ISSN: 2745-3673
http://www.Rcsj.ir

مجله‌ی علمی پژوهش در علوم رایانه (سال چهارم)
شماره ۱۴ / تابستان ۱۳۹۸ / ص ۱۰-۱

تولید آرایه پوشش بهینه با ترکیب الگوریتم‌های توده ذرات و تبرید شبیه‌سازی شده

سجاد اسفندیاری[۱]، وحید رافع[۲]، محمود فرخیان[۳]

[۱] مدرس، دانشگاه آزاد اسلامی واحد هرسین، گروه مهندسی کامپیوتر، هرسین.

[۲] دانشیار، دانشکده فنی و مهندسی، گروه مهندسی کامپیوتر، دانشگاه اراک، اراک.

[۳] مربی، دانشکده مهندسی، گروه مهندسی کامپیوتر، دانشگاه شهید چمران اهواز.

نام نویسنده مسئول:
سجاد اسفندیاری

چکیده

در سیستم‌های نرم‌افزاری زمانی که اجزای سیستم با هم ترکیب می‌شوند ممکن است خطاهای غیر منتظره‌ای رخ دهد. غالبا بررسی تمام خطاها در سیستم‌های بزرگ غیر ممکن است آزمون ترکیباتی روشی است که هدف آن مشخص کردن خطاهای بوجود آمده از ترکیب اجزا سیستم است یکی از مسائل رایج کاهش نمونه آزمون‌ها در آزمون ترکیباتی، آرایه پوشش است که به دو روش محاسباتی و استفاده از الگوریتم‌های فرامکاشفه‌ای قابل تولید هستند که راهکارهای مبتنی بر الگوریتم‌های فرامکاشفه‌ای رایج‌تر و قوی‌تر هستند اما بدلیل استفاده از ساختار پیچیده قالبا در پیکربندی‌های بزرگ دچار مشکل می‌شوند و توان پشتیبانی تا قوه ۶ را دارند راهکار پیشنهادی با بهره‌گیری از دو الگوریتم بهینه‌سازی توده ذرات و تبرید شبیه‌سازی شده و استفاده از ساختمان داده مناسب راهکاری پیشنهادی را تبدیل به یک راهکار قدرت‌مند کرده است نتایج نشان می‌دهد راهکار پیشنهادی در بهینه‌سازی پیکربندی‌های کوچک و بزرگ بسیار قدرت‌مند است و همچنین قوه تعامل تا ۲۲ را پوشش می‌دهد که از این لحاظ در بین تمامی راهکارهای موجود منحصر به فرد است.

واژگان کلیدی: آزمون نرم‌افزار، آزمون ترکیباتی، الگوریتم بهینه سازی توده ذرات، الگوریتم تبرید شبیه‌سازی شده.

مجلـه‌ی علمی پژوهش در علوم رایانه ، شماره ۱۴، تابستان ۱۳۹۸، ص ۱۰- ۱

ISSN: 2745-3673
http://www.Rcsj.ir

مقدمه

هدف از تست نرم‌افزار، یافتن خطاهای موجود در نیازمندی‌ها، طراحی و پیاده‌سازی آن است که این کار باعث افزایش اعتماد به کیفیت نرم افزار می‌شود. فرایند تست، شامل تولید داده‌های تست، دادن این داده‌ها به عنوان ورودی به نرم‌افزار تحت بررسی و سپس بررسی نتایج حاصل از تست نرم‌افزار است. این فرایند را می‌توان به شکل دستی انجام داد ولی این کار برای نرم‌افزارهای بزرگ و پیچیده احتمال کشف خطا را به شدت پائین آورده و نیز هزینه را بالا می‌برد. به این دلیل خودکارسازی تولید نمونه آزمون‌ها از اهمیت بالایی برخوردار است. معمولا حدود ۵۰٪ از هزینه مصرفی در چرخه تولید نرم افزار، به این بخش تعلق دارد. آزمون نرم افزار می‌تواند در هر مرحله از چرخه حیات تولید نرم افزار صورت گیرد، با این تفاوت که در هر مرحله ماهیت متفاوتی خواهد داشت. دنباله آزمونی که به عنوان ورودی آزمون در نظر گرفته می‌شوند، بسیار بزرگ هستند و عملاً بررسی تمامی آنها امکان پذیر نیست. با حذف نمونه‌های آزمون افزونه می‌توان به زیر مجموعه‌ای از دنباله آزمون رسید که ضمن پوشش کامل مشخصات، تعداد خطای بیشتری را نیز آشکار کند.

روش ساختاری و روش عملکردی دو دسته‌بندی مهم از روش‌های تست هستند. در تست ساختاری واحد نرم‌افزار به شکل یک «جعبه سفید» در نظر گرفته می‌شود. در این روش انتخاب نمونه آزمون‌ها بر مبنای کد پیاده‌سازی سیستم و به هدف اجرای عبارات خاص، شاخه‌های برنامه یا مسیرهای خاص است. در تست عملکردی، سیستم به عنوان یک «جعبه سیاه» در نظر گرفته می‌شود. انتخاب نمونه آزمون‌ها در این روش، بر مبنای نیازمندی‌ها یا طراحی خصوصیات نرم‌افزار است[1].

