Method for Identification of Suitable Persons in Collaborators' Networks

Pavla Dráždilová, Alisa Babskova, Jan Martinovič, Kateřina Slaninová, and Štěpán Minks

VŠB - Technical University of Ostrava,
Faculty of Electrical Engineering and Computer Science,
17. listopadu 15/2172, 708 33 Ostrava, Czech Republic
{pavla.drazdilova,jan.martinovic,alisa.babskova.st,
katerina.slaninova,min111}@vsb.cz

Abstract. Finding and recommendation of suitable persons based on their characteristics in social or collaboration networks is still a big challenge. The purpose of this paper is to discover and recommend suitable persons or whole community within a developers' network. The experiments were realized on the data collection of specialized web portal used for collaboration of developers - Codeplex.com. Users registered on this portal can participate in multiple projects, discussions, adding and sharing source codes or documentations, issue a release, etc. In the paper we deal with strength extraction between the developers based on their association with selected terms. We have used the approach for extraction of initial metadata, and we have used modified Jaccard coefficient for description of the strength of relations between developers. Proposed method is usable for creation of derived collaborators' subnetwork, where as input is used the set of words, which will describe the area or sphere, wherein we want to find or recommend suitable community and the words specify relation between the developers in the network. Obtained subnetwork describe a structure of developers' collaboration on projects, described by selected term.

1 Introduction

Recently the concept of social networks and online communities is becoming still more and more popular. As a result, the number of their users significantly increasing. Reasons for communication between people and creation of social networks in our time are various: study, hobby, work, games and programming is not the exception.

OSS (Open Source Software) is a example of a dynamic network, as well as a prototype of complex networks emerging on the Internet. By working through the Internet, interactions between developers can be considered as relations in the synthetic network of collaborators. These relations arise when the developers join the project and begin to communicate with others. OSS network consists of two entities - developers and projects. An examples of such OSS social network established on the basis of interaction between the participants is CodePlex.

Many programmers on the Internet are looking for interesting ideas, or assistance when implementing their own solutions. Online collaboration is no longer a novelty in our times and it is run by people all over the world. However, searching for suitable and capable people who could implement a particular idea at reasonable deadlines and high quality is an eternal problem.

In this paper we try to determine the strength of relationship or similarity between CodePlex developers in the context of projects they work on. To determine the context, we used project key words, which in the case of the CodePlex are extracted from project descriptions. We would find some developers or some community, which is specified by key words, for a recommendation.

Some related work dealing with the recommendation in the social network. In the article [1] authors studies people recommendations designed to help users find known, offline contacts and discover new friends on social networking sites. Other approach is in the article [4], where authors examine the dynamics of social network structures in Open Source Software teams but data were extracted monthly from the bug tracking system in order to achieve a longitudinal view of the interaction pattern of each project.

2 CODEPLEX

CodePlex is Microsoft's open source project hosting web site. You can use CodePlex to find open source software or create new projects to share with the world. Codeplex.com has 11 years old, it is ranked 2,107 in the world, a low rank means that this website gets lots of visitors. Its primary traffic from United States and is ranked 3,175 in United States. It has 104 subdomains with traffic. It has 136,500 visitors per day, and has 436,800 pageviews per day. CodePlex is mainly used by developers for collaboration on projects, sharing source codes, communication and software development. Generally, registered users can participate in multiple projects, discussions, adding the source code and documentation, issue a release, etc. Some of the users have defined a specific role within the project for which they work. Each user has his own page, where he can share information about himself, his projects on which he currently works, and the most recent activities. The CodePlex projects themselves can be considered as a very interesting source of information. In addition to the list of users and roles, CodePlex enables register keywords, add description of the project, the number of visits, status, date of creation, url and other information about the project. All activities are carried out on CodePlex by a particular user within a specific project.

Database which was created as a result of data obtained from CodePlex.com, consists of 6 main tables: User, Project, Discussions, RecentActivity, Membership and SourceCode (see Table 1).

