Constructing and Evaluating an Evolving Web-API Network for Service Discovery

Olayinka Adeleye[1(✉)], Jian Yu[1], Sira Yongchareon[1], and Yanbo Han[2]

[1] Department of Computer Science, Auckland University of Technology, Auckland 1010, New Zealand
{olayinka.adeleye,jian.yu,sira.yongchareon}@aut.ac.nz
[2] Beijing Key Laboratory for Large-scale Stream Data Processing, North China University of Technology, Beijing, China
yhan@ncut.edu.cn

Abstract. Web-APIs enable cross-organizational functionality integration over the Web and thus are the foundation of modern distributed service-based systems. However, despite the rapid increase in the number of Web-APIs available on the Internet, the discovery and uptake of appropriate Web-APIs by businesses on a Web scale is still a great challenge. One of the main reasons is that Web-APIs registered on directories such as ProgrammableWeb.com are in general *isolated*, as they are registered by diverse providers independently and progressively. In this paper, we present a method for analyzing the Web-API ecosystem and propose a complex-network-based approach for building an evolving social network for Web APIs. We conduct our analysis in two phases: First, from the complex network perspective, we investigate mashups and Web-APIs interactions and analyze the Web-API popularity distribution using the popular ProgrammbleWeb dataset. Second, we quantitatively measure the Preferential Attachment mechanism which is a key driver of an evolving network. Based on our analysis, we propose an approach to construct an evolving Web-API social network based on the theoretical procedure of the Barabási-Albert complex network model. Results presented in this work will not only provide insight into the topology of the Web-API ecosystems but also serve as a practical guide for designing an evolving-network-based solution for service discovery.

Keywords: Web APIs · Complex network analysis
Preferential attachment · Evolving networks · ProgrammableWeb

1 Introduction

The emergence of Web 2.0 coupled with the rapid development in Web service technology has led to a continual increase in the number of Web services and

This work was supported in part by the National Key Technology R&D Program of China (2017YFC0804406) and the National Natural Science Foundation of China (61672042).

C. Pahl et al. (Eds.): ICSOC 2018, LNCS 11236, pp. 603–617, 2018.
https://doi.org/10.1007/978-3-030-03596-9_44

their compositions. Nowadays many real-world applications such as online social media, online shopping, weather forecast, and disaster prevention [14,15] invoke web services via accessible endpoints to implement their functionalities. Modern web services with features such as *RESTful architecture*, *JSON data*, and/or *JavaScript interface* are usually called *Web-APIs*[1] in order to distinguish from the traditional SOAP-based web services; and multiple Web-APIs can be quickly composed into a webpage or application called *mashup*. This shortened software development life cycle leads to the formation of the so-called Web service ecosystem [5,16], where new services emerge, some old ones perish, and service vendors and developers collaborate to develop innovative software solutions. A typical representation of an evolving service ecosystem is ProgrammableWeb[2], which is currently the largest online Web-API directory, with over 19,000 Web-APIs belonging to more than 400 predefined categories, and over 6,000 mashups as at May 2018. It also provides information such as date of introduction, profile, and developers. The perishing of some existing Web-APIs and the emergence of new ones coupled with their dynamic collaborations drive the evolution of this service ecosystem over time [12].

For most service ecosystems, one of the main issues is the isolation of Web-APIs, which limits their discoverability. For instance, in ProgrammableWeb, Web-APIs have categories, and several Web-APIs can be involved in one mashup, but there is no *direct* connection or relationship between two Web-APIs. The reason behind this is that Web-APIs are usually registered by diverse service providers independently over time, and the connections or social relationships between Web-APIs are never directly created.

In this paper, we propose an evolving-complex-network-based approach to constructing a social network for Web-APIs based on their popularity in the service ecosystem. To achieve this, we first study the underlying topology of ProgrammableWeb and analyze the popularity distribution of Web-APIs in the ecosystem, using the dataset in a period of thirteen years (2005–2017). Then, we measure the Preferential Attachment (PA) of the ecosystem, which is the key mechanism that governs the evolution of many existing real world networks [11]. And finally, we incorporate our findings into the construction of an evolving social network of Web-APIs using the well-established Barabási-Albert model [3] in complex networks.

The main contribution of this paper includes:

1. We analyzed the popularity distribution of Web-APIs on ProgrammbleWeb based on mashup-API relationships and measured the PA mechanism which defines the topology of the ecosystem. To the best of our knowledge, this is the first time that PA is measured for a service ecosystem.
2. We designed and implemented an evolving social network model for ProgrammableWeb Web-APIs which facilitates service discovery and serves as a

[1] https://en.wikipedia.org/wiki/Web_API. Note that in this paper, we coin "Web" and "APIs" together as one term "Web-APIs" to emphasize the atomicity of this term.
[2] http://www.programmableweb.com.

stepping stone for developing advanced evolving network models for service ecosystems.