یکی از معیارهای سنجش استراتژی‌های مختلف در تولید دنباله آزمون کمینه، پوشش آرایه (CA) است پوشش آرایه با نماد CA(N; t,p,v) نشان داده می‌شود یک آرایه با تعداد N سطر و k ستون است که هر یک از پارامترهای ورودی v مقدار را می‌پذیرد. در این آرایه هر زیر آرایه t ستونی باید پوشش کامل را برای v مقدار ارضاء کند. هدف اصلی یافتن N با کمترین مقدار ممکن و در کمترین زمان می‌باشد (Colbour). این مسئله یک مسئله NP است و برای تولید این آرایه استراتژی‌های مختلفی وجود دارد که این استراتژی‌ها به دو دسته کلی زیر تقسیم می‌شوند[1]:

- استراتژی‌های مبتنی بر محاسبات محض
- استراتژی‌های مبتنی بر هوش مصنوعی

دسته اول همیشه تعداد ثابتی برای یک پیکربندی مشخص تولید می‌کنند و در خصوص دسته دوم از آنجایی که الگوریتم‌های مبتنی برهوش مصنوعی بر پایه تصادف می‌باشند لذا برخلاف استراتژی های مبتنی بر محاسبات محض قطعیتی برای این الگوریتم‌ها وجود ندارد و معمولا بهترین مقدار را در چند تکرار برای هر پیکربندی در نظر خواهند گرفت.

در بین الگوریتم‌های مبتنی بر محاسبات محض، الگوریتم‌های Jenney، TConfig، PICT، TVG، IPOG قادر به تولید دنباله آزمون تا $t = 6$ با سرعت بالا می‌باشند که سایز مستخرج شده (بدست آمده) از استراتژی‌های یاد شده قابل مقایسه با سایر الگوریتم‌ها در بسیاری از پیکربندی‌ها نمی‌باشند از این دسته الگوریتم TVG با وجود اینکه نتایج مطلوبی را در $t = 2$ و $t = 3$ تولید خواهد کرد اما حداکثر تا $t = 4$ نتایج آن در دسترس می‌باشد.

قویترین استراتژی‌های مبتنی بر هوش مصنوعی در تولید دنباله آزمون کمینه الگوریتم SA، استراتژی PSTG (مبتنی بر الگوریتم PSO)، استراتژی HSS (مبتنی بر الگوریتم HS) و کوکو (فاخته)، و GS (مبتنی بر GA) می‌باشند. الگوریتم SA برای $t = 2$ و $t = 3$ بهترین نتایج ممکن را در بسیاری از پیکربندی‌ها استخراج خواهد کرد اما برای $t \geq 4$ نتایجی در دسترس نیست. استراتژی PSTG و کوکو قادر خواهد بود تا $t = 6$ دنباله آزمون بهینه را تولید کند که در $t = 4$، $t = 5$ و $t = 6$ تقریبا بهترین نتایج را در بسیاری از پیکربندی‌ها استخراج می‌کند ولی برای $t \geq 7$ نمی‌تواند دنباله آزمون را تولید کند. دیگر الگوریتم قدرتمند مبتنی بر هوش مصنوعی استراتژی HSS می باشد این استراتژی قادر به تولید دنباله آزمون تا $t = 15$ می‌کند و استراتژی GS توانایی تولید دنباله آزمون تا $t = 20$ را دارد.

در این پژوهش سعی شده است با ترکیب الگوریتم بهینه سازی توده ذرات و الگوریتم تبرید شبیه‌سازی شده، دنباله آزمون را تا $t = 22$ را تولید کند. نتایج استخراج شده با استراتژی‌های مهم و دردسترس مقایسه می‌شود و برتری این پژوهش نسبت به بسیاری از استراتژی ها در حوزه تولید خودکار دنباله آزمون نشان داده می‌شود.

ادامه مقاله به این صورت است که در بخش دوم با عنوان تعاریف و مفاهیم بنیادی به تشریح مفاهیم همچون پوشش آرایه و الگوریتم بهینه‌سازی توده ذرات و الگوریتم تبرید شبیه‌سازی شده می‌پردازیم. بخش سوم مروری بر روش‌های ارائه شده در زمینه خودکار سازی تولید دنباله آزمون در ساختار پوشش آرایه است. بخش چهارم به تشریح کامل راه‌حل پیشنهادی پرداخته می‌شود که شامل شرح مراحل پیاده‌سازی و تحلیل پارامترهای ورودی راهکار می‌باشد. بخش آخر نیز به جمع‌بندی و نتیجه‌گیری تخصیص داده می‌شود و همچنین ویژگی‌های روش پیشنهادی، معایب و محدودیت‌ها بیان شده است و در ادامه مباحث و کارهایی که می‌توان بر روی آن متمرکزشد، مطرح شده است.

