**ارائه یک رویکرد جدید جهت اولویت بندی موارد آزمون مبتنی بر الگوریتم جستجوی گرانشی در برنامه­ها و بازی­های رایانه­ای**

**محمد شمس­الدینی1\***

1. **کارشناس ارشد نرم­افزار**

Email: shamsadini\_mohammad@yahoo.com

چکیده

 امروزه محیط کسب و کار در حال تغییر می­باشد و بازار یک عامل کلیدی برای دستیابی به موفقیت است. برای موفقیت بیشتر، کیفیت باید حداکثر شود و حداقل هزینه و زمان تحویل کوتاه باشد. به طور کلی آزمون، یکی از روش‌های اصلی تضمین کیفیت بازی و نرم‌افزار در صنعت بوده و هدف کلی صنعت نرم‌افزار و بازی، اطمینان ارائه نرم‌افزار و بازی با کیفیت بالا به کاربر می­باشد. برای اطمینان از کیفیت بالای بازی کامپیوتری، لازم است که نرم‌افزار آزمایش شود. آزمایش تضمین می‌کند که نرم‌افزار مطابق با مشخصات و الزامات کاربر باشد. در حالت ایده آل، تمام تنظیمات یک سیستم نرم‌افزاری باید مورد آزمایش قرار گیرند. ازاین‌رو در این پژوهش یک روش نوین و کارآمد جهت اولویت‌بندی موارد آزمون با استفاده از الگوریتم جستجوی گرانشی (GSA) ارائه می‌شود که سناریوی آزمون پیشنهادی شامل دو مرحله‌ی اصلی؛ یافتن دنباله‌ای از آزمون بهینه که مبتنی بر الگوریتم جستجوی گرانشی می‌باشد؛ که الگوریتم جستجوی گرانشی بر اساس قوانین موجود بین اجرام استفاده می‌کند و نیز شامل مرحله‌ی انتخاب مجموعه توالی از راه‌حل‌ها می‌باشد. در این پژوهش جهت روش­های ارزیابی از سه الگوریتم جستجوی گرانشی، تجمع ازدحام ذرات و ژنتیک استفاده شده است و برای ارزیابی روش پیشنهادی از سه معیار زمان، تعداد خطاهای تحت پوشش و تعداد آزمون­های بکار رفته در پوشش خطاها استفاده می­گردد. در سناریوی اول با افزایش تعداد تکرار، زمان رفع خطا کاهش پیدا می­کند و روش پیشنهادی و الگوریتم ژنتیک بدلیل پوشش مناسب فضای جستجو توانسته است با تکرار کم به جواب بهینه دست یابد. در سناریوی دوم با افزایش تعداد تکرار، زمان رفع خطا کاهش پیدا می­کند و الگوریتم تجمع ازدحام ذرات در تکرار پایین بدلیل اینکه جمعیت به سمت بهینگی عمومی حرکت می­کنند و هنوز بهینگی عمومی در موقعیت مناسب قرار نگرفته است، از کارایی پایین­تری برخوردار است.

**کلمات کليدي: الگوریتم جستجوی گرانشی (GSA)، اولویت‌بندی موارد آزمون، آزمون نرم‌افزار**

**1- مقدمه**

سیستم‌های نرم‌افزاری در طول عمر خود تغییر زیادی می‌کنند و اغلب چنین تغییراتی می‌تواند بر سیستم نرم‌افزاری تأثیر بگذارد. برای جلوگیری از تغییرات نامطلوب یا اشکالات غیرمنتظره، مهندسان نرم‌افزار قبل از استقرار یک نسخه جدید محصول، باید آزمون کلی عملکرد سیستم را تست کنند. آزمون نرم‌افزار فرایند ضروری اما گران است. یکی از روش‌های رایج برای ارزیابی کیفیت سیستم در یک توالی آزاد، آزمون رگرسیون است. در آزمون رگرسیون، مهندسان نرم‌افزار سیستم نرم‌افزاری را تأیید می‌کنند تا اطمینان حاصل شود که تغییرات جدید گسل‌های (خطا) جدیدی را نشان نمی‌دهند. با این ‌حال، سیستم‌های نرم‌افزاری مدرن به‌ طور مداوم تکامل پیدا می‌کنند و اندازه و پیچیدگی آن‌ها به ‌سرعت رشد می‌کند و بنابراین هزینه آزمون رگرسیون می‌تواند خیلی گران باشد [1].