The rest of this paper is organized as follows. Section 2 is the background and related work; Sect. 3 presents the analysis of the ProgrammableWeb ecosystem and also the analysis results including popularity distribution and PA; In Sect. 4, we present an approach to the construction of an Web-API evolving network and discuss its application in service discovery; Finally, Sect. 5 is the conclusion and future work.

2 Background and Related Work

In this section, we discuss the background of this work and the related work in complex network analysis, evolving Web service ecosystem analysis, and existing service social network construction approaches.

Over the years, complex networks have been extensively studied and several significant discoveries have been made including the well-acclaimed small-world networks [24] and scale-free networks [4]. Various mechanisms that governs a network's topology and evolution have been investigated and found ubiquitous among many real world networks. In particular, *preferential attachment* and *growth* have garnered special attention in evolving complex networks research [3,21], not only because they are fundamental to explaining the topological features observed in many real world networks but also because they have been empirically validated to be the drivers of many evolving networks. For instance, the topology of the Internet, the World Wide Web and the citation network have been investigated using evolving network models and shown to be fundamentally governed by the PA and growth mechanisms [1–3]. In terms of evolving network models, the PA and growth driven Barabási-Albert (BA) model [3] is the foundation of other models such as the fitness-based Bianconi-Barabási model [6].

There have been a number of studies investigating the evolutionary properties of service ecosystems and the complementary features of services and their compositions particularly on ProgrammableWeb. Weiss et al. [25], examined the structure of the mashup ecosystem using the ProgrammableWeb dataset. The authors analyzed the relationships of mashups and Web-APIs using a bipartite graph, and found that while the growth rate of new Web-APIs and mashups is linear, the distribution of mashups over APIs follows a power-law. Huang et al. [12] used a network analysis approach to study both the usage patterns and the evolution traces of Web-APIs in the ProgrammableWeb. The authors conducted their analysis based on two derived networks: the Composition-Service network, which is the same bipartite graph of Mashup-APIs used in [25] and the Service-Service network, which is a network of services that are used together in the same mashups. The authors found that the service popularity distribution is highly concentrated, which is consistent with the findings in [25], and they also found that the reuse rate of services is low and the advanced use of many services together is still rare, which provides evidence to our motivation

of building a social network for services/Web-APIs. To better present the ProgrammableWeb ecosystem, Lyu et al. [16] used a three-level hierarchical view to visualize it based on the Mashup-API graph, the tag graph, and domain graph. Wang et al. [23] also explored ProgrammableWeb data patterns from the user perspective with a User-API network.

As for the research on constructing service social networks, Fallatah et al. [9] proposed to add service-service, user-user, and user-service links to build a service social network. Based on the network, metrics such as user popularity, service market share, and user satisfaction can be measured. Simulation was done but how to build such network from real-world data was not discussed. Semantic information mined from service descriptions is a good reference for adding links among services. Wang et al. [22] used domain knowledge to calculate the degree of semantic match between any two services and then a threshold can be set to determine the number of links in the network. Similarly, Feng et al. [10] constructed three types of service networks based on the *subsume*, *sequential-total* (the output of service A covers the input of service B), and *sequential-part* (the output of service A partially covers the input of service B) semantic relations. Clearly such networks are static without considering any dynamical properties. From the evolving network perspective, Chen et al. [7] built a service social network partially based on the Bianconi-Barabási (BB) model. One limitation of their work is that the fitness parameter of an existing service node is calculated dynamically on the arrival of a new service, while the BB model requires a quenched/fixed fitness value for a node, which makes the closed-form solution of the BB model not applicable to this network.

3 Analysis and Results

In this section, we investigate the topology and dynamical mechanism of the ProgrammbleWeb registry. For the topology, we look at the popularity, or degree distribution of the Web-APIs based on the mashup-API bipartite graph. We first discuss data acquisition and processing, then we analyze the data in three steps: *visualization*, *model fitting*, and *comparison with existing classical network models* such the Poisson, exponential and log-normal distributions. For dynamical/evolving mechanisms, we investigate and measure *preferential attachment*.