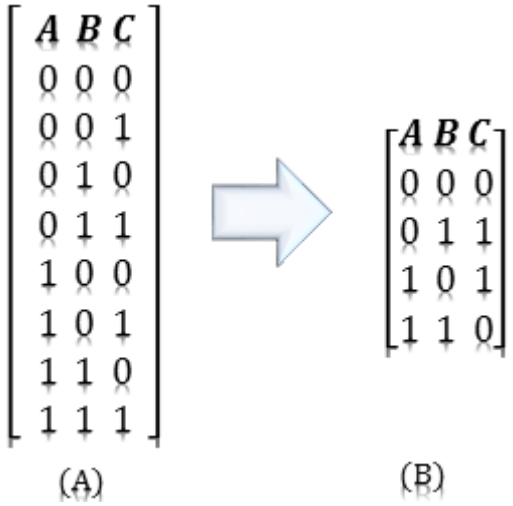
ISSN: 2745-3673
http://www.Rcsj.ir

تعاریف و مفاهیم بنیادی

آزمون پوشش آرایه[1]

دامنه ورودی بسیاری از برنامه های کاربردی محدود است. یعنی تعداد پارامترهای ورودی، کوچک و مقادیری که هر یک از پارامترها به خود می گیرند دارای مرز مشخصی هستند. هنگامی که این اعداد بسیار کوچک باشند (مثلا" ۳ پارامتر ورودی که هر یک ۲ مقدار (۰و ۱) مجزا می‌گیرند)، می توان تمام حالت های ورودی را در نظر گرفت و پردازش دامنه ورودی را به طور جامع، مورد آزمایش قرار داد (شکل(۱-a)) ولی، با رشد تعداد مقادیر ورودی و تعداد مقادیر مجزا جهت هر عنصر داده ای، آزمون جامع غیر عملی و امکان ناپذیر می شود. آزمون پوشش آرایه را می توان در مورد مسائلی به کار برد که در آنها دامنه ورودی نسبتا" کوچک است، ولی برای اجرای آزمون جامع بیش از حد بزرگ است.

برای نشان دادن اختلاف میان معیار پوشش t-way و روش سنتی "یک عنصر ورودی درهر نوبت"، سیستمی را در نظر بگیرید که دارای سه ورودی z و y و x که هر یک از این عناصر ورودی دارای دو مقدار (۰و ۱) هستند. که $۲^۳$=۸ مورد آزمون متفاوت، امکان پذیر است حال اگر بخواهیم مجموع تست پوششی سه پارامتر بولین فوق را با روش t-way محاسبه کنیم به شکل(۱-b) خواهیم رسید. در این معیار پوشش به جای در نظر گرفتن تمام ستون ها، ترکیب ۲ ستون را در نظر می‌گیرد (ستون های ۱و ۲، ستون‌های۱ و ۳، ستون‌های ۲و ۳). در صورتی که در تمامی این جفت ستون ها مقادیر ۰۰، ۰۱، ۱۰ و ۱۱ وجود داشت باشد به معنای پوشش کامل با معیار قوه ۲ خواهد بود [23].



شکل (۱): دنباله آزمون ۲-way با ورودی ۳ پارامتر بولین.

تعریف: CA(N: t, p, v) یک آرایه N*p بر روی v نماد است که هر زیر آرایه N*t از آن، تمام اشکال t تایی از آن v نماد را حداقل یکبار پوشش میدهد[2]. که تعداد کل پوشش ها نیز در این حالت از فرمول (۱) بدست می‌آید.

$$MAX_{Coverage} = \binom{p}{t} * v^t \qquad (۱)$$

به عنوان مثال CA (9: 2, 4, 3) یک آرایه ۴*۹ بر روی سه نماد (۰،۱،۲) که هر زیر آرایه ۲ * ۹ از آن تمام حالت‌های ممکن {۰, ۰}, {۰, ۱},{۰, ۲}, {۱, ۰}, {۱, ۱}, {۱, ۲}, {۲, ۰}, {۲, ۱}, {۲, ۲} را حداقل یکبار پوشش می‌دهد. همانطوری که ملاحظه می-شود تعداد سطرها از ۸۱ (۴^۳) به ۹ سطر کاهش پیدا می‌کند.

الگوریتم بهینه‌سازی توده ذرات (PSO)

الگوریتم PSO[2] یک الگوریتم جستجوی اجتماعی است که از روی رفتار اجتماعی دسته‌های پرندگان مدل شده است. در ابتدا این الگوریتم به منظور کشف الگوهای حاکم بر پرواز همزمان پرندگان و تغییر ناگهانی مسیر آنها و تغییر شکل بهینه‌ی دسته به کار گرفته شد.

