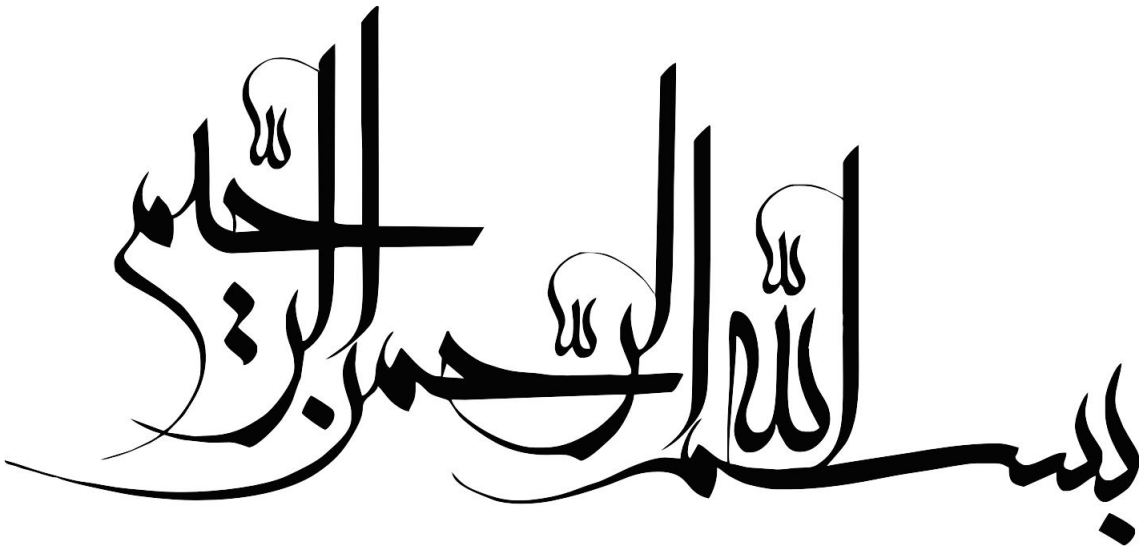
بسم الله الرحمن الرحيم

دانشگاه اصفهان

دانشکده مهندسی کامپیوتر

گروه مهندسی نرم‌افزار

پایان‌نامه‌ی کارشناسی ارشد
رشته مهندسی کامپیوتر گرایش نرم‌افزار

روشی جدید جهت تولید بهینه‌ی موارد آزمون بر اساس ماشین حالت UML

استاد راهنما:
دکتر بهمن زمانی

پژوهشگر:
امین رضائی

بهمن ماه ۱۳۹۲

کلیه حقوق مادی و معنوی مترتب بر دست‌آوردهای مطالعات، ابتکارات و نوآوری‌های ناشی از پژوهش موضوع این پایان‌نامه متعلق به دانشگاه اصفهان است. دانشجو موظف به رعایت آئین‌نامه و منشور اخلاق در پژوهش برای ارائه و یا چاپ مطالب مستخرج از پایان‌نامه خود می‌باشد.

دانشگاه اصفهان

دانشکده مهندسی کامپیوتر

گروه مهندسی نرم‌افزار

پایان‌نامه‌ی کارشناسی ارشد رشته مهندسی کامپیوتر گرایش نرم‌افزار آقای امین رضایی تحت عنوان

روشی جدید جهت تولید بهینه‌ی موارد آزمون بر اساس ماشین حالت UML

در تاریخ ..... توسط هیات داوران زیر بررسی و با درجه ..... به تصویب نهایی رسید.

۱- استاد راهنمای پایان‌نامه	دکتر بهمن زمانی	با مرتبه علمی استادیار	امضا
۲- استاد داور داخل گروه	دکتر افسانه فاطمی خوراسگانی	با مرتبه علمی استادیار	امضا
۳- استاد داور خارج از گروه	دکتر ناصر قدیری	با مرتبه علمی استادیار	امضا