اولویت‌بندی مورد آزمون (TCP) با هدف ترتیب موارد آزمون انتزاعی، به‌منظور افزایش سرعت، تشخیص گسل‌ها و تشخیص خطا را افزایش می‌دهد. نشان داده‌ شده است که تکنیک‌های اولویت‌بندی موارد آزمون برای بهبود فعالیت‌های آزمون رگرسیون مفید است. با اولویت‌بندی، میزان تشخیص خطا بهبود یافته است؛ بنابراین آزمون‌ها اجازه می‌دهند تا قبل از مرحله آزمایش سیستم، گسل‌ها را شناسایی کنند [2].

با توجه به مطالبی که بیان شد، مي‌توان گفت که با مسائل زير روبرو هستیم:

مسئله‌ی اول؛ یافتن دنباله‌ای مناسب و بهینه از آزمون‌ها با استفاده از الگوریتم جستجوی گرانشی (GSA) که تمام خطاها در حداقل زمان اجرایی پوشش داده می‌شوند.

مسئله‌ی دوم؛ استفاده از الگوریتم جستجوی گرانشی (GSA) باعث کاهش همپوشانی خطاها، تعداد آزمون‌ها و زمان اجرا می‌شود.

اهدافی که در اینجا به دنبال آن هستیم به شرح زیر می‌باشد:

* کاهش زمان اجرا
* کاهش تعداد آزمون‌ها
* کاهش تعداد آزمون‌های همپوشان

آزمون نرم‌افزار یکی از تکنیک‌های اصلی برای دستیابی به نرم‌افزار با کیفیت بالا است. اولویت‌بندی موارد آزمون (TCP) تلاش می‌کند تا اثربخشی شناسایی خطا را با برنامه‌ریزی موارد مهم آزمون که قبلاً اجرا می‌شود، جایی که اهمیت با برخی از معیارها یا استراتژی‌ها تعیین می‌شود را بهبود بخشد [3]. آزمون نرم‌افزار برای شناسایی حضور گسل‌ها و رفع نواقص (ناهنجاری‌ها) قبل از مرحله اجرا؛ که باعث خرابی نرم‌افزار می‌شوند، اجرا می‌گردد. بااین‌حال، آزمون نرم‌افزار یک کار وقت‌گیر و گران‌قیمت است که تقریباً 50 درصد از منابع توسعه سیستم نرم‌افزاری را مصرف می‌کند. مرحله آزمایشی شامل پیدا کردن اشکالات و حذف نقص در صورت امکان است. آزمون‌های مختلفی وجود دارد که آزمایش‌کنندگان نرم‌افزار بر اساس الزامات آن‌ها مانند جهش، رگرسيون، امنيت، آزمايش بار و غيره انجام می‌دهند. آزمون می‌تواند به‌صورت دستی یا به‌صورت خودکار با استفاده از ابزار آزمون انجام شود که بر اساس یافته‌های انجام‌شده ثابت‌شده است که آزمون نرم‌افزار خودکار بهتر از آزمایش دستی است. بااین‌حال، ابزارهای کمی برای آزمون داده‌های آزمایشی، امروزه در دسترس می‌باشند. ازاین‌رو آزمون نرم‌افزار و اولویت‌بندی موارد آزمون جهت اطمینان از کیفیت نرم‌افزار و رفع خطاها و نواقص اهمیت و ضرورت بسیار بالایی دارد [5].

**2- روش انجام تحقیق**

در این پژوهش هدف یافتن مجموعه توالی بهینه از آزمون‌ها می‌باشد که باید برای سه شرط زیر برقرار باشد

* پوشش دهی تمام خطاهای موجود
* برطرف کردن خطا‌ها در کمترین زمان ممکن
* بهره‌گیری از کمترین آزمون

در ابتدا باید نگاهی به سناریو‌های موجود انداخت. جزء اصلی هر سناریو آزمون‌ها می‌باشند. هر آزمون از دو جز مهم تشکیل‌شده است که عبارت‌اند از:

