علوم رایانش و فناوری اطلاعات
نشریه علمی انجمن کامپیوتر ایران
مجلد ۱۵، شماره ۲، ۱۳۹۶
صفحات ۳۶- ۴۶
مقاله عادی

نشریه علوم رایانش و فناوری اطلاعات

مدل‌سازی تکامل شبکه‌های ارتباطات کلاسی نرم‌افزارهای شی‌گرا

مجتبی صادقیان محمد خوانساری فرید دهقان

دانشکده علوم و فنون نوین، دانشگاه تهران، تهران، ایران

چکیده

شبکه‌ی ارتباطات کلاس‌های نرم‌افزارها با نگهداری و توسعه نرم‌افزار تکامل پیدا می‌کند. مطالعه تغییرات ویژگی‌های این شبکه‌ها به فهم روند طراحی و قواعد حاکم بر رشد نرم‌افزار کمک می‌کند. هدف ما در این مقاله، مدل‌سازی تکامل شبکه ساختار نرم‌افزارهای شی‌گرا، با بررسی خصوصیات شبکه در روند تغییرات مهندسی نرم‌افزار (ایجاد و حذف کلاس‌ها) است. سیستم‌های نرم‌افزاری شی‌گرا به صورت شبکه‌ای جهت‌دار که در آن گره‌ها متناظر با کلاس‌ها و یال‌ها متناظر با ارتباطات بین آن‌هاست (شامل انجمنی، ارث بری، تحقق و تجمع)، مدل‌سازی می‌شوند. مطالعات ما نشان داده شبکه نرم‌افزار دارای خصوصیات جهان کوچک بوده و گره‌های ایجاد شده و حذف شده دارای توزیع‌های درجه ورودی قانون توانی و درجه خروجی نرمال هستند. ارزیابی مدل پیشنهادی بر روی داده‌های واقعی پروژه متن باز EGit (با ۱۳نسخه و میانگین ۷۲۰ کلاس) انجام می‌شود. نتایج این مدل‌سازی به پیش‌بینی روند توسعه نرم‌افزار و استخراج قواعد حاکم بر تکامل نرم‌افزار به توسعه‌دهندگان کمک می‌کند.

کلمات کلیدی: کاوش مخازن کد منبع نرم‌افزار، تحلیل شبکه، تکامل شبکه، شبکه‌ی ارتباطات کلاسی نرم‌افزاهای شی‌گرا.

۱- مقدمه

تکامل سیستم‌های نرم‌افزاری شی‌گرا، به تغییرات حاصل از فرایند نگهداری[۱] و حفظ عملکرد سیستم و یا اضافه کردن عملکردهای جدید با تغییر نیاز اطلاق می‌شود. مدل‌سازی تکامل نرم‌افزارها به فهم روند طراحی و توسعه نرم‌افزار، تضمین کیفیت روند توسعه، پیش‌گویی تغییرات آینده و یافتن الگوهای تغییر بهینه در هر سیستم نرم‌افزاری کمک می‌کند. نتایج مدل‌سازی می‌تواند بصورت مستقیم در فعالیت‌های مهندسی بر پایه مدل جهت تست، طراحی و توسعه سیستم‌های نرم‌افزاری استفاده شود [۱].

روش‌های مختلفی برای مدل‌سازی تکامل سیستم‌های نرم‌افزاری ارائه شده است. تحلیل آماری کدها از جمله روش‌های شناخته شده برای مطالعه تکامل نرم‌افزار هست [۲]، مدل‌های آماری با استفاده از تعداد خطوط کد[۲] نرم‌افزار [۳]، رشد اندازه بخشی از بسته‌ها و یا فایل کلاس‌های اساسی نرم‌افزار را مدل‌سازی می‌کنند. مدل‌های آماری قادر به مدل‌سازی ساختار کلی نرم‌افزار از جمله ارتباطات بین کلاس‌ها نیستند [۴]، [۵]. نمایش ساختار سیستم‌های نرم‌افزار شی‌گرا به شکل شبکه امکان مطالعه رفتار، ویژگی‌های ساختاری و تغییرات ساختار کلاسی نرم‌افزار را فراهم می‌کند [۶]، [۷].

کاوش مخازن کد منبع سیستم‌های نرم‌افزاری[۳] از منابع اصلی شناسایی الگوهای تغییرات پویای سیستم‌های نرم‌افزاری است. کاوش فایل‌های نرم‌افزاری مخازن کد و استخراج نمودار کلاسی UML پروژه‌های نرم‌افزاری، امکان نمایش شبکه‌ی ساختار نرم‌افزار در سطح کلاس را فراهم می‌کند [۱]، [۸].

در سال‌های اخیر، مطالعات فراوانی بر روی شبکه‌های ساختاری نرم‌افزارها صورت گرفته است وخواص توزیع درجات توانی، جهان کوچکی، ساختار انجمنی را بررسی کرده است [۹]. برای نمونه لی و همکاران با مشاهده خاصیت پیمانه‌ای شبکه‌های نرم‌افزار واقعی، روش اتصال پیمانه‌ای بجای پیوست تک گره برای شبیه‌سازی تکامل شبکه نرم‌افزار ارائه کردند. در این روش به منظور اتصال یال‌های ورودی و خروجی در شبکه جهت‌دار به هنگام اتصال گره‌های جدید به شبکه اولیه، از روش اتصال ترجیحی[۴] استفاده شده است [۷]. احتمال اتصال یال‌های جدید در روش اتصال پیمانه‌ای با درجه‌های ورودی و خروجی گره‌ها (میزان وابستگی کلاس‌ها) متناسب است. تعداد یال‌های اتصالی متناسب با میزان همبستگی و

جفت‌شدگی در شبکه است. شبکه تولید شده توسط روش اتصال پیمانه‌ای خاصیت جهان کوچکی را از خود نشان می‌دهد. در این روش تنها رشد سیستم‌های نرم‌افزاری برای مدل‌سازی تکامل شبکه‌های نرم‌افزاری در نظر گرفته شده و تغییراتی چون حذف کلاس‌ها نادیده گرفته شده است [۷]. لذا قادر به شبیه‌سازی رفتار واقعی سیستم‌های نرم‌افزاری نیست. چیکلیز و همکاران مدل پیش‌گویی بر پایه شبکه و با ترکیب اطلاعات پیشین تکامل نرم‌افزار و قواعد مرتبط با حوزه شبکه (برای نمونه خصوصیات مقیاس آزادی و قوانین وراثت بین کلاس‌ها) برای تکامل نرم‌افزارها ارائه شده است [۱۰]. آنها فرایندهای رشد نرم‌افزار برای نمونه ایجاد و حذف ارتباط و مفهوم ساختار انجمنی را در مدل‌سازی لحاظ کرده‌اند. این مدل در یک حلقه تکراری با محاسبه تعداد نسخه‌ها جهت شبیه‌سازی، اتصال ترجیحی، رونوشت رفتار گذشته و تاثیر دادن سن یال‌ها و قواعد حوزه، اقدام به اضافه کردن گره‌های جدید می‌کند. توجه صرف به ارتباطات بین کلاس‌ها و نمونه‌برداری از توزیع‌های تجمعی حذف/ایجاد یال‌ها از جمله محدودیت‌های این مدل‌سازی است.

جدول ۱- روش‌های پیشگویی بکار برده شده برای مدل‌سازی تکامل ساختار نرم‌افزار

مرجع	نام مقاله	روش پیشنهادی	مزایا	معایب	مجموعه داده‌ها
[۱۱]	تکامل شبکه سیستم عامل لینوکس (۲۰۱۷)	کاوش مخازن کد نرم‌افزار با استفاده از معیارهای شبکه‌های پیچیده	تحلیل بر پایه رخداد	عدم در نظر گرفتن کلاس‌ها (اصلی‌ترین مولفه سیستم‌های شی‌گرا)	LOS
[۱۳]	چهارچوبی برای کسب، مدل‌سازی ایستا و تحلیل تکامل مدل‌های نرم‌افزاری (۲۰۱۶)	مدل‌سازی ایستا داده‌های کسب شده براساس مدل ترکیبی ARMA-GARCH	شبیه‌سازی رفتار ایستا پروژه‌های نرم‌افزاری	حجم بالای اطلاعات و محاسبه پیچیده ماتریس محاسبات	Java
[۱۴]	پیش‌بینی روند تکامل نرم‌افزار Java با به کار گیری مدل‌های شبکه‌ای (۲۰۱۵)	پیشگوی بر پایه شبکه و داده‌های تکامل نرم‌افزار	تاثیر دادن سن یال‌ها مدل‌سازی اضافه کردن گره‌های جدید و تغییر ارتباطات	توجه صرف به ارتباطات بین کلاس‌ها (یال‌های شبکه) نمونه‌برداری از توزیع‌های تجمعی حذف/ ایجاد یال‌ها	aTunes FreeCol JEdit Jmol Weka
[۷]	روش اتصال پیمانه‌ای برای تکامل شبکه‌های نرم‌افزاری (۲۰۱۳)	روش اتصال ترجیحی بسته‌های کلاس‌ها	نمایش خاصیت پیمانه‌ای در شبکه‌های نرم‌افزار و مدل‌سازی اتصال ترجیحی	عدم در نظر گرفتن تغییرات انجمن‌های موجود در نرم‌افزار (حذف کلاس‌ها)	Eclipse

گروه دیگری از تحقیقات به مطالعه و تحلیل روند، رفتار و خصوصیات تکامل نرم‌افزار از دید شبکه می‌پردازند. برای نمونه زایو و همکاران تکامل شبکه سیستم عامل لینوکس[۵] را در نسخه‌های انتشاری مورد مطالعه قرار دادند [۱۱]. آن‌ها نسخه‌های انتشاری سیستم عامل را بصورت شبکه‌ی جهت‌دار نمایش دادند، که در آن توابع با گره‌ها و فراخوانی بین آنها با یال‌ها نمایش داده شده است. نتایج تحلیل نشان‌دهنده رشد خطی اندازه سیستم عامل به همراه نزول یکنواخت ضریب خوشه‌بندی است. توزیع درجه‌های ورودی و خروجی شبکه ساختاری هسته لینوکس توانی است. همچنین مطالعات آنها نشان دهنده تکامل پیمانه‌های توابع براساس هفت دنباله از تغییرات شامل مداومت، رشد، انقباض، تولد، تقسیم، حذف و ادغام است. تحلیل آماری این تغییرات، الگوی و نسبت تغییرات پیمانه‌های توابع را آشکار ساخت. جمع‌بندی از روش‌های بکار برده شده برای مدل‌سازی تکامل سیستم‌های نرم‌افزار در جدول ۱ نمایش داده شده است.

