)

[3] Modularity is one measure of network structure. A detected module is also called a community or a cluster. Networks with high modularity have dense connections between the nodes within modules but sparse connections between nodes in different modules.

Domain	Unique	Total ↓	%
stackoverflow.com	1,303,641	2,179,791	0.598
microsoft.com	292,001	838,332	0.348
jsfiddle.net	691,280	757,489	0.913
github.com	371,401	679,592	0.547
wikipedia.org	70,549	397,721	0.177
google.com	175,870	374,030	0.470
php.net	50,221	288,377	0.174
oracle.com	71,681	271,933	0.264
android.com	25,029	198,683	0.126
jquery.com	9,052	185,464	0.049

Table 1 Top 10 Referenced Domains

4 URL Sharing Activities in Stack Overflow — Basic Statistics

First, we check if a valuable knowledge network can emerge from URL sharing activities in Stack Overflow.

What types of URLs have been shared? We identify 6,261,227 unique URLs from all the posts and comments in Stack Overflow. These URLs have been referenced 13,068,707 times in total. Table 1 lists the top 10 most frequently referenced domain names ordered by the Total column. The Total column shows the number of times that the unique URLs (shown in Unique column) from a domain have been referenced. The column % gives the uniqueness percentage, i.e. Unique/Total.

We can see that the domain *stackoverflow.com* stands out for both unique and total references, which means that developers reference Stack Overflow much more than any other domain. More than one sixth of the unique URLs reference to Stack Overflows own knowledge units. Furthermore, Stack Overflow internal URLs have higher uniqueness percentage than that of external URLs, except *jsfiddle.net*. *jsfiddle.net* is an online platform for rapid prototyping, testing, and sharing Javscript, CSS, and HTML code. Stack Overflow users often reference code examples shared on *jsfiddle.net* instead of embedding long code fragments in Stack Overflow posts. These code examples are usually unique for answering specific questions.

What is the size of the knowledge network? As of Mar. 16th, 2015, Stack Overflow has 8,978,719 questions and 15,074,572 answers, i.e., 1 question has about 1.68 answers. The distribution of answers to questions is similar to what was observed in (Treude et al. 2011) (see Section 5 of their paper) [4]. As such, the knowledge network we built from the data dump consists of 8,978,719 knowledge units, among which 6,600,392 are isolated knowledge units. The rest 2,378,327 non-isolated knowledge units have in total 2,271,053 associations. The number of non-isolated units and their associations reported here are calculated using the *post_links* table of Stack Over-

[4] We do not list the detailed answer-to-question distribution, since the focus of this work is on those non-isolated knowledge units and their associations.

flow's official data dump, which consists of all Stack Overflow's internal references. The number of domain references shown in Table 1 is calculated by scanning Stack Overflow's *posts* table and *comments* table, which consist of all Stack Overflow's questions, answers, and comments. We have to point out that there exists minor inconsistencies among the officially released *posts* and *post_links* table. For example, table *post_links* says that post ID 2572 is linked to post ID 209329. However, both posts cannot be found from the *posts* table (they have been removed from Stack Overflow). As a result, the total number of associations (2,271,053) reported here is slightly different from the total number of references to *stackoverflow.com* reported in Table 1. For consistency, we will use the number 2,271,053 as the total number of associations in the remainder of this paper.

In the early stage of Stack Overflow, its knowledge base is small and knowledge units are isolated. As Stack Overflow accumulates a sufficiently large amount of knowledge units, similar or correlated knowledge units start appearing (Fourney and Morris 2013). Such an accumulation leads to a time delay between the posting of a knowledge unit and its linking to other knowledge units. For instance, the first association in Stack Overflow was generated on Apr. 26th, 2010, about 24 months after the launch of Stack Overflow. We will discuss this further in Section 9.1 and Section 9.2.

How is the community feedback to non-isolated knowledge units versus isolated knowledge units? In Stack Overflow, users can archive *favorite* questions; an asker can *accept* an answer to his question as best answer; users can *upvote* questions and answers. In these cases, we consider the corresponding knowledge unit as having received positive feedback to the knowledge embodied in the knowledge units by the community.

To compare the community feedback to non-isolated knowledge units with that of isolated units, we control the number of answers of a knowledge unit to be no less than 5 to guarantee the popularity of the knowledge unit selected. We then randomly select 1000 non-isolated knowledge units (group 1) and 1000 isolated knowledge units (group 2) with this level of popularity. For each knowledge unit in the two groups, we count the number of *votes* (sum of *upvotes-downvotes* of all its posts) and the number of users who *favorite* the question. If the knowledge unit has an accepted answer, we consider it as one *upvote* given by the asker. We measure the normalized community feedback to a knowledge unit as $votes+favorites$ divided by the number of posts in the knowledge unit.

We find that the median normalized community feedback to a non-isolated knowledge unit is 14.96, which is much higher than that of an isolated knowledge unit (1.67). We use median rather than mean as the metric, because the feedback metric has a right-skewed distribution. To build confidence with the hypothesis that non-isolated knowledge units are of higher positive feedback compared with isolated knowledge units, we perform Wilcoxon-Mann-Whitney test to the two groups of feedback. We get U=89621.5 and p=$5.36e^{-222}$. Since $p \ll 0.05$, it means that the feedback difference between the two groups of knowledge units is statistically significant.

Implications. Our study indicates that the community feedback of non-isolated

knowledge units is significantly better than that of the isolated knowledge units. This implies that non-isolated knowledge units have attracted much more user attentions when compared to isolated knowledge units. That is to say, the non-isolated knowledge units can potentially receive much more visits. The good community feedback of non-isolated units, together with their highly correlated nature which will be studied in Section 6, indicates the knowledge density of a non-isolated sub knowledge network is generally larger than an isolated one. This can potentially introduce navigation difficulties for novice developers, which will be further discussed in Section 10.

Overall, the basic analysis shows that URL sharing in Stack Overflow gives rise to a valuable knowledge network that warrants the further study.

5 The Purposes of URL Sharing Activities — Understanding Why Users Share

Stack Overflow's internal URLs are shared by developers for certain reasons. In this section, we infer the purposes of URL sharing from the content of the referenced posts and the discussion context where URLs are shared.

5.1 Open Coding Process

We randomly select 1,100 different URL references. 100 of these URL references are used to develop the coding schema, and then the rest 1,000 URL references are used to code the URL sharing purposes of Stack Overflow users. In our context, the overall population size is 2,271,053, which is the overall number of knowledge associations as reported in Section 4. Our sample size for coding is 1,000. Considering a confidence level of 95%, the margin of error caused by our sample size is as small as 3.1%. For each URL reference, we retrieve the source knowledge unit where the URL is referenced and the target knowledge unit that the URL links to. We then read the posts and comments of the two knowledge units to identify, name and categorize why the user shares a particular URL in the source knowledge unit. This open coding process involves 3 stages and is performed by 3 persons (P1, P2, P3) who are all from computer science background with 5+ years of programming experience.

In Stage 1, P1 first develop a draft coding schema (i.e., categories) of purposes of URL sharing using 100 randomly selected URL references. Then P2 and P3 use the draft coding schema to categorize the same 100 URL references collaboratively, during which the draft schema is revised and refined. At the end of Stage 1, we obtain 5 categories of purposes of URL sharing. In Stage 2, P1 and P2 apply the resulting coding schema of Stage 1 to categorize another 1,000 URL references independently. They are instructed to take notes regarding the deficiency and ambiguity of the coding schema for categorizing certain URL references. The inter-rater agreement (Cohen's kappa) of this stage is 0.82. In Stage 3, P1, P2 and P3 discuss the coding results obtained in Stage 2 to revolve the disagreements between P1 and P2, and to revise the coding schema to resolve schema deficiencies and ambiguities. At the end of

Category	Frequency
1.Reference information for problem solving	31.4%
2.Reference existing answers	20.5% (2a: 13.9%, 2b: 6.6%)
3.Visited but not helpful	19.4%
4.Recommend related information	25.6%
5.Others	3.1%

Table 2 Purposes of URL Sharing

Stage 3, we obtain the final coding schema and the final coding results of 1,100 URL references.