* تعداد خطاهای تحت پوشش
* زمان رفع خطاها بر اساس دقیقه

در این پژوهش توالی آزمون‌هایی انتخاب می‌گردد که علاوه بر کاهش آزمون‌ها و همپوشانی آزمون‌ها، زمان اجرا نیز کاهش یابد. جهت یافتن توالی مورد نظر از الگوریتم جستجوی گرانشی استفاده می‌گردد. الگوریتم جستجوگرانشی بر اساس قوانین موجود بین اجرام استفاده می‌کند. از همین رو یک سیستم از اجرام را در نظر بگیرید که در یک فضا قرار دارند؛ که در این فضا هر جرم یک موقعیت مکانی دارد. برای محاسبه­ی جرم، ابتدا بر اساس تابع برازندگی، برازندگی هر جرم محاسبه می­گردد و برای محاسبه­ی شتاب یک جرم، ابتدا باید جمع نیرو­های اجرامی که بر این جرم نیرو وارد می­کنند را محاسبه نمود. بعد از محاسبه­ی نیرو­های وارده بر جرم موردنظر، شتاب جرم محاسبه می­گردد. بعد از محاسبه­ی شتاب جرم، سرعت حرکت جرم بر اساس ضریبی از سرعت قبلی به‌اضافه‌ی شتاب جرم به دست می­آید. درنهایت موقعیت جدید جرم بر این اساس محاسبه می­گردد.

3- پیشینه تحقیق

چن و همکارانش در سال 2018 رویکردی جهت اولویت‌بندی موارد آزمون بر اساس خوشه‌بندی مطابق شکل (1) پیشنهاد دادند [3]. در این پژوهش، بهبود اثربخشی آزمون نرم‌افزار شیءگرا در آزمون رگرسیون (ARS)، مبتنی بر تکنیک‌های خوشه‌بندی با استفاده از اطلاعات جعبه سیاه ارائه می‌شود. روش‌های خوشه‌بندی استفاده‌شده در این پژوهش؛ نمونه‌های خوشه‌بندی با توجه به تعداد اشیاء، با استفاده از الگوریتم‌های خوشه‌بندی K-means و K-medoids و با استفاده از الگوریتم خوشه‌بندی K-medoids، بر اساس یک روش شباهت دنباله‌ی فراخوانی شیء و روش خوشه‌بندی‌شده است. مطالعات تجربی و نتایج نشان می‌دهد که هر دو احتمال افزایش تشخیص زودهنگام و اثربخشی بالاتر از روش اولویت‌بندی تصادفی را دارد و تکنیک TCP پوشش روش را نشان می‌دهند.

|  |  |
| --- | --- |
|  |  |

**شکل (1). تصاویری از خوشه‌بندی**

سلطانی و همکارانش در سال 2017 به بررسی تحلیل تکنیک‌های اولویت‌بندی موارد آزمون پرداخته‌اند [4]. در این راستا یک تحلیل جامع از تکنیک‌های مختلف رگرسیون ارائه‌شده است؛ که عمدتاً بر اولویت‌بندی موارد آزمون تمرکز دارد. چندین تکنیک اولویت‌بندی در جزئیات موردبررسی قرار می‌گیرند. بسیاری از تکنیک‌ها را موردبررسی و مقایسه قرار می‌گیرند که برای اولویت‌بندی موارد آزمون مورداستفاده قرار می‌گیرند. هر تکنیک دارای مزایا و معایب خاص خود است. تمام تکنیک‌ها برای توضیح و نتیجه‌گیری تلاش می‌شود تا آزمایش‌کننده بتواند از هر تکنیک با توجه به نیازها و نیازهایش استفاده کند. تشخیص کیفیت نرم‌افزار و تشخیص خطا به‌منظور اثربخشی و کارایی بیشتر می‌تواند از طریق آزمون رگرسیون افزایش یابد.

بهویسی و همکارانش در سال 2017 رویکردی جهت بهبود عملکرد تکنیک‌های مبتنی بر پوشش جهت اولویت‌بندی موارد آزمون پیشنهاد دادند [5]. روش پیشنهادی مطابق شکل (2) بر اساس تکنیک‌های پوشش با استفاده از ساختار وابستگی بسته می‌باشد. موارد آزمون در خوشه‌ها بر اساس عملکرد، دسته‌بندی می‌شوند. ساختار وابستگیِ بسته، به‌منظور تسهیل آزمون مورد در هر خوشه اجرا می‌شود و اولویت‌بندی مورد آزمون از اطلاعات مربوط به پوشش تابع انجام می‌گیرد. نتایج نشان می‌دهد که روش پیشنهادشده در مقایسه با روش پوشش کد و برخی از تکنیک‌های پوشش در اولویت‌بندی موارد آزمون بهتر عمل می‌کند. علاوه بر این، روش فعلی، 97 درصد از درصد متوسط تشخیص خطا (APFD) و تعیین ضریب در همبستگی برابر با 8363/0 را به خود اختصاص می‌دهد که باعث افزایش عملکرد تکنیک‌های مبتنی بر پوشش می‌باشد.