مدل‌های تکامل شبکه‌ای پیشین، عمدتاً بر روی رشد شبکه متمرکز شده [۷] و یا اضافه/حذف شدن یال‌ها را در شبکه نرم‌افزار مدل‌سازی می‌کنند [۱۰]. مطالعات بر روی تغییرات نسخه‌های متوالی پروژه‌های شی‌گرا نشان داده است که الگوهای تغییرات هر دسته از مجموعه کلاس‌های شبکه از یکدیگر متفاوت هستند. تحلیل‌های آماری معیارهای مهندسی نرم‌افزار الگوی این تغییرات را در کلاس‌های نرم‌افزاری آشکار می‌کند [۱۱].

با در نظر گرفتن ایده‌ها و محدودیت‌های مدل‌سازی‌های پیشین و استفاده از معیارها تحلیل رشد نرم‌افزار، در این مقاله مدلی برای پیشگویی تکامل شبکه‌های نرم‌افزار بر پایه شبکه ساختار کلاس‌های نرم‌افزار و تحلیل تغییرات آماری کلاس‌ها در روند توسعه سیستم‌های نرم‌افزاری ارائه می‌شود. هدف مدل‌سازی تکامل شبکه‌های ارتباطات کلاسی نرم‌افزارهای شی‌گرا با استفاده از ابزارها و مفاهیم تحلیل شبکه‌های اجتماعی [۱۲] و مطالعه تغییرات شبکه‌ای ساختار در روند توسعه و نگهداری سیستم‌های نرم‌افزاری است. ما ازشبکه ارتباطات کلاس‌های نرم‌افزار که در آن گره‌ها متناظر با کلاس‌ها که دارای خصوصیت[۶] سن و یال‌ها نشان‌دهنده ارتباطات بین آنها (شامل انجمنی[۷]، ارثبری[۸]، تحقق[۹] و تجمع[۱۰]) است، برای مدل‌سازی تکامل سیستم‌های نرم‌افزاری استفاده می‌کنیم.

نوآوری مدل پیشنهادی ارائه شده، استفاده از شبکه‌های جهت‌دار نرم‌افزار به‌منظور نمایش انتقال اطلاعات و فراخوانی‌های بین پارامترهای کلاس‌ها است. علاوه بر این، ما از چهار ارتباط اساسی بین کلاس‌های نرم‌افزارهای شی‌گرا به منظور نمایش یال‌های کلاس‌ها در شبکه نرم‌افزار استفاده می‌کنیم. مدل پیشنهادی تغییرات گره‌های شبکه (ایجاد و حذف گره‌ها) را به منظور پیشگویی روند تکامل شبکه شبیه‌سازی می‌کند. در این مقاله، روند تکامل پروژه متن باز EGit[۱۱] از خانواده پروژه‌های eclipse مورد مطالعه و تحلیل قرار گرفته است. همچنین، صحت و دقت مدل ارائه شده در پیشگویی نسخه‌های متوالی پروژه EGit مورد ارزیابی قرار می‌گیرد.

در ادامه مقاله، مدل پیشنهادی در بخش ۲ شرح داده می‌شود، ارزیابی نتایج شبیه‌سازی مدل پیشنهادی بر روی شبکه‌های واقعی در بخش ۳ ارائه می‌شود. در نهایت بخش ۴، شامل نتیجه‌گیری و پیشنهادها برای کارها آتی است.

۲- مدل تکاملی شبکه ساختار نرم‌افزار

در این بخش نخست مجموعه داده‌های مورد استفاده (پروژه EGit) شرح داده می‌شود. شبکه ارتباطات کلاسی پروژه EGit توصیف و خصوصیات آماری این شبکه مورد مطالعه قرار می‌گیرد وتغییرات آماری کلاس‌ها در طول تکامل پروژه EGit ارزیابی می‌گردد. در نهایت براساس تحلیل‌های کسب شده، مدل تکامل پیشنهادی ارائه می‌گردد.

۲-۱- مجموعه دادگان

در این مقاله ما از پروژه‌ی متن باز eclipse جهت نمایش شبکه نرم‌افزار استفاده می‌کنیم. Eclipse یک محیط توسعه نرم‌افزاری چندزبانه برای محیط توسعه مجتمع با قابلیت اضافه کردن افزونه هست. این محیط توسعه در ابتدا با زبان جاوا و برای توسعه برنامه‌های این زبان استفاده می‌شده است. EGit یک سرویس از سوی تیم ارائه‌دهندگان eclipse برای سیستم کنترل نسخه Git است. Git یک[۱۲] SCM توزیع شده است که به توسعه‌دهندگان این امکان را می‌دهد تا یک کپی کامل از تمام گذشته هر نسخه از کد در اختیار داشته باشند. EGit انجام

جستجوها در گذشته کد را بسیار سریع و تطبیق‌پذیر می‌کند. این پشتیبانی از طریق مجموعه‌ای از افزونه‌های EGit_project ارائه شده است. قابلیت EGit بر پایه کتابخانه JGit است. ما از ۱۳ نسخه پروژه EGit به شرح جدول ۲ استفاده می‌کنیم.

۲-۲- شبکه‌ی ارتباطات کلاسی نرم‌افزار

کلاس‌ها اجزای اصلی در سیستم‌های نرم‌افزاری شی‌گرا هستند. در این مقاله ما از شبکه جهت‌دار ارتباطات بین کلاس‌های نرم‌افزار استفاده می‌کنیم. گره‌ها در این شبکه متناظر با کلاس‌ها هستند که با نام کلاس‌ها برچسب خورده و حاوی خصوصیت سن هر کلاس هستند. سن هر گره متناسب با تعداد گام‌های است که کلاس مفروض در نسخه‌های متوالی توسعه پروژه حضور دارد و دچار تغییر نشده است. برای نمونه در پروژه EGit یک کلاس خاص در نسخه v2.0.0 دارای سن یک و با فرض عدم‌تغییر همان کلاس در روند توسعه در نسخه v4.3 دارای سن ۱۳ خواهد بود.

جدول ۲- نسخه‌های پروژه EGit

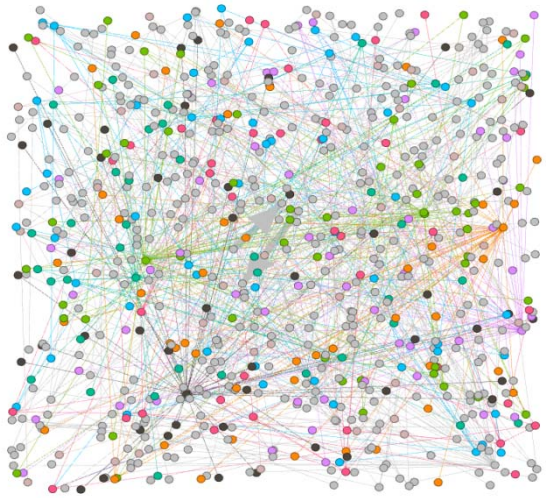
Num.	Tags	Release Date	Size
1	v2.0.0	2012-05-31 01:56:11	11.0 MB
2	v3.0.0	2013-05-08 17:04:52	11.8 MB
3	v3.1.0	2013-09-27 15:11:43	11.9 MB
4	v3.2.0	2013-11-13 18:42:22	12.2 MB
5	v3.3.0	2014-03-03 03:08:25	12.4 MB
6	v3.4	2014-05-06 02:32:29	13.7 MB
7	v3.5.0	2014-09-26 10:38:59	13.9 MB
8	v3.6.0	2014-11-12 19:41:59	14.0 MB
9	v3.7.0	2015-02-04 02:31:49	14.2 MB
10	v4.0.0	2015-03-23 21:36:36	14.3 MB
11	v4.1	2015-09-28 12:15:45	15.1 MB
12	v4.2	2015-11-11 01:32:52	15.2 MB
13	v4.3	2016-03-23 16:15:38	15.6 MB

یال‌های بین گره‌های شبکه ارتباطات جهت‌دار بین کلاس‌های سیستم نرم‌افزاری را نمایش می‌دهند. استفاده از یال‌های جهت‌دار در ساختار شبکه امکان نمایش دقیق‌تر جریان و انتقال اطلاعات (برای مثال ساختار فراخوانی‌ها، انتقال پیام و وراثت) را فراهم می‌کند [۱۵]. مدل پیشنهادی از چهار ارتباط اساسی بین کلاس‌های نرم‌افزار به نام‌های انجمنی، ارث بری، تحقق و تجمع (کسب شده از تجمیع ماتریس مجاورت ارتباطات در نمودار کلاسی UML) برای نمایش ارتباطات بین کلاس‌ها در شبکه نرم‌افزارهای شی‌گرا استفاده می‌کند. شکل ۱ نمونه‌ای از شبکه ساختار کلاسی نسخه v4.3.0 پروژه EGit را نمایش می‌دهد.