امضای مدیر گروه

سپاس‌گزاری

ای هستی بخش، وجود مرا بر نعمات بی‌کرانت توان شکر نیست. ذره ذره‌ی وجودم برای تو و نزدیک شدن به تو می‌تپد.
الهی مرا مدد کن تا دانش اندکم نه نردبانی باشد برای فزونی تکبر و غرور، نه حلقه‌ای برای اسارت و نه دست مایه‌ای برای تجارت، بلکه گامی باشد برای تجلیل از تو و متعالی ساختن زندگی خود و دیگران.
حال که توفیق جمع‌آوری و تهیه‌ی این مجموعه را یافته‌ام بر خود واجب می‌دانم از تمامی عزیزانی که در طی انجام این پژوهش از راهنمایی و یاری‌شان بهره‌مند گشته‌ام تشکر و قدردانی کنم و برای ایشان از درگاه پروردگار مهربان آرزوی سعادت و پیروزی نمایم.
در ابتدا صمیمانه‌ترین تقدیرها، تقدیم به خانواده عزیز و مهربانم که همواره حامی و مشوقم بوده‌اند و پیمودن روزهای سخت و آسان زندگی‌ام بدون دعای خیر و برکت وجودشان غیرممکن بود.
همچنین از استاد ارجمند و عزیزم، دکتر بهمن زمانی، که با سعه‌ی صدر و صبوری مرا راهنمایی نموده و با ارائه‌ی نظرات سازنده و رهنمودهای بی دریغشان در پیشبرد این پایان‌نامه مرا یاری رسانده‌اند و زحمت راهنمایی بنده را بر دوش داشته‌اند، کمال تشکر و قدردانی را دارم.
در آخر نیز، لازم است که از آقایان Stephan Weißleder و Felix Kurth بابت همکاری و راهنمایی‌های بی‌دریغ‌شان جهت انجام این پژوهش، کمال تشکر و قدردانی را داشته باشم.

تقدیم به خدایی که آفرید
جهان را، انسان را، عقل را، علم را، معرفت را، عشق را
و به کسانی که خداوند عشقشان را در وجودم دمید.
تقدیم به پدر و مادر بزرگوارم.

چکیده

آزمون یکی از فعالیت‌های اصلی و مهم در فرآیند تولید نرم‌افزار و نیز اصلی‌ترین روش جهت ارزیابی کیفیت نرم‌افزار تولیدی است. انجام آزمون به صورت دستی و سنتی دارای معایبی چون پیچیدگی زیاد، زمان‌بر بودن و مستعد خطا بودن است. بنابراین، خودکارسازی آزمون و تولید بهینه‌ی موارد آزمون با قدرت کشف خطای بالا و در زمان مناسب، بسیار مهم می‌باشد. آزمون مبتنی بر مدل، رویکرد جدیدی است که برای آزمون نرم‌افزار مورد استفاده قرار می‌گیرد و هدف از آن، تولید خودکار موارد آزمون از روی مدل آزمون می‌باشد.

در این پژوهش، روشی جدید جهت تولید بهینه‌ی موارد آزمون با استفاده از آزمون مبتنی بر مدل ارائه شده است. مدل آزمون مورد استفاده، متشکل از نمودار کلاس و ماشین حالت UML می‌باشد، که قیود و محدودیت‌های سیستم نیز توسط OCL بر روی این مدل درج می‌شوند. برای بیان قیود OCL، از ویژگی‌های نمودار کلاس استفاده می‌شود. همچنین اعمال موجود در نمودار کلاس می‌توانند دارای تعدادی پس/ پیش شرط باشند که به زبان OCL بیان می‌گردند. این اعمال می‌توانند به‌عنوان نتیجه‌ی انتقالاتِ ماشین حالت، مورد استفاده قرار گیرند. در روش ارائه شده، ابتدا توسط یک الگوریتم روبه‌جلو و عمق‌اول، مسیرهای انتزاعی از ماشین حالت، بر اساس معیارهای مشخصی انتخاب می‌شوند. سپس با استفاده از اجرای نمادین، مسیرهای انتزاعی تولید شده همراه با قیود موجود بر روی اجزای مسیر انتزاعی، به یک نمایش ریاضی در زبان AMPL تبدیل می‌شوند. آن‌گاه مدل ریاضی تولید شده، با استفاده از حل‌کننده‌های به‌روز و قدرت‌مند که با AMPL در ارتباط هستند، حل شده و داده‌های آزمون برای هر مسیر انتزاعی تولید می‌گردند. نهایتاً، این داده‌ها به موارد آزمون قابل اجرا تبدیل می‌شوند. جهت هدایت الگوریتم جست‌وجو و نیز به‌عنوان معیاری جهت سنجش کیفیت موارد آزمون تولیدی، از معیارهای پوشش استفاده شده است. همچنین، جهت افزایش قدرت کشف خطای الگوریتم، از تحلیل مقادیر مرزی جهت تولید داده‌های مرزی آزمون، استفاده شده است. با بهره‌گیری از حل‌کننده‌های مختلف، موارد آزمون از روی مسئله‌های گوناگون مانند مسئله‌های خطی، غیرخطی، تصمیم‌پذیر و تصمیم‌ناپذیر، که در مدل آزمون با استفاده از قیود OCL تعریف شده‌اند، تولید می‌گردد.