In CodePlex, we can see two types of entities: users and projects. Both are represented by tables that contain specific characteristics. The table User contains informations about users such as login, personalStatement, createdOn, lastVisit and url of user page. The table Project contains some characteristics of project in Codeplex: tags, date od created on, status, license, pageViews, count of visits, description and url of project page.

The undirect connection between the user and the project is implemented through activities within the scope of the project. These activities are in the database CodePlex

Table 1. The CodePlex database tables

Table	Number of lines
User	96251
Project	21184
Discussions	397329
RecentActivity	72285
Membership	126759
SourceCode	610917

divided into different types: SourceCode, Discussion, RecentActivity and Membership (see Table 2).

Table 2. The CodePlex activities

Activity	Meaning
SourceCode	records about added projects
Discussion	discussions about the project and the responses of individual users
RecentActivity	check-ins, task records, add Wiki information, notes about Release version etc
Membership	able to trace the users' participation in the projects and their assigned role

We can represent CodePlex as a bipartite graph of users and projects, where the edge between the user and the project is a user's activity in a project.

If we look at the data that we have in the Table User, we are not able to define the user's profile. It consists of the field of interest, what he deals with, the programming language he uses and at what level. PersonalStatement attribute is used to describe the user, but from the total set of our users downloaded, there was not a single one, who would fill it up. On the other hand, the project has enough information defined – which fields are concerned, how long it lasted, whether it is completed, which technology it is used, etc.

The main attribute, carrying the largest set of information, is the project Description – the description of the project itself.

Using activities such as user links to the projects, we are able to determine with some probability an area of specialization and a work of each user. For example, if a user is working on three projects written in .NET and one in Java, we could include him in .NET programmers with high probability, and less likely recommend him as a Java programmer.

In other words, terms or description of the project may not only help us to provide more information about projects, but also to determine the user's area of interests or abilities. As a result, the way we are able to compare user attributes determines the similarity to other network participants.

3 Collaborators network and Projects network

Whenever we think about collaboration between two persons, we not only look at the relationship itself, but also at the context. It is clear that depending on context, the strength of relationship changes. Therefore, we divide collaboration into two main parts *Developers' Relationship* and *Developers' Context*. We consider the relation between developers and the term describes the context between developers.

Developers have additional attributes. Usually it could be publications, teams, organizations, projects, etc. We called it attribute domain, in our case D_{CP_D} be a set of projects in Codeplex, then CP_D are attributes for all D_i developers, where objects is one developer's attributes described as $CP_{D_i} \subseteq D_{CP_D}$.

3.1 Developers' and Context Relationship

We describe a developers' relationship as commutative operation on cartesian product of developer's attribute $X\,x\,X$, where output is mapped to the set of real numbers $\mathbb{R}$.

We use Jaccard coefficient ([2] for evaluation of developers relations using their attributes.

$$Attribute_{Score}(CP_{D_i}, CP_{D_j}) = \frac{|CP_{D_i} \cap CP_{D_j}|}{|CP_{D_i} \cup CP_{D_j}|} \tag{1}$$

As we discussed above, every developer has it's attributes. Moreover, each project has a description text. If we use lexical analysis on this text, we can define a term set for every developer as T_{D_i} and this term set contains all terms of projects, which developer D_i participated. The extracted text is proceed to methods, which remove words that do not carry any important information. The main issue of this paper is not to describe this kind of methods. More could be found in [6, 3].

Term set T consists of all developers term sets $\{T_{D_0}, T_{D_1}, \ldots, T_{D_n}\} = T$, when the domain for terms T could be obtained as union of all terms extracted for each person $D_T = T_{D_0} \cup T_{D_1} \cup \ldots \cup T_{D_n}$.

The whole process of obtaining term sets is described in [5], so we just remiding $(t_k\ in\ T_{D_i})$ stands for the number of terms t_k by T_{D_i} and $(t_k\ in\ T)$ stands for the number of terms t_k in descriptions of all projects by T.