3.1 Data Acquisition and Processing

We collected the time-stamped raw data, which contains information regarding Web-APIs and mashups from *June* 2005 to *November* 2017 in ProgrammableWeb. Since the $ProgrammableWeb$ backend database is not publicly accessible, only its web pages can be employed for collecting the data. We used data scraping to crawl data from ProgrammableWeb web pages. The web pages are separated into two categories: Web-APIs and mashups, where every Web-API has properties including *name*, *description*, *publication date*, and *category*; similarly, each mashup also contains the above metadata plus the list of Web-APIs

Table 1. Summarize features of the programmableweb dataset

Number of Web APIs acquired	16,138
Number of Mashups acquired	5,883
Average number of Web APIs invoked by Mashups	2.1
Number of Mashups with less than 2 services	241
Number of Web APIs invoked in at least one Mashup	1,525

Table 2. Top 5 most consumed Web-APIs

Web APIs	Number of links
GoogleMap	2,072
Twitter	663
Youtube	557
Flickr	484
Facebook	377

invoked within it. Table 1 gives an overview of the ProgrammableWeb dataset. After pre-processing and removing redundant mashup points, we have 16,138 Web APIs and 5,883 Mashups for our analysis.

3.2 Affiliation Network of Web-APIs and Mashups

To extract the popularity distribution of Web-APIs in ProgrammableWeb, we model the ecosystem in the form of an *affiliation network* that depicts the invocation relation between mashups and Web-APIs. As shown in Fig. 1, technically, the network is a bipartite graph, where the edges indicate which Web-APIs are invoked by which mashups: $G = (M \uplus A, E)$ where M is the set of Mashups and A is the set of Web-APIs, and for any edge $(m, a) \in E, m \in M$ and $a \in A$.

Although there are over 16,000 Web-APIs in ProgrammableWeb, only 1,525 of them appear in one or more mashups. We found that the Google Map Web-API takes a center stage in the affiliation network, attracting 2,072 edges/mashup-consumption, which account for about 35 percent of the total mashups in the ecosystem. As shown in Table 2, Popular social media Web-APIs such as Twitter, Youtube, Flickr, and Facebook also appear 663, 557, 484, and 377 times respectively in the network. We also found that less than 7% of the Web-APIs involved in the network are consumed more than 100 times, and over 47% of the Web-APIs are used less than 4 times.

The complete affiliation network is visualized using the Force-Atlas 2 layout in Gephi[3] as shown in Fig. 2. The hubs as listed in Table 2 are clearly visible in the figure as disks with Google-Maps API being the largest one sitting at the bottom.

[3] https://gephi.org/.

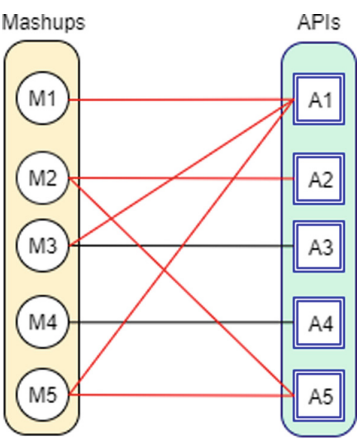


Fig. 1. Illustration of the Mashup-API bipartite graph

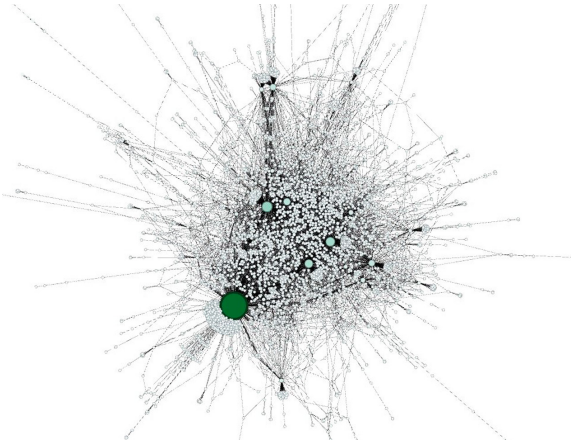


Fig. 2. Visualization of the Mashup-API affiliation network

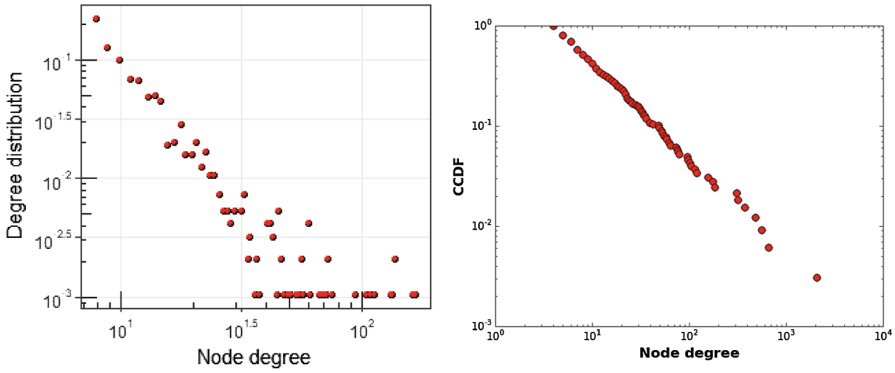
3.3 Web-API Nodes Degree Distribution

An integral part of analyzing the topology of a network is the plotting and fitting of its degree distribution $p(k)$. Networks with long-tailed degree distribution that follows a power-law are known to exhibit the *scale-free* topology. Most real networks such as the internet, WWW and the citation network are *scale-free* networks [3]. On the other hand, networks with exponentially-decaying-tail degree distribution are collectively referred to as *exponential Networks*.