[1] coverage array testing

ISSN: 2745-3673
http://www.Rcsj.ir

در PSO، ذرات در فضای جستجو جاری می‌شوند. تغییر مکان ذرات در فضای جستجو تحت تأثیر تجربه و دانش خودشان و همسایگانشان است. بنابراین موقعیت دیگر توده ذرات روی چگونگی جستجوی یک ذره اثر می‌گذارد . نتیجه‌ی مدل‌سازی این رفتار اجتماعی فرایند جستجویی است که ذرات به سمت نواحی موفق میل می‌کنند. ذرات از یکدیگر می‌آموزند و بر مبنای دانش بدست آمده به سمت بهترین همسایگان خود می‌روند اساس کار PSO بر این اصل استوار است که در هر لحظه هر ذره مکان خود را در فضای جستجو با توجه به بهترین مکانی که تاکنون در آن قرار گرفته است و بهترین مکانی که در کل همسایگی‌اش وجود دارد، تنظیم می‌کند.

هرذره با دنبال کردن ذرات بهینه در حالت فعلی، به حرکت خود درفضای مساله ادامه می‌دهد. گروهی از ذرات PSO درآغاز کار به‌صورت تصادفی به وجود می‌آیند و با به روز کردن نسل‌ها سعی در یافتن راه‌حل بهینه می‌نمایند. در هر گام، هر ذره با استفاده از دو بهترین مقدار به روز می‌شود. اولین مورد، بهترین موقعیتی است که تا کنون ذره موفق به رسیدن به آن شده است. موقعیت مذکور با pbest (بهترین محلی) شناخته و نگهداری می‌شود. بهترین مقدار دیگری که با نام بهترین عمومی توسط الگوریتم مورد استفاده قرار می‌گیرد، بهترین موقعیتی است که تا کنون توسط جمعیت ذرات بدست آمده است.این موقعیت با gbest نمایش داده می شود.

پس از یافتن بهترین مقادیر، سرعت و مکان هر ذره با استفاده از معادلات (۲) و (۳) به روز می‌شود.

$$v = v + c1 * rand * (pbest - position) + c2 * rand * (gbest - position) \quad (۲)$$

$$position = position + v \quad (۳)$$

سمت راست معادله (۲) از سه قسمت تشکیل شده است که قسمت اول، سرعت فعلی ذره است و قسمتهای دوم و سوم تغییر سرعت ذره و چرخش آن به سمت بهترین تجربه شخصی و بهترین تجربه گروه را به عهده دارند. اگر قسمت اول را در این معادله درنظر نگیریم، آنگاه سرعت ذرات تنها با توجه به موقعیت فعلی و بهترین تجربه ذره و بهترین تجربه جمع تعیین می‌شود. به این ترتیب، بهترین ذره جمع، در جای خود ثابت می‌ماند و سایرین به سمت آن ذره حرکت می‌کنند. در واقع حرکت دسته جمعی ذرات بدون قسمت اول معادله(۲)، پروس‌ای خواهد بود که طی آن فضای جستجو به تدریج کوچک می‌شود و جستجویی محلی حول بهترین ذره شکل می‌گیرد. در مقابل اگر فقط قسمت اول معادله (۳) را در نظر بگیریم، ذرات راه عادی خود را می‌روند تا به دیواره محدوده برسند و به نوعی جستجویی سراسری را انجام می‌دهند.

الگوریتم تبرید شبیه‌سازی شده (SA)

برای حل یک مسئله بهینه‌سازی، الگوریتم SA[۳] ابتدا از یک جواب اولیه شروع می‌کند و سپس در یک حلقه تکرار به جواب‌های همسایه حرکت می‌کند. اگر جواب همسایه بهتر از جواب فعلی باشد، الگوریتم آن را به عنوان جواب فعلی قرار می‌دهد (به آن حرکت می‌کند)، در غیر این صورت، الگوریتم آن جواب را با احتمال exp(-ΔE/T) به عنوان جواب فعلی می‌پذیرد. در این رابطه ΔE تفاوت بین تابع هدف جواب فعلی و جواب همسایه‌است و T یک پارامتر به نام دما است. در هر دما، چندین تکرار اجرا می‌شود و سپس دما به آرامی کاهش داده می‌شود. در گام‌های اولیه دما خیلی بالا قرار داده می‌شود تا احتمال بیشتری برای پذیرش جواب‌های بدتر وجود داشته باشد. با کاهش تدریجی دما، در گام‌های پایانی احتمال کمتری برای پذیرش جواب‌های بدتر وجود خواهد داشت و بنابراین الگوریتم به سمت یک جواب خوب همگرا می‌شود. الگوریتم SA یک الگوریتم غیرمقید می‌باشد که برای طراحی‌های سخت به کار می‌رود. شبه کد در شکل (۲) یک پیاده‌سازی از تبرید شبیه‌سازی شده را ارائه نشان می‌دهد. این الگوریتم از حالت s0 شروع می‌کند و تا رسیدن به حداکثر گام‌ها، Kmax یا رسیدن به حالتی با انرژی Emin یا کمتر از آن متوقف می‌شود. در این روند، صدا کردن تابع neighbor یک همسایه رندوم حالت فعلی را انتخاب می‌کند. تابع random یک عدد را در بازه (۰،۱) به صورت یکنواخت انتخاب می‌کند. زمان‌بندی تبرید با صدا کردن تابع temperature تعریف می‌شود، که در هر مرحله دمای سیستم را بر حسب زمان به دست می‌آورد.