5.2 Coding Schema

The final coding schema contains five general categories of purposes of URL sharing in Stack Overflow as follows:

1. Reference information that can help to solve the problem.
   There are four sub-categories: (a) The referenced post explains certain concept, approach, background knowledge, or the URL sharer's claim. (b) The referenced post gives working examples, e.g., code snippets. (c) The referenced post covers a sub-step for solving a complex problem. (d) The answer is adapted from the referenced post.
2. Reference existing answers to a question
   There are two sub-categories: (a) Confirmed duplicate. The question of the source knowledge unit is explicitly marked as duplicate by users with high reputation. Such questions can be easily recognized by the [duplicate] marker at the end of the question title. See an example at `http://stackoverflow.com/q/1725517`. (b) Possible duplicate. The URL sharer suggests that the question is duplicate to the referenced posts. However, the question asker and other users disagree with the suggestion. Possible duplicates can be judged by the comments or edit history of the question. See an example at `http://stackoverflow.com/q/23856165`.
3. Reference visited posts that cannot solve the problem.
   Before asking a question, a user may "search and research" existing posts. If he does not find answers, he may reference those visited posts to let the community know what he had tried but failed. Such URL references are usually contained in the body or the comments of the question. See an example at `http://stackoverflow.com/q/23935102`.
4. Recommend related information.
   The referenced post provides related information, although it does not directly answer the question.
5. Others.
   This category contains URL references that do not fit into the above 4 categories. For example, when a user asks a question with very poor descriptions, another

user may comment that the question is not clear and reference a good example that demonstrate how to ask question properly. See the reference embedded in the comments to this question `http://stackoverflow.com/q/22693431`.

While some of the above categories are distinctive (e.g., category **3** and category **2a**), we acknowledge that there are literal overlaps among some other categories. For example, a URL referencing a related concept can be category **1a** or category **4**. We regulate that if a URL reference is surrounded with problem-solving descriptions (e.g., `http://stackoverflow.com/a/13799882/2728388`), we categorize the URL reference as category **1a**. If the URL is suggested as "related" or "similar", but the URL sharer provides no or little explanation (e.g., the URL in the comment of this post `http://stackoverflow.com/q/28902097/2728388`), we categorize the URL reference as category **4**.

5.3 Coding Results

Table 2 summarizes our final coding results. As we can see, Stack Overflow users share URLs mainly to reference information that can help to solve a complex problem (category **1**) and to recommend related information that may be potentially helpful (category **4**). The other significant reason for URL sharing is to inform the community existing posts that one had tried but found not useful (category **3**). Although category **2** (reference existing answers) accounts for 20.4% of URL references, the real confirmed duplicates (category **2a**) are only 13.9%, while the rest 6.6% are possible duplicates.

Threats to validity. First, it is almost impossible to develop a perfect coding schema with no overlaps among the categories. Second, we only label 1,100 randomly-sampled URL references. Further studies are required to confirm and generalize our coding schema. However, we believe our 3-stage open coding process and the small margin of error caused by our sample size should produce reliable coding results.

5.4 How do the community norms of Stack Overflow shape users' URL sharing behavior?

Stack Overflow has a continuous community effort to identify and mark duplicate questions. Therefore, certain percentage of URLs links to existing questions that already have answers to the duplicate questions (category **2a**, 13.9%). However, as Stack Overflow encourages users "search and research" existing knowledge base before posting a new question [5], the percentage of confirmed duplicate questions is not very high. Furthermore, Stack Overflow encourages users sharing their "search and research" outcomes in their questions (Fourney and Morris 2013) so that the community knows what have been tried. This practice results in 19.4% (category **3**) of URL references in our sampled dataset. Finally, Stack Overflow accepts only

[5] How do I ask a good question? `http://stackoverflow.com/help/how-to-ask`

programming-related questions and has a strict policy on spam URLs [6]. As such, users have to be responsible for sharing relevant URLs. This explains why we do not find any junk URLs in our sampled dataset.

Implications. URL sharing connects different pieces of domain knowledge for various purposes. It also creates social contacts among users who hold related knowledge. Understanding the purposes of URL sharing and associating these purposes with user behaviors that Stack Overflow encourages inspire new search algorithms and bookmarking mechanisms to facilitate knowledge management and sharing in Q&A activities, which will be discussed further in Section 10.

6 Topological and Semantic Correlation — Analyzing Knowledge Interaction

Analyzing the purposes of URL sharing suggests that referencing or recommending related or duplicate information (i.e., category 1, category 2, and category 4 in Table 2) is the main reason for URL sharing. How will these purposes affect the topological correlations (i.e., degrees of knowledge units) or semantical correlations (i.e., semantics of the contents of knowledge units) among associated knowledge units? Will URL sharing behavior results in assortative (or disassortative) mixing (Newman 2002) of knowledge units in terms of their degree similarity (or dissimilarity) and semantic similarity (or dissimilarity)?

6.1 Analyzing Degree Correlations

Background. Given an association $ku_s \rightarrow ku_t$, the source knowledge unit ku_s seeks help and the target knowledge unit ku_t provides help. Generally, it holds that the most referenced knowledge units are most helpful for developers. Thus, if we regard the knowledge unit as a knowledge "helper", the indegree of the knowledge unit reflects the helper's "expertise" level. Degree correlations reveal the similarity (or dissimilarity) between the "expertise" levels of source and target knowledge units.

Method. To study the degree correlations of knowledge units, we construct the correlation profile (Maslov et al. 2004) of knowledge units. A correlation profile summarizes the relative frequency of the combination of expertise-levels between help seekers ku_s and help providers ku_t in the knowledge network. Figure 2 shows the resulting correlation profile in contour plot. The x axis and y axis represent the expertise-level of source and target knowledge units respectively, based on their indegree values. The z axis represents the relative frequency of the combination of expertise-levels between "help seeker" ku_s and "help provider" ku_t, and its value is quantified in the color bar on the right. The larger the z value, the more frequently the corresponding combination of expertise-levels appears.

[6] How does Stack Overflow handle spam? `http://meta.stackexchange.com/q/2765`

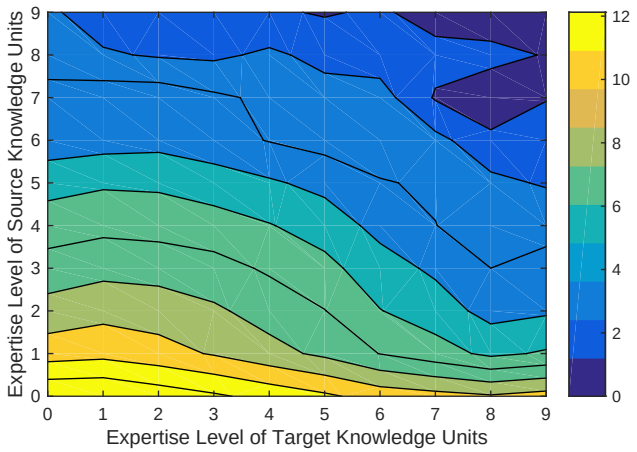

Fig. 2 Correlation Profile of Knowledge Units

We perform logarithmical data binning to indegree values and frequencies of expertise-level combinations. In our knowledge network, the smallest indegree value is 0 (i.e., knowledge units with only outgoing associations). Thus, we add 1 to all indegree values to be able to calculate logarithm. We obtain 10 expertise-levels for knowledge units using natural log transformation. The smallest indegree maps to expertise-level 0, and the largest indegree 6447 becomes 8.77, mapping to expertise-level 9. We accumulate the number of knowledge units falling into each expertise-level, and compute the relative frequencies of expertise-level combinations. The relative frequencies expertise-level combinations map to 13 levels after logarithmical data binning.

Results. We can understand Figure 2 from three perspectives. First, looking at the overall distribution of expertise-level combinations, we can see that the dominant combinations (the yellow and orange region) are the knowledge units with small indegree values, i.e., "novices". Novices seek a large amount of help but also provide much help. They contribute the most to the knowledge diffusion. On the other hand, there are very few associations among knowledge units with large indegree values (the dark blue region), i.e., "experts".

Second, looking at the *y-axis* which represent source knowledge units referencing others, we can see that source units with expertise-level 0-1 mainly reference to target units with low and middle expertise-level 0-5, sometimes reference target units with high expertise-level 6-9. Source units with middle expertise-level 2-5 sometimes reference to target units with expertise-level 0-7. Source units with high expertise-level 6-9 rarely reference other knowledge units. This follows our intuition that "experts" normally provide help, not seek help.

Third, looking at the *x-axis* which represents target knowledge units being referenced by others, similar expertise correlation analysis can be applied. Overall, we can see that with the increase of the expertise-level of the target knowledge units, the frequency of references drops drastically.

Implications. Knowledge units show neither assortative mixing nor disassortative mixing in terms of degree similarity or dissimilarity. That is, degree similarity or dissimilarity is not a decisive factor in URL sharing. This structural property, together with knowledge units' degree distribution properties discussed later, informs Q&A site designers that general search algorithm should be adapted to improve knowledge search in Stack Overflow, because network structure is one potential factor that affects the performance of search algorithms (e.g., PageRank), as pointed out by Zhang et al. (2007).