Plotting. To gain insight into the popularity of Web-APIs, we plot the degree distribution of the 1,525 Web-APIs based on their degrees in the affiliation network. As shown in Fig. 3, both the PDF (Probability Density Function) in log-log scale, linear binning, and the CCDF (Complementary Cumulative Distribution Function) in log-log scale are plotted.

In Fig. 3a, the small degree region demonstrates a log-linear relation between $p(k)$ and k ($\log p(k) \sim -\gamma \log k$, or $p(k) \sim k^{-\gamma}$), which is a typical feature of the scale-free network; while a plateau is formed at the large k region as typically we have only one copy of each large-degree node and this plateau affects our ability to estimate the degree exponent γ [3]. One way to extract information from the tail of the distribution is to use the CCDF (Fig. 3b), which enhances the statistical significance of the large-degree region, and if $p(k)$ follows the power-law, then the CCDF is also power-law: $P(k) \sim k^{-\gamma+1}$.

Fitting. In order to determine the best fit for the Web-API degree dataset, we first fit the data to four classical models including *Power-law*, *Exponential*, *Log-normal*, and *Poisson*. Figure 4 shows the result of the fitting when $k_{min} = 4$. We can see that both the power-law and the log-normal offer a good fit to the data, while the exponential and the Poisson fit poorly to the data.



(a) (b)

Fig. 3. Degree distribution plot of the Web-APIs nodes in the affiliation network 3a shows the Log-log plot (linear-binning) of the Web-APIs degree distribution 3b shows the CCDF plot of the degree distribution in log-log scale.

Table 3. Plausibility of fitting Power-law, Log-normal, Exponential, and Poisson models to the Web-API degree data

Parameter	Power-Law	Exponential	Log-normal	Poisson
γ	2.200837	-	-	-
p-value	0.783	0.000	0.667	0.000

To quantitatively measure the plausibility of each distribution, next we conducted a *goodness-of-fit* test based on the Kolmogorov-Smirnov (KS) distance which measures the difference between the model and the empirical data, and a $p\text{-}value \in [0, 1]$ is calculated to measure the model plausibility. The closer p is to 1, the more likely that the difference between the model and the empirical data is attributed to statistical fluctuations alone. If p is very small, the model is not a good fit to the empirical data [3].

Table 3 shows the resultant *p-values* for each distribution. Clearly, power-law is the most plausible fit ($p\text{-}value = 0.783$) and next to it is log-normal (0.667); for both exponential and Poisson, the *p-value* is zero.

Exponent Estimating. In the above, we have justified that the power-law model provides the best fit to our data, next we use MLE (Maximum Likelihood Estimation) to estimate the scaling parameter/degree exponent γ [17]:

$$\hat{\gamma} = 1 + n \left[\sum_{i=1}^{n} \ln \frac{k_i}{k_{min}} \right]^{-1} \tag{1}$$

where k_i, $i = 1 \ldots n$ are the observed values of k such that $k_i \geq k_{min}$, k_{min} represents the minimum degree of node in the network.

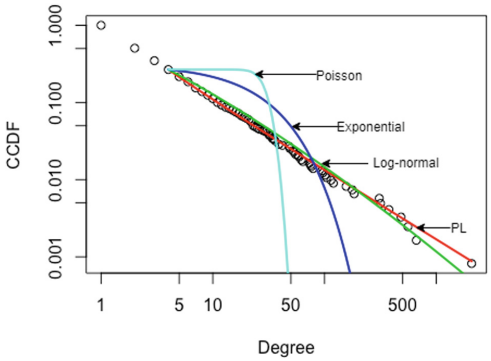


Fig. 4. Fitting Power-law (PL), Log-normal, Exponential, and Poisson models to the Web-API degree data

The assumption for estimating the parameter is that $\gamma > 1$, since the case of $\gamma \leq 1$ does not exist in real world [8].

When $k_{min} = 1$, the appropriate estimator for γ was given as:

$$\frac{\zeta'(\hat{\gamma})}{\zeta(\hat{\gamma})} = -\frac{1}{n}\sum_{i=1}^{n} \ln k_i \tag{2}$$

where $\zeta(\hat{\gamma})$ is the Riemann Zeta function.

