Journal of Information Technology Management
ISSN: 2008-5893
EISSN: 2423-5059
Vol. 9, No. 3; PP. 507-530
Fall 2017

University of Tehran
Faculty of Management

Estimation of total Effort and Effort Elapsed in Each Step of Software Development Using Optimal Bayesian Belief Network

Fatemeh Zare Baghiabad [1], Hasan Khademi Zare [2], Mohammad Saber Fallahnezhad [3], Fazlolah Adibnia [4]

Abstract: Accuracy in estimating the needed effort for software development caused software effort estimation to be a challenging issue. Beside estimation of total effort, determining the effort elapsed in each software development step is very important because any mistakes in enterprise resource planning can lead to project failure. In this paper, a Bayesian belief network was proposed based on effective components and software development process. In this model, the feedback loops are considered between development steps provided that the return rates are different for each project. Different return rates help us determine the percentages of the elapsed effort in each software development step, distinctively. Moreover, the error measurement resulted from optimized effort estimation and the optimal coefficients to modify the model are sought. The results of the comparison between the proposed model and other models showed that the model has the capability to highly accurately estimate the total effort (with the marginal error of about 0.114) and to estimate the effort elapsed in each software development step.

Key words: ***Bayesian belief network, Enterprise resource planning, Machine learning methods, Software effort estimation, Optimization.***

1. Ph.D. Candidate Industrial Engineering, Yazd University, Yazd, Iran
2. Associate Prof., Dep. of Industrial Engineering, Yazd University, Yazd, Iran
3. Associate Prof., Dep. of Industrial Engineering, Yazd University, Yazd, Iran
4. Assistant Prof., Dep. of Computer Engineering, Yazd University, Yazd, Iran

Submitted: 17 / November / 2016
Accepted: 21 / May / 2017
Corresponding Author: Hasan Khademi Zare
Email: Hkhademiz@yazd.ac.ir

مدیریت فناوری اطلاعات
Journal of Information Technology Management
دانشکدهٔ مدیریت دانشگاه تهران
دورهٔ ۹، شمارهٔ ۳
پاییز ۱۳۹۶
صص. ۵۳۰- ۵۰۷

تخمین تلاش کلی نرم‌افزار و تلاش صرف‌شده در هر مرحلهٔ تولید با استفاده از شبکهٔ بیزی بهینه

فاطمه زارع باقی‌آباد[۱]، حسن خادمی زارع[۲]، محمدصابر فلاح‌نژاد[۳]، فضل‌الله ادیب‌نیا[۴]

چکیده: دقت در تخمین تلاش لازم برای تولید نرم‌افزار، موجب شده است که تخمین تلاش هنوز به‌عنوان موضوع چالش‌انگیزی مطرح باشد. علاوه‌بر تخمین تلاش کلی، تعیین تلاش صرف‌شده در هر مرحله از تولید نیز به‌دلیل برنامه‌ریزی منابع اهمیت دارد؛ زیرا تخصیص نادرست منابع می‌تواند به شکست پروژه منجر شود. در این مقاله، یک شبکهٔ بیزی برای تخمین تلاش برپایهٔ مؤلفه‌های مؤثر و فرایند تولید ارائه شده است. در این مدل بین مراحل تولید، حلقه‌های تکرار در نظر گرفته شده که میزان تکرار آنها برای پروژه‌های مختلف، متفاوت خواهد بود و موجب می‌شود که درصد تلاش صرف‌شده در هر مرحلهٔ تولید برای هر پروژه به‌صورت منحصربه‌فرد تعیین شود. معیار خطای حاصل از تخمین تلاش، بهینه‌سازی شده و ضریب‌های بهینه برای اصلاح مدل به‌دست می‌آیند. نتایج مقایسهٔ مدل پیشنهادی با مدل‌های دیگر تخمین نشان می‌دهد مدل پیشنهاد شده، علاوه‌بر توانایی در تخمین دقیق تلاش کلی (با خطا ۰/۱۱۴)، قابلیت بالایی نیز برای تعیین تلاش هر مرحله تولید دارد.

واژه‌های کلیدی: برنامه‌ریزی منابع سازمان، بهینه‌سازی، تخمین تلاش نرم‌افزار، روش‌های یادگیری ماشین، شبکهٔ بیزی.

۱. دانشجوی دکتری مهندسی صنایع، دانشکدهٔ فنی و مهندسی، دانشگاه یزد، یزد، ایران
۲. دانشیار گروه مهندسی صنایع، دانشکدهٔ فنی و مهندسی، دانشگاه یزد، یزد، ایران
۳. دانشیار گروه مهندسی صنایع، دانشکدهٔ فنی و مهندسی، دانشگاه یزد، یزد، ایران
۴. استادیار گروه مهندسی کامپیوتر، دانشکدهٔ فنی و مهندسی، دانشگاه یزد، یزد، ایران

تاریخ دریافت مقاله: ۱۳۹۵/۰۸/۲۷
تاریخ پذیرش نهایی مقاله: ۱۳۹۶/۰۲/۳۱
نویسندهٔ مسئول مقاله: حسن خادمی زارع
E-mail: Hkhademiz@yazd.ac.ir

مقدمه

تخمین تلاش نرم‌افزار، فرایند تخمین تلاش (نفر ـ ماه) موردنیاز برای تولید یک سیستم نرم‌افزاری است. موضوع مهم در تخمین هزینهٔ نرم‌افزار، دقت تخمین است. هرچند مدل‌های زیادی در این زمینه پیشنهاد شده‌اند، هنوز دقت تخمین موضوع چالش‌انگیزی محسوب می‌شود (اختر، ۲۰۱۳). متغیرهای نامطمئن زیادی موجب بی‌دقتی در تخمین تلاش می‌شوند. برای نمونه، عواملی مانند روش تولید، سازمان تولیدکننده و زبان برنامه‌نویسی، موجب می‌شود که اندازهٔ نرم‌افزار به‌صورت یک عدد فازی بیان شود (خان، شیخ و نعمان، ۲۰۱۱). همچنین متغیرهای مهم دیگری هستند که به‌طور دقیق در آخر پروژه مشخص می‌شوند (فیونتاجا، بوراجو، لوپز و اکن، ۲۰۱۳). بنابراین ارائهٔ مدلی برای تخمین دقیق تلاش نرم‌افزار، به‌عنوان یک مسئله مطرح است.

همچنین علاوه‌بر تخمین تلاش کلی، تخمین تلاش صرف‌شده در هر مرحله از فرایند تولید نیز اهمیت زیادی دارد؛ زیرا برنامه‌ریزی صحیح منابع سازمان که یکی از عوامل مهم موفقیت پروژه محسوب می‌شود (خانلری و کفایی، ۱۳۹۳؛ گمنام سفیددارینی، ناصرزاده، روحانی و قاهردوست، ۱۳۹۳) به زمان و تلاش صرف‌شده در هر مرحله از پروژه بستگی دارد. با مشخص شدن تلاش صرف‌شده در هر مرحله، منابع می‌توانند به‌گونه‌ای برنامه‌ریزی و تخصیص داده شوند که درست در زمان نیاز، در دسترس باشند.

با توجه به سند چشم‌انداز جمهوری اسلامی ایران در افق سال ۱۴۰۴، کشور باید تا سال ۱۴۰۴ در حوزه‌های تولید و خدمات صنعت فناوری اطلاعات و ارتباطات در خاورمیانه، مقام اول را کسب کند (رئیسی وانانی و گنجعلی‌خان حاکمی، ۱۳۹۴)، توجه به حوزهٔ تولید نرم‌افزار حائز اهمیت زیادی است. این تحقیق در نظر دارد تلاش (نفر ـ ماه) کلی مورد نیاز برای تولید نرم‌افزار و نیز تلاش لازم در هر مرحله از فرایند تولید را برآورد کند تا مدیر علاوه‌بر برآوردی که از زمان و هزینهٔ کلی پروژه دارد، بتواند برنامه‌ریزی مناسبی برای منابع سازمان انجام دهد. در این تحقیق سعی بر آن است تا با پاسخ به به سؤال‌های پژوهشی زیر، اهداف ذکر شدهٔ بالا تحقق یابد.