شبکه ساختار کلاسی نرم‌افزار به صورت زیر نمایش داده می‌شود.

$$N = (V, E) \quad (۱)$$

که در آن G گراف جهت‌دار ساختار کلاس نرم‌افزار، $V = \{v_1, v_2, \dots\}$ مجموعه گره‌های شبکه که نمایانگر کلاس‌های نرم‌افزار و $E = \{e_{i,j}, \dots\}$ مجموعه یال‌های شبکه که در آن $e_{i,j}: v_i \rightarrow v_j$ نشان دهنده ارتباطات از کلاس مبدا v_i به کلاس مقصد v_j است. خواص و رفتار شبکه ساختار کلاسی نرم‌افزار در ادامه تحلیل می‌گردد. ویژگی اساسی شبکه‌های ارتباطات کلاسی نرم‌افزار EGit الگوی مشابه توزیع درجه توانی گره‌ها (مشاهده شده در شبکه‌های مقیاس آزاد[۱۳]) است. وجود خصوصیات شبکه‌های مقیاس آزاد و توزیع توانی معمولاً منسوب به حضور اتصال ترجیحی است. در زمینه نرم‌افزارهای شی‌گرا، این مدل تکامل نمایانگر آن است که کلاس‌های که نقش مرکزی در سیستم دارند (در نقش <<خدا[۱۴]>> که سرویس‌هایی را به کلاس‌های مشتری ارائه می‌کند) نقش جذب‌کننده را برای کلاس‌های جدید دارند. با تکامل گام به گام نرم‌افزار این پدیده، موجب اتصال درصد زیادی از کلاس‌های جدید به کلاس‌های با درجه ورودی بالا می‌شود و در نتیجه گره‌های (کلاس‌های) مهم در شبکه مهم باقی می‌مانند که این امر نشانه پایداری طراحی سیستم‌های نرم‌افزاری هست، به این معنی است که رفتار بسیاری از اشیاء[۱۵] در طول زمان پایدار خواهند بود و نیازی به تغییر نیست [۷]، [۱۴].



شکل ۱- شبکه ساختار کلاس پروژه EGit نسخه v4.3.0 با ۷۹۹ گره و ۸۴۲ یال

تفاوت اساسی بین شبکه‌های ساختاری نرم‌افزار و دیگر شبکه‌های پیچیده همبستگی[۱۶] منفی بین درجه‌های ورودی و خروجی در شبکه‌های نرم‌افزاری است [۷]. عدم‌تقارن بین توزیع درجه‌های ورودی و خروجی نشان می‌دهد که گره‌های با درجه بزرگتر غالباً دارای درجه خروجی کوچک هستند و برعکس. به علاوه، گره‌های با درجه ورودی بزرگ معمولا وظایف ساده و بنیادی را انجام می‌دهند. کلاس‌های متناظر با این گره‌ها مکرراً استفاده می‌شوند و ضرورتاً وابسته به دیگر کلاس‌ها نیستند. در مقابل، گره‌های با درجه خروجی بزرگتر معمولا به گره‌های زیادی وابسته‌اند و دارای ساختار داخلی پیچیده‌ای هستند و بنابراین احتمال وابستگی به دیگر کلاس‌ها بسیار اندک است [۱۶].

مطالعه بر روی شبکه‌های نرم‌افزارهای شی‌گرا نشان‌دهنده ساختار انجمنی[۱۷] در این شبکه‌ها است [۹]. شبکه نرم‌افزار در طول توسعه و تکامل خود این ساختار انجمنی را حفظ می‌کند و این به دلیل توسعه نرم‌افزارهای شی‌گرا براساس فرضیه <<لگو[۱۸]>> است. این فرضیه بیانگر ساخت نرم‌افزار بر پایه تعداد زیادی از انجمن‌های کوچک و مستقل از هم است [۱].

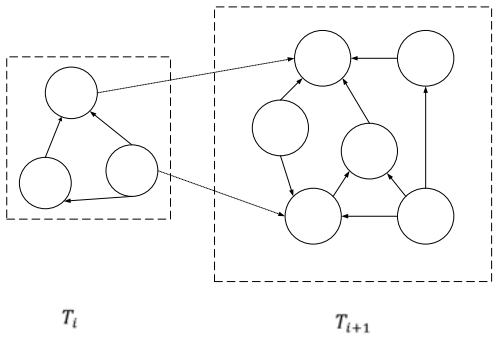
۲-۳- تغییرات کلاس‌های نرم‌افزارهای شی‌گرا

در این مقاله ما توسعه و تکامل سیستم‌های نرم‌افزاری شی‌گرا را براساس دو تغییر اساسی ایجاد و حذف کلاس‌ها مطالعه می‌کنیم.استخراج و شناسایی این تغییرات در نسخه‌های متوالی پروژه‌های نرم‌افزار با استفاده از مخازن کد منبع و نگاشت آنها

به شبکه ساختاری نرم‌افزار به ما در فهم الگوی تکامل ساختار نرم‌افزارهای شی‌گرا کمک می‌کند.

۲-۳-۱- ایجاد کلاس‌های جدید

اساس رشد سیستم‌های نرم‌افزاری بر پایه اضافه شدن مجموعه کلاس‌های جدید به ساختار پروژه نرم‌افزار به‌منظوراضافه کردن عملکردهای جدید از سوی کاربران و توسعه‌دهندگان است، (شکل ۲ اضافه شدن کلاس‌ها به شبکه نرم‌افزار را نمایش می‌دهد). برای نمونه وابستگی maven به اجزاء نمایه‌ساز در نسخه v4.2.0 پروژه EGit توسط توسعه‌دهندگان به عنوان ابزاری برای خودکارسازی ساخت اضافه گردیده است.



شکل ۲- ایجاد کلاس‌های جدید

مقایسه کلاس‌های نسخه‌های متوالی پروژه‌های نرم‌افزاری شی‌گرا می‌تواند به شناسایی کلاس‌های تازه اضافه شده کمک کند. بصورت عملی با مقایسه لیست کلاس‌های نرم‌افزار در دو نسخه متوالی T_i و T_{i+1} می‌توان تعداد کلاس‌های جدید اضافه شده به نسخه متوالی به همراه لیست کامل این کلاس‌ها را کسب کرد، (جدول ۳).

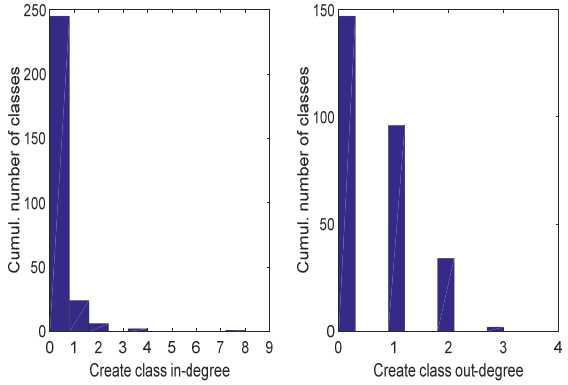
جدول ۳- دنباله تعداد کلاس‌های اضافه شده

Projects	Tags	Creation
EGit	v2.0.0.	-
	v3.0.0	49
	v3.1.0	4
	v3.2.0	34
	v3.3.0	16
	v3.4.0	15
	v3.5.0	21
	v3.6.0	6
	v3.7.0	9
	v4.0.0	8
	v4.1.0	89
	v4.2.0	5
	v4.3.0	31

مشاهد رفتار گره‌های متناظر با کلاس‌های جدید در شبکه نسخه متوالی (T_{i+1}) پروژه نرم‌افزار، به استخراج الگوی اتصال گره‌های جدید به شبکه کمک می‌کند. برای مدل‌سازی اتصال کلاس‌های جدید به شبکه نرم‌افزار ما خصوصیات (توزیع درجات ورودی/ خروجی) تمام کلاس‌های ایجاد شده (در ۱۳ نگارش متوالی) را مطالعه می‌کنیم. در هرگام تکامل شبکه نرم‌افزار، با شناسایی گره‌های متناظر با کلاس‌های ایجاد شده (با استفاده از برچسب هر گره) تعداد همسایه‌های ورودی و خروجی گره‌ها شمارش و نمودار فراوانی تجمعی برای درجه‌های ورودی و خروجی مجموع تمام کلاس‌های ایجاد شده در روند تکامل شبکه نرم‌افزار EGit ترسیم می‌شود.

شکل ۳ مجموع تعداد کلاس‌های ایجاد شده (اضافه شده به شبکه) از هر درجه برای درجات ورودی و خروجی را بصورت مجزا نمایش می‌دهد. نمودار حاصله توزیع درجه‌های ورودی/خروجی تمام کلاس‌های ایجاد شده در اتصال به شبکه‌های پیشین برای پروژه EGit را آشکار می‌کند.

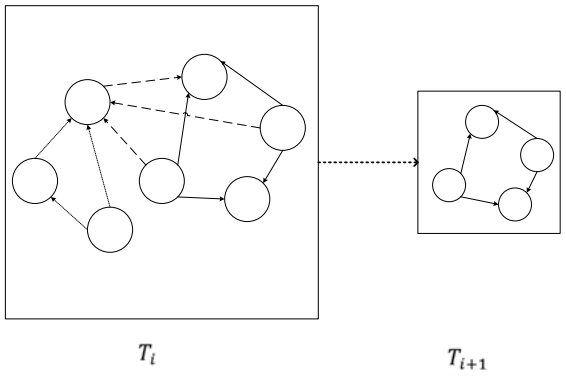
مدل تکامل پیشنهادی با بهره‌گیری از توزیع‌های درجات گره‌ها روند ایجاد گره‌های جدید در شبکه نرم‌افزار را نشان می‌دهد.