همچنین، یک تبدیل مدل به مدل بر روی ماشین حالت مربوط به مدل آزمون، جهت تبدیل شبه‌حالت تاریخچه‌ی عمیق و کم‌عمق به شبه‌حالت انتخاب، ارائه می‌گردد. با استفاده از این تبدیل مدل به مدل، می‌توان انتظار داشت که موارد آزمون تولیدی از روی مدل تبدیل شده، تعداد خط بیشتری را در کد منبع مربوط به مدل آزمون مورد پوشش قرار دهند. بنابراین می‌توان انتظار داشت که نرخ کشف خطای بالاتری، نسبت به موارد آزمون تولیدی از روی مدل اصلی، داشته باشند.

کلمات کلیدی:

آزمون مبتنی بر مدل، آزمون خودکار، ماشین حالت UML، زبان برنامه‌ریزی ریاضی AMPL، حل‌کننده‌ی قیود.

فهرست مطالب

عنوان **صفحه**

عنوان **صفحه**

عنوان **صفحه**

فهرست شکل‌ها

عنوان **صفحه**

عنوان **صفحه**

فهرست جدول‌ها

عنوان **صفحه**

فصل ۱

توصیف کلیات پژوهش

۱.۱ مقدمه

امروزه با توجه به پیچیده‌تر شدن سیستم‌های کامپیوتری، استفاده از مدل به عنوان یک ضرورت به ویژه در مهندسی نرم‌افزار مطرح می‌شود. برخلاف برخی از فرآیندهای توسعه‌ی نرم‌افزار که محور اصلی آن‌ها کد می‌باشد، در مهندسی نرم‌افزار مدل‌رانده[۱]، مدل محصول اصلی می‌باشد که فرآیند توسعه‌ی نرم‌افزار را پیش می‌برد. هدف نهایی از مهندسی نرم‌افزار مدل‌رانده، تولید خودکار نرم‌افزار از روی مدل مربوطه می‌باشد [۱].

آزمون نرم‌افزار[۲]، بخشی مهم از فرآیند توسعه‌ی نرم‌افزار و عمومی‌ترین روش جهت اطمینان از کیفیت نرم‌افزار می‌باشد که حدود ۵۰٪ از تلاش و بودجه‌ی پروژه صرف انجام این عمل می‌گردد [۲]. شدت خطاها در نرم‌افزار می‌توانند در محدوده‌ای از یک اشتباه کوچک تا اشتباهی که منجر به انهدام سیستم شود، قرار گیرند. گاهی اوقات این خطاها می‌توانند سبب خسارات جانی یا مالی جبران‌ناپذیری شوند. برای مثال، خطا در سیستم‌های نرم‌افزاری مربوط به کنترل ترافیک هوایی، سیستم‌های پزشکی و یا سیستم‌های نظامی، ممکن است موجب مرگ انسان‌ها شود. هدف آزمون نرم‌افزار، یافتن این خطاها می‌باشد.

انجام آزمون به صورت دستی و سنتی دارای معایبی چون پیچیدگی زیاد، زمان‌بر بودن و مستعد خطا بودن

---

[۱] Model Driven Software Engineering (MDSE)

[۲] Software Testing

است. همچنین به طور معمول، هنگامی که یک کار پیچیده به انسان محول شود این امکان وجود دارد که آن کار به طور ناقص و همراه با خطا انجام شود. از طرفی، آزمون جامع و کامل نرم‌افزار در عمل امکان‌پذیر نمی‌باشد، به همین دلیل، آزمون نمی‌تواند اثباتی از درستی نرم‌افزار ارائه دهد. بنابراین، خودکارسازی آزمون و تولید بهینه‌ی موارد آزمون[۱] با قدرت کشف خطای بالا و در زمان مناسب، بسیار مهم می‌باشد.