۱. چگونه می‌توان مدلی برای تخمین تلاش نرم‌افزار ارائه داد که مؤلفه‌های مؤثر بر تلاش و فرایند تولید را با هم درنظر بگیرد؟
۲. چگونه شبکهٔ بیز می‌تواند عدم‌قطعیت‌های موجود در ماهیت تخمین تلاش نرم‌افزار، مانند حلقه‌های بازنگری بین مراحل تولید را پوشش دهد؟
۳. چگونه می‌توان میزان تکرار حلقه‌های بازنگری را برای هر پروژه مطابق با شرایط آن تعیین کرد؟

۴. چگونه می‌توان بازه‌های گسستهٔ مربوط به گره‌های شبکهٔ بیزی را برای افزایش دقت تخمین به‌صورت پیوسته دید؟
۵. چگونه تلاش کلی و تلاش موردنیاز در هر مرحله از فرایند تولید برآورد می‌شود؟
۶. چگونه خطای تلاش تخمینی بهینه‌سازی شده و مدل به‌روزرسانی می‌شود؟

پیشینهٔ نظری پژوهش

پروژهٔ تولید نرم‌افزار شامل مراحل تولید و چیدمانی از ترتیب انجام آنها، بسته به نیازها و هدف مدنظر است. مدل‌های مختلف تولید نرم‌افزار شامل مدل خطی[1]، آبشاری[2]، توسعهٔ سریع[3]، نمونه‌سازی[4]، افزایشی[5]، حلزونی[6] و چابک[7] است (خادمی‌زارع و زارع باقی‌آباد، ۱۳۹۰). در این تحقیق، از روش آبشاری استفاده می‌شود؛ زیرا مدل آبشاری پروژه‌محور بوده و به‌دلیل ساختارمندی می‌تواند آنچه طراحی شده را به‌خوبی پیاده‌سازی کند. همچنین این مدل در پروژه‌های بدون ساختار، انعطاف‌پذیری لازم را در برابر تغییرات پیش‌آمده دارد. برای تخمین بهتر زمان و هزینه، می‌توان عوامل متغیر و احتمالی را به مدل وارد کرد. همچنین کاربر درطول دورهٔ تولید نرم‌افزار، حضور پررنگی داشته و نظرهای او در حین پروژه اعمال می‌شود (هوتل، ۲۰۰۸).

مدل‌های مختلف تخمین تلاش نرم‌افزار به دو دسته روش‌های مدل‌محور و خبره‌محور دسته‌بندی می‌شوند. مدل‌های مدل‌محور می‌توانند به روش‌های آماری مانند رگرسیون و روش‌های هوشمند مانند یادگیری ماشین طبقه‌بندی شود (ایدری، آمازال و آبران، ۲۰۱۵؛ لوپزمارتین، ۲۰۱۵).

امروزه، توجه به روش‌های یادگیری ماشین در حال افزایش است؛ زیرا این روش‌ها، قابلیت مدل‌کردن روابط بین تلاش نرم‌افزار و مؤلفه‌های مؤثر بر آن را دارند. براساس تحقیق لوپز مارتین (۲۰۱۵) که در آن، تواتر استفاده از روش‌های یادگیری ماشین در زمینهٔ تخمین تلاش نرم‌افزار، در مقاله‌های منتشر شده بین سال‌های ۱۹۹۱ تا ۲۰۱۰ بررسی شده است، روش استدلال برمبنای نمونه[8] در ۳۷ درصد مقاله‌ها و شبکه‌های عصبی در ۲۶ درصد مقاله‌ها استفاده

1. Linear model
2. Waterfall model
3. Rapid Application Development
4. Prototyping model
5. Incremental model
6. Spiral model
7. Agile Models
8. Case-based Reasoning (CBR)

شده‌اند. همچنین الگوریتم ژنتیک به تعداد زیاد و به شکل ترکیبی با روش‌های دیگر یادگیری ماشین به‌کار رفته بود، اما شبکه‌های بیزی به تعداد کم در حد یک یا دو مقاله در زمینهٔ تخمین تلاش استفاده شده بودند. بنابراین، هنوز کاربرد این روش در تخمین تلاش نرم‌افزار محدود است و این نیاز برای کشف قابلیت‌های آن وجود دارد. شبکهٔ بیزی برای پایگاه‌های دادهٔ کوچک و ناکامل مناسب است. روابط علت و معلولی شبکهٔ بیزی به خبرگان کمک می‌کند داده‌ها را تکمیل کرده و آنها را دردسترس نگه دارند (زارع، خادمی‌زارع و فلاح‌نژاد، ۲۰۱۶). همچنین امکان یادگیری ساختاری و ترکیب منابع مختلف دانش را فراهم می‌کند. با توجه به استفادهٔ محدود از شبکه‌های بیز در زمینهٔ تخمین تلاش نرم‌افزار و قابلیت‌های ذکر شده در بالا و نیز قابلیت شبکهٔ بیزی برای درنظرگرفتن عدم‌قطعیت‌ها، به‌دلیل ماهیت احتمالی گره‌ها، در این تحقیق از شبکهٔ بیزی برای تخمین تلاش نرم‌افزار استفاده شده است.

یک شبکهٔ بیزی شامل گراف و جدول‌های احتمال است که گره‌ها معرف متغیرهای احتمالی و کمان‌ها گویای روابط علت و معلولی هستند. این احتمالات برای گره‌های والد بیان‌کنندهٔ احتمالات اصلی و برای گره‌های فرزند نشان‌دهندهٔ احتمالات شرطی هستند. احتمالات پیشین و شرطی، قسمتی از طراحی شبکه‌اند و در فرایند تحلیل تغییر نمی‌کنند. معمولاً توزیع پیشین توسط خبرگان و احتمالات شرطی توسط پایگاه دادهٔ تحت فرایند یادگیری به‌دست می‌آیند (پرکوسیچ، سوآرز، آلمیدا و پرکوسیچ، ۲۰۱۵). تلاش موردنیاز برای تولید نرم‌افزار، یک متغیر پیوسته است در حالی که شبکه‌های بیزی با متغیرهای گسسته کار می‌کنند. بنابراین، تخمین تلاش باید توسط چند بازهٔ گسسته تقریب زده شود (پندهارکار، سوبرامانیان و راجر، ۲۰۰۵). تخمین براساس بازه‌های گسسته موجب کاهش دقت می‌شود، به‌طوری که انتخاب روش گسسته‌سازی و تعداد بازه‌ها بر دقت تخمین بسیار تأثیر می‌گذارد (فرناندزدیگو و تورالبامارتینز، ۲۰۱۲).

شبکهٔ بیزی یک گراف بدون حلقه است و حلقه‌های بازنگری‌ای که گاهی در مدل‌سازی مفید هستند را پوشش نمی‌دهد (آسیتالو، ۲۰۰۷). با توجه به دو ضعف ذکر شده در شبکهٔ بیزی، شامل بازه‌های گسسته و در نظر نگرفتن حلقه‌های بازنگری، تحقیق حاضر بازه‌های مربوط به گره‌های شبکهٔ بیزی را به کمک اعداد فازی به‌صورت پیوسته در نظر می‌گیرد، همچنین حلقه‌های بازنگری را بین مراحل تولید با درنظرگرفتن معیار میزان معرفی عیب[1] به شبکهٔ بیزی اضافه کرده است.

1. Defect Introduction Rate (DIR)

پیشینهٔ تجربی پژوهش

بعضی از مدل‌های تخمین تلاش براساس مؤلفه‌های مؤثر بر هزینه، برخی دیگر براساس فرایند تولید و بعضی دیگر براساس هر دو، تلاش را تخمین می‌زنند. مدل‌هایی که تلاش را براساس فرایند تولید یا مؤلفه‌ها و مراحل تخمین می‌زنند، این مزیت را دارند که علاوه‌بر تلاش کلی، می‌توانند برآوردی از تلاش صرف‌شده در هر مرحله از تولید نیز داشته باشند و مطابق آن برنامه‌ریزی مناسبی برای منابع انجام دهند. ابتدا مدل‌های مربوط به تخمین براساس مؤلفه‌ها، سپس مدل‌های مربوط به تخمین براساس فرایند تولید و بعد مدل‌های تخمین براساس مؤلفه‌ها و فرایند تولید بیان می‌شود.

تخمین برپایهٔ مؤلفه‌های مؤثر بر تلاش

یک مدل سه سطحی شبکهٔ بیزی برمبنای مؤلفه‌های کوکومودو برای تخمین بهره‌وری نرم‌افزار (DSI/MM)[1] ارائه شده است (استاملوس، آنجلیس، دیمو و ساکلاریس، ۲۰۰۳). در سطح اول، ۱۵ مؤلفهٔ کوکومودو به چهار گروه محصول، کامپیوتر، پرسنل و پروژه دسته‌بندی می‌شود. در سطح دوم، این چهار گروه به دو دسته عوامل فنی و انسانی طبقه‌بندی می‌شود. در سطح سوم، براساس این دو دسته، تلاش موردنیاز نرم‌افزار برآورد می‌گردد. تمام گره‌های مربوط به شبکه دارای ۵ بازهٔ گسسته خیلی کم، کم، متوسط، زیاد و خیلی زیاد است. گسسته بودن بازه‌ها موجب تفکیک دفعی داده‌ها شده و دقت را کاهش می‌دهد. یک مدل دو سطحی شبکهٔ بیزی دیگر برای تخمین تلاش نرم‌افزار برمبنای سه مؤلفهٔ روش تولید نرم‌افزار، ابزارهای مهندسی نرم‌افزار و تجربهٔ کار با این ابزارها طراحی شده است (پندهارکار و همکاران، ۲۰۰۵). روش تولید شامل دو مؤلفهٔ چرخهٔ عمر ایجاد سیستم و طراحی کاربردی سریع است. نوع ابزار شامل ابزارهای مهندسی نرم‌افزار به‌کمک کامپیوتر و ابزارهای تسهیل مهندسی اطلاعات می‌شود و تجربهٔ کار با ابزار نیز شامل سه بازهٔ کم، متوسط و زیاد است. در این مدل، تخمین شبکهٔ بیزی با ورود اطلاعات جدید به‌روزرسانی می‌شود. در این مقاله بازه‌های مؤلفهٔ تجربهٔ کار با ابزار گسسته است که موجب کاهش دقت می‌شود. همچنین آزمون و کنترل مدل نیز انجام نشده است. در مقالهٔ دیگر یک مدل شبکهٔ بیزی سه سطحی برای تخمین تلاش نرم‌افزار بر پایهٔ مؤلفه‌های کوکومودو طراحی شده است (زارع، خادمی‌زارع و فلاح‌نژاد، ۲۰۱۶). برخلاف مدل‌های قبلی، گره‌های شبکهٔ بیزی در این مدل دارای بازه‌های پیوسته است. همچنین معیار خطا بهینه‌سازی شده و با استفاده از

1. Delivered source structures per man-month

ضریب بهینهٔ حاصل، شبکهٔ بیزی اصلاح می‌شود. تنها اشکال مدل این است که به میزان تلاش صرف شده در هر مرحلهٔ تولید توجه نشده است.