[2] Particle Swarm Optimization
[3] Simulated Annealing

ISSN: 2745-3673
http://www.Rcsj.ir

Let s = s0
For k = 0 through k_max (exclusive)
T ← temperature (k/k_ma)
Pick a random neighbor, snew ← neighbor(s)
If P (E(s), E (s_new), T) ≥ random (0, 1)
s ← s_new
Output: the final state s

شکل (۲): الگوریتم تبرید شبیه‌سازی شده

پیشینه پژوهش

پژوهش‌های زیادی برای حل این مسئله وجود دارد که به دو گروه اصلی تقسیم می‌شوند [3]:

- روش‌های ریاضی[۴]
- روش‌های محاسباتی[۵]

روش‌های ریاضی از برخی از ساختارهای ترکیباتی مربوط به توابع ریاضی مانند آرایه متعامد۶ استنتاج شده‌اند اما روش‌های محاسباتی عمدتاً از استراتژی‌های حریصانه یا تکنیک‌های فرامکاشفه‌ای برای تولید آرایه پوشش بهینه در فضای جستجو استفاده می‌کنند. روش‌های ریاضی آرایه پوشش بهینه را فقط برای پیکربندی‌های خاصی تولید می‌کنند از استراتژی‌های موجود در روش‌های ریاضی به Combinatorial Test Servicesو (TConfig) که بر اساس تعمیم آرایه متعامد ساخته شده‌اند می‌توان اشاره کرد. عمده‌ترین مشکل راهکارهای مبتنی بر آرایه متعامد این است که توانایی ساخت آرایه پوشش برای پیکربندی‌های کوچک و خاص را دارند که این محدودیت باعث استفاده از روش‌های محاسباتی شده است.

راهکار اغلب روش‌های محاسباتی به این صورت است که در ابتدا تمام ترکیبات ممکن با توجه به مشخصات ورودی تولید می‌شود پس از آن نمونه آزمون‌ها برای پوشش این ترکیبات ساخته می‌شوند. تفاوت اصلی بین استراتژی‌های مختلف روش محاسباتی ساختن نمونه آزمون است. دو روش اصلی برای ساخت نمونه آزمون در روش‌های محاسباتی وجود دارد [4]:

- یک-آزمون-در-یک-زمان[۷]
- یک-پارامتر-در-یک-زمان[۸]

در روش یک-آزمون-در-یک-زمان در هر مرحله یک نمونه آزمون یا یک مجموعه کامل از نمونه آزمون‌ها را تولید می‌کند سپس نمونه آزمونی که بیشترین ترکیبات را پوشش دهد انتخاب می‌شود این نمونه آزمون یک سطر از دنباله آزمون نهایی (آرایه پوشش) را تشکیل می‌دهد. این روش نیز به طور کلی به دو دسته زیر تقسیم می‌شوند [2]:

- راهکارهای مبتنی بر محاسبات محض
- راهکارهای مبتنی بر هوش مصنوعی

استراتژی‌های مبتنی بر محاسبات محض با استفاده از محاسبات و عبارات ریاضی نمونه آزمون را مشخص می‌کند و این امر باعث می‌شود که نتایج حاصل قطعی باشد بدین معنا که اگر چندین بار الگوریتم بر روی یک پیکربندی خاص اعمال گردد نتایج هیچ تغییری نکند که معروف‌ترین استراتژی‌هایی بر اساس این روش AETG، mAETG، PICT، DDA، CTE-XL، TVG، Jenny، GTWay و ITCH است [5].

دسته‌ی دیگری از روش‌های مبتنی بر محاسبات الگوریتم‌های مبتنی بر هوش مصنوعی هستند. این الگوریتم‌ها بر اساس روش یک-آزمون-در-یک-زمان گسترش یافته‌اند. با کاربرد روز افزون مهندسی نرم‌افزار مبتنی بر جستجو۹ در حل بهینه مسائل نرم‌افزار، استفاده از الگوریتم‌های فرامکاشفه‌ای در تولید آرایه پوشش از سایر روش‌ها پرطرفدارتر است. الگوریتم‌های فرا مکاشفه‌ای با یک مجموعه تصادفی از راه‌حل‌ها شروع می‌شوند سپس به‌منظور بهبود این راه‌حل‌های اولیه یک سری تغیرات بر روی آن‌ها انجام می‌شود و در آخر بهترین راه‌حل

---

[6] Mathematical method
[5] Computational method
[6] Orthogonal arrays
[7] One-test-at-a-time
[8] One-parameter-at-a-time
[9] Search Based Software Engineering

ISSN: 2745-3673
http://www.Rcsj.ir

انتخاب می‌شود این مراحل تا زمانی که تمام ترکیبات پوشش داده شوند ادامه می‌یابد. تعدادی از الگوریتم‌های فرا مکاشفه‌ای که تاکنون استفاده شده‌اند SA، TS، GA، ACA، PSO، CS، HSS، TLBO، H-B، BA و BPTS هستند [6, 7].