6.2 Analyzing Semantic Correlations

Background. We exemplify the analysis of semantic correlations using the sub knowledge network reachable from the "seed" knowledge unit (question ID=53513), i.e., we traverse the knowledge network staring from the seed and collect knowledge units that are directly and indirectly linked to the seed. This seed knowledge unit discusses *how to check if a list is empty in Python* [7]. We aim to understand the following two questions:

- How semantically similar are the seed knowledge unit and the knowledge units directly associated with it?
- In the knowledge network reachable from the seed knowledge unit, will there be latent knowledge clusters, if so, how these knowledge clusters are connected and how semantically similar these knowledge clusters are?

6.2.1 Directly Associated Knowledge Units

Method. The body and comments of the posts in a knowledge unit are first transformed into a text document. We pre-process the text documents by filtering away stop words, discarding code snippets, and removing all HTML tags. The text documents are then transformed into tf-idf (term frequency-inverse document frequency) vectors (Manning et al. 2008). The semantic similarity between the two knowledge units are measured by the cosine similarity of their tf-idf vectors.

Results. Figure 3 shows the seed knowledge unit (the central node) and all its 12 directly associated knowledge units. In this example, all the 12 knowledge units reference the seed knowledge unit. We label the uses of these 12 URL references according to coding schema in Table 2. The results are: 2 knowledge units reference the seed unit as it helps solve their problems (category 1). 4 knowledge units reference the seed unit as existing answers to their questions (category 2), including 3 knowledge units explicitly marked as [duplicate] (category 2a) and 1 knowledge unit suggested as possible duplicate (category 2b). 2 knowledge units recommend the seed knowledge unit as related information (category 4). 4 knowledge units list the seed knowledge units as visited but not helpful to their questions (category 3).

[7] Visit the web service we built `http://knowledge-so.appspot.com` to see more examples.

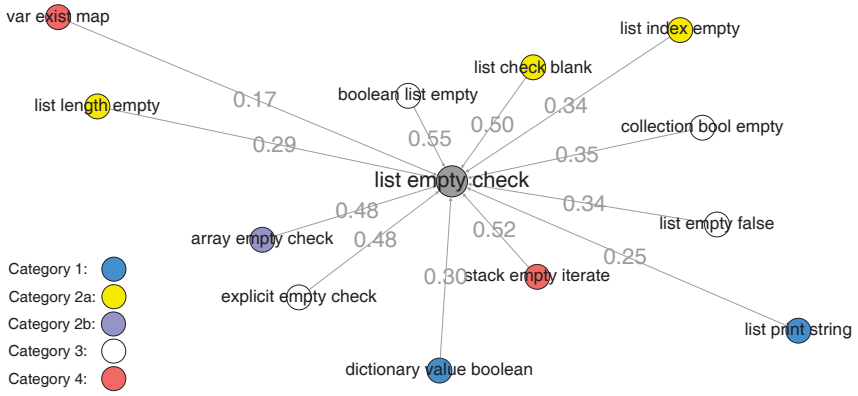

Fig. 3 Directly Associated Knowledge Units. The central node (seed knowledge unit) discusses *how to check if a list is empty in Python*. The color of its directly associated nodes denotes the URL sharing purpose according to Table 2. The three words on top of the nodes are the top words given by tf-idf. The length of an edge is inversely proportional to the cosine similarity between the two knowledge units it connects.

The length of an edge is inversely proportional to the semantic similarity between the two knowledge units it connects. We label each knowledge unit with the three words (excluding the word "python") that have the highest tf-idf values, and label the edges with the cosine similarities between the two knowledge units. We can see that not all knowledge units are of equal semantic similarity to the seed knowledge unit. The smallest cosine similarity is 0.17, and the largest is 0.55. Some knowledge units discuss topics very similar to that of the seed knowledge unit, such as those labeled with words "list", "empty". Others discuss more remotely related topics such as "imap", "var", "exist".

Implications. Although directly associated knowledge units exhibit assortative mixing of semantic topics, degree of semantic similarities between associated knowledge units do not correlate with specific purposes of URL sharing. Surprisingly, knowledge units referencing the seed knowledge unit as "visited but not helpful" have relatively higher semantic similarities with the seed unit than many other knowledge units with other categories of purposes. These results call for innovations in non-keywords-based knowledge search systems that could more accurately assess the information value and relevance of the knowledge units, beyond simple term frequency similarity.

6.2.2 Knowledge Clusters

Method. We use breadth-first graph traversal to collect all the knowledge units reachable from the seed knowledge unit. Surprisingly, the resulting sub knowledge network is of huge size, containing 1.04 million knowledge units (43.7% of all non-isolated knowledge units) and 1.32 million associations (57.8% of all associations). In order to unveil the underlying topological and semantic correlations in the knowledge network reachable from the seed, we limit the breadth-first traversal to obtain a sub knowledge network of 300 knowledge units and 369 associations.

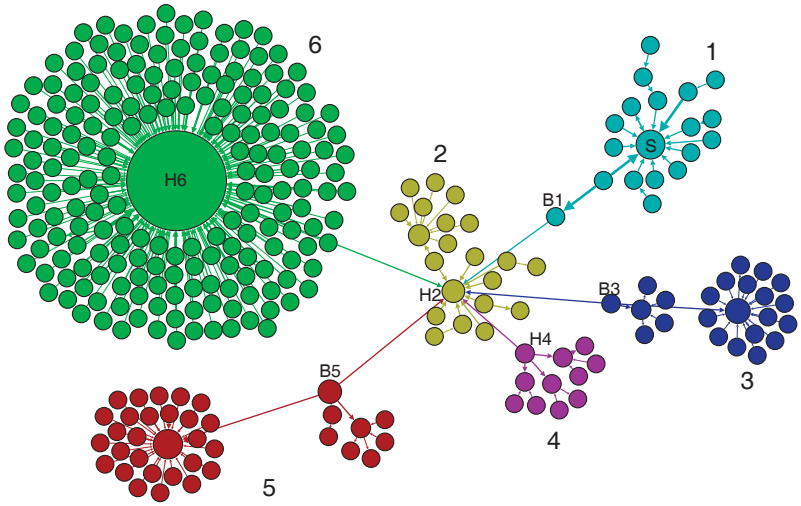

Fig. 4 Clustering Results of a Sub Knowledge Network Reachable from the Seed. The seed knowledge unit 'S' is the same one as shown in Figure 3. Nodes in the same cluster share the same color.

TopicID	Dirichlet Parameter	Top Words
0	119.49245	list empty python check false boolean value true object array
1	12.8508	index answer reason execute len key blank type list python
2	9.77833	variable implicit collect python effect nonzero complex implement

Table 3 Topics of Knowledge Cluster 1

ClusterID	Top Words
1	list empty python check false boolean value true object array
2	python check test method list catch key index string find
3	python attribute way check object method function exist type
4	type python function check method argument class code differ
5	valid check file window function way string directory exist
6	input user number integer valid python string error way check

Table 4 Top 1 Topic of the Six Detected Knowledge Clusters

To find out whether knowledge units form latent knowledge clusters, we perform community detection in the resulting sub knowledge network using the *Louvain Method* (Blondel et al. 2008) implemented in Gephi (Bastian et al. 2009). The *Louvain Method* adopts modularity maximization iteratively to partition the graph into a finite number of disjoint clusters that are considered as communities. Then, we apply the Latent Dirichlet Allocation (LDA) (Blei et al. 2003) topic modeling to extract topics for the detected clusters of knowledge units. For each knowledge cluster, we detect 10 topics and run LDA for 1000 iterations.

Results. The community detection result is shown in Figure 4. The seed knowledge unit is marked as *S*. The size of the nodes is proportional to their indegree. Nodes in the same community (i.e., knowledge cluster) share the same color. We have two interesting observations: (i) Knowledge units in this resulting sub knowledge network can be very well partitioned into six communities. This indicates that Stack Overflow's knowledge network exhibits high degree of modularity (modularity coefficient is 0.528 in this subnetwork). (ii) Knowledge clusters are connected to one another through either some hub nodes such as S, H2, H4 and H6, or some betweenness nodes such as B1, B3 and B5. The hub nodes are the knowledge units with high indegree, while the betweenness nodes represent the knowledge that are common in two or more knowledge clusters. These hub nodes and betweenness nodes form a structure skeleton for this sub knowledge network.

Table 3 summarizes the top 3 topics with the largest Dirichlet parameter for the knowledge cluster 1 that contains the knowledge units in Figure 3. We can see that the first topic dominates and should be considered as the main topic. The top words of the first topic, such as "python", "list", "empty", "check" match particularly well with the semantics of the knowledge units shown in Figure 3. Table 4 lists the top 1 topic with the largest Dirichlet parameter of all the six knowledge clusters. We can see the topics of these six knowledge clusters are similar to certain extent, but are of apparent difference as well. Directly connected knowledge clusters exhibit assortative mixing. As the distance between knowledge clusters increases, the semantic similarity of knowledge clusters decreases.