We can evaluate association between the selected term $t_k \in D_T$ and a developer $D_i \in D$:

$$R(T_{D_i}, t_k) = \frac{(t_k\ in\ T_{D_i})}{(t_k\ in\ T) + |T_{D_i}| - (t_k\ in\ T_{D_i})} \tag{2}$$

We normalize $R(T_{D_i}, t_k)$ such that $R_{Norm}(T_{D_i}, t_k) \in < 0, 1 >$:

$$R_{Norm}(T_{D_i}, t_k) = \frac{R(T_{D_i}, t_k)}{MAX(R(T_{D_i}, t_1), \ldots, R(T_{D_i}, t_{|T_{D_i}|}))} \tag{3}$$

Evaluation of the whole relationship context of two persons D_i and D_j has two steps. First, we compute association between D_i and select term t_k, and between the second developer D_j and t_k separately. Afterwards, because each part is already evaluated by real number, we combine both results in the same way; we can combine the whole result in equation one. In CodePlex we see the description text for the developer as the all description of all projects he is working on, joined together. We obtain equation for the $Context_{Score}$:

$$Context_{Score}(T_{D_i}, T_{D_j}, t_k) = R_{Norm}(T_{D_i}, t_k)\, R_{Norm}(T_{D_j}, t_k) \tag{4}$$

3.2 Collaboration – Whole Score

The last step is to define Score, which consists of $Attribute_{Score}$ and $Context_{Score}$:

$$Score(CP_{D_i}, CP_{D_j}, T_{D_i}, T_{D_j}, t_k) = Attribute_{Score}(CP_{D_i}, CP_{D_j})\, Context_{Score}(T_{D_i}, T_{D_j}, t_k) \tag{5}$$

This equation evaluates the relation between developers depending on the selected words, which represent the context. So we get a evaluation for the new subnet, which is specified by selected terms.

3.3 Construction of the Collaborators Graph

To describe the network of collaboration, we use standard weighted graph $G_D(V_D, E_D)$, where weighted function is defined as $w_D : E_D(G) \mapsto \mathbb{R}$, when $w_D(e) \geq 0$.

The determination of set D is simple, because objects of vertices set V_D match with objects of set D, so $V_D = D$. However, we can do the same with all the possible pairs from set D to assign a set of edges E_D; it is better to design the algorithm to each implementation at first, and to reduce the number of useless computations. In addition, we must choose term t_k for function w_D, which reflects the context. Because only the commutative operations are used, we do not need to take into consideration the order of attribute objects in function parameters. Moreover E_D is two-object set, where the order of objects does not matter, so the evaluating is done just once.

When we construct graph based on developers' projects relationship, we use $Attribute_{Score}(CP_{D_i}, CP_{D_j})$ as w_D, where no term is needed, then simply $V_D = D$, which means that every developer is a vertex in the graph. Then, for each developer $D_i \in D$ we find collaborators D_{i_C} and for each collaborator $D_j \in D_{i_C}$ we create two-object set $\{D_i, D_j\}$, which corresponds with an edge in the graph. Equation 1 is then used to evaluate the edge.

The function $Score(CP_{D_i}, CP_{D_j}, T_{D_i}, T_{D_j}, t_k)$ is used for evaluating the edges in the context of the term. The only difference is, that majority of developers has not chosen term in their description text, so the result will be 0 and no edge would exists. Hence, we first determine subset of developers $D_{t_k} \subseteq D$ for those that have a term in their description text, followed by the same steps described in the last paragraph to compute developers' projects relationship. Then, the term t_k is used for computation of the second part in $Context_{Score}(T_{D_i}, T_{D_j}, t_k)$. Finally, we calculate the whole *Score* by multiplication of both parts.