الگوریتم SA نتایج بهینه‌ای را برای پیکربندی‌های کوچک (t<۳) تولید می‌کند. الگوریتم GA و TS نیز وقتی ابعاد مسئله بالا برود (t>۳) قادر به تولید جواب نیستند. الگوریتم H-B تا t = ۱۰ پشتیبانی می‌کند اما به دلیل زمان بر بودن فرآیند جستجو فقط برای آرایه-های پوشش محدودی توانایی تولید دارد. الگوریتم CS و PSO توانایی پشتیبانی تا t=۶ را دارند. الگوریتم PSO مشکلاتی از قبیل تنظیم کردن پارامتر و همگرایی زودرس که باعث گیر افتادن در بهینه محلی می‌شود، را دارد که این عوامل تاثیر بسزایی بر توانایی بهینه‌سازی آن می‌گذارد. مشکل تمام این استراتژی‌ها محاسبات سنگین آن‌ها است و اینکه در تولید دنباله آزمون نهایی از دقت بالایی برخوردار نیستند. برای مثال الگوریتم ژنتیک از پیچیدگی مراحل ادغام و جهش رنج می‌برد. ACA هنگامی که تعداد مورچه‌ها زیاد می‌شود با مشکلات متعددی روبه‌رو می‌شود. TS از مکانیسم بروزرسانی لیست ممنوعه رنج می‌برد. علاوه بر این، استراتژی‌ها با این مسأله مواجه هستند که باید یک توازنی بین دقت الگوریتم و سرعت آن برقرار کنند.

روش یک-پارامتر-در-یک-زمان آرایه پوشش را با اضافه کردن هر پارامتر مرحله به مرحله می‌سازد و سپس با اضافه کردن ترکیبات ممکن برای پارامترهای موجود مراحل را تا زمانی که پوشش کامل شود ادامه می‌دهد. IPOG، IPOG-s و IPOG-D استراتژی‌های رایجی هستند که بر اساس این روش گسترش داده شده‌اند [8, 9].

راهکار پیشنهادی

راهکار جدیدی که در این پژوهش ارائه شده است استفاده ترکیبی الگوریتم‌های جستجوی SA و PSO در حوزه تولید دنباله آزمون است. در این پژوهش به کمک استراتژی پوشش آرایه (CA) سعی در تولید دنباله آزمون با کمترین تعداد را دارد که در این راستا جهت تولید دنباله آزمون کمینه با تغییرات در الگوریتم PSO و الگوریتم SA و در نهایت ترکیب این دو الگوریتم موفق به تولید دنباله آزمونی شده است که تقریبا در بسیاری از پیکربندی‌ها نتایج بهتر و یا برابر را تولید می‌کند. یکی از زمان‌برترین مراحل تولید دنباله آزمون تابع ارزیابی می‌باشد که برابر با پوشش‌های جدیدی است که در دنباله آزمون وجود ندارد و هنگامی که تعداد کل پوشش‌های جدید به فرمول (۱) برسد به معنای پوشش کامل می‌باشد و الگوریتم در آن زمان پایان می‌پذیرد و در صورتی که تعداد تکرار الگوریتم پایان یابد و پوشش به حدنصاب نرسد الگوریتم قادر به تولید دنباله آزمون نخواهد بود.