To understand why this sub knowledge network exhibits these six knowledge clusters, we analyze users' URL sharing purposes among the seed unit, hub units, and betweenness units. Take the knowledge unit S, B1, H2 and H6 as an example. The seed unit S discusses *best way to check if a list is empty in Python*. The betweenness unit B1 (ID=11786157) discusses how to write Python code such that *in a nested if-statements, if certain list indexes exist, then run a function*. The knowledge unit (ID=23365216) between B1 and S can be easily solved by combining the answers in S and B1, and thus is marked by the community as [duplicate] of both S and B1.

In an answer of B1, one user mentions *try block* is the perfect candidate for handling the issue, which naturally introduces the knowledge of *exceptions* in Python. Another user suggests that to better understand Python *exceptions*, one has to know *EAFP programming principle* in Python. Thus, this user recommends a URL referencing the knowledge unit H2 (ID=11360858) that contains thorough knowledge on *EAFP principle* from its definition to usage. As such, the two knowledge clusters centred on S and H2 are bridged by the unit B1 that discusses issues relevant the knowledge of both clusters.

The knowledge cluster centred on H6 (ID=23294658) is the largest cluster in this sub-network. The question of H6 asks how to write Python code *asking the user for input until they give a valid response*. One user gives a comprehensive answer (ID=2728388), which not only answers the question, but also covers a wide range of relevant Python knowledge such as *while-loop*, *exception handling*, and *user input checking*. The discussion on exception handling in this answer recommends the unit H2 for further information on *EAFP principle*, which connects the two clusters.

All the discussions in this answer are exemplified with working code snippets. As a result, this answer is highly recognized by the community and is regarded as a *community wiki level* answer. Thus, in many knowledge units regarding Python *while-loop*, *exception handling*, and *user input checking*, users recommend H6 as it likely has an answer to the questions. Interestingly, many knowledge units that were asked before H6 but received poor answers (e.g., question 10563920 and 24174288) are marked as [duplicate] of H6.

Implications. Navigating a large knowledge network can be challenging due to the reachability of the network and assortative mixing of semantic topics among associated knowledge units. High modularity of the knowledge network, semantic topics of knowledge clusters, and the structure skeleton formed by hub and betweenness knowledge units can be leveraged to improve the navigability of the knowledge network when knowledge users perform user-centered exploratory search (Marchionini 2006) in the knowledge base.

7 The Structure of Knowledge Network — Detecting Stability

We have seen how knowledge units correlate. In particular, we observed an emergent structure skeleton in the knowledge network formed in Stack Overflow. As this knowledge network is generated by a large community of users with little central coordination, we wonder if the structure of the knowledge network has become stable, in terms of the nodes' degree distributions, which motivates the study in this section.

Background. The structure of a complex network is stable if the network is scale-free. A scale-free network should produce a power law degree distribution. A power law defines a functional relationship between two scalar quantities x and y:

$$y \propto x^{-\alpha} \tag{1}$$

where α is the scaling parameter. When performing log transformation, $log(y)$ becomes a linear function of $log(x)$. It is important to mention that real-world empirical distributions rarely fit into power law over the whole range of x values. Rather, observed power laws hold only over a limited range (Wagner et al. 2014). Hence, we have to determine a minimum value x_{min} so that the distribution of y follows a power law with $x \geq x_{min}$.

To detect a power law, a traditional method is to use linear fitting for the linear function between $log(y)$ and $log(x)$ (Broder et al. 2000). Recent works (Clauset et al. 2009; Meusel et al. 2014) suggest that simple linear fitting can lead to unreliable results. Thus, we adopt the strict power law detection method proposed in (Clauset et al. 2009).

Method. We study indegree and outdegree distribution of Stack Overflow's knowledge network separately. Degree distribution is a function describing the number of knowledge units (i.e., y) in the network with a particular degree (i.e., x). We first

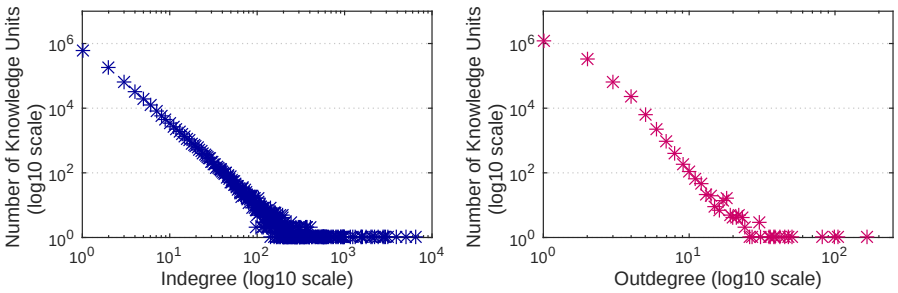

Fig. 5 Degree Distributions of Stack Overflow's Knowledge Network

transform the degree distribution of knowledge units into Complementary Cumulated Distribution Function (CCDF). For a degree value x, CCDF cumulates all the knowledge units whose degree is no less than x and divide the number of such knowledge units by the total number of knowledge units to get a probability, represented as $P(X \geq x)$. As suggested in (Clauset et al. 2009), we use *maximum likelihood estimation* to estimate α, and we minimize *Kolmogorov-Smirnov (KS) statistic*, which quantifies a distance between the observed CCDF data and the theoretical power law model in Eq. 1, to find x_{min}.

We evaluate the power law fitting using statistical hypothesis testing. The null hypothesis is that the indegree (or outdegree) distribution follows power law. The alternative hypothesis is the indegree (or outdegree) distribution does not follow power law. First we generate a large number (10,000) of power law distributed synthetic data sets using the computed x_{min} and α. For each synthetic data set, we fit the data to its theoretical power law model in Eq. 1 and calculate the corresponding *KS statistic*. Then we count the fraction of all the synthetic data sets whose resulting *KS statistics* are greater than the *KS statistics* of our empirical indegree (or outdegree) distribution data. This fraction is the *p-value* that quantifies the plausibility of the hypothesis. If *p-value* is no greater than the significance level, we reject the null hypothesis and accept the alternative hypothesis.

Results. Figure 5 gives indegree and outdegree distributions of the knowledge network. *x-axis* shows the degree value of the knowledge units, *y-axis* is the number of knowledge units with a particular degree. Visually both indegree and outdegree distributions seem to exhibit power laws when plotted in *log-log* space.

In Figure 6, we show the indegree (and outdegree) CCDF distribution (blue asterisks) and the power law fitting line (black line) obtained. For indegree, the power law fitting reports α=2.6 and x_{min}=26. We see the power law fitting line coincides with the indegree CCDF distribution data very well. For outdegree, the power law fitting reports α=2.8 and x_{min}=1. However, we see that the fitting line deviates from the outdegree CCDF distribution data.

The hypothesis testing shows that the *p-value* for indegree distribution is 0.71, while it is 0.00 for outdegree distribution. Therefore, we accept the null hypothesis for indegree distribution. That is, the indegree distribution of the knowledge network

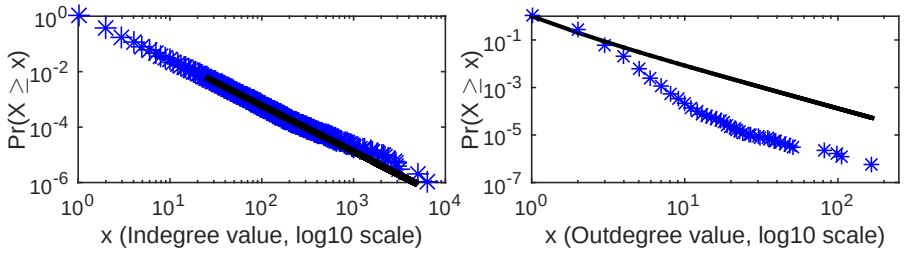

Fig. 6 Indegree and Outdegree CCDF

follows power law. In contrast, we reject the null hypothesis for outdegree distribution and accept the alternative hypothesis. That is, the outdegree distribution of the knowledge network does not follow power law.