شکل ۳- فراوانی درجه‌های کلاس‌های تازه اضافه شده

۲-۳-۲- حذف کلاس‌های قدیمی

قدیمی شدن برخی از عملکردها در روند توسعه نرم‌افزار و یا عدم‌نگهداری طولانی مدت منجر به حذف کلاس‌های از ساختار نرم‌افزار می‌شود، برای نمونه حذف ارجاع دهنده HoverManager در نسخه v3.6.0. شکل ۴ نشان‌دهنده مفهوم حذف کلاس‌ها در شبکه نرم‌افزار است.



شکل ۴- حذف کلاس‌های قدیمی

کلاس‌های حذف شده به آسانی قابل شناسایی‌اند (با مقایسه نام کلاس‌ها در نسخه‌های متوالی). دنبال کردن لیست کلاس‌های حذف شده در نسخه‌های متوالی توسعه نرم‌افزارهای شی‌گرا امکان تحلیل الگوی حذف این کلاس‌ها در ساختار

شبکه نرم‌افزار را برای ما فراهم می‌کند. همانند تحلیل ایجاد کلاس‌های جدید، به منظور مدل‌سازی حذف کلاس‌ها در روند تکامل شبکه نرم‌افزار، مجموع تمام کلاس‌های حذف شده پیشین شناسایی و تحلیل می‌شوند. جدول ۴ تعداد کلاس‌های حذف شده در گام‌های توسعه پروژه نرم‌افزار (۱۳ نگارش پروژه EGit) را به ما نمایش می‌دهد.

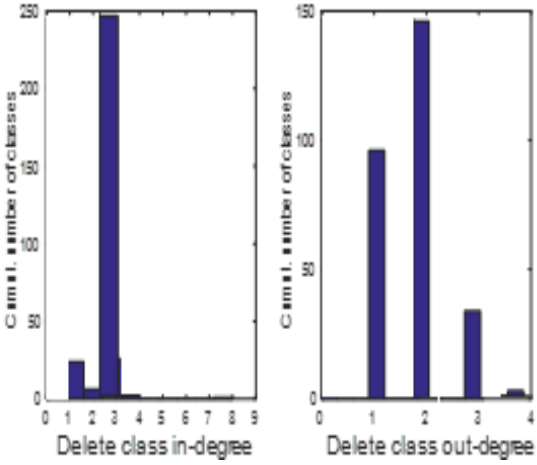
مطالعات ما بر روی شبکه کلاس‌های نرم‌افزار نشان‌داد احتمال حذف شدن کلاس‌ها وابسته به درجات گره‌های متناظر در شبکه سیستم‌افزاری است.

جدول ۴- دنبالع تعداد کلاس‌های حذف شده

Projects	Tags	Deletion
EGit	v2.0.0	-
	v3.0.0	5
	v3.1.0	3
	v3.2.0	16
	v3.3.0	10
	v3.4.0	2
	v3.5.0	6
	v3.6.0	0
	v3.7.0	1
	v4.0.0	1
	v4.1.0	5
	v4.2.0	0
	v4.3.0	12

شکل ۵، مجموع تعداد کلاس‌های حذف شده (گره‌های حذف شده از شبکه) برای درجات ورودی و خروجی را بصورت مجزا نمایش می‌دهد. نمودار حاصله توزیع درجه‌های ورودی/خروجی تمام کلاس‌های حذف شده در اتصال به شبکه‌های پیشین برای پروژه EGit را آشکار می‌کند.

الگوهای توزیع‌های بدست آمده به مدل‌سازی حذف گره‌ها در روند تکامل شبکه کمک می‌کند.



شکل ۵- توزیع درجه‌های کلاس‌های حذف شده

۲-۴- مدل تکامل ساختاری پیشنهادی

در این بخش مدل تکامل پیشنهادی بر پایه توزیع‌های کسب شده از دو نوع تغییر، ایجاد یا حذف گره‌های متناظر با کلاس‌ها ارائه می‌گردد. شبکه اولیه برای اعمال الگوهای تغییرات در روش پیشنهادی، نمودار UML ارتباطات کلاسی پروژه واقعی نرم‌افزار است. برخلاف مدل‌سازی‌های پیشین ما از ۴ ارتباط اساسی بین کلاس‌های نرم‌افزاری (شامل انجمنی، ارث بری، تحقق و تجمع) برای ترسیم ارتباطات بین گره‌های شبکه استفاده کردیم، که باعث شده شبکه ترسیم شده شباهت بیشتری به مدل واقعی سیستم نرم‌افزاری از جنبه نمایش ارتباطات کلاسی داشته باشد. فرایند کلی مدل‌سازی تغییرات ساختار نرم‌افزار در شکل ۶ نمایش داده شده است.

شکل ۶- مراحل مدل پیشنهادی تکامل ساختاری

در اینجا تغییرات اشاره به فرایند ایجاد و حذف کلاس‌های نرم‌افزاری دارد. در ادامه مدل‌سازی هر یک از دو تغییر اساسی در روند توسعه نرم‌افزار بر روی شبکه نرم‌افزار تشریح می‌شود.

۲-۴-۱- مدل‌سازی ایجاد گره‌های جدید

ایجاد کلاس‌های جدید در روند رشد نرم‌افزار شامل اضافه شدن مجموعه‌ای از کلاس‌ها به ساختار اصلی نرم‌افزار است. مجموعه کلاس‌های جدید می‌تواند تک کلاس و یا تعدادی از کلاس‌ها (بسته) باشند که با یکدیگر ارتباط دارند. تعداد گره‌های تازه اضافه شده به شبکه نرم‌افزار با محاسبه توزیع احتمالی گسسته بر روی مجموعه دنباله داده‌های تعداد کلاس‌های ایجاد شده در هر گام تکاملی (جدول ۳) بدست می‌آید. تابع جرم احتمال برای وقوع هر یک از اعداد دنباله تعداد کلاس‌های ایجاد شده توسط رابطه ۱ محاسبه می‌شود.

$$P(C) = Num.ofCreatedClasses / Sum(Num.ofCreatedClasses) \quad (۲)$$

که در آن Num.of Created Classes تعداد کلاس‌های ایجاد شده در هر نسخه انتشاری پروژه نرم‌افزار و P(Num.of Created Classes) فراوانی نسبی تعداد کلاس‌های ایجاد شده است.

هر کلاس در مجموعه کلاس‌های تازه ایجاد شده با توجه به نقش و عملکردی که در سیستم دارد با مجموعه‌ای از کلاس‌های نرم‌افزار ارتباط خواهد داشت، بنابراین گام بعدی تشخیص تعداد و نوع ارتباط کلاس‌های تازه ایجاد شده (درجه ورودی/خروجی گره‌های متناظر در شبکه نرم‌افزار) است.

با فرض تک عضوی نبودن مجموعه گره‌های تازه ایجاد شده، مطالعات نشان می‌دهد که زیر شبکه ایجاد شده در روند توسعه نرم‌افزار توزیع درجه ورودی/خروجی توانی دارد (شکل ۳). لذا با کسب دنباله درجات ورودی/خروجی و نگاشت آن بر روی توزیع احتمال توانی، دنباله درجات ورودی/خروجی مجموعه گره‌های تازه ایجاد شده اضافه شده به ساختار شبکه اصلی نرم‌افزار محاسبه می‌شود.

فرض کنید $P(x) = Cx^n$ توزیع توانی استخراج شده از دنباله درجات کلاس‌های تازه ایجاد شده است که در آن $x \in [x_0, x_1]$. دنباله تعداد درجات ورودی/خروجی برای هریک از کلاس‌های جدید ایجاد شده توسط رابطه ۳ محاسبه می‌شود:

$$\text{Rand\_PowerLaw}(x) = [(x_1^{n+1} - x_0^{n+1})y + x_0^{n+1}]^{\frac{1}{(n+1)}} \qquad (۳)$$

که در آن y متغیر با توزیع یکنواخت با بازه $[0,1]$، n توان توزیع، x_0, x_1 محدوده توزیع و x متغیر توزیع شده توانی (درجه گره‌های تازه ایجاد شده) است.

برای اتصال گره‌های تازه ایجاد شده به شبکه نرم‌افزار احتمال اتصال به گره‌های همسایه در شبکه نرم‌افزار محاسبه می‌شود. احتمال انتخاب هر گره متناسب با مقادیر تابع جرم احتمال برای انتخاب گره‌های شبکه نرم‌افزار به عنوان همسایه‌های گره‌های تازه ایجاد شده است. بررسی‌های ما نشان داده است که احتمال انتخاب هر گره همسایه متناسب با توزیع احتمال درجه‌های ورودی/خروجی و سن کلاس‌ها است. بنابراین برای همسایه‌های خروجی (کلاس‌هایی که به آنها وابسته‌اند) داریم:

$$p_{Out\_neighbor} = p_{Network\_class\_indegree} * p_{Network\_class\_age} \qquad (۴)$$

که در آن $p_{Network\_class\_indegree}$ احتمال شرطی انتخاب گره با درجه ورودی خاص (احتمال رخداد گره i با درجه ورودی indegree در دنباله درجات ورودی کلاس‌های تازه ایجاد شده به شرط احتمال رخ داد همان گره با درجه ورودی مشخص در شبکه نرم‌افزار، فرمول ۵)، $p_{Network\_class\_age}$ احتمال شرطی انتخاب گره با سن خاص (احتمال رخ داد گره با سن age در دنباله سن کلاس‌های تازه ایجاد شده به شرط احتمال رخ داد همان گره با سن مشخص در شبکه نرم‌افزار) و $p_{Out\_neighbor}$ مقدار تابع جرم احتمال برای انتخاب گره‌ها به عنوان همسایه خروجی است.

$$p_{Network\_class\_indegree} = \frac{p_{\text{degree i between New class indegree}}}{p_{\text{degree i in whole network}}} \qquad (۵)$$

بطور مشابه برای همسایه‌های ورودی گره‌های تازه ایجاد شده (کلاس‌های که وابسته به کلاس‌های جدیداند) داریم:

$$p_{In\_neighbor} = p_{Network\_class\_outdegree} * p_{Network\_class\_age} \qquad (۶)$$

که در آن $p_{Network\_class\_outdegree}$ احتمال شرطی انتخاب گره با درجه خروجی خاص (احتمال رخداد گره با درجه خروجی out_degree در دنباله درجات خروجی کلاس‌های تازه ایجاد شده به شرط احتمال رخ داد همان گره با درجه خروجی مشخص در شبکه نرم‌افزار، فرمول ۷، $p_{Network\_class\_age}$ احتمال شرطی انتخاب گره با سن خاص و $p_{In\_neighbor}$ مقدار تابع جرم احتمال برای انتخاب گره‌ها به عنوان همسایه ورودی است.

$$p_{Network\_class\_outdegree} = \frac{p_{\text{degree i between New class outdegree}}}{p_{\text{degree i in whole network}}} \qquad (۷)$$

با در نظر گرفتن تعداد درجه‌های ورودی/خروجی هر گره تازه ایجاد شده و براساس مقادیر کسب شده برای تابع جرم احتمال همسایه‌ها و توزیع احتمال گسسته، همسایه‌های هریک از گره‌های اضافه شده به شبکه نرم‌افزارتعیین می‌شوند.