Otherwise, when $k_{min} > 1$, the appropriate estimator for γ is:

$$\frac{\zeta'(\hat{\gamma}, k_{min})}{\zeta(\hat{\gamma}, k_{min})} = -\frac{1}{n}\sum_{i=1}^{n} \ln k_i \tag{3}$$

Using the method described in [8], which is also based on the KS distance, we can find the optimal k_{min} with respect to each data point and select the value that gives the minimal KS distance between the CCDF of our data and the fitted model. The resultant γ value for the CCDF is around 2.2 (or the γ value for the PDF is 3.2), which is close to that of the Internet ($\gamma = 3.42$) [3].

3.4 Measuring Preferential Attachment

Real-world networks reach their current size by adding new nodes to the network progressively, and a common phenomenon occurs, where new nodes tend to connect to existing nodes with high degree. This phenomenon is called *Preferential Attachment (PA)* [4]. If the probability that a newly arrive node connects to an existing node i is proportional to the degree of that node k_i, or

$$\Pi(k_i) = \frac{k_i}{\sum_j k_j} \tag{4}$$

Table 4. Preferential attachment measurement

Node span	α (Newman)	α (PAFit)
10	0.97 ± 0.05	1.09 ± 0.06
20	0.96 ± 0.05	1.08 ± 0.06
50	0.95 ± 0.07	1.06 ± 0.07
100	0.94 ± 0.06	1.05 ± 0.08
Monthly	0.96 ± 0.09	1.03 ± 0.09

then we call it *linear-PA*. The combination of growth and linear-PA play a critical role in shaping a network's topology and are responsible for the emergence of the scale-free property [3].

We can use the exponent α to classify different types of PA

$$\Pi(k) \sim k^{\alpha} \tag{5}$$

if α is 1, then PA is *linear*; if α is less than 1, then PA is *sub-linear*; otherwise PA is *super-linear* [3].

We aim to detect the presence of PA in the Web-API node set of the affiliation network of ProgammableWeb and also measure its α value. To do so, we can examine the degree increase of a node i between a fixed span Δt: $\Delta k_i = k_i(t + \Delta t) - k_i(t)$. For example, if $\Delta t = 5$, $k_i(t + \Delta t)$ is the degree of node i after five new nodes joined the affiliation network. The relative change $\Delta k_i / \Delta t$ should follow

$$\frac{\Delta k_i}{\Delta t} \sim \Pi(k_i) \tag{6}$$

Actually, to reduce the noise we can measure the cumulative preferential attachment:

$$\pi(k) = \sum_{k_i=0}^{k} \Pi(k_i) \tag{7}$$

We employ both the *PAFit* method [20] and Newmans's method [18] to estimate PA. As we can see in Table 4, Node Spans 10, 20, 50, 100, and monthly all output consistent results of $\alpha \approx 1$, which demonstrates the existence of linear-PA, or scale-free property, of Web-APIs in the ProgrammableWeb affiliation network.

4 Constructing an Evolving Web-API Network

In this section, we propose a complex-network-based approach to constructing an evolving social network for Web-APIs. We first discuss the limitation of the projection-based approach that projects the affiliation network to an one-mode API-API network; then, we present the evolving network model, and the strategy and procedure we use to construct the social network for ProgrammableWeb APIs; then we discuss the topological property of the constructed network; and finally we discuss an application of the constructed network in service discovery.

4.1 Limitation of the Projection-Based Approach

In order to build a social network for service ecosystems such as ProgrammableWeb, a simple approach is directly applying one-mode projection to mashup-API affiliation network to derive an API-API network. This approach has been used in [13,16]. Figure 5 illustrates how to project an affiliation network onto an one-mode API-API network.

The limitation of this approach is apparent: *Only Web-APIs used in mashups (suppose every mashup contains at least two Web-APIs) will appear on the projected network.* For example, the ProgrammableWeb affiliation network contains only 1,525 Web-APIs, which is less than 10% of the total 16,138 Web-APIs on the registry. Furthermore, the popularity/number-of-links of a Web-API node on the projected network is discounted as there are mashups that use only one Web-API, and such links are not counted in the projected network (for example the link between M_5 and A_6 in Fig. 5).

4.2 Network Model and Construction Strategy

As discussed in Sect. 3.4, the combination of *growth* and *PA* are the two generic mechanisms that drive many real-world networks, and we have validated the presence of both growth and PA in the ProgrammableWeb affiliation network. Based on that, we aim to build a growing/evolving network of Web-APIs that preserve both the topology properties of affiliation network and the popularity information of the Web-API nodes, while including all the Web-APIs in the ecosystems.