تخمین برپایهٔ فرایند تولید نرم‌افزار

در زمینهٔ تخمین براساس فرایند تولید نرم‌افزار، محققان یک شبکهٔ بیزی برای تخمین تلاش نرم‌افزار برمبنای فازهای فرایند تولید نرم‌افزار ارائه کردند (بیبی و استاملوس، ۲۰۰۴). آنها از فرایند یکپارچهٔ منطقی برای تولید نرم‌افراز بهره بردند. فازهای این روش شامل فاز شروع، بسط، ساخت و گذر است. جریان‌های کاری که در هر فاز تکرار می‌شود و تولید افزایشی را نشان می‌دهند، شامل مدل‌سازی تجاری، تعیین نیازها، تحلیل و طراحی، پیاده‌سازی، آزمون، استقرار و پشتیبانی است. محققان دیگری یک فرایند چند مرحله‌ای برای ساخت یک شبکهٔ بیزی نیمه خودکار برای تخمین تلاش نرم‌افزار ارائه دادند. در این تحقیق، ابتدا یک مدل خودکار براساس شش فاز تولید شامل پروپوزال، تعیین نیازها، طراحی، کد نویسی، نصب و خاتمهٔ پروژه ساخته شده و در مرحلهٔ دوم، این مدل توسط خبرگان تعدیل می‌شود (فیونتتاجا و همکاران، ۲۰۱۳).

تخمین برپایه مؤلفه‌ها و فرایند تولید نرم‌افزار

در زمینهٔ تخمین براساس مؤلفه‌ها و فرایند تولید، محققان یک الگوریتم سه مرحله‌ای برای تخمین زمان و قیمت نرم‌افزار در شرایط عدم قطعیت ارائه دادند (زارع باقی‌آباد و خادمی زارع، ۲۰۱۵). ابتدا با استفاده از روش توابع کارکردی، اندازهٔ نرم‌افزار برآورد شده، سپس با روش کوکومودو، تلاش نرم‌افزار در سه حالت خوش‌بینانه، محتمل و بدبینانه به‌صورت یک عدد فازی تخمین زده شده است. در مرحلهٔ سوم، با بهره‌مندی از نظر خبرگان، تلاش تخمین زده شده با روش کوکومودو در مراحل تولید نرم‌افزار که به روش آبشاری درنظرگرفته شده تقسیم می‌شود. در روش آبشاری، حلقه‌های بازنگری بین مراحل به‌صورت احتمالی دیده می‌شود. به بیانی، مراحل تولید و حلقه‌های بازنگری به‌صورت یک شبکهٔ گرت دیده شده و زمان انجام پروژه با در نظرگرفتن تکرارها بین مراحل تولید محاسبه می‌شود. این مقاله، روشی برای بهینه‌سازی خطا و اصلاح مدل ارائه نکرده و میزان تکرار بین مراحل برای تمام پروژه‌ها یکسان در نظرگرفته شده است.

با توجه به مقالات بررسی‌شده، ضعف‌هایی که در این مقالات دیده شده شامل موارد زیر است:

- درنظرگرفتن بازه‌های شبکهٔ بیزی به‌صورت گسسته که موجب کاهش دقت تخمین می‌شود.
- در نظر نگرفتن حلقه‌های بازخورد بین مراحل تولید که موجب دور شدن مدل از واقعیت می‌شود؛ چرا که در واقع، تکرار بین مراحل تولید انجام می‌شود. همچنین در یک مقاله که حلقه‌های بازخورد در نظرگرفته شده، مقدار تکرار برای تمام پروژه‌ها بدون توجه به ویژگی‌های پروژه، یکسان در نظرگرفته شده است.
- عدم بهینه‌سازی معیار خطا که موجب می‌شود مدل پیشنهادی دیگر اصلاح نشده و مطابق ویژگی‌های بهینه به‌روزرسانی نشود.
- عدم محاسبهٔ تلاش صرف شده در هر مرحلهٔ تولید برای مقالاتی که براساس مؤلفه‌های مؤثر، تلاش کلی را تخمین زده‌اند. چنین مقالاتی برای برنامه‌ریزی منابع و کنترل پروژه، ایده‌ای ارائه نمی‌دهند. همچنین مقالاتی که براساس مراحل کار کرده‌اند نیز، به محاسبهٔ تلاش مراحل نپرداخته‌اند.

هدف این تحقیق تخمین تلاش کلی و تلاش هر مرحله از تولید نرم‌افزار است؛ به طوری که ضعف‌های مطرح شده در مقالات قبلی را پوشش دهد و به تخمین‌های بهتری دست یابد. در این تحقیق برای تخمین تلاش نرم‌افزار، یک مدل سه سطحی شبکهٔ بیزی برمبنای مؤلفه‌های روش کوکومودو ارائه شده است. در سطح اول، مؤلفه‌های کوکومودو در چهار گروه محصول، کامپیوتر، پرسنل و پروژه قرار می‌گیرند. در سطح دوم، این گروه‌ها و نیز اندازهٔ نرم‌افزار با توجه به ضریب تأثیرشان روی تلاش مراحل، وارد پنج گرهٔ مراحل تولید می‌شود. مراحل تولید شامل تعیین نیازها، طراحی و تحلیل، کدنویسی، آزمون و پشتیبانی است که به روش آبشاری در نظر گرفته شده است. در سطح سوم با استفاده از مراحل، تلاش موردنیاز برای تولید نرم‌افزار برآورد می‌شود. معیار خطای نتیجه شده از تخمین پروژه‌های مرحلهٔ یادگیری، بهینه‌سازی شده و ضرایب بهینه برای اصلاح مدل به‌دست می‌آید (شکل ۱).

به‌طور خلاصه، قوت‌های مدل پیشنهادی عبارت‌اند از:

- درنظر گرفتن حلقه‌های بازخورد بین مراحل تولید و نیز به‌دست آوردن تعداد تکرار آنها برای هر پروژه با توجه به سطح کیفیت و قابلیت‌های موردنیاز مشتری؛
- در نظرگرفتن همزمان مؤلفه‌ها و مراحل برای به‌دست آوردن درصد تلاش صرف‌شده در هر مرحله و تخصیص منابع در زمان موردنیاز؛
- در نظرگرفتن بازهٔ گره‌های شبکهٔ بیزی به‌صورت پیوسته برای به‌دست آوردن دقت بیشتر.

شکل ۱. مدل پیشنهادی برای تخمین تلاش مورد نیاز برای تولید نرم‌افزار

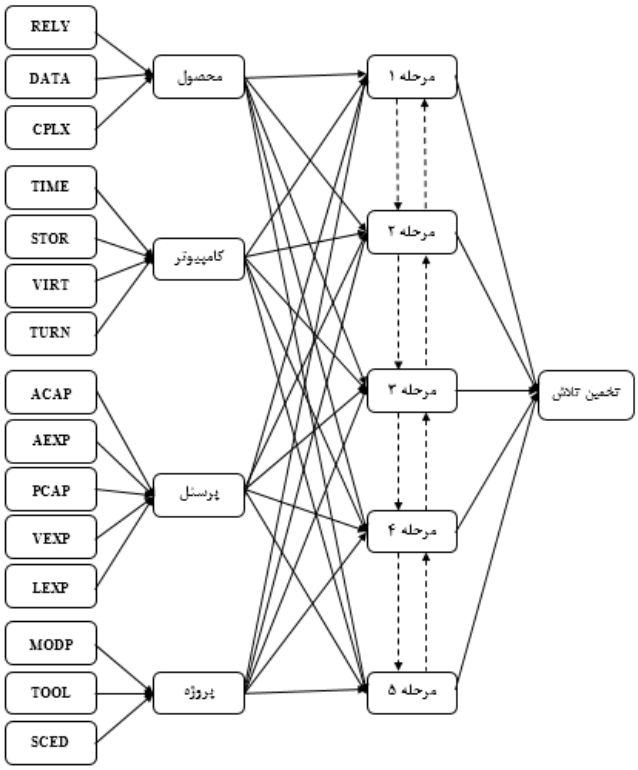
روش‌شناسی پژوهش

این تحقیق از نوع توسعه‌ای بوده و دارای مدل ریاضی است. مراحل انجام کار برای تخمین تلاش (نفر_ ساعت) موردنیاز برای پروژهٔ تولید نرم‌افزار در شکل ۱ نشان داده شده و در ادامه شرح داده می‌شود.