یکی از روش‌های آزمون خودکار نرم‌افزار، آزمون مبتنی بر مدل[۲] می‌باشد. در آزمون مبتنی بر مدل، مدلی از رفتار مورد انتظار سیستم یا مدل محیطی سیستم، ایجاد می‌شود. سپس با استفاده از این مدل، موارد آزمون می‌توانند به صورت خودکار ایجاد گردند. زبان مدل‌سازی یکنواخت (UML)[۳] یکی از پرکاربردترین زبان‌های مدل‌سازی می‌باشد که برای مدل‌سازی و بیان رفتار سیستم می‌توان از آن استفاده نمود. در این پژوهش نیز از زبان مدل‌سازی UML، جهت توصیف رفتار سیستم استفاده شده است. برای بیان محدودیت‌های سیستم نیز، از زبان قید شئ OCL[۴] استفاده شده است.

به‌دلیل این‌که UML قابلیت اجرایی شدن و بررسی قیود OCL و نیز تولید داده‌هایی که منجر به برآورده شدن این قیود شوند را ندارد، نیاز است که الگوریتمی جهت اجرایی شدن و تبدیل مدل و قیود تعریف شده بر روی آن تعریف گردیده و موارد آزمون قابل اجرا تولید گردند. همچنین، نیاز است که موارد آزمون تولیدی دارای قدرت کشف خطای بالایی بوده و در مدت زمان قابل قبولی تولید گردند. الگوریتم ارائه شده جهت تولید موارد آزمون نیز، باید قدرت حل مسئله‌های مختلف از نظر پیچیدگی را داشته باشد.

در این پژوهش، مدل آزمون و همچنین قیود موجود بر روی آن، که با استفاده از UML و OCL توصیف شده‌اند، تبدیل به یک مسئله‌ی ریاضی در زبان مدل‌سازی ریاضی AMPL[۵] شده و با استفاده از حل‌کننده‌های[۶] به‌روز و قدرت‌مند که توسط AMPL پشتیبانی می‌شوند، حل شده و موارد آزمون تولید می‌گردد. علاوه‌بر این، در این پژوهش یک تبدیل مدل آزمون، جهت افزایش کیفیت موارد آزمون تولیدی، ارائه گردیده است.

در این فصل، ابتدا در بخش ۲.۱ مسئله‌ی پژوهش معرفی می‌گردد. سپس در بخش ۳.۱ راه‌کار ارائه شده در این پژوهش جهت حل مسئله‌ی مطرح شده و تولید بهینه‌ی موارد آزمون به اختصار، مطرح می‌شود. راه‌کار ارائه شده شامل الگوریتمی جهت تولید موارد آزمون انتزاعی و همچنین راه‌کاری جهت تولید داده‌های آزمون و موارد آزمون اجرایی می‌باشد. در ادامه، در بخش ۴.۱ نحوه‌ی ارزیابی راه‌کار ارائه شده مشخص می‌گردد. بخش ۵.۱ ساختار پایان‌نامه و فصل‌بندی آن را مشخص می‌کند.

۲.۱ بیان مسئله

مهندسی نرم‌افزار مدل‌رانده، ایده‌ای است که توسط صنعت و دانشگاه مورد استقبال قرار گرفته است و نمونه‌های موفقیت‌آمیز زیادی از استفاده‌ی مهندسی نرم‌افزار مدل‌رانده هم در صنعت و هم به صورت تحقیقات دانشگاهی گزارش شده است [۳]. هدف اصلی مهندسی نرم‌افزار مدل‌رانده تولید کد قابل اجرا از روی مدل سیستم می‌باشد. آزمون نرم‌افزار یکی از فعالیت‌های اصلی و مهم در فرآیند توسعه‌ی نرم‌افزار می‌باشد. هدف از این پژوهش خودکارسازی این فعالیت مهم در جهت دست‌یابی به اهداف مهندسی نرم‌افزار مدل‌رانده و همچنین افزایش کیفیت موارد آزمون تولیدی می‌باشد.