**شکل (2). چارچوب**

میراندا و همکارانش در سال 2017 [6] پیشنهاد رویکردهای خاصی (scope-aided) برای اولویت‌بندی، انتخاب و به حداقل رساندن آزمون‌های جعبه سفید را داده‌اند؛ که با استفاده از محدودیت‌های احتمالی محدوده دامنه ورودی جدید، با استفاده از تغییر مجدد یا انتخاب موارد آزمون، موردتوجه قرار می‌گیرد. رویکرد scope-aided مستلزم استفاده از اطلاعات یا شهود در مورد نحوه استفاده از نرم‌افزار در هر زمینه یا دامنه جدید جهت هدایت تمرکز اولویت‌بندی، انتخاب و به حداقل رساندن آزمون‌های موجود است و به‌این‌ترتیب آن‌ها برای آزمون مجدد بیشتر مقرون‌به‌صرفه می‌باشند. شناسایی محدودیت‌های استفاده مجدد اولین گام اساسی است و تنها این است که قطعاً نیاز به مداخله‌ی انسانی دارد. پس از محدودیت محدوده شده، رویکرد می‌تواند به‌طور کامل خودکار باشد و در این مقاله از تکنیک‌های اجرای نمادین پویا استفاده می‌شود. نتایج حاصل از ارزیابی تجربی امیدوارکننده می‌باشند؛ اگرچه در هر مطالعه‌ای به‌طور مداوم پیروی نمی‌کند، اما به‌طور خاص در مواردی که فقط تعداد محدودی از موارد آزمون را می‌توان اجرا کرد. به‌عنوان‌مثال، در مواردی که تنها 25٪ از موارد آزمون اجرا می‌شود، برای مثال اولویت‌بندی با کمک محدوده درگشت‌های محدوده یافت شده سریع‌تر از هر روش دیگر است.

تاهات و همکارانش در سال 2017 مدلی بر اساس اولویت‌بندی مجموعه آزمون رگرسیون پیشنهاد دادند [7]. در این پژوهش؛ مدل پیشنهادی بر مبنای یک کلاس تغییرات که به‌طور اصلاح‌نشده می‌باشد؛ ارائه می‌شود (فقط کد منبع اصلاح می‌شود). پس از شناسایی عناصر، مدل مربوط به کد منبع اصلاح‌شده و اطلاعات در طی اجرای مدل جمع‌آوری می‌شود. در این پژوهش، روش اولویت آزمون مبتنی بر مدل ارائه‌شده است که در آن اطلاعات مربوط به مدل سیستم و رفتار آن برای تعیین اولویت‌بندی مجموعه آزمون جهت آزمایش مجدد سیستم استفاده می‌شود. نتایج مطالعات تجربی نشان می‌دهد که تمام روش‌های ارائه‌شده بر اساس مدل اولویت‌بندی آزمون، اثربخشی تشخیص زودهنگام اولیه را در مقایسه با مرتب‌سازی تصادفی بهبود می‌بخشد. هنگام مقایسه با یکدیگر، برخی از روش‌ها به‌طورکلی بهتر از روش‌های دیگر انجام می‌شود.

حجی و همکارانش در سال 2016 یک ابزار نسخه‌ی نمایشی جهت آزمون سیستم‌های قابل تنظیم با ویژگی IDE ارائه دادند [8]. در این پژوهش با استفاده از گسترش ویژگی IDE، هدف این است که تکنیک‌های آزمون جدیدی را برای سیستم‌های قابل تنظیم اعمال شود. درون ویژگی IDE، چندین استراتژی برای بهینه‌سازی فرایند آزمون ادغام می‌شوند. استراتژی‌های نمونه‌گیری چندگانه به‌منظور کاهش مجموعه تنظیمات آزمایش‌شده به‌طور قابل‌قبول، مانند نمونه‌برداری T-wise، تنظیمات تصادفی و تنظیمات کاربر تعریف‌شده است. بااین‌حال، این استراتژی‌ها اغلب در عمل اعمال نمی‌شوند و آن‌ها نیازمند تلاش‌های دستی یا چارچوب آزمایش تخصصی می‌باشند. در ابزار ویژگی IDE، تمام استراتژی‌های ذکرشده یکپارچه می‌شوند و تلاش‌های دستی با خودکار سازی روند تولید و آزمایش تنظیمات کاهش داده می‌شود. علاوه بر این، نیز برای پشتیبانی از آزمون واحد جهت جلوگیری از اجراهای آزمایش بارگیری شده و آزمون آگاهانه متغیر، ارائه می‌شود. به‌طورکلی با این استراتژی‌ها و الگوریتم‌های یکپارچه، هدف این است که آزمون‌های سیستم‌های قابل تنظیم را بدون در نظر گرفتن اندازه و ماهیت این سیستم‌ها تسهیل دهیم.