Implications. The structure of the knowledge network with respect to indegree distribution is scale free. That is, regardless of how large the network grows, the shape of indegree distribution remains largely the same and thus stable. This structural stability can be explained using preferential attachment theory (Barabási and Albert 1999), i.e., "the rich get richer", as will be further covered in 9.3. For the knowledge community that Stack Overflow serves, this finding suggests that the knowledge units that developers care the most about are largely stable, although the size of the knowledge network keeps growing with the rapid growth of the users and the fast development of programming technologies. Furthermore, the indegree's power law distribution implies that a small proportion of the knowledge units is attracting a large proportion of users' attention. These knowledge units are usually hub and betweenness units that form the structure skeleton of the knowledge network. This structure skeleton can be leveraged to improve knowledge accessibility and navigability when knowledge users perform user-centered exploratory search (Marchionini 2006) in the knowledge base.

8 The Dynamics of Knowledge Network — Revealing Evolution Trend

In the previous section, we demonstrated that the indegree distribution of the knowledge network in Stack Overflow has reached stability. A stable knowledge network does not appear randomly and suddenly. Instead, it results from the distributive knowledge of the community over time. In this section, we continue to investigate how the structure of the knowledge network with respect to its indegree distribution has been gradually formed from users' day-to-day URL sharing activities.

Background. The knowledge network in Stack Overflow has gone through an evolving process from the initial chaos to the current stability. We want to quantify the growth of the knowledge network over time. To that end, we divide its evolving pro-

cess into a sequence of time-ordered *states* and measure the *differences* between these states.

In this work, we represent a state as the indegree distribution of rank-ordered knowledge units with the highest indegree values in the knowledge network at a time point. These top-ranked knowledge units represent hub and betweenness nodes that form the structure skeleton of the knowledge network. The *difference* between the two states is measured as the divergence between the two distributions. When the divergence becomes or is close to zero, it implies the emergence of structural stability in the knowledge network.

Method. We get a sequence of states at a sequence of sampled time points. Because the number of new associations created over a fixed period of time is uneven, we calculate the indegree distribution of ranked-ordered knowledge units in the knowledge network at flexible time points when a fixed number of new associations (10,000 in this work) has been created (similar to (Wagner et al. 2014)), rather than computing distributions at fixed time intervals as in (Halpin et al. 2007). As a result, we get 227 states at 227 time points (i.e., 2,271,053 associations divided by 10,000).

At each sampled point, we select the top N ranked knowledge units with the highest indegree values, and transform these indegree values into a vector of N probabilities. The probability of the *ith* ranked knowledge units, denoted as $P(i)$, is computed as the indegree value of the knowledge unit at the *ith* ranked position divided by the sum of the indegree values of all the top N knowledge units. This probability vector represents the indegree distribution of a particular state.

We adopt the Jensen-Shannon divergence (JSD) to measure the divergence between the indegree distributions of the two states. JSD is based on the Kullback-Leibler divergence (KLD) but is symmetric and always computable (Lin 1991). Given two probability vectors P and Q, with M as their average vector, i.e., $\frac{1}{2}(P + Q)$, the JSD between P and Q is defined as:

$$\mathrm{JSD}(P \parallel Q) = \frac{1}{2}\mathrm{KLD}(P \parallel M) + \frac{1}{2}\mathrm{KLD}(Q \parallel M) \tag{2}$$

where

$$\mathrm{KLD}(P \parallel Q) = \sum_i P(i) \log_2(\frac{P(i)}{Q(i)}) \tag{3}$$

Inspired by Halpin et al. (2007), we implement two complementary ways to examine how the indegree distribution of the knowledge network has evolved into a scale-free distribution.

- We calculate the JS divergence between the distributions of every two consecutive states.
- We calculate the JS divergence between the distributions at each state and the final distribution at the last state (i.e., Mar. 16th, 2015).

Results. Figure 7 shows JS divergence results between consecutive states. Here we show the results for the top 10, top 50, and top 100 knowledge units with the highest

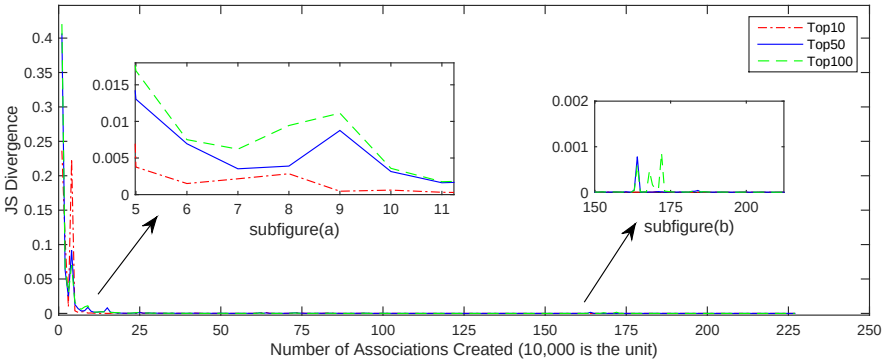

Fig. 7 JS Divergence between Consecutive States

indegree values [8]. For these three cases, we can see that they all start with high JSD values, which indicates that initial indegree distributions of the knowledge network are divergent between two consecutive states. After some changes in the beginning, the JSD values drop close to zero rapidly, before the total number of associations reaches 200,000, which is less than 10% of the total associations in the latest knowledge network. This happened about 12 months after the first association was created.

We zoom into the slopes before the JSD values converge (subfigure(a)). We find that the convergence speed for the top 10 knowledge units is the fastest, and the top 100 is the slowest, while the top 50 falls into the middle. This phenomenon is what we can expect. The more top knowledge units under investigation, the more time it requires for the community to develop some implicit consensus on which knowledge units are more worth referencing and sharing. Furthermore, even after the distribution converges to a stable distribution, there can still be small changes in the distribution. The subfigure(b) in Figure 7 shows an example, which reflects the burst of references to some knowledge units with high information value (e.g., the question (ID=14220321) that discusses AJAX in JavaScript).

Figure 8 shows JS divergence results between the distribution at each state and the final distribution. We can see that the divergence between the initial distributions and the final distribution is much larger than the divergences between the two consecutive initial distributions. Furthermore, compared with the convergence of the consecutive distributions, the convergence to the final distribution takes much longer time, after about 1,300,000 associations have been created. This happened about 39 months after the first association was created.

Implications. The indegree distributions of rank-ordered knowledge units converge quickly. Then, small changes between consecutive states accumulate over time and finally make the indegree distribution stable. The convergence and changes of indegree distribution of the knowledge network can be explained using information value theory (Halpin et al. 2007), as will be covered in Section 9.2.

[8] The results of other top N knowledge units are similar. Visit our web service at `http://knowledge-so.appspot.com` to see more results.

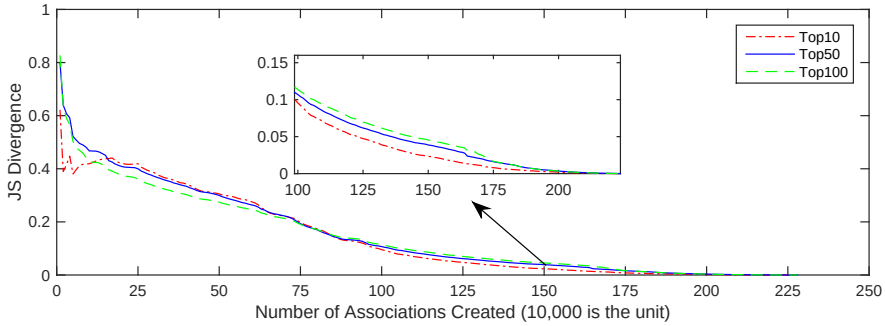

Fig. 8 JS Divergence between Each State and the Last State

9 An Explanatory Generative Model

From Section 4 to Section 8, we have progressively studied various aspects of Stack Overflow's knowledge network, from its basic statistics, reasons of its formulation from the view of developers, how knowledge units correlate, whether stability has been reached, to how the knowledge network evolves.

In this section, to integrate previous findings, we propose an explanatory generative model that models the factors that affect developers' URL sharing behavior in Stack Overflow. This model helps explain our empirical findings of the structure and dynamics of the knowledge network emerged from URL sharing behavior. Specifically, in Section 9.1, 9.2 and 9.3, we cover the three factors that affect the degree distribution of the knowledge network in Stack Overflow. In Section 9.4, we combine these factors into a model to simulate the process of Stack Overflow's knowledge diffusion process.

9.1 Baseline Probability — "To Reference or Not to Reference"

As Stack Overflow accumulates a large amount of knowledge units, similar or correlated knowledge units start appearing (Fourney and Morris 2013). As such, when asking a question or providing an answer in Stack Overflow, there will be a baseline probability P_r for a user to reference an existing knowledge unit. This baseline probability is determined by the priori knowledge of the users. A priori knowledge refers to what the users already know of and would like to share with others.