---

[۱] Test Cases
[۲] Model Based Testing (MBT)
[۳] Unified Modeling Language (UML)
[۴] Object Constraint Language (OCL)
[۵] A Mathematical Programming Language (AMPL)
[۶] Solver

UML یکی از پرکاربردترین و شناخته‌شده‌ترین زبان‌های مدل‌سازی می‌باشد که برای بیان قیود بر روی اجزای آن، از OCL استفاده می‌شود. از آن‌جا که UML و OCL قابلیت اجرایی شدن را ندارد، نیاز است که الگوریتمی جهت اجرایی شدن و تبدیل مدل آزمون و قیود تعریف شده بر روی آن به کد آزمون قابل اجرا ارائه گردد. از طرفی تولید بهینه‌ی موارد آزمون با قدرت کشف خطای بالا و در زمان مناسب، بسیار مهم می‌باشد. این پژوهش سعی در ارائه‌ی راه‌کار و الگوریتمی جهت تولید بهینه‌ی موارد آزمون از روی مدل UML دارد. موارد آزمون تولیدی باید در کم‌ترین زمان ممکن تولید گردند و نرخ کشف بالایی را نیز داشته باشند.

۳.۱ معرفی راه‌حل

رویکردی که در این پژوهش مورد استفاده قرار گرفته است، بر پایه‌ی الگوریتم جست‌وجوی عمق اول[۱] و روبه‌جلو[۲] بر روی ماشین حالت، جهت تولید سناریوی آزمون انتزاعی می‌باشد. آزمون انتزاعی مسیری در ماشین حالت می‌باشد که با استفاده از معیار خاصی که کیفیت موارد آزمون تولیدی را مشخص می‌کند انتخاب می‌شود. الگوریتم جست‌وجوی ماشین حالت، سعی در برآورده ساختن معیارهای کیفیت آزمون دارد. همچنین جهت اجرایی ساختن مدل ورودی و نیز سناریوهای انتزاعی آزمون و تولید بهینه‌ی داده‌های آزمون، از اجرای نمادین و حل‌کننده‌های قیود استفاده شده است. همچنین در این پژوهش سعی در حل مسئله‌های مختلف که از نظر پیچیدگی در گروه‌های مختلف قرار دارند، شده است. بدین معنی که راه‌کار ارائه شده نه تنها قادر به حل مسئله‌ها و قیود ساده می‌باشد، بلکه می‌تواند مسئله‌های پیچیده را نیز حل نموده و موارد آزمون را تولید نماید. علاوه‌بر این، از تبدیل مدل آزمون جهت افزایش کیفیت موارد آزمون تولیدی، استفاده شده است. از این طریق می‌توان کیفیت موارد آزمون تولیدی و قدرت کشف خطای آن‌ها را افزایش داد.

در ادامه در بخش ۱.۳.۱ به معرفی الگوریتم جست‌وجوی گراف، جهت تولید موارد آزمون انتزاعی پرداخته می‌شود. سپس در بخش ۲.۳.۱ در مورد نحوه‌ی اجرایی کردن مدل آزمون و تولید داده‌های آزمون صحبت خواهد شد و در بخش ۳.۳.۱ نیز، تبدیل مدل آزمون مورد بررسی قرار می‌گیرد.

۱.۳.۱ جست‌وجوی گراف

در این پژوهش، در ابتدا سعی می‌گردد که موارد آزمون انتزاعی با استفاده از الگوریتم جست‌وجوی گراف عمق‌اول و روبه‌جلو بر روی ماشین حالت و با هدف برآورده ساختن معیارهای پوشش ساختاری که تبدیل به اهداف آزمون شده‌اند، تولید گردند. برای دریافت مدل ورودی و تبدیل معیارهای پوشش به اهداف آزمون، از ابزار ParTeG [۳] استفاده شده است. همچنین با استفاده از این ابزار، اجزای مدل ورودی در یک ساختار میانی که توسط ابزار ارائه شده است ذخیره می‌شوند و پس از آن، عملیات تولید موارد آزمون با استفاده از ساختار داخلی، ادامه می‌یابد. با استفاده از الگوریتم جست‌وجوی گراف که در این پژوهش پیاده‌سازی شده است، سعی می‌گردد که اهداف آزمون تولیدی برآورده شوند. در این مرحله مسیرهای رفتاری از ماشین حالت استخراج شده که با پیموده شدن این مسیرها، اهداف آزمون برآورده می‌شوند. از آن‌جایی‌که اجزای موجود در این مسیرهای انتزاعی شامل قیودی

---

[۱] Depth First Search

[۲] Forward Search

[۳] ParTeG (PARtition TEst Generator) یک ابزار آزمون مبتنی بر مدل می‌باشد که توسط Stephan Weiβleder طراحی و پیاده‌سازی شده است. این ابزار در آدرس http://parteg.sourceforge.net قابل دسترس می‌باشد.