نگوچی و همکارانش در سال 2015 اولویت‌بندی موارد آزمون را برای آزمایش جعبه سیاه بر روی محصولات جدید با استفاده از ACO پیشنهاد داده‌اند [9]. چارچوب پیشنهادی شامل مراحل زیر می‌باشد:

• آزمایش‌کنندگان موارد آزمون یک محصول پیشین و هدف را به دسته‌های آزمون طبقه‌بندی می‌کنند.

• آزمایش‌کنندگان داده‌های عملکرد تاریخی را برای هر دسته جمع‌آوری می‌کنند.

• آزمایش‌کنندگان لیستی از محدودیت‌های قضیه بین دسته‌ها را ایجاد می‌کنند.

• چارچوب پیشنهادی یک سفارش طبقه‌بندی اولویت‌بندی را برای محصول قبلی با استفاده از ACO تولید می‌کند.

• چارچوب پیشنهادی سپس لیست دسته‌بندی اولویت‌بندی شده را برای محصول هدف با مقایسه لیست تولیدشده در مرحله 4 و لیست محصول هدف ایجاد می‌کند.

شارما و همکارانش در سال 2013 تکنیک‌های آزمون نرم‌افزار با استفاده از الگوریتم ژنتیک موردبررسی قرار داده‌اند [10]. در این مقاله، یک نظرسنجی با الگوریتم ژنتیک (GA) جهت رسیدگی به مسائل مختلف در حین آزمایش نرم‌افزار ارائه می‌شود. در این مقاله، کاربردهای GA در انواع مختلف آزمون نرم‌افزار موردبحث قرار می‌گیرد. GA نیز با فازی و همچنین با شبکه‌های عصبی در انواع آزمون استفاده می‌شود و نیز همچنین برنامه‌های GA در زمینه‌های مختلف آزمون مانند برنامه‌ریزی آزمون، حداقل موارد آزمون رگرسیون، آزمون مبتنی بر مدل و آزمون وب، بررسی خواهند شد. یافته شده است که با استفاده از GA، نتایج و عملکرد آزمون می‌تواند بهبود یابد.

**4- روش پیشنهادی**

در این پژوهش یک روش کارآمد جهت اولویت‌بندی موارد آزمون با استفاده از الگوریتم جستجوی گرانشی (GSA) ارائه می‌شود که سناریوی آزمون پیشنهادی شامل دو مرحله‌ی اصلی؛ یافتن دنباله‌ای از آزمون بهینه که مبتنی بر الگوریتم جستجوی گرانشی می‌باشد؛ که الگوریتم جستجوگرانشی بر اساس قوانین موجود بین اجرام استفاده می‌کند و نیز شامل مرحله‌ی انتخاب مجموعه توالی از راه‌حل‌ها می‌باشد.

در شکل (3) فلوچارت روش پیشنهادی آورده شده است.

سناریوهای آزمون

یافتن توالی بهینه مبتنی بر الگوریتم جستجوی گرانشی

موقعیت اولیه اجرام

محاسبه‌ی برازندگی اجرام

بروز رسانی نیروی گرانشی و جرم اجرام

محاسبه‌ی سرعت و شتاب

انتخاب بهترین اجرام

مجموعه توالی بهینه از راه‌حل‌ها

**شکل (3). فلوچارت روش پیشنهادی**

همانطور که در شکل (3) دیده می­شود، ابتدا سناریوهای آزمون قرار دارد. این سناریوها شامل آزمون­ها به همراه زمان رفع آنها می­باشد. در ادامه در جدول (1) نمونه­ای از این سناریو برای هشت آزمون و ده خطا آورده شده است.

همانطور که در جدول (1) نشان داده شده است، هر آزمون برخی از خطاها را تحت پوشش قرار داده است. برای مثال آزمون T1 خطاهای F2,F4,F7,F9 را تحت پوشش قرار داده است.