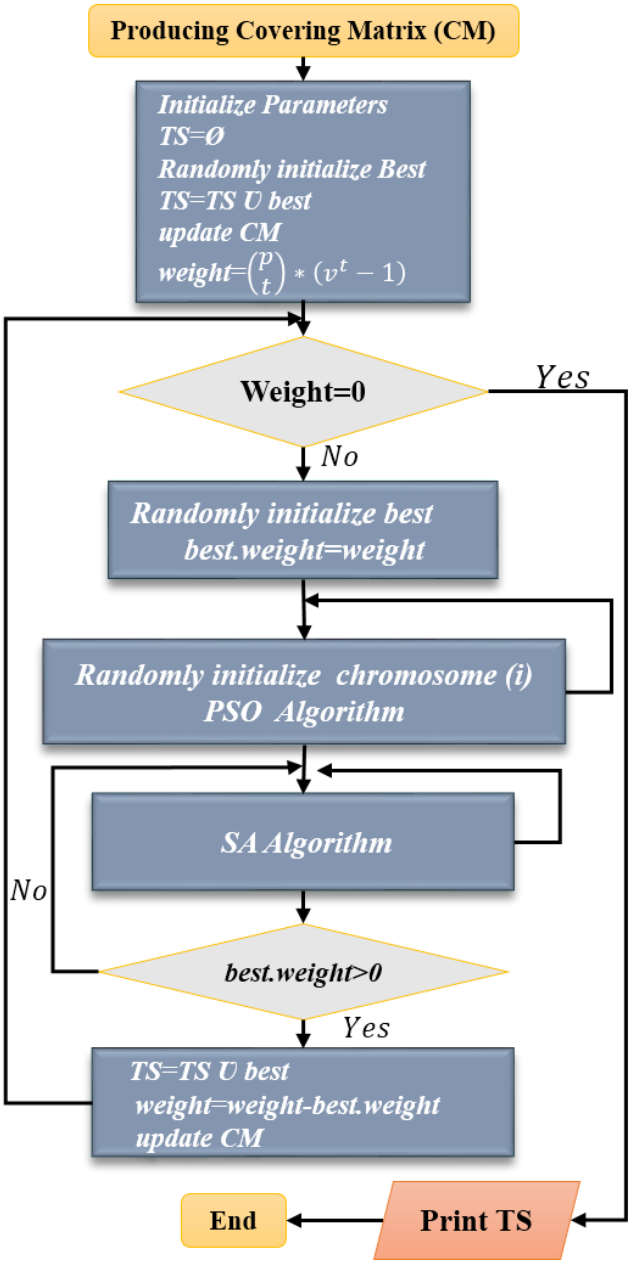
در الگوریتم تولید CA(N: t, p, v) هر ذره شامل دو قسمت داده و وزن می‌باشد که داده‌ها از یک آرایه p عنصری تشکیل شده است که مقادیر عناصر آن بین ۰ تا v-1 است و وزن آن از تابع ارزیابی بدست می‌آید. همانطوری که در شکل (۳) **Error! Reference source not found.** نشان داده شده است در گام اول از آنجایی که وزن سطر اول دنباله آزمون برای تمامی مقادیر $X_i(t)$ برابر $\binom{k}{t}$ خواهد بود لذا سطر اول به طور تصادف انتخاب می‌شود. الگوریتم پیشنهادی دارای دو حلقه اصلی است که اولین حلقه تعداد کل تکرار را مشخص می‌کند در پیاده سازی‌های معمول پس از پایان این حلقه به یک جواب نهایی برای مسئله خواهیم رسید اما ماهیت تولید دنباله آزمون به گونه‌ای است که در هر تکرار یک نمونه آزمون از دنباله آزمون را انتخاب می‌کند و این نمونه آزمون هیچ گونه تاثیری بر مراحل بعدی اجرای این حلقه ندارد به همین خاطر در ابتدای این حلقه بهترین مقدار(best) به طور تصادف انتخاب می‌شود. دیگر حلقه در الگوریتم پیشنهادی به تعداد جمعیت ذرات تکرار می‌شود که این مقدار با توجه به پارامترهای ورودی CA بین ۲۰ تا ۲۵۰ متغیر است که در داخل این حلقه پس از تولید مقدار جدید $X_i(t)$ و ایجاد همسایه برای best وزن هریک محاسبه می‌گردد و best با بهترین مقدار تعویض می‌شود. و در پایان این حلقه best به دنباله آزمون اضافه می‌گردد و تا زمانی که پوشش کامل شود الگوریتم ادامه خواهد داشت. تابع تولید همسایه یک تابع ساده و نحوه عملکرد آن به صورتی است که ۲ یا ۳ عنصر از k عنصر را بصورت تصادف انتخاب و برای هریک مقداری تصادفی بین ۰ و v-1 انتخاب می‌کند. از آنجایی که این تغییر بر روی بهترین مقدار انجام می‌گیرد این تابع بسیار موثر بوده و اصلی‌ترین دلیل برتری آن نسبت به دیگر پژوهش‌های مبتنی بر PSO است.

ارزیابی

در این بخش سعی داریم نتایج استراتژی پیشنهادی را با سایر استراتژی موجود مبتنی بر محاسبات و مبتنی بر PSO و SA مقایسه کنیم در این ارزیابی نمایش برجسته در جداول نمایانگر کمترین مقدار در پیکر بندی مربوطه است. مشخصات بستر سخت‌افزاری برای اجرای روش پیشنهادی عبارتند از: Windows 7، CPU i7QM ™core 2.20GHz و RAM 6GB. و پیاده سازی آن در محیط MATLAB R2013b صورت گرفته است.