Users may obtain a priori knowledge by search, system recommendation, or bookmarking. Users can always "search and research" existing knowledge base before asking a question or providing an answer. If no desired knowledge units are found, the user can choose to ask his own Stack Overflow question. While an asker is entering the question title, Stack Overflow can dynamically prompt some existing questions that may already have answers to the new question that the user wants to ask. If the asker finds good answers in these existing questions, he does not even need to post the new question. Otherwise, he can reference those existing posts in

his question to tell others what he already tried. Stack Overflow also allows users to bookmark a question as their "*Favorites*" if they find useful knowledge embodied in the question and its answers. They may reference to this bookmarked question in their answer if it can help with other questions.

9.2 Information Value — "Create or Change the Distribution"

Assume that a user decides to reference an existing knowledge unit. With a probability P_a, he references an existing knowledge unit based on the information value (Halpin et al. 2007) of the knowledge unit. A knowledge unit will be referenced with a probability P_{iv} proportional to its information value. In the context of developers' help-seeking process, the information value of a particular knowledge unit is judged by how much useful information is contained in the knowledge unit that can facilitate a developer to get the desired information he is searching for. The information value of a knowledge unit can be estimated using the attributes of the knowledge unit such as *votes*, *favorites*, and *best answer*, or using the semantics of the knowledge unit such as topics or programming-related entities and their relations.

The probability $P_a \times P_{iv}$ creates the initial degree distribution of knowledge units based on the information value of the knowledge units. At the early stage of Stack Overflow, all the knowledge units are either not referenced or only referenced a few times. At this stage, knowledge units with higher information value will have higher probability to be referenced. As such, a power law degree distribution of knowledge units could emerge after certain amount of URL sharing activities.

The probability $P_a \times P_{iv}$ can change the degree distribution by referencing the newly added high-value knowledge units and quickly move them up in the list of most referenced knowledge units. For example, the question (ID=14220321) was posted on Jan. 8th, 2013 and was first referenced on Jan. 10th 2013. As this question and its answers provide a good discussion about AJAX request in JavaScript, it was referenced frequently by other JavaScript-related questions and moved up to the top 10 most referenced knowledge unit on Nov. 26th, 2013. Such a burst of references to certain knowledge units due to their high information values leads to small changes in the degree distribution (see the subfigure(b) in Figure 7).

The probability $P_a \times P_{iv}$ can also affect the distribution by changing the relative ranking positions of knowledge units. For example, the knowledge unit (question ID=12573816) which discusses the C++ compiler's linker error ranked one position ahead of knowledge unit (question ID=218384) which discusses Java's null pointer exception before Jan. 31, 2015. However, during a short two-month period from Dec. 1st, 2014 to Jan. 31st, 2015, the knowledge unit (218384) had been referenced 365 times, which is 205 times more than that of the knowledge unit (12573816). This causes the swapping of the ranking positions between the two knowledge units.

9.3 Preferential Attachment — "Reinforce the Distribution"

The scale-free property of the indegree distribution of knowledge network, as shown in Section 7, indicates that new associations are more likely to be added to already

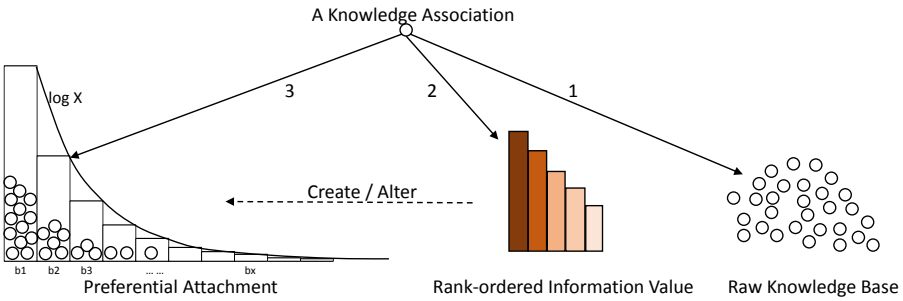

Fig. 9 Knowledge Diffusion Process through Social URL Sharing

referenced knowledge units. Therefore, the indegree of those most referenced knowledge units increases proportionally to the number of new associations being added over time. This behavior is well known as preferential attachment (Barabási and Albert 1999), i.e., "the rich get richer". When referencing an existing knowledge unit, with a probability $1 - P_a$, the user references one that has already been referenced. A highly referenced knowledge unit will appear in many posts. Thus, it is more likely to attract users' attention and be shared again. Therefore, we model a probability P_{ind} that an already referenced knowledge unit will be referenced proportionally to its indegree. Let $R(x)$ be the times that a knowledge unit x has been referenced. Then, the probability P_{ind} is $\frac{R(x)}{\sum R(i)}$, where $\sum R(i)$ is the sum of the times that all knowledge units have been referenced. The probability $(1 - P_a) \times P_{ind}$ means that a knowledge unit with higher indegree in the past will more likely be reinforced in the future.

9.4 Simulating Knowledge Diffusion Process

Imagine the process of referencing a knowledge unit as putting a ball into bins as shown in Figure 9. Here a bin represents a knowledge unit, and a ball represents a reference from a knowledge unit to another (i.e., a knowledge association). A ball will be added to bins through one of the three paths. With the probability $1 - P_r$, the user does not put a ball in any bins (i.e., path 1), which means he does not reference any existing knowledge units. With the probability P_r (baseline probability), he puts a ball in one of the existing bins. Now, he has two choices. First, with a probability P_a, he puts a ball in a bin based on the information value P_{iv} of the bin (i.e., path 2). Second, with a probability $1 - P_a$, he puts a ball in a bin based on the number of balls P_{ind} already in the bin (i.e., path 3). By linearly interpolating the path 2 and path 3, we can generate the probability distribution of a knowledge unit x that will be referenced as:

$$P(x) = P_r(x) \times (P_a \times P_{iv} + (1 - P_a) \times \frac{R(x)}{\sum R(i)})$$

The information value dimension $P_a \times P_{iv}$ explains the creation and change of the power law distribution, while the $(1-P_a)\times\frac{R(x)}{\sum R(i)}$ dimension explains the reinforcement and stability of the distribution.

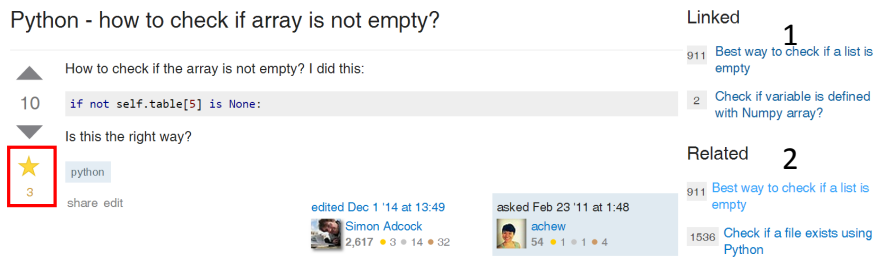

Fig. 10 A Stack Overflow Question Example

10 Discussion

So far, we have obtained a comprehensive understanding of the structure and dynamics of the knowledge network in Stack Overflow. Finally, in this section, we discuss the implications of our study for Q&A site design, general users of crowdsourced knowledge base, and knowledge representation.

10.1 Implications for Q&A Site Design

A user must know of a knowledge unit before he can share it with others. To improve this baseline probability P_r of URL sharing in our generative model covered in Section 9.1, Q&A sites like Stack Overflow need effective mechanisms for users to search and bookmark existing knowledge units.

Q&A sites like Stack Overflow usually rank the search results by simply keyword matching, recency of latest action, or votes. These simple ranking criteria do not take into account the semantic topics of the knowledge units, nor the attention that the knowledge units attract in the community. As URL sharing behavior creates a knowledge network exhibiting assortative mixing of semantic topics and structural stability, a *Topic-Sensitive PageRank* algorithm (Haveliwala 2002) would provide more sensible rankings of the search results. Topic-sensitive PageRank could be applied to capture the importance of knowledge units in the knowledge network with respect to a particular topic. As Stack Overflow's knowledge network exhibits different structural properties from the general Web, the original algorithm designed for the general Web must be adapted, as pointed out by Zhang et al. (2007).

Stack Overflow allows users to bookmark knowledge units by *Favorites* (see the red box in Figure 10). In addition to such knowledge bookmarking, remembering the knowledge "searched and researched" just-in-time is also important for knowledge diffusion, as "search and research" is a common practice that Stack Overflow encourages, as mentioned in Section 5.4. Existing knowledge remembrance mechanisms like CiteHistory tool (Fourney and Morris 2013) lack effective knowledge representation to manage and retrieve knowledge at semantic level. The information is usually represented as a list of items (e.g., URLs), while what users care about are types, properties and relationships of domain-specific entities discussed in the knowledge

units. A *knowledge graph* (Bordes and Gabrilovich 2014) would provide a semantic representation of bookmarked and just-in-time researched knowledge units (see also Section 10.3). The knowledge graph enables semantic search of the user's priori knowledge around domain-specific entities for knowledge sharing.