شبکهٔ بیزی برپایهٔ مؤلفه‌های کوکومودو و فرایند تولید نرم‌افزار

مدل سه سطحی شبکهٔ بیزی در شکل نشان داده شده است. در سطح اول، مؤلفه‌های کوکومودو در چهار گروه محصول، کامپیوتر، پرسنل و پروژه قرار می‌گیرند. در واقع، این مؤلفه‌ها دربردارندهٔ منابع سازمان و نیز خواسته‌های مشتری است. در پایگاه دادهٔ ناسا[1] که اطلاعات مؤلفه‌ها، اندازهٔ نرم‌افزار و تلاش صرف شده مربوط به ۶۰ پروژه واقعی آمده است، برای مؤلفه‌ها شش سطح خیلی‌کم (VL)، کم (L)، متوسط (Nom)، زیاد (H)، خیلی‌زیاد (VH) و فوق‌العاده زیاد (XH) در نظر گرفته شده است. شمایی از پایگاه داده ناسا در جدول ۱ و مقادیر مربوط به سطوح مؤلفه‌ها در جدول ۲ نشان داده شده است.

1. http://promise.site.uottawa.ca/SERopository/Datasets/cocomonasa-v1.arff



شکل ۲. شبکهٔ بیزی سه سطحی برپایهٔ مؤلفه‌های کوکومودو، مراحل تولید نرم‌افزار و حلقه‌های تکرار

جدول ۱. پایگاه داده کوکومو ـ ناسا

تلاش واقعی	اندازه	عوامل انسانی								عوامل فنی						
		پروژه			پرسنل					کامپیوتر				محصول		
		۱۵	۱۴	۱۳	۱۲	۱۱	۱۰	۹	۸	۷	۶	۵	۴	۳	۲	۱
۱۷۰	۳۲/۶	L	VH	VH	H	L	N	H	VH	H	L	VH	VH	H	VH	N
۳۲۴	۱۵۰	N	N	N	H	N	VH	VH	H	L	L	XH	N	H	L	N
۲۴۰۰	۳۰۲	N	VL	H	N	N	H	N	N	L	L	N	N	H	L	H
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
۱۵۵	۱۹/۳	H	L	L	H	N	H	H	H	H	N	N	VH	H	H	N

جدول ۲. عدد مربوط به هر سطح مؤلفه‌های به‌دست آمده از پایگاه دادهٔ ناسا

مؤلفه‌ها	خیلی کم	کم	متوسط	زیاد	خیلی زیاد	فوق‌العاده زیاد	بهره‌وری
ACAP	۱/۴۶	۱/۱۹	۱	۰/۸۶	۰/۷۱	–	۲/۰۶
PCAP	۱/۴۲	۱/۱۷	۱	۰/۸۶	۰/۷۰	–	۱/۶۷
AEXP	۱/۲۹	۱/۱۳	۱	۰/۹۱	۰/۸۲	–	۱/۵۷
MODP	۱/۲۴	۱/۱۰	۱	۰/۹۱	۰/۸۲	–	۱/۳۴
TOOL	۱/۲۴	۱/۱۰	۱	۰/۹۱	۰/۸۳	–	۱/۴۹
VEXP	۱/۲۱	۱/۱۰	۱	۰/۹۰	–	–	۱/۳۴
LEXP	۱/۱۴	۱/۰۷	۱	۰/۹۵	–	–	۱/۲۰
SCED	۱/۲۳	۱/۰۸	۱	۱/۰۴	۱/۱۰	–	e
STOR	–	–	۱	۱/۰۶	۱/۲۱	۱/۵۶	−۱/۲۱
DATA	–	۰/۹۴	۱	۱/۰۸	۱/۱۶	–	−۱/۲۳
TIME	–	–	۱	۱/۱۱	۱/۳۰	۱/۶۶	−۱/۳۰
TURN	–	۰/۸۷	۱	۱/۰۷	۱/۱۵	–	−۱/۳۲
VIRT	–	۰/۸۷	۱	۱/۱۵	۱/۳۰	–	−۱/۴۹
RELY	۰/۷۵	۰/۸۸	۱	۱/۱۵	۱/۴۰	–	−۱/۸۷
CPLX	۰/۷۰	۰/۸۵	۱	۱/۱۵	۱/۳۰	۱/۶۵	−۲/۳۶

در سطح دوم، این چهار گروه همراه با مؤلفهٔ اندازهٔ نرم‌افزار (هزار خط برنامه‌نویسی) با ضرایب اثرگذاری متفاوت به مراحل پنج‌گانهٔ تولید نرم‌افزار وارد می‌شوند. این ضرایب اثرگذاری با استفاده از پرسشنامه‌ای که توسط خبرگان تکمیل می‌شود، به‌دست می‌آیند. مراحل تولید به روش آبشاری شامل طرح‌ریزی، تحلیل و طراحی سیستم، کدنویسی، آزمون و پشتیبانی است که البته بسته به نرم‌افزار تحت تولید و منابع انسانی و فنی موجود، ممکن است بین این مراحل برگشت صورت گیرد تا کیفیت مطلوب به‌دست آمده و عیوب ایجاد شده رفع شوند. تعداد تکرار بین مراحل برای هر پروژه بسته به قابلیت‌های مورد نیاز سیستم نرم‌افزاری، متفاوت است. به بیانی، هر پروژه بسته به قابلیت‌های مختلف برای مؤلفه‌ها، بازخورد متفاوتی بین مراحل تولید دارد. در سطح سوم، تلاش صرف‌شده در مراحل، بر اساس تکرارهای مراحل به‌روز شده و طبق آن، تلاش نهایی تخمین زده می‌شود.

یادگیری شبکهٔ بیزی

در این گام، توابع عضویت مؤلفه‌ها، ضرایب تبدیل بین گروه‌ها و مراحل تولید، تعداد تکرار بین مراحل و نیز احتمالات شرطی بین سطوح تعیین می‌شود. پایگاه داده شامل ۶۰ پروژه است که از ۴۰ پروژه در مرحلهٔ یادگیری و ۲۰ پروژه دیگر در مرحلهٔ آزمون استفاده می‌شود. ابتدا روابط شبکهٔ بیزی به‌صورت زیر بیان می‌شود؛ گفتنی است نمادهای مطرح شده در این روابط در جدول ۳ درج شده است.

رابطهٔ ۱)
$$\pi_0(product) = \Sigma\, p(product|RELY, DATA, CPLX)\pi_0(RELY)\pi_0(DATA)\pi_0(CPLX)$$

رابطهٔ ۲)
$$\pi_0(computer) = \Sigma p(computer|TIME, STOR, VIRT, TURN)\ \pi_0(TIME)\pi_0(STOR)\pi_0(VIRT)\pi_0(TURN)$$

رابطهٔ ۳)
$$\pi_0(personnel) = \Sigma p(personnel|ACAP, AEXP, PCAP, VEXP, LEXP)\ \pi_0(ACAP)\pi_0(AEXP)\pi_0(PCAP)\pi_0(VEXP)\pi_0(LEXP)$$

رابطهٔ ۴)
$$\pi_0(project) = \Sigma p(project|MODP, TOOL, SCED)\pi_0(MODP)\pi_0(TOOL)\pi_0(SCED)$$

رابطهٔ ۵)
$$\pi_0(step1) = \Sigma p(step1|product, computer, personnel, project)\ \pi_0(product)\pi_0(computer)\pi_0(personnel)\pi_0(project)\pi_0(size)$$

رابطهٔ ۶)
$$\pi_0(step2) = \Sigma p(step2|product, computer, personnel, project)\ \pi_0(product)\pi_0(computer)\pi_0(personnel)\pi_0(project)\pi_0(size)$$

رابطهٔ ۷)
$$\pi_0(step3) = \Sigma p(step3|product, computer, personnel, project)\ \pi_0(product)\pi_0(computer)\pi_0(personnel)\pi_0(project)\pi_0(size)$$

رابطهٔ ۸)
$$\pi_0(step4) = \Sigma p(step4|product, computer, personnel, project)\ \pi_0(product)\pi_0(computer)\pi_0(personnel)\pi_0(project)\pi_0(size)$$

رابطهٔ ۹)
$$\pi_0(step5) = \Sigma p(step5|product, computer, personnel, project)\ \pi_0(product)\pi_0(computer)\pi_0(personnel)\pi_0(project)\pi_0(size)$$

رابطهٔ ۱۰)
$$\pi_0(effort) = \Sigma p(effort|step1, step2, step3, step4, step5)\ \pi_0(step1)\pi_0(step2)\pi_0(step3)\pi_0(step4)\pi_0(step5)$$

جدول ۳. تعریف نمادهای به‌کار رفته در مدل

نمادها	تعاریف
$\pi_0(CPLX)$	توزیع احتمال پیشین CPLX؛ برای نمونه: $\pi_0(CPLX = low) = 1, \pi_0(CPLX = other\ wise) = 0$
$p(product\|RELY, DATA, CPLX)$	احتمال شرطی محصول برحسب سه مؤلفهٔ RELY، DATA و CPLX
$\pi_0(product)$	توزیع احتمال پیشین محصول
β^{opt}	ضریب بهینه برای اصلاح تلاش صرف شده در هر مرحلهٔ تولید
$mean(effort)$	متوسط تلاش واقعی متعلق به سه بازهٔ کم، متوسط و زیاد که برابر با (۹۰۱/۹۹، ۹۵/۳۲، ۳۵/۲۳۶) است.
$est.effort$	تلاش تخمین زده شده
$MMRE$	متوسط قدرمطلق خطای نسبی
MRE	قدرمطلق خطای نسبی
$Act.effort$	تلاش واقعی