**جدول (1). نمونه­ای از سناریو­های آزمون و خطا**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  **F10** |  **F9** |  **F8** |  **F7** |  **F6** |  **F5** |  **F4** |  **F3** |  **F2** |  **F1** | **موارد خطا** |
| **موارد آزمون** |
|  0 |  1 |  0 |  1 |  0 |  0 |  1 |  0 |  1 |  0 |  T1 |
|  0 |  0 |  0 |  0 |  0 |  0 |  0 |  1 |  0 |  1 |  T2 |
|  0 |  0 |  1 |  1 |  0 |  1 |  0 |  0 |  0 |  1 |  T3 |
|  0 |  1 |  0 |  0 |  0 |  0 |  1 |  0 |  1 |  0 |  T4 |
|  1 |  0 |  0 |  0 |  1 |  0 |  0 |  1 |  0 |  0 |  T5 |
|  0 |  0 |  0 |  1 |  0 |  0 |  0 |  0 |  0 |  1 |  T6 |
|  0 |  0 |  1 |  0 |  1 |  0 |  0 |  1 |  0 |  0 |  T7 |
|  1 |  0 |  0 |  0 |  0 |  0 |  0 |  0 |  1 |  0 |  T8 |

5- یافتن توالی بهینه مبتنی بر الگوریتم جستجوی گرانشی

الگوریتم جستجوی گرانشی جز الگوریتم­های تکاملی می­باشد که از طبیعت الهام گرفته است. این الگوریتم از قوانین موجود بین سیارات بهره گرفته است که برای اولین بار توسط آقای میرجلیلی برای حل مسائل بهینه­سازی استفاده گردیده است. در ادامه هریک از بخش­های الگوریتم شرح داده می­شود.

5-1- موقعیت اولیه اجرام

از آنجایی که در این پژوهش قصد داریم توالی بهینه از آزمون­ها را بیابیم، بنابراین جمعیت اولیه باید نگاشتی از آزمون­ها باشد. جهت بدست آوردن جمعیت اولیه از رابطه­ی (1) استفاده می­گردد.

$X\_{i}=\left(x\_{1},x\_{2},…,x\_{n}\right) n=1:N$ (1)

در رابطه­ی بالا N برابر تعداد آزمون­ها می­باشد.

مقادیر جمعیت اولیه به صورت تصادفی در بازه صفر و یک ایجاد می­گردد. جمعیت اولیه برای جدول (1) یک بردار یک در هشت می­باشد که در جدول (2) نشان داده شده است.

**جدول (2). نگاشت جمعیت اولیه برای هشت آزمون**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0.92 | 0.88 | 0.47 | 0.24 | 0.76 | 0.125 | 0.83 | 0.35 |

از آنجایی که در جدول (1) هشت آزمون وجود دارد، جمعیت اولیه تولید­شده در جدول (2) نیز هشت در نظر گرفته شده است که مقادیر هر خانه توسط یک عدد تصادفی بین صفر و یک تکمیل شده است. بعد از ایجاد جمعیت اولیه باید بر روی هر جمعیت عملیات باینری­سازی صورت گردید. جهت باینری کردن جمعیت اولیه از روش آستانه­گذاری استفاده
می­گردد. در این روش از یک حد آستانه برای باینری کردن استفاده می­شود. در پژوهش مورد نظر از حد آستانه­ی 5/0 بهره گرفته شده است. به عبارتی مقادیری از جمعیت که مقدار آنها بزرگتر یا مساوی حد آستانه باشند برابر یک و مقادیری از جمعیت که مقادیر آن از حد آستانه کوچکتر باشند مقدار صفر تعلق می­گیرد. برای جدول (2) عملیات باینری­سازی با حد آستانه 5/0 به صورت زیر می­باشد.

**جدول (3). عملیات باینری­سازی با حد آستانه 5/0**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 |

همانطور که در جدول (3) قابل مشاهده می­باشد، مقادیر به صفر و یک تبدیل گشته­اند که مقدار صفر بیانگر عدم استفاده از آزمون مورد نظر و یک بیانگر استفاده از آزمون مورد نظر می­باشد. براساس جدول (3) آزمون­های T2,T4,T7,T8 انتخاب شده و آزمون­های T1,T3,T5,T6 انتخاب نگردیده­اند. بعد از عملیات باینری­سازی باید برای هر جمعیت برازندگی آن را محاسبه نمود که در ادامه نحوه­ی محاسبه­ی برازندگی جمعیت اولیه آورده شده است.