مطالعه مجموع کلاس‌های اضافه شده (بسته‌های نرم‌افزاری) در هر گام نشان داده است که مجموعه کلاس‌های تازه ایجاد شده زیر شبکه‌ای متصل از گره‌ها با توزیع درجات ورودی/ خروجی توانی هستند. دنباله درجات ورودی/خروجی گره‌های شبکه مشابه درجه‌های اتصالی گره‌ها به شبکه نرم‌افزار از رابطه ۳ بدست می‌آید. در ادامه با استفاده از الگوریتم [۱۷] زیرشبکه جهت‌دار گره‌های تازه ایجاد شده ساخته می‌شود. الگوریتم مذکور با استفاده از دنباله‌ی درجات گراف نخست دنباله را به صورت نزولی مرتب کرده و سپس در هرگام بزرگترین عدد دنباله را حذف و از بقیه رئوس یک واحد کم می‌کند (مشتقات پیاپی دنباله) تا در مرحله اخر به دنباله‌ای می‌رسد که تمام جملاتش صفر است.

۲-۴-۲- مدل‌سازی حذف گره‌های قدیمی

حذف گره‌های شبکه نرم‌افزاری شامل دو مرحله است: ۱) تعیین تعداد گره‌های که باید حذف شوند. ۲) شناسایی و حذف گره‌ها از شبکه. مشابه مدل‌سازی ایجاد گره‌ها، تعداد گره‌های حذف شده در هر گام، با استفاده از توزیع احتمالی گسسته دنباله تعداد کلاس‌های حذف شده در پروژه‌ی واقعی نرم‌افزار (جدول ۴) محاسبه می‌شود.

مطالعات ما نشان‌دهنده وابستگی تابع الگوی حذف یک گره خاص به ۳ پارامتر متمایز است. پارامتر اول احتمال حذف مربوط به خود گره به شرط داشتن ویژگی‌های خاص (با معیارهای درجات و سن گره‌ها)، پارامتر دوم احتمال حذف گره به شرط داشتن همسایه‌های خروجی خاص و پارامتر سوم احتمال حذف گره به شرط داشتن همسایه‌های ورودی خاص است. تحلیل آماری دنباله درجات حذف شده نشان‌دهند توزیع درجه ورودی توانی و توزیع درجه خروجی نرمال است (شکل ۵). به‌صورت عملی احتمال حذف گره در شبکه توسط رابطه ۸ محاسبه می‌شود.

$$p_{Delete\_node} = p_{Delete\_Network\_node} + p_{Delete\_node\_outneighbor} + p_{Delete\_node\_inneighbor} \qquad (۸)$$

که در آن $p_{Delete\_Network}$ احتمال حذف گره با در نظر گرفتن ویژگی‌های شبکه‌ای، $p_{Delete\_outneighbor}$ احتمال حذف گره با در نظر گرفتن احتمال همسایه‌های خروجی، $p_{Delete\_inneighbor}$ احتمال حذف گره i با در نظر گرفتن احتمال همسایه‌های ورودی و p_{Delete_node} مقدار تابع جرم احتمال برای انتخاب گره‌ها جهت حذف شدن است.

ویژگی‌های شبکه‌ای گره متناسب با درجه ورودی، خروجی و سن گره است، لذا سه صفت درجات ورودی و درجات خروجی و سن کلاس‌ها درکل باهم تعیین‌کننده احتمال حذف یک گره $p_{Delete\_Network}$ هستند، رابطه ۹.

$$p_{Delete\_Network\_node} = p_{Delete\_class\_indegree} * p_{Delete\_class\_outdegree} * p_{Delete_{class_{age}}} \quad (۹)$$

که در آن $p_{Delete\_class\_indegree}$ احتمال شرطی حذف گره با درجه ورودی خاص، $p_{Delete\_class\_outdegree}$ احتمال شرطی حذف گره با درجه خروجی خاص، $p_{Delete\_class\_age}$ احتمال شرطی حذف گره با سن خاص (احتمال حذف گره با سن age در دنباله سن کلاس‌های حذف شده به شرط احتمال رخ داد همان گره با سن مشخص در شبکه نرم‌افزار) است.

$$p_{Delete\_class\_indegree} = \frac{cdf\_power-law(indegree)}{p(indegree)} \quad (۱۰)$$

$p_{Delete\_class\_indegree}$ احتمال حذف گره با درجه ورودی خاص یک احتمال شرطی برای انتخاب گره با درجه ورودی خاص از لیست دنباله درجات ورودی گره‌های حذف شده که توسط تابع توزیع تجمعی توانی به شکل $cdf\_power - law(indegree)$ (که از نگاشت بر روی دنباله درجات ورودی گره‌های حذف شده بدست می‌آید) محاسبه می‌شود به شرط انتخاب گره‌ای با درجه ورودی خاص در شبکه نرم‌افزار است.

$$p_{Delete\_class\_outdegree} = \frac{cdf\_Gaussian(outdegree)}{p(outdegree)} \quad (۱۱)$$

$p_{Delete\_class\_outdegree}$ احتمال حذف گره با درجه خروجی خاص یک احتمال شرطی برای انتخاب گره با درجه خروجی خاص از لیست دنباله درجات خروجی گره‌های حذف شده که توسط تابع توزیع تجمعی نرمال، $cdf\_Gaussian(outdegree)$ (که از نگاشت بر روی دنباله درجات خروجی گره‌های حذف شده بدست می‌آید) محاسبه می‌شود به شرط انتخاب گره‌ای با درجه خروجی خاص در شبکه نرم‌افزار است.

تحلیل روند حذف کلاس‌ها نشان‌دهنده همبستگی بین رفتار همسایه‌ها و گره حذف شده است. بنابراین، احتمال حذف یک گره وابسته به احتمال انتخاب همسایه‌های گره‌های حذف شده است، لذا داریم:

$$p_{Delete\_node\_inneighbor} = p_{node\_inneighbor\_outdegree} * p_{node_{inneighbor_{age}}} \quad (۱۲)$$

$$p_{Delete\_node\_outneighbor} = p_{node\_outneighbor\_indegree} * p_{node\_outneighbor\_age} \quad (۱۳)$$

که در آن $p_{Delete\_node\_inneighbor}$، $p_{node\_outneighbor\_indegree}$ احتمال‌های شرطی درجه‌های همسایه‌های گره‌های شبکه با درنظر گرفتن توزیع درجه‌های همسایه‌های گره‌های حذف شده در پروژه‌ی نرم‌افزاری واقعی (مجموعه دادگان) و $p_{node\_outneighbor\_age}$ احتمال شرطی سن همسایه‌های گره حذف شده (کسب شده از احتمال شرطی سن همسایه‌های گره‌های حذف شده در پروژه‌ی نرم‌افزاری واقعی است. در نهایت با محاسبه این احتمالات برای تمام گره‌های شبکه و استفاده از توزیع احتمال گسسته مدل قادر به شناسایی و حذف کلاس‌ها در روند توسعه نرم‌افزار خواهد بود.

۲-۵- تجمیع مدل‌سازی‌های تغییرات

فرایند شبیه‌سازی تکامل ساختار نرم‌افزار شامل تجمیع مدل‌سازی تغییرات حاصله بر روی شبکه اولیه (شبکه واقعی نسخه نرم‌افزار) و تکرار روند محاسبه احتمال‌ها و شناسایی گره‌ها با استفاده از توزیع‌های بدست آمده براساس تعداد گام‌های شبیه‌سازی است.

Algorithm: Evolution model
(N, list of changing features)

1. n.create, n.delete = Calculate_Changes_Number (list of all changes' number)
2. Delete_classes(N, n. delete, list of deleted classes' degrees & ages)
3. Find_neighbors(N, indegree, outdegree, list of classes' degrees & ages)
4. Create_classes(N, n. create, list of created classes' degrees & ages)
5. Aging_classes(N)
6. Return N

در گام اول تعداد کلاس‌های که باید حذف و ایجاد شوند محاسبه می‌شود. Calculate_Changes_Number تعداد کلاس‌های کاندیدای تغییرات ایجاد و حذف را با پارامتر ورودی لیست‌های تعداد کلاس‌های ایجاد شده و حذف شده مشخص می‌نماید (فرمول ۱). Delete_classes تابع حذف کلاس‌های کاندید ورودی تابع شبکه نرم‌افزار تعداد کلاس‌های کاندید حذف و لیست درجات و سن کلاس‌های حذف شده تا کنون است. Create_classes تابع ایجاد کلاس‌ها با پارامترهای ورودی شبکه نرم‌افزار، تعداد کلاس‌های کاندید برای ایجاد و لیست درجات و سن کلاس‌های ایجاد شده تا کنون است. Find_neighbors همسایه‌های گره‌های جدید در شبکه ساختاری نرم‌افزار را مشخص می‌کند. به منظور محاسبه سن کلاس‌ها، تابع Aging_classes تعریف شده است.