رابطه‌های ۱ تا ۱۰ بیان می‌کند احتمال پیشین گرهٔ والد طبق تئوری بیز، برابر احتمال شرطی گرهٔ والد برحسب گره‌های فرزند ضرب در احتمال پیشین گره‌های فرزند است. برای نمونه، رابطهٔ ۱ به این صورت شرح داده می‌شود: طبق جدول ، مؤلفهٔ RELY می‌تواند دارای ۵ حالت، مؤلفهٔ DATA می‌تواند دارای ۴ حالت و مؤلفهٔ CPLX می‌تواند دارای ۶ حالت باشد. به بیان دیگر، این سه مؤلفه می‌توانند ۱۲۰= ۶ × ۴ × ۵ حالت داشته باشند. برای این که احتمال مؤلفهٔ محصول (product) کم باشد، باید جمع احتمال این ۱۲۰ حالت را محاسبه کرد. به بیانی برای حالت اول، احتمال این که محصول کم باشد برابر است با احتمال شرطی محصول = کم به شرطی که مؤلفهٔ RELY خیلی کم، مؤلفهٔ DATA کم و مؤلفهٔ CPLX خیلی کم باشد، ضرب در احتمال این که مؤلفهٔ RELY خیلی کم، مؤلفهٔ DATA کم و مؤلفهٔ CPLX خیلی کم باشد. در جدول ۱، سطح هر مؤلفه تعیین شده و طبق آن، احتمال پیشین مؤلفه‌ها تعیین می‌شود. به‌طور مثال، برای پروژهٔ اول در این جدول، مؤلفه RELY دارای سطح N است، یعنی احتمال این که مؤلفهٔ RELY متوسط باشد برابر ۱ و احتمال سایر حالت‌ها برابر صفر است به بیانی $\pi_0(RELY) = nominal = 1$ و $\pi_0(RELY) = O.W = 0$ است. برای محاسبهٔ این ۱۲۰ احتمال شرطی، در رابطهٔ ۱ باید از تابع عضویت گروه محصول استفاده کرد. کمترین مقداری که گروه محصول می‌تواند بگیرد وقتی است که مؤلفه RELY، خیلی کم، مؤلفهٔ DATA کم و مؤلفه CPLX خیلی کم باشد که مقدار هر یک طبق جدول ۲ به‌ترتیب برابر

۰/۷۵، ۰/۹۴ و ۰/۷ است. بیشترین مقدار نیز موقعی است که مؤلفهٔ RELY، خیلی زیاد، مؤلفه DATA خیلی زیاد و مؤلفهٔ CPLX فوق‌العاده زیاد باشد که مقدار هر یک طبق جدول ۲ به‌ترتیب برابر ۱/۴، ۱/۱۶ و ۱/۶۵ است. همچنین موقعی مقدار محصول در حالت نرمال است که هر سه مؤلفه در حالت نرمال و دارای مقدار ۱ باشند. بنابراین کمترین مقدار محصول برابر با ۰/۴۹۴ و بیشترین حالت برابر با ۲/۶۸ است که می‌توان به‌صورت عدد فازی (۲/۶۸، ۱، ۰/۴۹۴) نشان داد. حال برای هر یک از ۱۲۰ حالت مقدار سه مؤلفه را از جدول ۲ به‌دست آورده، درهم ضرب می‌کنیم. بدین ترتیب میزان تعلق آن به بازهٔ کم، متوسط و زیاد را طبق محاسبات منطق فازی اعداد مثلثی (رابطه‌های ۱۱ و ۱۲) محاسبه می‌شود.

$$\begin{cases} \dfrac{x-0.494}{1-0.494}, & Nominal \\ 1-\dfrac{x-0.494}{1-0.494}, & Low \\ 0, & High \end{cases}$$

رابطهٔ ۱۱)

$$\begin{cases} \dfrac{2.68-x}{2.68-1}, & Nominal \\ 1-\dfrac{2.68-x}{2.68-1}, & High \\ 0, & Low \end{cases}$$

رابطهٔ ۱۲)

برای نمونه، احتمال شرطی‌ای که محصول کم، متوسط و زیاد باشد، به شرط این که مؤلفهٔ RELY خیلی زیاد، مؤلفهٔ DATA خیلی زیاد و مؤلفهٔ CPLX فوق‌العاده زیاد باشد، به‌ترتیب برابر ۰، ۰ و ۱ است. احتمالات شرطی بین سایر گره‌های والد و فرزند نیز به همین گونه محاسبه می‌شود. مؤلفهٔ دیگر اندازهٔ نرم‌افزار است که به توضیح نیاز دارد، این مؤلفه برابر با هزار خط برنامه‌نویسی است. در بین ۴۰ پروژهٔ مرحله یادگیری، کمترین اندازه ۲/۲ و بیشترین اندازه ۴۲۳ است. میانهٔ این ۴۰ پروژه، یعنی ۳۲ نیز برای اندازهٔ متوسط در نظر گرفته می‌شود. احتمال پیشین مؤلفهٔ اندازه برای هر پروژه از طریق رابطه‌های ۱۳ و ۱۴ محاسبه می‌شود.

$$\begin{cases} \dfrac{x-2.2}{32-2.2}, & Nominal \\ 1-\dfrac{x-2.2}{32-2.2}, & Low \\ 0, & High \end{cases}$$

رابطهٔ ۱۳)

$$\begin{cases} \frac{423-x}{423-32}, & Nominal \\ 1-\frac{423-x}{423-32}, & High \\ 0, & Low \end{cases}$$

رابطهٔ ۱۴)

برای محاسبهٔ ضرایب تبدیل تلاش گروه‌ها به مراحل تولید، پرسشنامه‌ای طراحی شد و ۴۵ خبره در زمینهٔ مهندسی نرم‌افزار آن را تکمیل کردند که متوسط نظرها در جدول ۴ نشان داده شده است.

جدول ٤. تأثیر مؤلفه‌های چهارگانه بر مراحل پنج‌گانهٔ تولید

تأثیر گروه‌ها بر مراحل	طرح‌ریزی	تحلیل و طراحی	کدنویسی	آزمون	پشتیبانی
محصول	۳/۱۰۷	۴/۶۴۲	۳/۶۳۴	۳/۳۰۲	۳/۹۱۵
کامپیوتر	۱/۳۱۶	۲/۶۳۲	۲/۷۸۳	۲/۹۹۱	۳/۳۱
پرسنل	۱/۷۱۹	۲/۷۵۹	۲/۶۰۵	۲/۲۹۷	۱/۵۱۶
پروژه	۲/۱۵۴	۵	۴/۳۰۹	۴	۳/۶۳۴
اندازهٔ نرم‌افزار	۲	۴	۵	۵	۵

آخرین پارامتری که در مرحلهٔ یادگیری شبکه باید تعیین شود، تعداد تکرار بین مراحل تولید است. برای این منظور از معیاری به نام میزان معرفی عیب استفاده می‌شود. این معیار ضریب وجود داشتن عیب نسبت به حالت نرمال را بیان می‌کند. برای نمونه، طبق جدول ۵، اگر مؤلفهٔ RELY دارای سطح VH (خیلی زیاد) باشد، در مرحلهٔ طرح‌ریزی ۰/۷، در مرحلهٔ تحلیل و طراحی ۰/۶۹ و در مرحلهٔ کدنویسی ۰/۶۹ نسبت به حالت نرمال عیب خواهد داشت. برای ۱۴ مؤلفهٔ دیگر نیز این جدول‌ها در مقالهٔ دولانی چالانی (۱۹۹۷) محاسبه شده است. طبق این جدول‌ها می‌توان برای هر پروژهٔ خاص، درصد وجود عیب را برای این سه مرحله محاسبه کرد.

در این مقاله، تعداد تکرار بین مراحل برابر با جمع اعشار اعداد بزرگ‌تر از ۱ برای ۱۵ مؤلفهٔ کوکومو در نظرگرفته شده است. این معیار موجب می‌شود تعداد تکرار بین مراحل برای هر پروژه متفاوت باشد و هر پروژه طبق نیازهای خاص مشتریان و منابع دردسترس، به تعداد برگشت‌های متفاوتی بین مراحل نیاز داشته باشد. این جدول‌ها فقط برای سه مرحله محاسبه شده‌اند و دو مرحلهٔ آزمون و پشتیبانی را شامل نمی‌شوند. بنابراین، برای دو حالت دیگر، از نظر خبرگان ذکر شده در مقالهٔ زارع باقی‌آباد و خادمی زارع (۲۰۱۵) استفاده شده است. تعداد تکرارهای سه مرحلهٔ اول بر اساس معیار DIR و دو مرحلهٔ بعد طبق نظر خبرگان محاسبه شده است که در جدول ۶ مشاهده می‌شود. برای محاسبهٔ تعداد تکرار مراحل طبق شکل ۲، اگر از مرحلهٔ دوم به مرحلهٔ اول

بازگشت صورت گیرد، علاوه‌بر مرحلهٔ اول، مرحلهٔ دوم نیز دوباره انجام خواهد شد. بنابراین برای مرحلهٔ دوم، تعداد تکرار برابر با تعداد تکرار مرحلهٔ دوم به‌اضافهٔ تعداد تکرار مرحلهٔ اول است. نتایج این محاسبات در جدول ۷ درج شده است و درنهایت، تعداد تکرار کلی هر مرحله برابر یک بار اجرای خود مرحله به‌اضافهٔ تعداد تکرار خواهد بود.