10.2 Implications for Developers using Crowdsourced Knowledge

Through years of accumulation, Stack Overflow has become a huge body of crowdsourced programming knowledge that complements traditional technical documentation (Parnin et al. 2012). This body of crowdsourced knowledge does not just help particular question askers or just Stack Overflow registered users. Instead, it can be accessed and used by anybody who has programming issues or wants to learn programming. For example, a person who wants to learn EAFP principle and exception handling in Python can visit the knowledge units H2 and H6 and other associated knowledge units shown in Figure 4 in Section 6.2.

Although the knowledge is highly likely out there, it may not be easily accessible in most situations. Developers, especially less experienced ones, who search the knowledge base in Stack Overflow are often faced with two difficulties: *navigability of knowledge network* and *search query formulation or reformulation*. When searching Stack Overflow, it is very likely that a user lands on a knowledge unit that is relevant but does not fully satisfy his information needs. To find the needed information, the user has to explore the knowledge units associated (directly or indirectly) with the current knowledge unit or research with a new query. However, navigating Stack Overflow's knowledge network can be challenging due to the reachability of the network and different degrees of assortative mixing of semantic topics among associated knowledge units. Furthermore, for novice developers who hold insufficient knowledge for certain programming aspects, it would be difficult for them to precisely express his information needs.

When viewing a knowledge unit, Stack Overflow currently lists directly-associated knowledge units under "Linked" with question title and votes (see Figure 10). Looking at this list, the user has no idea about where the knowledge unit he is viewing stands in the larger context of the knowledge network, and have little information scent about where he can go, which knowledge unit(s) may lead to the knowledge he needs, or how he may reformulate his query to improve the search results. That is, knowledge accessibility becomes a critical issue for exploiting the crowdsourced knowledge in Stack Overflow. Furthermore, the difficulty in finding the needed information may also result in duplicate questions.

We believe high modularity of the knowledge network, semantic topics of knowledge clusters, and the structure skeleton formed by hub and betweenness knowledge units can be leveraged to support user-centered exploratory search (Marchionini 2006). The key idea, inspired by the observations in Section 6.2, is to make hidden knowledge clusters, their hub and betweenness unit, their topics and their correlations easily accessible in an *interactive knowledge location map*.

Figure 3, Figure 4 and Table 4 in Section 6.2 show some preliminary elements for building an interactive knowledge location map for large knowledge network.

This knowledge location map is an analogy of location map that one can find in big shopping malls, in which shops (knowledge units) are "clustered" based on the goods (problems) they sell (solve), and some descriptions (topics) of each "cluster" are displayed aside. Location map also clearly tell shoppers (knowledge users) "You are here" in a big mall (knowledge network) and help them determine where they can go and how to get there. The extracted topics of available knowledge units or cluster, centered around programming-specific entities (also discussed in Section 10.3), could help with query reformulation. As such, interactive knowledge location map would improve knowledge accessibility for users of the crowdsourced knowledge in Stack Overflow. Considering the fact that one quarter of the total knowledge units are connected (see Section 4), which is already a non-trivial proportion, plus the fact that more users' attention is there on the non-isolated knowledge units (see the implications at the end of Section 4), we believe the proposed knowledge location map is necessary for developers' exploratory search within Stack Overflow. Such a knowledge location map will be an additional channel that can provide extra assistance to developers' information seeking process without impeding the user's normal information search.

10.3 Implications for Software Engineering Knowledge Representation and Search

The knowledge network in Stack Overflow is built by the crowdsourced power of its users. It can be time consuming to purely rely on several human experts to digest and then integrate the large amount of information generated by millions of users into the knowledge network, as evidenced by the fact that there was a time delay between the posting of a knowledge unit and the referencing of the knowledge unit in other posts. This motivates advanced knowledge management in Stack Overflow that goes beyond human manual efforts. One such advanced technique is to develop entity-centric knowledge network/graph that organizes the knowledge in Stack Overflow centered on software-specific entities. A further motivation for developing entity-centric knowledge management systems is that existing search systems are mainly document-centric and topic-based; topics mainly capture linguistic properties of the knowledge units (i.e., documents), but they may not accurately represent the information value of the units, such as types, properties and relations of programming-related entities discussed in the units. As such, searching by topics often does not lead to knowledge units that meet the users' information needs. This is evident in those "visited but not helpful" URL references in Table 2, shown in Section 5.

Entity-centric search (or semantic search) (Cucerzan 2007; Ferré and Hermann 2011) is an open but far-reaching research area; it can better understand the users' information needs and determine the information value of the knowledge units. Unlike document-centric search that is built on a hyperlink network of documents, entity-centric search is built on a knowledge graph that capture the types, properties and relationships of entities for a particular domain. Entity-centric search can enable more productive and pleasant user experience by providing direct, aggregated and in-depth answers around domain-specific entities that meet the user's information needs. Suc-

cessful examples include Google's Direct Answer and Answer Panel and Facebook's Entity Graph.

The fast development and effective quality-control mechanism of Stack Overflow make it a well-accepted and trustworthy knowledge base for developers. As such, the knowledge network in Stack Overflow provides a sold basis for constructing a programming-specific knowledge graph that could more accurately capture the information value and relevance of the knowledge units. One potential work is to develop domain-specific named entity recognition (NER) (Tjong Kim Sang and De Meulder 2003) techniques to recognize different categories of programming-related entities embodied in the body and comments of knowledge units, such as programming languages, frameworks, tools, APIs, tutorials, and application features. One pilot study performing software-specific NER has been done by Ye et al. (2016). A further direction is to develop domain-specific relation resolution (Amer-Yahia et al. 2014) techniques to derive from the knowledge units semantic relations between the recognized programming-related entities, for example, an API implements an application feature, a code snippet fixes a bug of an API. The resulting programming-specific knowledge graph would enable semantic search of programming-related entities.

11 Conclusions

In this paper, we presented a systematic study of the structure and dynamics of the knowledge network formed in Stack Overflow. This study shows: 1) Users share URLs for diverse categories of purposes. 2) These URL sharing behaviors create a complex knowledge network with high modularity, assortative mixing of semantic topics, and a structure skeleton consisting of highly recognized knowledge units. 3) The structure of the knowledge network with respect to indegree distribution is scale-free (i.e., stable), in spite of the ad-hoc and opportunistic nature of URL sharing activities, while the outdegree distribution of the knowledge network is not scale-free. 4) The indegree distributions of the knowledge network converge quickly, with small changes over time after the convergence to the stable distribution. We demonstrated that the knowledge network is a natural product of URL sharing behavior that Stack Overflow supports and encourages, and proposed an explanatory model based on information value and preferential attachment theories to explain the underlying factors that drive the formation and evolution of the knowledge network in Stack Overflow. In addition to our empirical study, we discussed the implications of our empirical findings for new search algorithm and bookmarking mechanism in Q&A site design, for improving knowledge accessibility in crowdsourced knowledge base, and for far-reaching research on programming-specific knowledge graph and semantic search.

References

M. Allamanis, C. Sutton, Why, when, and what: analyzing stack overflow questions by topic, type, and code, in *Proceedings of the 10th Working Conference on Mining Software Repositories*, IEEE Press, 2013, pp. 53–56. IEEE Press

S. Amer-Yahia, F. Bonchi, C. Castillo, E. Feuerstein, I. Mendez-Diaz, P. Zabala, Composite retrieval of diverse and complementary bundles (2014)

A. Anderson, D. Huttenlocher, J. Kleinberg, J. Leskovec, Discovering value from community activity on focused question answering sites: a case study of stack overflow, in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge Discovery and Data Mining*, ACM, 2012, pp. 850–858. ACM

K. Bajaj, K. Pattabiraman, A. Mesbah, Mining questions asked by web developers, in *Proceedings of the 11th Working Conference on Mining Software Repositories*, ACM, 2014, pp. 112–121. ACM

A.-L. Barabási, R. Albert, Emergence of scaling in random networks. Science **286**(5439), 509–512 (1999)

A. Barua, S.W. Thomas, A.E. Hassan, What are developers talking about? an analysis of topics and trends in stack overflow. Empirical Software Engineering, 619–654 (2014)

M. Bastian, S. Heymann, M. Jacomy, et al., Gephi: an open source software for exploring and manipulating networks. ICWSM **8**, 361–362 (2009)