می‌باشند که با OCL بیان شده است، نیاز است که این قیود، اجرایی شده و داده‌هایی برای اجرایی شدن و برآورده شدن آن‌ها ایجاد گردد.

۲.۳.۱ اجرایی کردن مدل ورودی و قیود موجود بر روی مدل

نمایش داخلی ایجاد شده از مدل ورودی در مرحله‌ی قبل، با استفاده از یک تبدیل مدل به متن، به یک نمایش رسمی ریاضی و قابل اجرا تبدیل می‌گردد. سپس سعی می‌شود که قیود موجود در مسئله حل شده و داده‌های آزمون تولید گردند. جهت دست‌یابی به این مهم، قیود موجود بر روی مسیرهای رفتاری، که توسط الگوریتم جست‌وجوی گراف تولید گردیدند، تبدیل به مدلی ریاضی در زبان برنامه‌ریزی ریاضی AMPL می‌شود. هر برنامه‌ی AMPL به دو بخش مدل و داده نیاز دارد. مدل AMPL با تبدیل ماشین حالت به AMPL و داده‌ی AMPL با تبدیل هر مسیر انتزاعی به AMPL تولید می‌شود. در هنگام ارائه‌ی الگوریتم‌های مربوط به این تبدیل‌ها، نیاز است که معنای مدل اصلی پس از تبدیل حفظ شود. پس از ایجاد مدل رسمی و ریاضی از سیستم، این مدل با استفاده از تعدادی از حل‌کننده‌های متعدد و به‌روز که توسط AMPL پشتیبانی می‌شوند، مانند CPLEX، Minos، Gecode[۱]، Jacop[۲]، Couenne[۳] و Cbc[۴]، حل شده و داده‌های آزمون تولید می‌گردند.

۳.۳.۱ تبدیل مدل آزمون

جهت افزایش کیفیت موارد آزمون تولیدی با استفاده از معیار پوشش تمامی انتقالات در آزمون مبتنی بر مدل، یک تبدیل مدل به مدل ارائه شده است. این بدان معنی است که تبدیل مدل[۵] می‌تواند به عنوان یک رویکرد برای افزایش کیفیت موارد آزمون تولیدی استفاده شود. در این پژوهش، شبه‌حالت[۶] تاریخچه‌ی عمیق[۷] و تاریخچه‌ی کم‌عمق[۸] از ماشین حالت UML، با هدف افزایش کیفیت موارد آزمون تولیدی، به شبه‌حالت انتخاب[۹] تبدیل شده و انتقالاتی به آن افزوده می‌شود. برای انجام این تبدیل از زبان ویزارد اپسیلون[۱۰] [۴] استفاده شده است و تبدیل مدل به گونه‌ای انجام شده است که در حین تغییر ساختار مدل اصلی، معنای آن در مدل تبدیل شده حفظ گردد. از طریق این تبدیل مدل، می‌توان هنگام تولید موارد آزمون، مسیرهای اجرایی بیشتری از سیستم تحت آزمون را مورد توجه قرار داده و از این طریق پوشش کد منبع سیستم تحت آزمون را افزایش داد. با افزایش پوشش کد منبع سیستم، قدرت کشف خطای موارد آزمون تولیدی افزایش یافته و می‌توان گفت که کیفیت آن‌ها نیز بهبود یافته است.