جدول ۵. DIR برای مؤلفهٔ RELY در سه سطح خیلی زیاد، متوسط و خیلی کم

RELY	طرح‌ریزی	تحلیل و طراحی	کدنویسی
VH	۰/۷	۰/۶۹	۰/۶۹
N	۱	۱	۱
VL	۱/۴۳	۱/۴۵	۱/۴۵

دولانی‌چالانی (۱۹۹۷)

جدول ٦. میزان بازگشت بین مراحل برای ٤٠ پروژه

	حداقل ٤٠ پروژه	متوسط ٤٠ پروزه	حداکثر ٤٠ پروژه
تعیین نیازها	۰/۰۴	۰/۳۲	۰/۶۷
طراحی و تحلیل	۰/۰۵	۰/۴۲	۰/۸۸
کدنویسی	۰/۰۵	۰/۴۷	۰/۹۹
آزمون	۰/۱۴	۰/۲۸	۰/۴۲
پشتیبانی	۰/۴۱	۰/۵۴	۰/۶۸

جدول ۷. تعداد تکرار مراحل برای ٤٠ پروژه

	حداقل ٤٠ پروژه	متوسط ٤٠ پروزه	حداکثر ٤٠ پروژه
تعیین نیازها	۰/۰۴	۰/۳۲	۰/۶۷
طراحی و تحلیل	۰/۰۵+۰/۰۴=۰/۰۹	۰/۴۲+۰/۳۲=۰/۷۴	۰/۸۸+۰/۶۷=۱/۵۵
کدنویسی	۰/۰۵+۰/۰۵=۰/۱	۰/۴۷+۰/۴۲=۰/۸۹	۰/۹۹+۰/۸۸=۱/۸۷
آزمون	۰/۱۴+۰/۰۵=۰/۱۹	۰/۲۸+۰/۴۷=۰/۷۵	۰/۴۲+۰/۹۹=۱/۴۱
پشتیبانی	۰/۴۱	۰/۵۴	۰/۶۸

حال با داشتن احتمالات پیشین، مؤلفه‌های ورودی و احتمالات شرطی طبق رابطه‌های ۱ تا ۱۰، احتمال پیشین تلاش به‌دست می‌آید که با ضرب در متوسط تلاش واقعی هر یک از بازه‌های کم، متوسط و زیاد، تلاش تخمینی محاسبه می‌شود (رابطهٔ ۱۵).

$$est.effort = \Sigma\pi_0(effort) * mean(effort)$$ رابطهٔ ۱۵)

در این مرحله، با مقایسه تلاش واقعی و تلاش تخمینی، معیار خطا برای ۴۰ پروژه محاسبه می‌شود. معیار خطا به‌صورت زیر تعریف می‌شود:

$$MRE_i = \frac{|\, est.effort_i - Act.effort_i|}{Act.effort_i}$$ رابطهٔ ۱۶)

$$MMRE = \frac{\Sigma_{i=1}^{N} MRE_i}{N}$$ رابطهٔ ۱۷)

$$A = \Sigma_{i=1}^{N} Y_i, \begin{cases} Y_i = 1, if\ , MRE_i \leq 0.25 \\ Y_i = 0, if\ , MRE_i \geq 0.25 \end{cases}$$ رابطهٔ ۱۸)

$$Error = MMRE - \frac{A}{N}$$ رابطهٔ ۱۹)

در روابط بالا، MRE معرف قدرمطلق خطای نسبی؛ MMRE نشان‌دهندهٔ متوسط قدرمطلق خطای نسبی؛ A تعداد پروژه‌هایی که قدرمطلق خطای نسبی آنها کمتر از ۰/۲۵ است؛ N تعداد پروژه‌ها که در این مرحله برابر ۴۰ است و Error معیار خطا است. همان‌طور که از رابطهٔ ۱۹ مشخص است، معیار خطا می‌تواند منفی باشد. با استفاده از روابط بالا، معیار خطا برای ۴۰ پروژه برابر ۰/۹۰۵۷ به‌دست می‌آید.

بهینه‌سازی معیار خطا

در این مرحله، معیار خطا با استفاده از سه الگوریتم بهینه‌سازی علف‌های هرز[1]، ژنتیک[2] و اجتماع پرندگان[3] بهینه می‌شود تا ضریب بهینه برای اصلاح مدل به‌دست آید. در مقالاتی که تاکنون در مورد بهینه‌سازی معیار خطا در زمینه تخمین تلاش نرم‌افزار کار شده، از الگوریتم‌های ژنتیک و اجتماع پرندگان استفاده شده است (زارع، خادمی‌زارع و فلاح‌نژاد، ۲۰۱۶). در این تحقیق نیز علاوه‌بر این دو الگوریتم، از الگوریتم علف‌های هرز که یک الگوریتم جدید بوده و قابلیت وفق‌پذیری و استواری زیادی دارد (زو، لیو، چن، هی و وو، ۲۰۱۵) استفاده می‌شود. از آنجا که

1. Invasive Weed Optimization (IWO)
2. Genetic Algorithm (GA)
3. Particle Swarm Optimization (PSO)

یکی از اهداف این مقاله، محاسبهٔ درصد تلاش صرف شده در هر مرحلهٔ تولید است، تلاش وارد شده به هر مرحله تولید در یک ضریب β ضرب شده و سعی می‌شود این ضریب به‌گونه‌ای بهینه گردد که معیار خطا کمینه شود. به بیانی، بهینه شدن ضریب β علاوه‌بر این که موجب افزایش دقت می‌شود، به محاسبهٔ دقیق‌تر تلاش صرف شده در هر مرحله و تخصیص مناسب‌تر منابع کمک می‌کند. با بهینه شدن β_1 تا β_5، احتمالات شرطی مراحل به شرط گروه‌ها به‌روز شده و به همین ترتیب، احتمالات پیشین مراحل، احتمال شرطی تلاش به شرط مراحل و نیز احتمال پیشین تلاش و درنتیجه مقادیر تخمین به‌روز می‌شوند. در این مقاله از سه الگوریتم اجتماع پرندگان، ژنتیک و علف‌های هرز استفاده می‌شود. توضیحات مربوط به الگوریتم‌های ژنتیک و اجتماع پرندگان برای تخمین تلاش در مقالهٔ زارع و همکاران (۲۰۱۶) آمده است.

الگوریتم علف‌های هرز (IWO)

این الگوریتم توسط محرابیان و لوکاس در سال ۲۰۰۶ با الگوگیری از رفتار علف‌های هرز ارائه شد (محرابیان و لوکاس، ۲۰۰۶). این الگوریتم مانند علف‌های هرز، از قابلیت وفق‌پذیری و استواری قوی برخوردار است و می‌تواند جمعیت تصادفی از اعضا تولید کند. مؤلفه‌های مهم آن، دانه، تابع برازش، علف و حداکثر جمعیت علف‌ها هستند. دانه نشان‌دهندهٔ یک عضو یا یک راه‌حل قبل از ارزیابی برازش است. تابع برازش، ارزش هر راه‌حل (دانه) را تعیین می‌کند. با استفاده از تابع برازش، یک دانه به یک علف یا یک جواب تولید شده در جمعیت فعلی به یک جواب در جمعیت بعدی تبدیل می‌شود. حداکثر تعداد علف‌ها نیز به حداکثر تعداد جمعیت اشاره دارد که تعداد راه‌حل‌های بد بیشتر از آن باید حذف شوند (رضوی‌فر، پالاد و زیو، ۲۰۱۵). الگوریتم علف‌های هرز شامل چهار مرحلهٔ ایجاد جمعیت[1]، تولید مجدد[2]، پخش جمعیت تولید شده در فضا[3] و حذف رقابتی[4] است (زو و همکاران، ۲۰۱۵). ابتدا یک جمعیت اولیه از علف‌های هرز تولید می‌شود. این علف‌ها برازش شده و مطابق با مقدار برازش، دانه تولید می‌کنند. دانه‌های تولید شده و علف‌ها تولید جمعیت جدید کرده، مجدد جمعیت جدید برازش شده و تعداد اعضای بد بیشتر از حداکثر جمعیت حذف می‌شود. به این ترتیب الگوریتم پیش می‌رود تا به شرط توقف برسد.

---

1. Initialize a population
2. Reproduction
3. Spatial dispersal
4. Competitive exclusion

یافته‌های پژوهش

نتایج بهینه‌سازی با استفاده از سه الگوریتم در جدول ۸ نشان داده شده است. همان‌طور که مشخص است، الگوریتم علف‌های هرز (IWO) توانست خطای کمتری به‌دست آورد، بنابراین β_1 تا β_5 به‌دست آمده به‌عنوان ضریب اصلاحی، در مدل وارد می‌شود.