این تابع در هر گام شبیه‌سازی شبکه نرم افزار یک واحد به خصوصیت سن گره‌های شبکه اضافه می‌کند (برای کلاس‌های ایجاد شده در هرگام نخست سن واحد را تخصیص می‌دهد). گام‌های ۱ تا ۶ برای ایجاد شبکه نرم‌افزار در هر گام توسعه (نسخه‌های متوالی) تکرار می‌شوند و در هر گام ورودی شبکه نرم‌افزار شبکه ایجاد شده در مرحله قبلی است.

پیچیدگی زمانی الگوریتم پیشنهادی وابسته به مرحله ساخت زیرشبکه (بسته‌های نرم‌افزاری) است که برابر با پیچیدگی زمانی الگوریتم مرجع [۱۷] یا همان $o(n \log n)$ است.

۳- نتایج شبیه‌سازی

ارزیابی مدل پیشنهادی با استفاده از ابزارها و سنجه‌های حوزه علوم شبکه با بکارگیری دادگان واقعی انجام می‌گیرد. عملکرد مدل پیشنهادی در مقایسه با شبکه‌های شبیه‌سازی شده توسط مدل اتصال پیمانه‌ای [۷] ارزیابی می‌شود.

با توجه به اینکه سایر روش‌های مدل‌سازی مطالعه شده بر پایه تحلیل توابع نرم‌افزار [۱۱]، تحلیل آماری تکامل نرم‌افزار [۱۳] و یا مدل‌سازی‌های وابسته به قواعد دامنه مجموعه داده‌ها [۱۴] هستند، مدل اتصال پیمانه‌ای [۷] نزدیک‌ترین مدل‌سازی به مدل پیشنهادی ما بر پایه شبیه‌سازی تکامل شبکه‌های ارتباطات کلاس‌های نرم‌افزار است.

مجموعه‌ای از نسخه‌های پیشین مجموعه دادگان (در حدود ۶۰٪ مجموعه نسخه‌های در دسترس نرم‌افزار برای هر پروژه) برای محاسبه توزیع خصوصیات شبکه (پارامترهای مدل پیشنهادی شامل توزیع درجات و سن) استفاده می‌شود. سایر نسخه‌ها به عنوان مجموعه داده تست استفاده می‌شوند. متغیرهای روش اتصال پیمانه‌ای به منظور مقایسه با مدل پیشنهادی به شرح جدول ۵ است.

جدول ۵- متغیرهای مدل اتصال پیمانه‌ای

n_0	اندازه اولیه شبکه
g	نرخ رشد شبکه
m	شیب خط رشد گره‌ها و یال‌های شبکه (میانگین تعداد یال‌های اضافه شده با اضافه شدن تک گره)
Γ	نرخ جفت‌شدگی[۱۹]- درجه عمومی استفاده شده در نرم‌افزار (نسبت تعداد یال‌هایی که گره جدید را به شبکه اضافه می‌کنند بر تعداد کل یال‌های گره - مجموع یال‌های داخلی بسته گره و یال‌های که به شبکه نرم‌افزار متصل می‌کنند.)

مقادیر این متغیرها برای پروژه EGit در جدول ۶ نمایش داده شده است. معیارهای که در ادامه مورد مطالعه قرار می‌گیرند نماینده ساختار همبندی شبکه نرم‌افزار هستند، در نتیجه می‌توانند به‌عنوان شاخصی از میزان شباهت شبکه پیش‌بینی شده در مقایسه با نسخه نهایی سیستم نرم‌افزار واقعی در نظر گرفته شوند.

جدول ۶- مقادیر متغیرهای مربوط به مدل اتصال پیمانه‌ای در مجموعه دادگان

Parameters	EGit project
n_0	10
$g = \frac{1}{n_0}$	0.1
m	9.021
Γ	0.4445

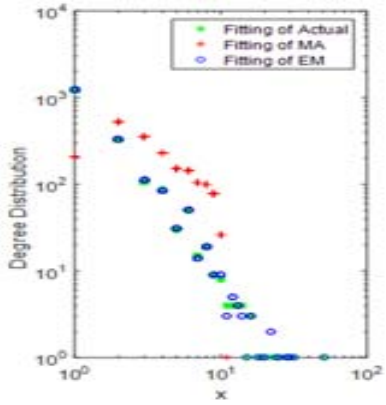
توجه: در این بخش ما از نمادهای زیر استفاده می‌کنیم.

- شبکه‌های واقعی (Actual Software Networks): Actual
- روش اتصال پیمانه‌ای (Modular Attachment model): MA
- مدل پیشنهادی (The Proposed Evolution Model): EM

این نکته قابل توضیح است که اندازه شبکه برای پیش‌بینی تکامل پروژه در مدل پیشنهادی، شبکه اولیه نرم افزار EGit (نسخه v2.0.0) است.

۳-۱- توزیع درجات

اولین معیار سنجش دقت توزیع درجه‌های شبکه‌های تولید شده توسط مدل پیشنهادی و روش اتصال پیمانه‌ای در مقایسه با شبکه واقعی نرم‌افزار است. شکل ۷.



شکل ۷- توزیع درجات شبکه‌های واقعی (Actual)، شبکه‌های تولید شده توسط روش اتصال پیمانه (MA) و مدل پیشنهادی (EM)

همانگونه که شکل ۷ نمایش می‌دهد، درجه شبکه‌های واقعی توزیع توانی دارند که مطابق با شبکه‌های تولید شده توسط مدل پیشنهادی است. در مقایسه با شبکه‌های تولید شده توسط مدل اتصال پیمانه‌ای، شبکه‌های تولیدی مدل پیشنهادی دارای تطابق دقیق‌تری با توزیع درجات شبکه‌های واقعی eclipse است.

از دید مهندسی نرم‌افزار کلاس‌های با قابلیت استفاده مجدد بیشتر محبوب هستند، که منجر به طراحی سیستم‌های نرم‌افزاری با درجه ورودی بزرگ می‌شوند. از سوی دیگر کلاس‌های با ساختار داخلی پیچیده اغلب به دلیل سختی نگهداری نرم‌افزار بسیار محبوب نیستند، بنابراین توزیع ارتباطات بین کلاس‌های نرم‌افزاری از توزیع توانی پیروی می‌کند (شکل ۷).

کلاس‌های پایه برای عملکرد ساختار نرم‌افزار حیاتی هستند. شکست برخی از گره‌های اساسی با درجه ورودی بالا در شبکه نرم‌افزار به شکنندگی اساسی عملکرد سیستم و نقص عملکرد منجر می‌شود، بنابراین حفظ عملکرد بدون خطای کلاس‌های اساسی بسیار مهم خواهد بود.

۳-۲- همبستگی بین درجات ورودی و خروجی

در شبکه‌های شبیه‌سازی و واقعی نرم‌افزار، ما از ضریب همبستگی برای سنجش رابطه بین درجه ورودی و خروجی استفاده می‌کنیم. نتایج ضریب همبستگی برای پروژه EGit در جدول ۷ نمایش داده شده است. نتایج نشان‌دهنده عدم‌تقارن بین توزیع درجه ورودی و خروجی است.

جدول ۷- ضریب همبستگی در پروژه EGit

Correlation coefficient	actual network	Modular attachment model	The Proposed Evolution Model
EGit v4.3.0	-0.2443	-0.1967	-0.2511

دلیل این عدم‌تقارن این است که گره‌های با درجه ورودی بالا نماینده کلاس‌های پایه و ساده در سیستم‌های نرم‌افزاری بوده، لذا به‌صورت مکرر استفاده شده و کلاس‌های دیگر به آنها وابسته هستند. در مقابل، گره‌ها با درجه خروجی بالا معمولا نماینده کلاس‌های هستند که وابسته به گره‌های زیادی بوده و ساختار داخلی پیچیده‌ای دارند و در نتیجه احتمال وابستگی دیگر کلاس‌ها کمتر هست [۷].

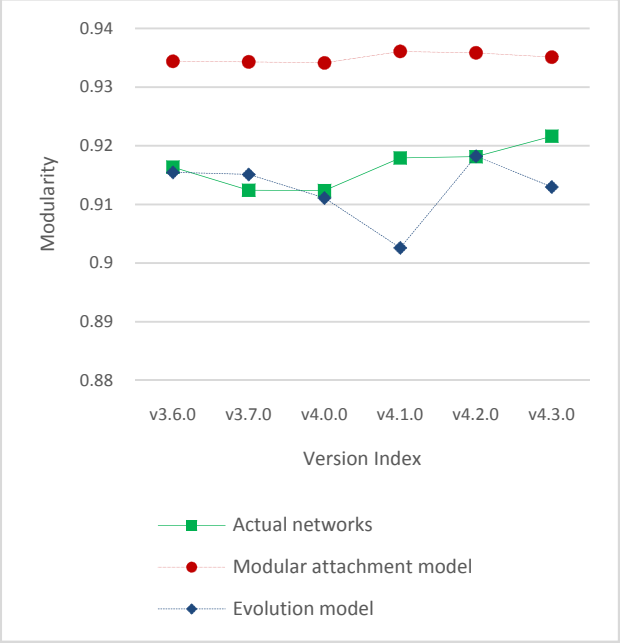
۳-۳- خاصیت پیمانه‌ای

ساختار انجمنی شبکه‌های نرم‌افزار همواره در طول تکامل حفظ می‌شود. ما از معیار پیمانه‌ای[۲۰] برای سنجش خاصیت پیمانه‌ای شبکه‌های شبیه‌سازی شده و شبکه‌های واقعی استفاده می‌کنیم [۹]. پیمان‌های (Q) یک معیار استاندارد برای تعیین قدرت ساختار یک انجمن در شبکه است. مقدار پیمان‌های بال نشان می‌دهدکه انجمن‌های به وضوح تعریف شده‌ای در شبکه وجود دارد. تعریف پیمان‌های به شکل زیر است، فرمول ۱۴.