5-2- محاسبه برازندگی جمعیت اولیه

برازندگی هر جمعیت نشان دهنده­ی میزان شایستگی یک جمعیت نسبت به جمعیت دیگر می­باشد. برازندگی می­تواند در دو حالت کمینه­سازی و بیشنه­سازی مورد استفاده قرار بگیرد. در حالت بیشینه­سازی جمعیتی بهتر است که برازندگی آن بیشتر باشد ولی در مسائل کمینه­سازی جمعیتی بهتر است که میزان برازندگی آن کمتر باشد. در این پژوهش از روش کمینه­سازی استفاده گردیده است. تابع برازندگی در این پژوهش میزان زمان صرف شده برای پوشش خطاها (رفع خطاها) و تعداد
آزمون­های بکار رفته می­باشد که باید کمینه گردد. جهت محاسبه­ی برازندگی از رابطه­ی (2) بهره گرفته می­شود.

$Fitness=\sum\_{i=1}^{T\\_select}Time\_{i}+\left(N\_{Fault}-N\_{Fault-select}\right)\*penalty+\left(\frac{T\_{select}}{T\_{Totla}}\right)\*penalty$ *(2)*

در رابطه­ی بالا $T\\_select$ برابر تعداد آزمون­های انتخاب شده، $Time\_{i}$برابر زمان رفع خطای آزمون iام، $N\_{Fault}$ برابر کل خطاها، $N\_{Fault-select}$ برابر خطاهای تحت پوشش، $penalty$ میزان جریمه و در نهایت $T\_{Totla}$ برابر تعداد کل آزمون­ها می­باشد.

براساس جدول (3)که عملیات باینری­سازی روی جمعیت اولیه صورت گرفته است، مقادیر زیر بدست می­آید که در جدول (4) بیان گردیده است.

**جدول (4). مقادیر بدست آمده بعد از عملیات باینری­سازی**

|  |  |  |  |
| --- | --- | --- | --- |
| زمان | تعداد خطاها | خطاهای تحت پوشش | آزمون |
| 14 | 9 | F1,F2,F3,F6,F4,F8,F9,F10 | T2,T4,T7,T8 |

با فرض اینکه مقدار penalty برابر ده باشد، مقدار برازندگی برای جمعیت اولیه­ی مورد نظر به صورت رابطه (4) محاسبه می­گردد.

$Fitness=14+\left(10-9\right)\*10+\left(\frac{4}{8}\right)\*10=29$ *(3)*

بعد از محاسبه­ی برازندگی جمعیت اولیه، عملیات بروزرسانی نیروی گرانشی و جرم اجرام صورت می­گیرد. بعد از بدست آوردن نیروی گرانشی و جرم اجرام، سرعت و شتاب هر جمعیت محاسبه می­گردد. بعد از محاسبه­ی سرعت و شتاب، بار دیگر عملیات انتخاب جمعیت بهتر (جمعیتی که برازندگی کمتری داشته باشد) انتخاب می­شود.

**6- ارزیابی و نتایج روش پیشنهادی**

آزمایش‌ها در سه سناریو مختلف دسته‌بندی می‌شود. سپس بعد از انجام آزمایش‌ها، جهت ارزیابی سیستم پیشنهادی از سه معیار زمان، تعداد خطاهای تحت پوشش و تعداد آزمون­های بکار رفته در پوشش خطاها استفاده می­گردد.

6-1- محیط شبیه­سازی

جهت اجرا و شبیه­سازی روش پیشنهادی از سیستمی با مشخصات زیر استفاده می­گردد.

Operation System= windows 10 pro 64-bit

RAM= 8G

CPU= Core i7 8400

Matlab Version= 2016b

6-2- طراحی آزمایشات

جهت ارزیابی روش پیشنهادی آزمایشاتی صورت گرفته است که در شکل (4) آورده شده است.

سناریو آزمون

سناریو اول

سناریو دوم

سناریو سوم

روش های ارزیابی

ژنتیک

تجمع ازدحام ذرات

الگوریتم جستجوی گرانشی

معیار­های ارزیابی

زمان

تعداد خطاهای تحت پوشش

تعداد آزمون

**شکل (4). فلوچارت ارزیابی آزمایشات**

6-3- سناریو آزمون

جهت ارزیابی روش پیشنهادی از سه سناریو مختلف استفاده گردیده است که مشخصات این سه سناریو در زیر آورده شده است.