We base our model on the Barabási-Albert evolving network model [3]. For the *growth* aspect, it involves continuous addition of new nodes (Web-APIs) into the network, therefore increasing the number of nodes in the network throughout its life span. To do so, first, we initialize the network, starting with fully connected m_0 number of nodes. At every time step, we add a new node with m links.

To incorporate *PA*, we dynamically estimate the probability that a link of the new node connects to an existing node i depends on its degree k_i using the linear-PA equation (Eq. 4) described in Subsect. 3.4.

Figure 6 illustrate the network's growth and PA mechanism (assuming $m_0 = 4, m = 1$): at time 0, four fully connected nodes form the initial network; at time 1, *Node 5* joins the network; based on linear-PA, as *Node 1–4* each has the same degree, each of them has the same probability to attract *Node 5* to connect to it, which is 3/12, or 1/4; at time 2, based on linear-PA, the probabilities for *node 1–5* to attract the new node are $[3/K, 3/K, 4/K, 3/K, 1/K]$ where $K = 13$ is the total degree of the network and the numerator is the degree of each existing node.

Based on the growth and the linear-PA, older nodes always have better chance to attract links than newer nodes as they have better degree/popularity. Specific to ProgrammableWeb, to preserve the popularity information in the affiliation network, we need to define a strategy on *when* a Web-API joins the growing

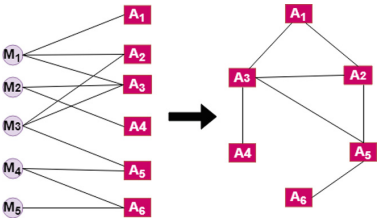


Fig. 5. Affiliation network projection

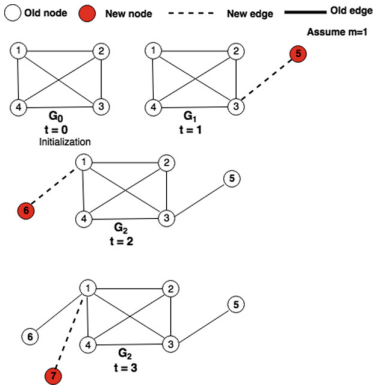


Fig. 6. Illustration of Web-API network growth procedure

network; i.e., we need to sort the full list of Web-APIs and put them onto the network one-by-one (or step-by-step) based on their position in the sorted list. A simple strategy is stated below:

- First, We sort the Web-API nodes based on their degree (or popularity) in the affiliation network in a descending order so that higher-degree nodes are in the front to produce L_1;
- Then, for Web-APIs that do not appear in the affiliation network we sort them based on their date of publication/birth in an ascending order so that older nodes are in the front to produce L_2;
  (it is worth noting that *date-of-birth* is also used in [19] as a measure of popularity.)
- Finally, we just append L_2 in the end of L_1 and put the node into the network one-by-one based on their order in the list.

The complete network construction procedure is described below:

Procedure

Input Parameters (N: number of nodes, m_0: number of initial nodes, m: number of links added at each time step)

1. Creating node list: Sort Web-APIs nodes based on popularity and date-of-birth;
2. Initializing network: Start with a fully connected m_0 number of most popular nodes;
3. Growth: At each time step, a new node with m number of links is added and connected to m number of already existing nodes in the network, where $m \leq m_0$;

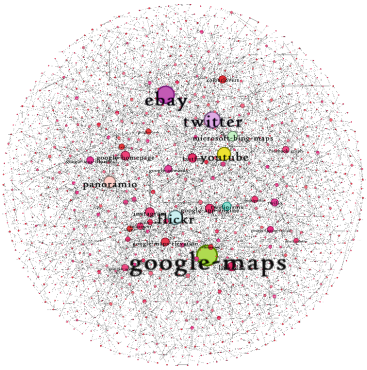


Fig. 7. Overview of the Web-API network

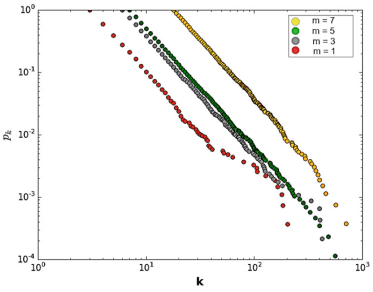


Fig. 8. Degree distribution of the Web-API Network; from bottom to top: m = 1 (red), m = 3 (grey), m = 5 (green) and m = 7 (yellow). (Color figure online)

4. Preferential attachment: With probability $\Pi(k_i)$, the new node connects to an already existing node i with degree (k_i). Probability $\Pi(k_i)$ is estimated dynamically based on Eq. (5).
5. After all N nodes join the network, we obtain the popularity-based, evolving Web-API network.