$$Q = \sum(e_{ii} - a_i^2) \qquad (۱۴)$$

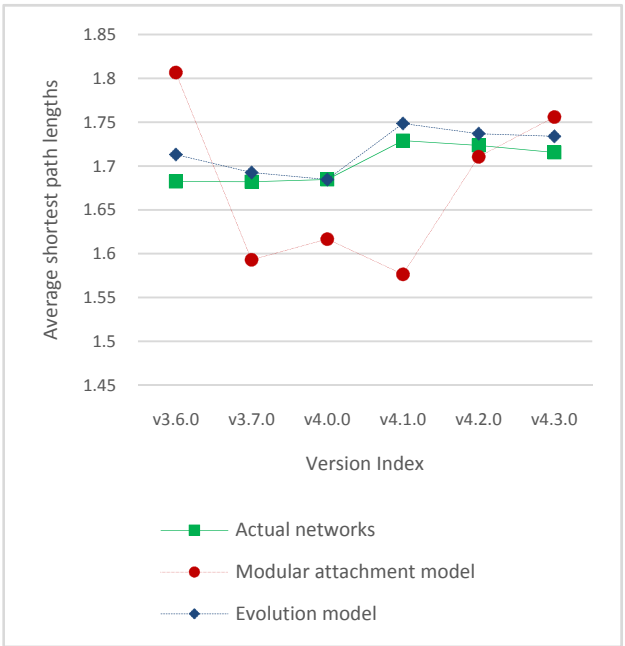
که درآن e_{ii} نسبت یال‌های بین گره‌های درون انجمن i به کل یال‌های گراف و a_i نشان‌دهنده نسبت تمام یال‌های است که از مرز انجمن عبور می‌کنند. شکل ۸ مقادیر پیمانه‌ای محاسبه شده توسط الگوریتم لوین[۲۱] [۱۸] برای شبکه‌های شبیه‌سازی شده و شبکه‌های واقعی نمایش می‌دهد. مقادیر پیمانه‌ای شبکه‌های

شبیه‌سازی شده توسط مدل تکاملی تفاوت اندکی با شبکه واقعی نرم‌افزارها دارد در حالی که مقادیر پیمانه‌ای در شبکه تولید شده توسط مدل اتصال پیمان‌های اختلاف قابل توجهی دارد.



شکل ۸- مقادیر پیمانه‌ای شبکه‌های واقعی و شبیه‌سازی شده

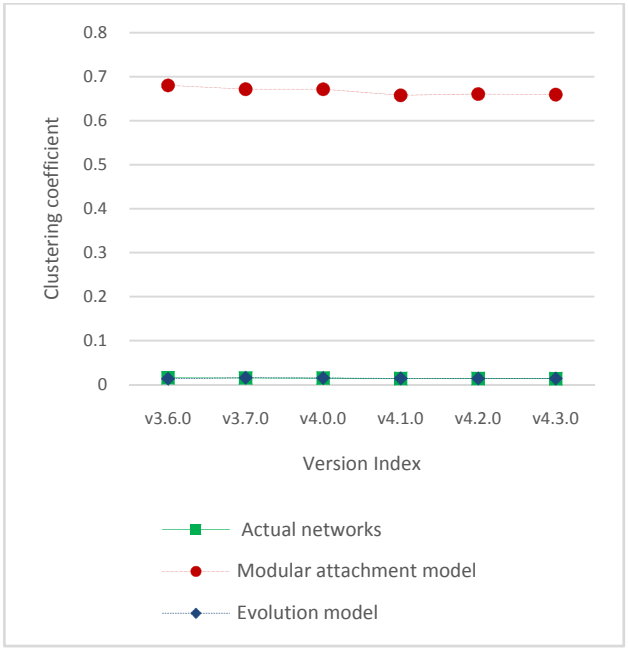
از دیدگاه مهندسی نرم‌افزار، نرم‌افزارهای با خاصیت پیمانه‌ای بالا نشان‌دهنده اصل حداقل جفت‌شدگی و حداکثر انسجام هستند، لذا بصورت ذاتی پیمانه‌هایی با ساختار قوی در شبکه نرم‌افزار ایجاد می‌گردند. علاوه بر این ساختار پیمانه‌های سیستم‌های نرم‌افزاری می‌توانند به تقویت همکاری داخل کلاسی با حداقل ارتباطات خارج پیمانه‌ای کمک کند [۹].



شکل ۹- میانگین طول کوتاه‌ترین مسیر در شبکه‌های واقعی و شبیه‌سازی شده

۳-۴- جهان کوچکی

خوشه‌بندی یکی از ویژگی‌های اساسی شبکه‌های جهان کوچک است. مقادیر ضریب خوشه‌بندی[۲۲] برای شبکه‌های شبیه‌سازی شده و شبکه‌های واقعی در شکل ۹ نمایش داده شده است.



شکل ۱۰- مقادیر ضریب خوشه‌بندی شبکه‌های واقعی و شبیه‌سازی شده

خوشه‌بندی معیاری برای سنجش همبستگی پیمانه‌های شبکه‌های نرم‌افزاری است. میانگین طول کوتاه‌ترین مسیر از دیگر معیار مهم در شبکه‌های جهان کوچک هست. این معیار نشان‌دهنده سرعت ارتباطات در سیستم‌های نرم‌افزاری است. شکل ۱۰ نشان‌دهنده مقادیر میانگین طول مسیر[۲۳] برای شبکه‌های واقعی و شبیه‌سازی شده است.

همانطور که نتایج نشان دادند که شبکه‌های نرم‌افزاری با وجود داشتن اندازه بزرگ (تعداد گره‌ها و یال‌های زیاد) دارای میانگین طول مسیر کوتاه هستند، بنابراین انتقال داده‌های بین کلاس‌ها و پیمانه‌ها بسیار موثرتر است.

خوشه‌بندی بالا و میانگین طول مسیر کوتاه از ویژگی‌های شبکه‌های جهان کوچک هستند که در طول توسعه و تکامل شبکه نرم‌افزار و شبیه‌سازی مدل تکامل پیشنهادی حفظ شده‌اند.

مدل پیشنهادی در مقایسه با مدل اتصال پیمانه‌ای در شکل‌گیری سیستم‌های نرم‌افزاری واقعی بسیار موفق‌تر عمل کرده است. استفاده از تحلیل آماری تغییرات ساختار نرم‌افزار در مدل‌سازی شبکه نرم‌افزاری منجر به مدل‌سازی بهتری نسبت به مدل اتصال پیمانه‌ای شده است بگونه‌ای که شبکه پیش‌بینی شده دارای ساختاری بسیار مشابه به شبکه واقعی نرم‌افزار از لحاظ معیارهای مشخص‌کننده شبکه است.

۴- نتیجه‌گیری

کاوش مخازن کد سیستم‌های نرم‌افزاری متن باز امکان مطالعه و تحلیل روند تکامل نرم‌افزار را فراهم کرده است. مدل‌سازی‌های تکامل سیستم‌های نرم‌افزار مبتنی بر شبکه، صرفاً بر پایه رشد و تغییر اتصالات شبکه‌های نرم‌افزاری بوده و سنجه‌های مهندسی نرم‌افزار را کمتر در نظر می‌گرفتند. در این مقاله، با کاوش

مخازن کد نرم‌افزار و مطالعه تکامل سیستم‌های نرم‌افزاری شی‌گرا و استفاده از خصوصیات شبکه کلاسی نرم‌افزار و تغییرات مهندسی نرم‌افزار، مدلی برای پیشگویی تکامل شبکه‌های نرم‌افزاری ارائه گردید.

شبکه جهت‌دار کلاس‌های نرم‌افزار که گره‌ها متناظر با کلاس‌ها و یال‌ها نشان‌دهنده چهار ارتباط اساسی انجمنی، ارث بری، تحقق و تجمع بین آنهاست، به عنوان پایه مدل‌سازی استفاده شد. مطالعه تغییرات آماری تغییرات ساختاری نرم‌افزار (ایجاد، حذف کلاس‌ها) در نسخه‌های متوالی، الگوی تغییرات کلاس‌های نرم‌افزار را آشکار می‌کند. مدل پیشنهادی با تجمیع الگوهای تغییر اساسی بر روی شبکه کلاسی نرم‌افزار، نسخه‌های آتی نرم‌افزار را شبیه‌سازی می‌کند. به‌منظور نمایش کارایی و برتری مدل پیشنهادی، قدرت شبیه‌سازی مدل بر روی پروژه‌ی واقعی متن باز eclipse به نام EGit (۱۳ نگارش مختلف) و در مقایسه با مدل اتصال پیمانه‌ای ارزیابی شد. نتایج نشان‌دهنده توزیع درجه توانی زیر شبکه‌های اضافه شده (مجموعه کلاس‌های تازه اضافه شده) و درجات ورودی و توزیع نرمال درجات خروجی گره‌های حذف شده است.

شبکه تولید شده توسط مدل پیشنهادی نسبت به مدل رشد اتصال پیمانه‌ای توزیع درجه شباهت بیشتری با شبکه واقعی نرم‌افزار داشته و ضریب همبستگی بین درجه‌های منفی و مثبت در شبکه شبیه‌سازی شده توسط مدل پیشنهادی به مدل شبکه واقعی نزدیک‌تر است. شبکه تولید شده توسط مدل پیشنهادی از لحاظ معیارهای جهان کوچکی (میانگین طول کوتاهترین مسیر و خوشه‌بندی) دارای ارزش مقادیر نزدیک‌تری به شبکه‌های واقعی در مقایسه با شبکه شبیه‌سازی شده مدل اتصال پیمانه‌ایی است. ضریب پیمانه‌ای و اندازه‌های انجمن‌های شبکه مدل پیشنهادی در مقایسه با شبکه اتصال ترجیحی تطابق نزدیک‌تری به شبکه نرم‌افزار واقعی دارند. اندازه شبکه تولید شده توسط مدل پیشنهادی نسبت به شبکه ایجاد شده توسط اتصال ترجیحی تطابق بهتری با شبکه واقعی نرم‌افزار دارد.