* سناریو اول: در این سناریو تعداد آزمون­ها ده عدد، تعداد خطاها ده عدد و مجموع زمان­ها برابر 87 می­باشد.
* سناریو دوم: در این سناریو تعداد آزمون­ها 100 عدد، تعداد خطاها 50 عدد و مجموع زمان­ها برابر 1020 می­باشد.
* سناریو سوم: در این سناریو تعداد آزمون­ها 1000 عدد، تعداد خطاها 400 عدد و مجموع زمان­ها برابر 10510 می­باشد.

6-4- روش­های ارزیابی

جهت ارزیابی روش پیشنهادی از دو روش تکاملی ژنتیک و تجمع ازدحام ذرات استفاده گردیده است که در ادامه مشخصات این روش­ها آورده شده است.

* الگوریتم ژنتیک: الگوریتم ژنتیک براساس عملگر­های ترکیب و جهش اقدام به بهبود جمعیت اول می­کنند.
* الگوریتم تجمع ازدحام ذرات: الگوریتم تجمع ازدحام ذرات براساس بهینگی محلی و سراسری اقدام به بهبود جمعیت اولیه می­کنند.
* الگوریتم جستجوی گرانشی(روش پیشنهادی) : الگوریتم جستجوی گرانشی براساس براساس قوانین موجود بین سیارات از قبیل جرم اجرام، شتاب و نیروی گرانشی اقدام به بهبود جمعیت اولیه می­کنند.

6-5- معیارهای ارزیابی

جهت ارزیابی روش پیشنهادی از سه معیار زمان، تعداد خطاهای تحت پوشش و تعداد آزمون­های بکار رفته در پوشش خطاها استفاده می­گردد.

* زمان: جهت ارزیابی معیار زمان باید به سراغ آزمون­های انتخابی رفت، زیرا هر آزمون براساس تعداد خطاهای قابل پوشش، زمانی برای رفع آنها سپری می­کند. در نهایت مجموع زمان­های انتخابی به عنوان نتیجه این معیار در نظر گرفته می­شود. این معیار باید کمینه گردد.
* تعداد خطای تحت پوشش: منظور از تعداد خطاهای تحت پوشش ، تعداد خطاهایی است که توسط آزمون­های انتخابی برطرف می­گردند. این معیار باید بیشینه گردد.
* تعداد آزمون­ها: هدف الگوریتم­ها یافتن دنباله­ای مناسب از آزمون­هاست که علاوه بر پوشش دادن تمامی خطاها، زمان نیز کاهش یابد. برای کاهش یافتن زمان، تعداد آزمون­ها بیاد کمینه گردد.

**10- مراجع**

1. Azizi, M. and H. Do, A Collaborative Filtering Recommender System for Test Case Prioritization in Web Applications. Computer Science > Software Engineering, 2018: p. 1.

2. Srikanth, H., L. Williams, and J. Osborne, System Test Case Prioritization of New and Regression Test Cases International Symposium on Empirical Software Engineering, 17-18 Nov. 2005 p. 1.

3.. Chen, J., et al., Test case prioritization for object-oriented software: An adaptive random sequence approach based on clustering The Journal of Systems and Software, January 2018. 135: p. 107.

4. Sultan, Z., et al., Analytical Review on Test Cases Prioritization Techniques: An Empirical Study. International Journal of Advanced Computer Science and Applications 2017. 8(2): p. 293.

5. Bhavyasri, G., P.A. AnandaRao, and D.P.R. Raju, Enhancing the Performance of Coverage-Based Techniques in Test Case Prioritization. International Journal of Scientific Research in Computer Science, Engineering and Information Technology, 2017. 2(5): p. 468-473.

6. Miranda, B. and A. Bertolino, Scope-aided Test Prioritization, Selection and Minimization for Software Reuse. Journal of systems and software, September 2017. 131: p. 528.

7. Tahat, L., et al., State-based models in regression test suite prioritization. Software Quality Journal, September 2017. 25(3): p. 703.

8. Al-Hajjaji, M., et al., Tool demo: testing configurable systems with FeatureIDE. International Conference on Generative Programming: Concepts and Experiences, March 2017. 52(3): p. 173.

9. Noguchi, T., et al., History-Based Test Case Prioritization for Black Box Testing on a New Product using Ant Colony Optimization. Software Testing, Verification and Validation (ICST), 2015 IEEE 8th International Conference on, 07 May 2015: p. 1.

10. Sharma, C., S. Sabharwal, and R. Sibal, A Survey on Software Testing Techniques using Genetic Algorithm. International Journal of Computer Science, January 2013. 10(1): p. 381-393.