An overview of the constructed ProgrammableWeb API network containing all its Web-APIs is shown in Fig. 7, with popular nodes labelled. Next, we analyze the topological property of this network.

4.3 Topological Properties of the Web-API Network

In this subsection, we discuss three main topological property of the constructed Web-API network in terms of *degree distribution*, *network diameter*, and *clustering coefficient*.

Degree Distribution. Figure 8 shows the degree distribution (log-binning) with $N = 16138$, $m_0 = 4$, and $m = 1$(red, $\gamma = 2.782$), $m = 3$ (grey, $\gamma = 2.823$), $m = 5$ (green, $\gamma = 3.05$) and $m = 7$ (yellow, $\gamma = 3.002$).

Theoretically, we can use the continuum theory [3] to estimate the degree exponent of the degree distribution

$$p(k) \approx 2m^{1/\beta}k^{-\gamma} \tag{8}$$

where $\beta = 1/2$ is the dynamical exponent and $\gamma = 1/\beta + 1 = 3$.

So the constructed Web-API network is a scale-free network with degree exponent $\gamma \approx 3$.

Network Diameter. Network diameter d is the maximum of the shortest distances between any two nodes. If a network's diameter is proportional to $\ln N$, then it is a *small-world network* [24]. Theoretically, the expected value of the diameter of the Web-API network is [3]:

$$\langle d \rangle \sim \frac{\ln N}{\ln \ln N} \tag{9}$$

If we plug in $N = 16138$, we get $\langle d \rangle \approx 4.27$. For $m = 5$ and $m = 7$, the actual diameters are 6 and 5 respectively, which are consistent with the expected value. As d grows slower than $\ln N$, the Web-API network is *ultra-small*.

Clustering Coefficient. A clustering coefficient measures the density of links in a node's immediate neighborhood, and the average clustering coefficient of a network can be obtained by averaging over all its nodes. Theoretically, the average clustering coefficient of the Web-API network is [3]:

$$\langle C \rangle \sim \frac{(InN)^2}{N} \tag{10}$$

If we plug in $N = 16138$, we get $\langle C \rangle \approx 0.006$. For $m = 5$ and $m = 7$, the actual clustering coefficient are 0.005 and 0.007 respectively, which are consistent with the theoretical value.

4.4 Applying Web-API Network in Service Discovery

In this section, we discuss one application of the Web-API network in service discovery. Consider a new service consumer, who wants to leverage different Web-APIs from different domains (say Dictionary, Translation and Social) to create a mashup that allows users to find the meaning of a word in English, translate to French and post it on social media. In order to navigate through the Web-API network and discover require Web-APIs, a *link-as-you-go* approach [7] can be used. As illustrated in Fig. 9, the user can *zoom-in* to the network, starting from *Twitter API* and then navigate through by following social links to discover *Oxford dictionary (OX)* and then *Google Translate (GT)* API. Such user activity pattern is very similar to surfing the WWW; just in our case the user is surfing the service network.

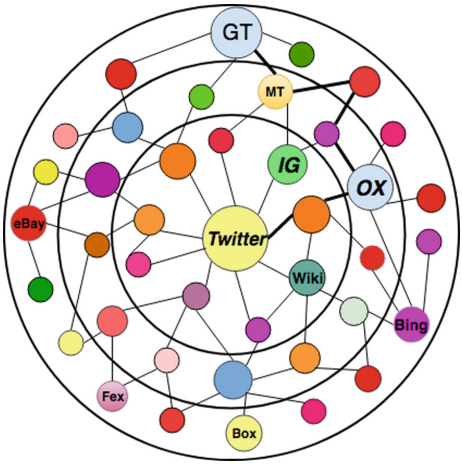


Fig. 9. Web APIs Discovery with *link-as-you-go* approach.

5 Conclusion and Future Work

In this paper, we propose an evolving-network-based approach for constructing a social network for Web-APIs based on their popularity in the service ecosystem. We achieve this by first studying the underlying topology of the ProgrammableWeb service ecosystem from the complex network perspective and analyze the popularity distribution and preferential attachment of Web-APIs in the ecosystem; then, we incorporate our findings into the construction of an evolving social network of Web-APIs using the well-established Barabási-Albert model. In the future we want to do further research in the following directions: (1) As the construction strategy of the network is purely based on node popularity, we plan to investigate other strategies that consider factors such as node fitness and similarity; (2) The clustering coefficient of the constructed network is low compared to real-world networks; we plan to investigate strategies that can build better clustered networks.