D.M. Blei, A.Y. Ng, M.I. Jordan, Latent dirichlet allocation. the Journal of Machine Learning Research, 993–1022 (2003)

V.D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment **2008**(10), 10008 (2008)

A. Bordes, E. Gabrilovich, Constructing and mining web-scale knowledge graphs: KDD 2014 tutorial, in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, 2014, pp. 1967–1967. ACM

A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, J. Wiener, Graph structure in the web. Computer networks **33**(1), 309–320 (2000)

C. Cattuto, V. Loreto, L. Pietronero, Semiotic dynamics and collaborative tagging. Proceedings of the National Academy of Sciences **104**(5), 1461–1464 (2007)

A. Clauset, C.R. Shalizi, M.E. Newman, Power-law distributions in empirical data. SIAM review **51**(4), 661–703 (2009)

S. Cucerzan, Large-Scale Named Entity Disambiguation Based on Wikipedia Data., in *EMNLP-CoNLL*, vol. 7, 2007, pp. 708–716

S. Ferré, A. Hermann, Semantic search: Reconciling expressive querying and exploratory search, in *The Semantic Web–ISWC 2011*, 2011, pp. 177–192

A. Fourney, M.R. Morris, Enhancing Technical Q&A Forums with CiteHistory, in *Proceedings of the Seventh International Conference on Weblogs and Social Media, ICWSM 2013*, 2013

W.-T. Fu, T. Kannampallil, R. Kang, J. He, Semantic imitation in social tagging. ACM Transactions on Computer-Human Interaction (TOCHI) **17**(3), 12 (2010)

P. Fugelstad, P. Dwyer, J. Filson Moses, J. Kim, C.A. Mannino, L. Terveen, M. Snyder, What makes users rate (share, tag, edit...)?: predicting patterns of participation in online communities, in *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*, ACM, 2012, pp. 969–978. ACM

S.A. Golder, B.A. Huberman, Usage patterns of collaborative tagging systems. Journal of information science **32**(2), 198–208 (2006)

C. Gómez, B. Cleary, L. Singer, A study of innovation diffusion through link sharing on stack overflow, in *Mining Software Repositories (MSR), 2013 10th IEEE Working Conference on*, IEEE, 2013, pp. 81–84. IEEE

L. Guerrouj, S. Azad, P.C. Rigby, The influence of App churn on App success and StackOverflow discussions, in *Software Analysis, Evolution and Reengineering (SANER), 2015 IEEE 22nd International Conference on*, 2015, pp. 321–330

H. Halpin, V. Robu, H. Shepherd, The complex dynamics of collaborative tagging, in *Proceedings of the 16th international conference on World Wide Web*, ACM, 2007, pp. 211–220. ACM

A.E. Hassan, R.C. Holt, The small world of software reverse engineering, in *Reverse Engineering, 2004. Proceedings. 11th Working Conference on*, 2004, pp. 278–283

T.H. Haveliwala, Topic-sensitive pagerank, in *Proceedings of the 11th international conference on World Wide Web*, ACM, 2002, pp. 517–526. ACM

A. Java, X. Song, T. Finin, B. Tseng, Why we twitter: understanding microblogging usage and communities, in *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, ACM, 2007, pp. 56–65. ACM

J. Lin, Divergence measures based on the shannon entropy. Information Theory, IEEE Transactions on **37**(1), 145–151 (1991)

P. Louridas, D. Spinellis, V. Vlachos, Power laws in software. ACM Trans. Softw. Eng. Methodol. **18**(1), 2–1226 (2008)

L. Mamykina, B. Manoim, M. Mittal, G. Hripcsak, B. Hartmann, Design lessons from the fastest Q&A

site in the West, in *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM, 2011, pp. 2857–2866. ACM

C.D. Manning, P. Raghavan, H. Schütze, *Introduction to Information Retrieval* (Cambridge University Press, New York, NY, USA, 2008). ISBN 0521865719, 9780521865715

G. Marchionini, Exploratory search: from finding to understanding. Communications of the ACM **49**(4), 41–46 (2006)

S. Maslov, K. Sneppen, A. Zaliznyak, Detection of topological patterns in complex networks: correlation profile of the internet. Physica A: Statistical Mechanics and its Applications **333**, 529–540 (2004)

R. Meusel, S. Vigna, O. Lehmberg, C. Bizer, Graph structure in the web—revisited: a trick of the heavy tail, in *Proceedings of the companion publication of the 23rd international conference on World wide web companion*, International World Wide Web Conferences Steering Committee, 2014, pp. 427–432. International World Wide Web Conferences Steering Committee

M.E. Newman, Assortative mixing in networks. Physical review letters **89**(20), 208701 (2002)

N. Novielli, F. Calefato, F. Lanubile, The challenges of sentiment detection in the social programmer ecosystem, in *Proceedings of the 7th International Workshop on Social Software Engineering*, ACM, 2015, pp. 33–40. ACM

A. Pal, S. Chang, J.A. Konstan, Evolution of Experts in Question Answering Communities., in *ICWSM*, 2012

C. Parnin, C. Treude, L. Grammel, M.-A. Storey, Crowd documentation: Exploring the coverage and the dynamics of api discussions on stack overflow. Georgia Institute of Technology, Tech. Rep (2012)

R. Pressman, *Software Engineering: A Practitioner's Approach*, 7th edn. (McGraw-Hill, Inc., New York, NY, USA, 2010)

J. Preusse, J. Kunegis, M. Thimm, S. Staab, T. Gottron, Structural Dynamics of Knowledge Networks., in *ICWSM*, 2013

M.M. Rahman, S. Yeasmin, C.K. Roy, Towards a context-aware ide-based meta search engine for recommendation about programming errors and exceptions, in *Software Maintenance, Reengineering and Reverse Engineering (CSMR-WCRE), 2014 Software Evolution Week-IEEE Conference on*, IEEE, 2014, pp. 194–203. IEEE

C. Rosen, E. Shihab, What are mobile developers asking about? a large scale study using stack overflow. Empirical Software Engineering, 1–32 (2015)

M. Squire, Should We Move to Stack Overflow?: Measuring the Utility of Social Media for Developer Support, in *Proceedings of the 37th International Conference on Software Engineering - Volume 2*. ICSE '15 (IEEE Press, Piscataway, NJ, USA, 2015), pp. 219–228

S. Subramanian, L. Inozemtseva, R. Holmes, Live API Documentation, in *Proceedings of the 36th International Conference on Software Engineering*. ICSE 2014 (ACM, New York, NY, USA, 2014), pp. 643–652. ISBN 978-1-4503-2756-5

J. Sunshine, J.D. Herbsleb, J. Aldrich, Searching the State Space: A Qualitative Study of API Protocol Usability, in *Proceedings of the 22Nd International Conference on Program Comprehension*. ICPC 2015, 2015

E.F. Tjong Kim Sang, F. De Meulder, Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition, in *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, Association for Computational Linguistics, 2003, pp. 142–147. Association for Computational Linguistics

C. Treude, O. Barzilay, M.-A. Storey, How do programmers ask and answer questions on the web?: Nier track, in *Software Engineering (ICSE), 2011 33rd International Conference on*, IEEE, 2011, pp. 804–807. IEEE

B. Vasilescu, A. Serebrenik, P. Devanbu, V. Filkov, How social Q&A sites are changing knowledge sharing in open source software communities, in *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*, ACM, 2014, pp. 342–354. ACM

C. Wagner, P. Singer, M. Strohmaier, B.A. Huberman, Semantic stability in social tagging streams, in *Proceedings of the 23rd international conference on World Wide Web*, International World Wide Web Conferences Steering Committee, 2014, pp. 735–746. International World Wide Web Conferences Steering Committee

S. Wang, D. Lo, L. Jiang, An empirical study on developer interactions in stackoverflow, in *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, ACM, 2013, pp. 1019–1024. ACM

S. Wang, D. Lo, B. Vasilescu, A. Serebrenik, Entagrec: an enhanced tag recommendation system for software information sites, in *Software Maintenance and Evolution (ICSME), 2014 IEEE International Conference on*, IEEE, 2014, pp. 291–300. IEEE

D. Ye, Z. Xing, C.Y. Foo, Z.Q. Ang, J. Li, N. Kapre, Software-specific Named Entity Recognition in Software Engineering Social Content, in *The 23rd IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER 2016)*, 2016. Accepted to appear. Preprint available at: `http://yedeheng.weebly.com/uploads/5/0/3/9/50390459/saner2016.pdf`

J. Zhang, M.S. Ackerman, L. Adamic, Expertise networks in online communities: structure and algorithms, in *Proceedings of the 16th international conference on World Wide Web*, ACM, 2007, pp. 221–230. ACM