۴.۱ روش ارزیابی

الگوریتم‌های ارائه شده در این پژوهش، به صورت یک ابزار و در قالب افزونه‌های Eclipse، پیاده‌سازی شده‌اند. ابزار پیاده‌سازی شده و الگوریتم‌های مربوطه با استفاده از سه مسئله‌ی علمی مورد ارزیابی قرار گرفته‌اند. در

---

[۱] https://code.google.com/p/ampl/downloads/list?q=gecode
[۲] http://jacop.osolpro.com/
[۳] https://projects.coin-or.org/Couenne
[۴] https://projects.coin-or.org/Cbc
[۵] Model Transformation
[۶] Pseudo State
[۷] Deep History
[۸] Shallow History
[۹] Choice Pseudo State
[۱۰] Epsilon Wizard Language (EWL)

این ارزیابی‌ها، راه‌کار ارائه شده در این پژوهش از جنبه‌هایی مانند زمان تولید موارد آزمون، درصد برآورده‌سازی اهداف آزمون، کیفیت داده‌های آزمون تولید شده و قدرت تشخیص خطای موارد آزمون تولیدی، مورد سنجش قرار می‌گیرد. همچنین نشان داده می‌شود که قابلیت استفاده از حل‌کننده‌های مختلف، یکی از بزرگ‌ترین مزایای استفاده از راه‌کار ارائه شده جهت تولید بهینه‌ی آزمون واحد[1] می‌باشد.

۵.۱ ساختار پایان‌نامه

این پایان‌نامه به صورت ذیل سازمان‌دهی شده است. در فصل ۲، مفاهیم پایه‌ای مرتبط با راه‌کار ارائه شده در این پژوهش، شرح داده می‌شود. این فصل، به شرح مفاهیم مرتبط با آزمون مبتنی بر مدل، روش‌های مدل‌سازی مورد استفاده جهت توصیف سیستم و همچنین پیش‌نیازهای ریاضی جهت توصیف سیستم و استفاده از حل‌کننده‌ها جهت حل مسئله، می‌پردازد. در فصل ۳ فعالیت‌ها و تلاش‌های انجام گرفته در زمینه‌ی آزمون مبتنی بر مدل شرح داده شده و پیشینه‌ی آن مورد بررسی قرار می‌گیرد. در ادامه و در فصل ۴ به بیان راه‌کار ارائه شده در این پژوهش، به‌منظور تولید بهینه‌ی موارد آزمون، پرداخته می‌شود. راه‌کار پیشنهادی، در فصل ۵ از جنبه‌های مختلف، مورد ارزیابی و سنجش قرار می‌گیرد. در نهایت در فصل ۶، دست‌آوردهای حاصل از این پژوهش بیان شده و پیشنهادهایی جهت انجام فعالیت‌های آتی ارائه می‌گردد.

[1] Unit Test

فصل ۲

آزمون مبتنی بر مدل، روش‌های مدل‌سازی و حل قیود

۱.۲ مقدمه

در این فصل، مفاهیم پایه‌ای مرتبط با راه‌کار ارائه شده در این پژوهش، شرح داده می‌شود. ابتدا در بخش ۲.۲ مهندسی نرم‌افزار مدل‌رانده معرفی می‌شود و در بخش ۳.۲ در مورد زبان‌های مدل‌سازی توضیحاتی ارائه می‌گردد. سپس در بخش ۴.۲، در مورد آزمون نرم‌افزار، اهمیت آن و همچنین آزمون سنتی و خودکار نرم‌افزار توضیحاتی ارائه می‌گردد. در نهایت نیز در بخش‌های ۵.۲ و ۶.۲ به ترتیب مطالبی در مورد پیش‌نیازهای ریاضی و زبان مدل‌سازی ریاضی AMPL، ذکر می‌گردد.

۲.۲ مهندسی نرم‌افزار مدل‌رانده

امروزه با توجه به پیچیده‌تر شدن سیستم‌های کامپیوتری، استفاده از مدل به عنوان یک ضرورت به ویژه در مهندسی نرم‌افزار مطرح می‌شود. برخلاف برخی از فرآیندهای توسعه‌ی نرم‌افزار که محور اصلی آن‌ها کد می‌باشد، در مهندسی نرم‌افزار مدل‌رانده، مدل محصول اصلی می‌باشد که فرآیند توسعه‌ی نرم‌افزار را پیش می‌برد. هدف نهایی از مهندسی نرم‌افزار مدل‌رانده، تولید خودکار نرم‌افزار از روی مدل مربوطه می‌باشد [۱].