References

1. Albert, R., Jeong, H., Barabási, A.-L.: Internet: diameter of the world-wide web. Nature **401**(6749), 130 (1999)
2. Barabási, A.-L.: Network science: luck or reason. Nature **489**(7417), 507 (2012)
3. Barabási, A.-L.: Network Science. Cambridge University Press, Cambridge (2016)
4. Barabási, A.-L., Albert, R.: Emergence of scaling in random networks. Science **286**(5439), 509–512 (1999)
5. Barros, A.P., Dumas, M.: The rise of web service ecosystems. IT Prof. **8**(5), 31–37 (2006)
6. Bianconi, G., Barabási, A.-L.: Competition and multiscaling in evolving networks. EPL (Europhys. Lett.) **54**(4), 436 (2001)

7. Chen, W., Paik, I., Hung, P.C.K.: Constructing a global social service network for better quality of web service discovery. IEEE Trans. Serv. Comput. **8**(2), 284–298 (2015)
8. Clauset, A., Shalizi, C.R., Newman, M.E.J.: Power-law distributions in empirical data. SIAM Rev. **51**(4), 661–703 (2009)
9. Fallatah, H., Bentahar, J., Asl, E.K.: Social network-based framework for web services discovery. In: 2014 International Conference on Future Internet of Things and Cloud (FiCloud), pp. 159–166. IEEE (2014)
10. Feng, Z., Lan, B., Zhang, Z., Chen, S.: A study of semantic web services network. Comput. J. **58**(6), 1293–1305 (2015)
11. Hébert-Dufresne, L., Allard, A., Marceau, V., Noël, P.-A., Dubé, L.J.: Structural preferential attachment: network organization beyond the link. Phys. Rev. Lett. **107**(15), 158702 (2011)
12. Huang, K., Fan, Y., Tan, W.: An empirical study of programmable web: a network analysis on a service-mashup system. In: 2012 IEEE 19th International Conference on Web Services, Honolulu, HI, USA, 24–29 June 2012, pp. 552–559 (2012)
13. Huang, K., Fan, Y., Tan, W.: Recommendation in an evolving service ecosystem based on network prediction. IEEE Trans. Autom. Sci. Eng. **11**(3), 906–920 (2014)
14. Kavitha, R., Anuvelavan, S.: Weather master: mobile application of cyclone disaster refinement forecast system in location based on gis using geo-algorithm. Int. J. Sci. Eng. Res. **6**, 88–93 (2015)
15. Lee, J., Niko, D.L., Hwang, H., Park, M., Kim, C.: A GIS-based design for a smartphone disaster information service application. In: 2011 First ACIS/JNU International Conference on Computers, Networks, Systems and Industrial Engineering (CNSI), pp. 338–341. IEEE (2011)
16. Lyu, S., Liu, J., Tang, M., Kang, G., Cao, B., Duan, Y.: Three-level views of the web service network: an empirical study based on programmableweb. In: 2014 IEEE International Congress on Big Data (BigData Congress), pp. 374–381. IEEE (2014)
17. Muniruzzaman, A.N.M.: On measures of location and dispersion and tests of hypotheses in a pare to population. Calcutta Stat. Assoc. Bull. **7**(3), 115–123 (1957)
18. Newman, M.E.J.: Clustering and preferential attachment in growing networks. Phys. Rev. E **64**(2), 025102 (2001)
19. Papadopoulos, F., Kitsak, M., Serrano, M.Á., Boguná, M., Krioukov, D.: Popularity versus similarity in growing networks. Nature **489**(7417), 537 (2012)
20. Pham, T., Sheridan, P., Shimodaira, H.: PAFit: a statistical method for measuring preferential attachment in temporal complex networks. PloS one **10**(9), e0137796 (2015)
21. Pham, T., Sheridan, P., Shimodaira, H.: Joint estimation of preferential attachment and node fitness in growing complex networks. Sci. Rep. **6**, 32558 (2016)
22. Wang, H., Feng, Z., Chen, S., Xu, J., Sui, Y.: Constructing service network via classification and annotation. In: 2010 Fifth IEEE International Symposium on Service Oriented System Engineering (SOSE), pp. 69–73. IEEE (2010)
23. Wang, J., Chen, H., Zhang, Y.: Mining user behavior pattern in mashup community. In: IEEE International Conference on Information Reuse and Integration, IRI 2009, pp. 126–131. IEEE (2009)
24. Watts, D.J., Strogatz, S.H.: Collective dynamics of small-world networks. Nature **393**(6684), 440 (1998)
25. Weiss, M., Gangadharan, G.R.: Modeling the mashup ecosystem: structure and growth. R&D Manag. **40**(1), 40–49 (2010)