جدول ۸. معیار خطای پروژه‌های مرحلهٔ یادگیری قبل و بعد از بهینه‌سازی

الگوریتم	خطای قبل از بهینه‌سازی	خطای بعد از بهینه‌سازی	β^{opt} به‌دست آمده از بهینه‌سازی
IWO	۰/۹۰۵۷	۰/۳۹۱۳	(۰/۳، ۰/۰۱۴، ۰/۲۱۵، ۲/۰۷۲، ۰/۳۸۵)
GA		۰/۴۶۴۱	(۰/۲، ۰/۸، ۰/۸۶۷، ۰/۸، ۰/۸)
PSO		۰/۴۰۱۴	(۰/۲۹۸، ۰/۸۸، ۰/۱۹۹، ۰/۹۳۷، ۰/۵۸)

آزمون مدل

در این مرحله با استفاده از ضریب بهینهٔ به‌دست آمده از مرحلهٔ بهینه‌سازی معیار خطا، تلاش ۲۰ پروژه باقی‌مانده برآورد می‌شود. نتایج مرحلهٔ آزمون را در جدول ۹ مشاهده می‌کنید.

جدول ۹. معیار خطای پروژه‌های مرحلهٔ آزمون قبل و بعد از بهینه‌سازی

الگوریتم	خطا قبل از بهینه‌سازی	β^{opt} به‌دست آمده از بهینه‌سازی پروژه‌های یادگیری	خطای بعد از بهینه‌سازی
IWO	۰/۲۱۶۸	(۰/۳، ۰/۰۱۴، ۰/۲۱۵، ۲/۰۷۲، ۰/۳۸۵)	۰/۱۱۳۶

با توجه به پایگاه داده و جدول ۶ برای پروژهٔ اول، تعداد تکرار اجرای مرحلهٔ اول برابر ۱/۶۷، مرحلهٔ دوم برابر ۲/۴۹، مرحلهٔ سوم برابر ۲/۶۶، مرحلهٔ چهارم برابر ۲/۱۲ و مرحلهٔ پنجم برابر ۱/۵۴ است. ضرایب تبدیل تلاش گروه‌ها به مراحل نیز در جدول ۴ نشان داده شده است. همچنین ضرایب بهینه برای اصلاح تقسیم تلاش در مراحل در جدول ۹ آمده است. بنابراین برای پروژهٔ اول، ضریب تلاش وارد شده به مرحلهٔ اول برابر با (ضریب تبدیل گروه محصول ×

ضریب تلاش گروه محصول + ضریب تبدیل گروه کامپیوتر × ضریب تلاش گروه کامپیوتر+ ضریب تبدیل گروه پرسنل × ضریب تلاش گروه پرسنل + ضریب تبدیل گروه پروژه × ضریب تلاش گروه پروژه + ضریب تبدیل سایز × سایز) × تعداد تکرار × ضریب بهینه مرحلهٔ اول است. با توجه به جدول ۲، ضریب تلاش گروه محصول از حاصل ضرب سه مؤلفهٔ RELY، DATA و CPLX به‌دست می‌آید که برای پروژهٔ اول برابر ۱/۳۳ است. به همین ترتیب، ضریب تلاش گروه‌های کامپیوتر، پرسنل و پروژه ۱/۴۶، ۰/۶۸ و ۰/۷۴ به‌دست می‌آیند. بنابراین طبق رابطهٔ بالا برای پروژهٔ اول، ضریب تلاش مرحلهٔ اول برابر است با:

(۳۲/۶ × ۲+۰/۷۴ × ۲/۱۵۴+۰/۶۸ × ۱/۷۱۹+۱/۴۶ × ۱/۳۱۶+۱/۳۳ × ۳/۱۰۷) × ۰/۳۸۵ × ۱/۶۷ = ۴۷/۵۸۸

به همین ترتیب، ضریب تلاش مرحله‌های ۲ تا ۵ برابر با ۷۵۳/۲۱۹، ۱۰۱/۱۴۴، ۵/۲۳۱ و ۸۰/۸۴۷ به‌دست می‌آید. طبق ضرایب تلاش به‌دست آمده، درصد تلاش صرف‌شده در مراحل به‌ترتیب ۴/۸، ۷۶، ۱۰/۲، ۰/۵۲۹ و ۸/۱۸ است. همچنین تلاش واقعی برابر با ۱۷۰ نفر ــ ماه و تلاش تخمینی برابر با ۱۷۸/۴۱۹ است که طبق درصدهای به‌دست آمده، در مراحل صرف می‌شود و بر اساس آن، می‌توان منابع انسانی و فنی را برنامه‌ریزی کرد و به خوبی تخصیص داد.

اعتبارسنجی مدل

مقایسه با مدل شبکهٔ بیزی برپایهٔ مؤلفه‌های کوکومودو

در این بخش، مدل شبکهٔ بیزی که تنها براساس مؤلفه‌های کوکومودو طراحی شده است، بررسی می‌شود. مؤلفه‌های این مدل همان ۱۵ مؤلفهٔ کوکومودو بررسی شده در مدل پیشنهادی بوده و پایگاه دادهٔ آن نیز همان پایگاه داده مدل پیشنهادی است. معیار خطا با استفاده از الگوریتم ژنتیک بهینه‌سازی شده و ضریب بهینه برای اصلاح توزیع احتمال پیشین تلاش ارائه شده است. نتایج معیار خطا برای پروژه‌های مرحلهٔ آزمون قبل و بعد از بهینه‌سازی و نیز ضریب بهینه برای اصلاح مدل، در جدول ۱۰ نشان داده شده است. البته این مقاله، ایده‌ای برای میزان تلاش صرف شده در هر مرحله از تولید نرم‌افزار ارائه نمی‌دهد.

جدول ۱۰. نتایج آزمون شبکهٔ بیزی برپایهٔ مؤلفه‌های کوکومودو

الگوریتم بهینه‌سازی	معیار خطا قبل از بهینه‌سازی	λ^{opt} حاصل از بهینه‌سازی پروژه‌های یادگیری	معیار خطا بعد از بهینه‌سازی
GA	۰/۴۷	(۱ ،۰ ،۰/۲)	−۰/۰۸۲

زارع و همکاران (۲۰۱۶)

مقایسه با مدل گرت برپایهٔ مؤلفه‌های کوکومودو و فرایند تولید

در این قسمت، با استفاده از پیاده‌سازی مدل گرت ارائه‌شده در مقالهٔ زارع باقی‌آباد و خادمی‌زارع (۲۰۱۵) روی پایگاه دادهٔ مدل پیشنهادی، یعنی همان پایگاه دادهٔ کوکومو ـ ناسا، معیار خطا محاسبه می‌شود. معیار خطا نیز مشابه معیار خطای مدل پیشنهادی در نظرگرفته شده است. در این قسمت معیار خطا برای ۶۰ پروژه محاسبه می‌شود؛ زیرا مدل گرت دارای فرایند یادگیری نیست. برای تک‌تک پروژه‌ها، براساس ۱۵ مؤلفه و نیز اندازهٔ نرم‌افزار، تلاش خوش‌بینانه، محتمل و بدبینانه براساس رابطه‌های ۳ تا ۵ مقالهٔ زارع باقی‌آباد و خادمی‌زارع (۲۰۱۵) محاسبه شده و با توجه به درصدهای مندرج در جدول ۵ مقالهٔ مذکور، بین مراحل تولید تقسیم می‌شود. محاسبات گرت روی شبکهٔ مراحل انجام شده و نفرـ ماه نهایی به‌دست می‌آید. نتایج تخمین با مقادیر واقعی پایگاه داده کوکومو ـ ناسا مقایسه شده و معیار خطای $MMRE - \frac{A}{N}$ برابر با ۰/۶۵۷ به‌دست آمد.

تحلیل عددی

در دو قسمت قبل سعی شد مدل پیشنهادی با مدل‌هایی که از نظر معیار خطا به هم مشابهت دارند، مقایسه شود. از مقایسهٔ مدل‌ها نتایج زیر به‌دست آمد:

- مدل پیشنهادی با دقت خوبی (۰/۱۱۳۶) توانست تلاش موردنیاز برای تولید نرم‌افزار را تخمین بزند. علاوه‌بر این که برای هر پروژه با پیچیدگی‌ها و نیازهای متفاوت مشتریان، می‌تواند تلاش صرف شده در هر مرحله را نیز تعیین کند و با این کار دقت تخصیص منابع در زمان موردنیاز را افزایش داده و ریسک دیرکرد پروژه را کاهش دهد.
- خطای مدل پیشنهادی قبل از بهینه‌سازی برای دسته پروژه‌های آزمون، کمتر از خطای مدل شبکهٔ بیزی بر پایهٔ مؤلفه‌های کوکومودو بود. به بیانی ۰/۲۱۶۸ < ۰/۴۷ است. بعد از بهینه‌سازی خطای مدل پیشنهادی نصف شد و به ۰/۱۱۳۶ رسید، اما خطای مدل شبکهٔ بیزی بر پایهٔ مؤلفه‌ها به ۰/۰۸۲− رسید؛ شایان ذکر است که این مدل، هیچ ایده‌ای برای درصد تلاش صرف شده در مراحل و برنامه‌ریزی منابع که یکی از اهداف این مقاله است، ارائه نمی‌کند.
- در مقایسهٔ مدل پیشنهادی با مدل گرت برپایهٔ مؤلفه‌های کوکومودو و فرایند تولید، خطای مدل گرت ۰/۶۵۷ > ۰/۱۱۳۶ به‌دست آمد. خطای مدل پیشنهادی قبل و بعد از بهینه‌سازی در مقایسه با خطای مدل گرت کمتر است، ضمن این که در مدل گرت،

درصد تلاش صرف شده در ۸ مرحلهٔ تولید برای تمام پروژه‌ها برابر با ۳، ۵، ۲۰، ۱۷، ۲۵، ۲۴، ۱۱ و ۵ در نظر گرفته شده است. درصورتی که هر پروژه‌ای با ویژگی متفاوت، به تعداد تکرارهای متفاوتی بین مراحل و در نتیجه درصد تلاش‌های صرف شدهٔ متفاوتی بین مراحل نیاز دارد. علاوه براین، بهینه‌سازی و به‌روزرسانی مدل انجام نشده است.