غنی کردن ارتباطات شبکه نرم‌افزاری مورد استفاده با بهره‌گیری از سایر ارتباطات بین کلاسی بصورت مستقیم در بهبود دقت تحلیل رفتار شبکه‌های نرم‌افزاری تاثیرگذار است. مدل پیشنهادی از چهار ارتباط اساسی بین کلاس‌های نرم‌افزاربرای نمایش ارتباطات گره‌ها و تحلیل شبکه نرم‌افزار استفاده کرده است. اضافه کردن سایر ارتباطات بین کلاس‌های نرم‌افزار (برای نمونه چندگانگی[۲۴] و ترکیب‌بندی[۲۵]) به نمایش واقعی‌تر روابط بین گره‌های شبکه نرم‌افزار کمک می‌کند. در مدل‌سازی الگوی دو دسته تغییرات سیستم‌های نرم‌افزاری در شبکه کلاسی توسط مدل پیشنهادی میزان تاثیر صفات گوناگون برای نمونه توزیع درجات و سن بصورت خطی در نظر گرفته شده است. سنجش میزان تاثیر و اهمیت هر یک از این صفات در مدل‌سازی تغییرات با استفاده از مدل‌های مارکوف[۲۶] و بازخوردی[۲۷] در افزایش دقت مدل‌سازی تغییرات تاثیرگذار خواهد بود.

مراجع

[1] T. Mens, and S. Demeyer, *Software Evolution*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008.

[2] S. W. Thomas, B. Adams, A. E. Hassan, and D. Blostein, "Studying software evolution using topic models," *Sci. Comput. Program*, vol. 80, pp. 457–479, Feb. 2014.

[3] J. Liebig, A. Von Rhein, C. Kastner, S. Apel, J. Dorre, and C. Lengauer, "Scalable analysis of variable software," in Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering. ACM, 2013, pp. 81–91.

[4] G. Rasool, and N. Fazal, "Evolution Prediction and Process Support of OSS Studies: A Systematic Mapping," *Arab. J. Sci. Eng.*, May 2017.

[5] L. Madeyski, and M. Kawalerowicz, "Software engineering needs agile experimentation: a new practice and supporting tool," in Software Engineering: Challenges and Solutions. Springer, 2017, pp. 149–162.

[6] L.-Z. Zhu, B.-B. Yin, and K.-Y. Cai, "Generating Mechanisms for Evolving Software Mirror Graph," *J. Mod. Phys.*, vol. 03, no. 09, pp. 1050–1059, 2012.

[7] H. Li, H. Zhao, W. Cai, J.-Q. Xu, and J. Ai, "A modular attachment mechanism for software network evolution," *Phys. Stat. Mech. Its Appl.*, vol. 392, no. 9, pp. 2025–2037, May 2013.

[8] G. Canfora, M. Di Penta, and L. Cerulo, "Achievements and challenges in software reverse engineering," *Commun. ACM*, vol. 54, no. 4, p. 142, Apr. 2011.

[9] L. Šubelj, and M. Bajec, "Community structure of complex software systems: Analysis and applications," *Phys. Stat. Mech. Its Appl.*, vol. 390, no. 16, pp. 2968–2975, Aug. 2011.

[10] T. Chaikalis, and A. Chatzigeorgiou, "Forecasting Java Software Evolution Trends Employing Network Models," *IEEE Trans. Softw. Eng.*, vol. 41, no. 6, pp. 582–602, Jun. 2015.

[11] G. Xiao, Z. Zheng, and H. Wang, "Evolution of Linux operating system network," *Phys. Stat. Mech. Its Appl.*, vol. 466, pp. 249–258, Jan. 2017.

[12] T. Hellmann, and M. Staudigl, "Evolution of social networks," *Eur. J. Oper. Res.*, vol. 234, no. 3, pp. 583–596, May 2014.

[13] H. Shariat Yazdi, L. Angelis, T. Kehrer, and U. Kelter, "A framework for capturing, statistically modeling and analyzing the evolution of software models," *J. Syst. Softw.*, vol. 118, pp. 176–207, Aug. 2016.

[14] T. Chaikalis, and A. Chatzigeorgiou, "Forecasting Java Software Evolution Trends Employing Network Models," *IEEE Trans. Softw. Eng.*, vol. 41, no. 6, pp. 582–602, Jun. 2015.

[15] W. Pan, B. Li, Y. Ma, and J. Liu, "A Novel Software Evolution Model Based on Software Networks," in *Complex Sciences*, vol. 5, J. Zhou, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 1281–1291.

[16] H. Li, L.-Y. Hao, R. Chen, X. Ge, and H. Zhao, "Symmetric Preferential Attachment for New Vertices Attaching to Software Networks," *New Gener. Comput.*, vol. 32, no. 3–4, pp. 271–296, Aug. 2014.

[17] H. Kim, C. I. Del Genio, K. E. Bassler, and Z. Toroczkai, "Constructing and sampling directed graphs with

given degree sequences," *New J. Phys.*, vol. 14, no. 2, p. 023012, Feb. 2012.

[18] Y. Jiang, Y. Huang, P. Li, S. Gao, Y. Zhang, and Y. Yan, "How to detect communities in large networks," in International Conference on Intelligent Computing. Springer, 2015, pp. 76–84.

مجتبی صادقیان مدرک کارشناسی و کارشناسی‌ارشد خود را به ترتیب از دانشگاه تبریز در سال ۹۳ و دانشگاه تهران در سال ۹۵ دریافت نموده است. پایان‌نامه ایشان در مورد تحلیل و مدل‌سازی تکامل شبکه‌های نرم‌افزاری متن باز شی‌گرا با استفاده از مفاهیم شبکه‌های پیچیده و مهندسی نرم‌افزار است.

آدرس پست‌الکترونیکی ایشان عبارت است از:

m.sadeghian1991@ut.ac.ir

محمد خوانساری مدرک کارشناسی، کارشناسی‌ارشد و دکتری خود را از از دانشگاه صنعتی شریف در مهندسی کامپیوتر به ترتیب در سال‌های ۱۳۷۵، ۱۳۷۷ و ۱۳۸۷ اخذ کرد. او یکی از مشارکت‌کنندگان اصلی راه‌اندازی مرکز پژوهشی فناوری اطلاعات و ارتباطات پیشرفته دانشگاه صنعتی شریف (AICT) و دارای یک بورس تحقیقاتی از موسسه DAAD آلمان است. از سوابق اجرایی-پژوهشی ایشان می‌توان به مدیریت طرح ملی نرم‌افزارهای آزاد/متن‌باز (لینوکس فارسی)، رئیس پژوهشکده فناوری اطلاعات و رئیس مرکز تحقیقات مخابرات ایران (پژوهشگاه ارتباطات و فناوری اطلاعات)، عضو هیات علمی پردیس بین‌الملل دانشگاه صنعتی شریف (کیش) و رئیس مرکز فناوری اطلاعات و فضای مجازی دانشگاه تهران اشاره کرد. ایشان مؤسس گرایش بین‌رشته‌ای مهندسی سامانه‌های شبکه‌ای در کارشناسی ارشد مهندسی فناوری اطلاعات می‌باشند. از شهریور ۱۳۹۰ تاکنون، ایشان عضو هیئت علمی دانشکده علوم و فنون نوین دانشگاه تهران هستند. زمینه‌های پژوهشی ایشان عبارتند از: علوم شبکه و شبکه‌های پیچیده، شبکه‌های سنسور بی‌سیم چندرسانه‌ای و سلامت، نرم‌افزارهای آزاد/متن‌باز، سیستم‌های اطلاعات سلامت.

آدرس پست‌الکترونیکی ایشان عبارت است از:

m.khansari@ut.ac.ir

فرید دهقان دانشجوی دکتری رشته مهندسی فناوری اطلاعات است و در زمینه تحلیل تعامل در توسعه نرم‌افزار بر بستر گیت هاب کار می‌کند. به تحلیل شبکه‌های اجتماعی تعاملی و داده‌های بزرگ و نیز تحلیل گرافی احتمالاتی و داده‌کاوی علاقه دارد.

آدرس پست‌الکترونیکی ایشان عبارت است از:

farid.dehgan@ut.ac.ir

اطلاعات بررسی مقاله:

تاریخ ارسال: ۱۳۹۶/۰۶/۱۳

تاریخ اصلاح: ۱۳۹۶/۰۷/۱۰

تاریخ قبول شدن: ۱۳۹۷/۰۱/۲۲

نویسنده مرتبط: دکتر محمد خوانساری، دانشکده علوم و فنون نوین، دانشگاه تهران، تهران، ایران.

[1]Maintenance
[2]Number of Code Lines
[3]Mining Software Code Repositories
[4]Preferential Attachment
[5]Linux Operating System (LOS)
[6]Attribute
[7]Association
[8]Generalization
[9]Realization
[10]Aggregation
[11]https://github.com/eclipse/egit
[12]Source Code Management
[13]Scale-Free
[14]God
[15]Objects
[16]Correlation
[17]Community
[18]Lego
[19]Coupling
[20]Modularity
[21]Louvain
[22]Clustering Coefficient
[23]Average Shortest Path Lengths
[24]Multiplicity
[25]Composition
[26]Markov Model
[27]Feedback Model