3.4 Construction of the Projects Graph

We consider as well as developers, as well as projects. We define *Projects' Relationship* and *Projects' Context*.

We use Jaccard coefficient for evaluation of projects relations using their attributes - $Attribute_{Score}$.

$$Attribute_{Score}(CP_{P_i}, CP_{P_j}) = \frac{|CP_{P_i} \cap CP_{P_j}|}{|CP_{P_i} \cup CP_{P_j}|} \tag{6}$$

We evaluate association between the selected term $t_k \in D_T$ and a project $P_i \in P$:

$$R(T_{P_i}, t_k) = \frac{(t_k \; in \; T_{P_i})}{(t_k \; in \; T) + |T_{P_i}| - (t_k \; in \; T_{P_i})} \tag{7}$$

We compute equation for the $Context_{Score}$:

$$Context_{Score}(T_{P_i}, T_{P_j}, t_k) = R_{Norm}(T_{P_i}, t_k) \, R_{Norm}(T_{P_j}, t_k) \tag{8}$$

The last step is to calculate $Score$, which consists of $Attribute_{Score}$ and $Context_{Score}$:

$$Score(CP_{P_i}, CP_{P_j}, T_{P_i}, T_{P_j}, t_k) = Attribute_{Score}(CP_{P_i}, CP_{P_j}) \, Context_{Score}(T_{P_i}, T_{P_j}, t_k) \tag{9}$$

To describe the network of projects, we use standard weighted graph $G(V_P, E_P)$, where weighted function is defined as $w_P : E_P(G) \mapsto \mathbb{R}$, when $w_P(e) \geq 0$. We consider $V_P = P$ and we use $Score$ for edges evaluation in the new graph $G(V_P, E_P)$.

4 Experiments

For the basic computation of the collaboration, we chose the terms ”iphone”, ”wp7”, ”android” and apply it to the formula 3.

The results were limited to the collaborators with whose the person has collaborated together on the project at least once. We show centrality value of selected nodes in the Table 3. These centralities characterize the position of vertices in the network.

Table 3. Centralities of developers

User	Degree	Weighted degree	Closeness	Betweenness
raja4567	34	7,69353	1,92	4948,5
modder	4	0,0033	2,879	4143
raouf	7	0,0366	2,879	0

We show in the Table 4 values of $Attribute_{Score}$ for person with nickname modder, in the Table 5 for person with nickname raja4567 and in the Table 6 for person with nickname raouf.

We can immediately notice that even though ”modder” and ”raouf” do not participate on many projects with ”raja4567” (they have one common project), the AtributeScore is 0.01176471 and 0.01204819. For example ”shankar00” participate on 2 projects with ”raja4567” and the AtributeScore is 0.02469136. User ”modder” has only one common project with ”shankar00”, but AtributeScore is strong, probably because ”shankar00” not cooperate with many other persons.

Table 4. Collaborators of ”modder”

Number	Collaborators	Projects	Common projects	$Attribute_{Score}$
1	**modder**	5	5	1
2	doln	1	1	0,2
3	draculus	1	1	0,2
4	FreQi	1	1	0,2
...				
13	**shankar00**	3	1	0,1666667
...				
25	**raja4567**	81	1	0,01176471

Table 5. Collaborators of ”raja4567”

Number	Collaborators	Projects	Common projects	$Attribute_{Score}$
1	**raja4567**	81	81	1
2	senux	7	5	0,0602
3	atechnikality	8	5	0,0595
4	sagarjena	9	5	0,059
...				
15	**shankar00**	2	2	0,025
...				
408	**raouf**	3	1	0,012
...				
429	modder	5	1	0,01176471

Table 6. Collaborators of ”raouf”

Number	Collaborators	Projects	Common projects	$Attribute_{Score}$
1	**raouf**	3	3	1
2	bvencel	1	1	0,33
3	KathyWu	1	1	0,33
4	srikanth602	1	1	0,33
5	Lickie	1	1	0,33
...				
15	**raja4567**	81	1	0,01204819

4.1 Key Terms Computation for Developers

At first, we have calculated the keywords for the ”modder”, ”raja4567” and ”raouf”. We have selected only the some terms for illustration (see Table 7). For comparison we marked some terms (bold text), which was used as a context between developers.