اساس روش‌های مدل‌رانده، تبدیلِ مدل است [۵]. تبدیلِ مدل، عامل متمایز کننده‌ی مهندسی نرم‌افزار مدل‌رانده از روش‌های سنتی است، که از مدل به عنوان طرح پیش‌نویس طراحی استفاده می‌کنند. تبدیل، یک نگاشت است که محصولی را به عنوان ورودی می‌پذیرد و محصول دیگری را به عنوان خروجی تولید می‌کند. تبدیل می‌تواند بصورت دستی یا خودکار انجام شود. در تبدیل دستی، توسعه‌دهنده وظیفه‌ی دریافت مدل ورودی و اِعمال تغییرات و انجام تبدیل را بر عهده دارد. در این روش، تضمین سازگاری محصول خروجی با مدل ورودی برعهده‌ی توسعه‌دهنده می‌باشد. در تبدیل خودکار از قوانین تبدیل برای اعمال تغییرات استفاده می‌شود؛ بنابراین، سازگاری محصول خروجی با مدل ورودی تضمین می‌شود. به عنوان مثال‌هایی از این قوانین می‌توان به نمایه[۱] و الگو[۲] اشاره نمود که اِعمال نمایه یا الگو روی یک مدل، نوعی تبدیل مدل محسوب می‌شود [۶].

تبدیل مدل به کد، یکی از انواع تبدیل می‌باشد که به آن تولید کد یا مهندسی رو به جلو نیز گفته می‌شود. تولید کد یکی از ویژگی‌هایی است که مهندسی نرم‌افزار مدل‌رانده را، از دیگر فرآیندهای توسعه‌ی نرم‌افزار متمایز می‌کند.

۳.۲ زبان‌های مدل‌سازی

مدل یک نمایش با سطح تجرید بالا از سیستم می‌باشد و یک نمونه از فرامدل یا زبان مدل‌سازی به حساب می‌آید. زبان‌های مدل‌سازی بسیاری وجود دارند که می‌توان از آن‌ها برای ایجاد مدل آزمون بهره برد. به عنوان مثال می‌توان به ماشین حالت مجرد[۳]، زبان مدل‌سازی یکنواخت (UML)، زبان قید شئ (OCL) و Object-Z اشاره نمود. در این پژوهش از زبان مدل‌سازی یکنواخت و زبان قید شئ، استفاده شده است. در ادامه در بخش‌های ۱.۳.۲ و ۲.۳.۲، به ترتیب به توصیف زبان مدل‌سازی یکنواخت و زبان قید شئ، خواهیم پرداخت.

۱.۳.۲ زبان مدل‌سازی یکنواخت (UML)

UML برای اولین بار در سال ۱۹۹۸ توسط بوچ[۴]، رامبو[۵] و جی‌کابسون[۶] ارائه شد [۷]. UML برپایه‌ی امکان فرا شئ[۷] [۸] بنا نهاده شده است، که یک زبان خاصِ طراحی شده برای تعریف زبان‌های مدل‌سازی مبتنی بر شئ می‌باشد. در زبان مدل‌سازی یکنواخت، چندین نمودار با نشانه‌گذاری گرافیکی برای توصیف ویژگی‌های ساختاری و رفتاری سیستم تعریف شده است. به عنوان نمونه می‌توان به نمودار کلاس[۸] و ماشین حالت[۹] به ترتیب برای توصیف ویژگی‌های ساختاری و رفتاری یک سیستم، اشاره نمود. هر نمودار UML دارای یک فرامدل[۱۰] می‌باشد. فرامدل مشخص می‌کند که هر مدل، شامل چه اجزایی می‌باشد و این اجزا به چه صورت با هم در ارتباط هستند. بنابراین هر مدل باید منطبق بر فرامدل مربوط به خود باشد.

در این پژوهش، جهت ایجاد مدل آزمون، از نمودار کلاس و ماشین حالت UML استفاده شده است که در ادامه به اختصار معرفی می‌گردند. ابزارهای زیادی برپایه‌ی UML وجود دارند که برای ایجاد و ویرایش نمودارها، مورد استفاده قرار می‌گیرند. در این پژوهش از ابزار مدل‌سازی Papyrus جهت ایجاد مدل آزمون استفاده شده است.

---

[۱] Profile
[۲] Pattern
[۳] Abstract State Machine
[۴] Booch
[۵] Rumbaugh
[۶] Jacobson
[۷] Meta Object Facility (MOF)
[۸] Class Diagram
[۹] State Machine
[۱۰] Meta Model