نتیجه‌گیری

در این مقاله یک شبکهٔ بیزی سه سطحی برپایهٔ مؤلفه‌های کوکومودو و فرایند تولید برای تخمین تلاش کلی نرم‌افزار و تلاش هر مرحله از تولید آن ارائه شده است. تمام بازه‌های موجود در گره‌های شبکه، به‌صورت پیوسته به شکل اعداد فازی مثلثی در نظر گرفته شده‌اند. در سطح اول، مؤلفه‌های کوکومودو بر اساس چهار گروه محصول، کامپیوتر، پرسنل و پروژه وارد می‌شوند. در سطح دوم، گروه‌ها همراه با مؤلفهٔ اندازهٔ نرم‌افزار با ضریب تبدیل‌هایی که توسط خبرگان تعیین شده است، به پنج مرحلهٔ تولید وارد می‌شوند. بعد از اعمال حلقه‌های بازگشت بین مراحل و به‌روزرسانی احتمالات شرطی، تلاش نهایی برآورد می‌شود. معیار خطا با استفاده از الگوریتم‌های متاهیوریستیک بهینه‌سازی شده و ضریب بهینه برای اصلاح تبدیل تلاش گروه‌ها به مراحل، به‌دست آمده است. بهینه‌سازی مدل موجب می‌شود که خطا از ۰/۲۱۶۸ به ۰/۱۱۳۶ کاهش یابد. همچنین مقایسهٔ این مدل با مدل گرت نشان داد این مدل، دقت بیشتری برای تخمین تلاش داشته است؛ ضمن این که برای هر پروژه، درصد تلاش صرف شدهٔ متفاوتی در هر مرحلهٔ تولید به دست می‌دهد که موجب برنامه‌ریزی بهتر منابع شده و ریسک دیرکرد پروژه را کاهش می‌دهد.

در این مقاله اثر کیفیت نرم‌افزار در قالب تعداد تکرار مراحل تولید دیده شده است. توسعهٔ مدل‌های دیگری که در تخمین تلاش نرم‌افزار، سطح کیفیت موردنیاز نرم‌افزار را در نظر بگیرند، می‌تواند به‌عنوان یک زمینهٔ تحقیقاتی مطرح شود. همچنین استفاده از ابزارهای مختلف داده‌کاوی برای ساخت خودکار شبکهٔ بیزی و محاسبات سریع‌تر احتمالات شرطی برپایهٔ پایگاه داده پیشنهاد می‌شود؛ زیرا موجب صرف وقت کمتر و کاهش اشتباهات محاسباتی می‌شود. همچنین رویکرد شبکهٔ بیزی ارائه شده می‌تواند در قالب یک محصول نرم‌افزاری برای تعیین قیمت نرم‌افزارهای مختلف، تجاری‌سازی شود.

فهرست منابع

خادمی زارع، ح.، زارع باقی‌آباد، ف. (۱۳۹۰). یک الگوریتم ترکیبی برای تخمین زمان و هزینهٔ پروژه‌های تولید نرم‌افزار در شرایط فازی، *هفتمین کنفرانس بین‌المللی مدیریت پروژه*، تهران، ایران

خانلری، ا.، کفائی، ا. (۱۳۹۳). بررسی تأثیر ابعاد ساختاری بر موفقیت سیستم برنامه‌ریزی منابع سازمانی در شرکت‌های ایرانی دارندهٔ این سیستم، *نشریهٔ مدیریت فناوری اطلاعات*، ۶ (۱)، ۷۰-۴۷.

رئیسی وانانی، ا، گنجعلی‌خان حاکمی، ف. (۱۳۹۴). طراحی سیستم استنتاج فازی ـ عصبی انطباقی برای ارزیابی استقرار سیستم هوشمندی کسب‌وکار در صنعت تولید نرم‌افزار، *نشریهٔ مدیریت فناوری اطلاعات*، ۷(۱)، ۱۰۴-۸۵.

گمنام سفیددارینی، م.، ناصرزاده، م. ر.، روحانی، س.، قاهردوست، ع. ر. (۱۳۹۳). بررسی آثار متقابل عوامل بحرانی شکست پروژه‌های پیاده‌سازی ERP در صنایع ایران. *نشریهٔ مدیریت فناوری اطلاعات*، ۶(۴)، ۶۷۴-۶۴۹.

Akhtar, N. (2013). Perceptual Evolution for Software Project Cost Estimation using Ant Colony System. *International Journal of Computer Applications*, 81 (14), 23-30.

Bibi, S. & Stamelos, I. (2004, September). Software Process Modeling with Bayesian Belief Networks. *Proceedings of the 10th International Software Metrics Symposium,* Chicago, USA.

Fernández-Diego M. & Torralba-Martínez, J.M. (2012, September). Discretization Methods for NBC in Effort Estimation: An Empirical Comparison based on ISBSG Projects. *Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM),* ACM, Lund, 103-106.

Fuentetaja, R., Borrajo, D., López C. L. & Ocón, J. (2013). Multi-step Generation of Bayesian Networks Models for Software Projects Estimations. *International Journal of Computational Intelligence Systems,* 6 (5), 796-821.

Gomnam Sefiddarboni, M., Naserzadeh, S.M.R., Rouhani, S. & Ghaherdoost, A.R. (2015). Investigating mutual effects of critical failure factors of ERP implementation in Iranian industries with Grey-based DEMATEL method. *Journal of Information Technology Management,* 6(4), 649-674. (*in Persian*)

Hotle, M. (2008). Waterfalls, Products and Projects: A Primer to Software Development Methods. Available in: https://www.gartner.com/doc/611708/ waterfalls-products-projects-primer-software.

Idri, A., Amazal, F. A. & Abran, A. (2015). Analogy-based software development effort estimation: A systematic mapping and review. *Information and Software Technology*, 58, 206-230.

Khan, J., Shaikh, Z. A. & Nauman, A. B. (2011). Development of intelligent effort estimation model based on fuzzy logic using Bayesian networks. *Software Engineering, Business Continuity, and Education*, 257, 74-84.

Khanlari, A. & Kafaee, O. (2014). Investigating the impact of Organizational Structure on ERP post-implementation success, a study on Iranian Firms. *Journal of Information Technology Management*, 6(1), 47-70. (*in Persian*)

Lopez-Martin, C. (2015). Predictive accuracy comparison between neural networks and statistical regression for development effort of software projects. *Applied Soft Computing*, 27, 434-449.

Mehrabian A. R. & Lucas, C. (2006). A novel numerical optimization algorithm inspired from weed colonization. *Ecological informatics,* 1(4), 355-366.

Pendharkar, P. C., Subramanian G. H. & Rodger, J. A. (2005). A Probabilistic Model for Predicting Software Development Effort. *IEEE Transactions on software engineering*, 31 (7), 615-624.

Perkusich, M., Soares, G., Almeida H. & Perkusich, A. (2015). A procedure to detect problems of processes in software development projects using Bayesian networks. *Expert Systems with Applications*, 42 (1), 437-450.

Raeesi Vanani, I. & Ganjalikhan Hakemi, F. (2015). Designing an Adaptive Nuero-Fuzzy Inference System for Evaluating the Business Intelligence System Implementation in Software Industry. *Journal of Information Technology Management*, 7(1), 85-104. (*in Persian*)

Stamelos, I., Angelis, L., Dimou P. & Sakellaris, E. (2003). On the use of Bayesian belief networks for the prediction of software productivity. *Information and Software Technology,* 45 (1), 51–60.

Uusitalo, L. (2007). Advantages and challenges of Bayesian networks in environmental modelling. *Ecological modelling*, 203 (3/4), 312–318.

Zare, F., Zare H. K. & Fallahnezhad, M. S. (2016). Software effort estimation based on the optimal Bayesian belief network. *Applied Soft Computing*, 49, 968-980.

Zarebaghiabad, F. & Khademizare, H. (2012). Proceedings of the 7th International Project Management Conference, Tehran, Iran. (*in Persian*)

Zarebaghiabad, F. & Khademizare, H. (2015). A Three- Stage Algorithm for Software Cost and Time Estimation in Fuzzy Environment. *International Journal of Industrial Engineering & Production Research*, 26, (3), 193-211.