In the Figure 1 is whole network of collaborators for the selected terms ”iphone”, ”wp7”, ”android”. The edge weights are evaluated by *Score*. This subnetwork has 199 connected components (communities) with collaborating developers.

Second part of Figure 1 shows graph of the connected component which contain selected and highlighted developers. We can see that selected developers are not in the one community of

Table 7. Key Terms for the persons "modder", "raja4567" and "raouf"

number	t_k	t_k *in* $T_{P_{modder}}$	t_k	t_k *in* $T_{P_{raja4567}}$	t_k	t_k *in* $T_{P_{raouf}}$
1	mediascout	1	licens	1	torchlight	1
2	ne	0,7800623	distribut	0,7	resx	0,7017544
3	sal	0,5980892	contributor	0,6934211	crunch	0,6028985
4	movi	0,4556041	work	0,6413794	decenc	0,333333
5	rapid	0,4191964	term	0,5994475	svt	0,3301587
6	rockethub	0,401282	notic	0,5570145	blackberri	0,2729659
7	myne	0,401282	modif	0,5359043	empti	0,2396313
...						
14					**android**	0,1438451
...						
143	**wp7**	0,1152854				
...						
1305			**android**	0,01322994		
...						
2572			**iphon**	0,00643989		
...						
2587			**wp7**	0,006402954		

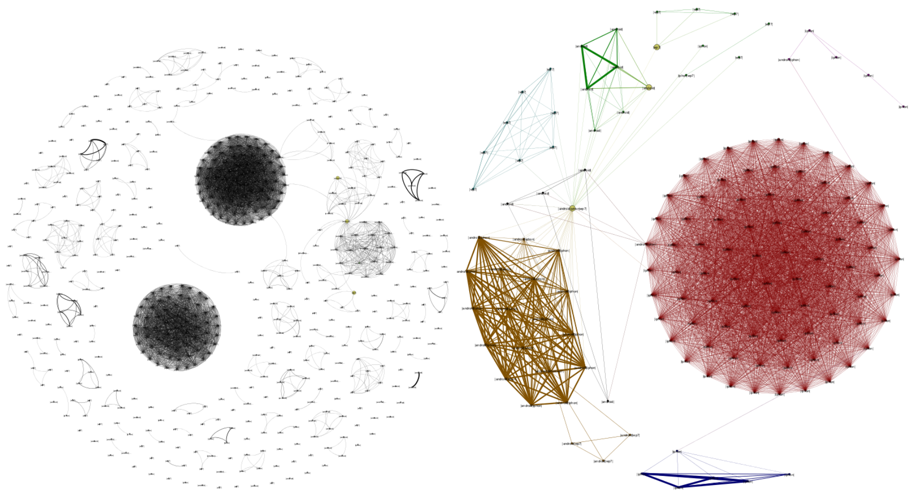

Fig. 1. Synthetic collaborators network for the terms *"iphone", "wp7", "android"* and selected subnetwork with developers "modder", "raja4567" and "raouf"

collaborators. They are connected, but the relation is too weak. They are not suitable for recommendation.

We used our algorithm for spectral clustering [7] and we detect communities of more collaborated developers. Than we can recommend the "green" community, which contain developers with the stronger relation in the context of selected words.

4.2 Subnetwork of Projects

We chose the terms ”iphone”, ”wp7”, ”android” and create subnetwork of projects. The graph of this subnetwork (see Figure 2) is not connected (contain 243 connected components) and most components are isolated vertices. When we extend the selected terms and create the new subnetwork in the context with terms ”iphone”, ”wp7”, ”android” + ”silverlight”, than is obvious a importance of the term ”silverlight” which connect more projects.

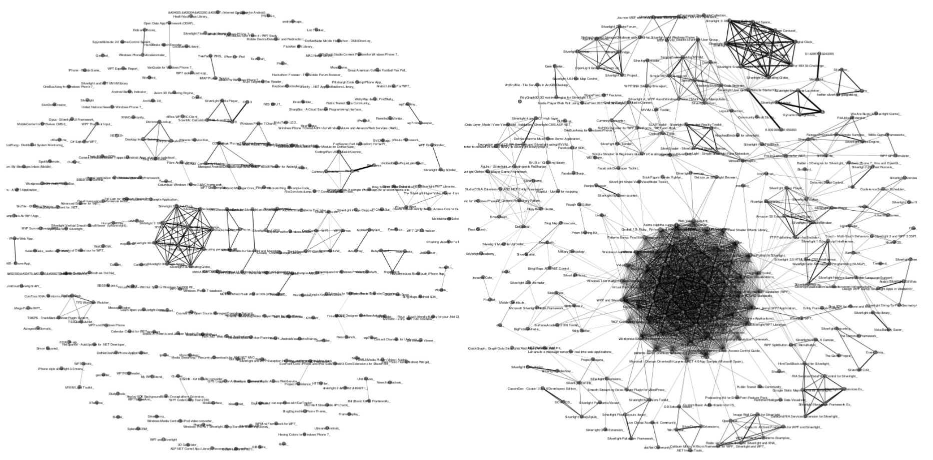

Fig. 2. Extracted subnetwork of projects for selected terms ”iphone”, ”wp7”, ”android” and the other subnetwork have terms ”iphone”, ”wp7”, ”android” + ”silverlight”

The Figure 1 and the Figure 2 were visualized using the program Gephi [1].

5 Conclusion

Research presented in this article is oriented to the strength extraction between persons based on their context in the CodePlex. The method was presented using the data collection from the CodePlex database, which contains information of the activities of developers in the project. The proposed method is usable for the development of collaboration network. The description of this network is based on the set of terms (as the input), which are used in the description of projects by the given developer. Using this method, we have obtained the new weight in the synthetic collaborators network. By means of the set of selected term, belonging to one (or more) persons, we can construct the subnetwork with only the context-related collaborators. This subnetwork can be very helpful in searching of the persons who are interested in the same area, defined by the selected term. It is usable for members of the project management, who need to find suitable developers specialized to certain area.

[1] http://gephi.org/

Acknowledgment

This work was supported by SGS, VSB – Technical University of Ostrava, Czech Republic, under the grant No. SP2012/151 Large graph analysis and processing and by the Bio-Inspired Methods: research, development and knowledge transfer project, reg. no. CZ.1.07/2.3.00/20.0073 funded by Operational Programme Education for Competitiveness, co-financed by ESF and state budget of the Czech Republic.

References

1. J. Chen, W. Geyer, C. Dugan, M. Muller, and I. Guy. Make new friends, but keep the old. *Proceedings of the 27th international conference on Human factors in computing systems CHI 09*, (1):201, 2009.
2. M.-M. Deza and E. Deza. *Dictionary of Distances*. Elsevier Science, Amsterdam, The Netherlands, Oct. 2006.
3. M. Konchady. *Text Mining Application Programming*. Charles River Media, 1 edition, May 2006.
4. Y. Long and K. Siau. Social network structures in open source software development teams. *Journal of Database Management*, 18(2):25–40, 2007.
5. S. Minks, J. Martinovic, P. Drazdilova, and K. Slaninova. Author cooperation based on terms of article titles from dblp. In *IHCI2011*, 2011.
6. M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.
7. L. Vojacek, J. Martinovic, K. Slaninova, P. Drazdilova, and J. Dvorsky. Combined method for effective clustering based on parallel som and spectral clustering. In *DATESO*, pages 120–131, 2011.