ISSN: 2745-3673
http://www.Rcsj.ir

همانطور که در جدول (۱) مشاهده می‌شود استراتژی پیشنهادی را با دو استراتژی مبتنی بر محاسبات IPOG و IPOG-D که در ابزار ACTS قرار دارند و استراژی‌های مبتنی بر الگوریتم‌های فرامکاشفه‌ای PSO و CS و استراتژی PICT که مبتنی بر محاسبات است مقایسه می‌گردد. در جدول (۱) پیکربندی ساده CA (N; 2, p, 2) که در آن $3 \leq p \leq 15$ است در نظر گرفته شده است. همانطور که ملاحظه می‌شود در این مقایسه استراتژی پیشنهادی در تمامی موارد بهترین عملکرد را دارد و بعد از آن استراتژی IPOG در هشت مورد بهترین نتیجه را تولید می‌کند در این پیکرندی استراتژی‌های مبتنی بر محاسبات نتایج قابل قبولی را تولید می‌کنند.



شکل (۳): الگوریتم راهکار پیشنهادی

ISSN: 2745-3673
http://www.Rcsj.ir

جدول (۱) :اندازه دنباله آزمون برای CA (N; 2, p, 2) و $3 \leq p \leq 15$

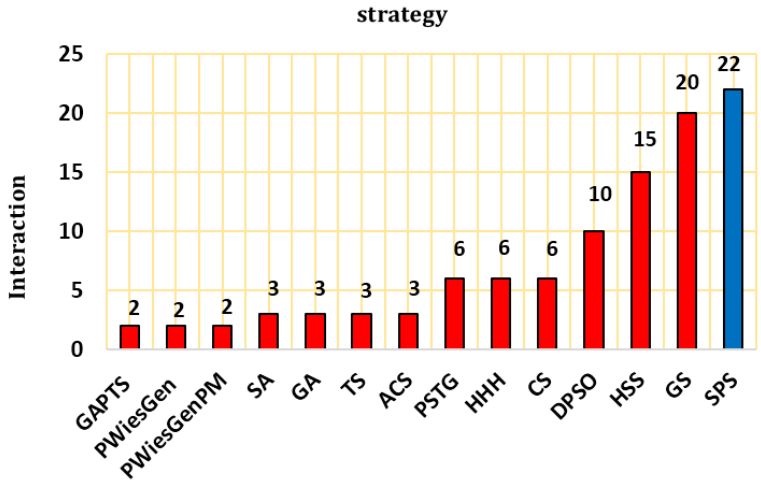
p	IPOG-DN	IPOGN	PICTN	CSN	PSTGN	SPSN
۳	۶	۴	۴	۴	۴	۴
۴	۶	۶	۵	۶	۶	۵
۵	۶	۶	۷	۶	۶	۶
۶	۶	۷	۶	۷	۷	۶
۷	۸	۷	۷	۷	۷	۷
۸	۸	۸	۷	۸	۸	۷
۹	۸	۸	۹	۸	۸	۸
۱۰	۸	۸	۹	۸	۸	۸
۱۱	۹	۸	۹	۹	۹	۸
۱۲	۹	۸	۹	۹	۹	۸
۱۳	۹	۸	۹	۹	۹	۸
۱۴	۹	۱۰	۱۰	۹	۹	۹
۱۵	۱۰	۱۰	۱۰	۱۰	۱۰	۹

جدول (۲) ارزیابی را برای پیکربندی سنگین CA (N; 4, p, 5) که در آن $5 \leq p \leq 15$ است، نشان می‌دهد. همانطور که مشاهده می‌شود در این پیکربندی استراتژی‌های مبتنی بر الگوریتم‌های فرامکاشفه ای بسیار قوی‌تر از الگوریتم‌های مبتنی بر محاسبات هستند. که در این بین استراتژی SPS در تمامی موارد برتری خود را نسب به سایر استراتژی‌ها نشان می‌دهد.

جدول (۲) :اندازه دنباله آزمون برای CA (N; 4, p, 5) و $5 \leq p \leq 15$

p	IPOG-DN	IPOGN	PICTN	CSN	PSTGN	SPSN
۵	۱۲۵۰	۷۷۳	۸۱۰	۷۷۶	۷۷۹	۷۷۰
۶	۱۲۵۰	۱۰۵۸	۱۰۷۲	۹۹۱	۱۰۰۱	۹۸۴
۷	۱۸۵۸	۱۲۹۳	۱۲۷۹	۱۲۰۰	۱۲۰۹	۱۱۷۵
۸	۱۸۵۸	۱۵۱۱	۱۴۶۸	۱۴۱۵	۱۴۱۴	۱۳۷۰
۹	۲۱۱۰	۱۷۰۲	۱۶۴۳	۱۵۶۲	۱۵۷۰	۱۵۴۸
۱۰	۲۱۱۰	۱۸۶۹	۱۸۱۲	۱۷۳۱	۱۷۱۶	۱۶۳۷
۱۱	۲۴۸۰	۲۰۲۴	۱۹۵۷	۲۰۶۲	۱۹۰۲	۱۸۳۸
۱۲	۲۴۷۹	۲۱۵۰	۲۱۰۳	۲۲۲۳	۲۰۱۵	۱۹۶۶
۱۳	۲۸۶۹	۲۲۹۶	۲۲۳۸	۲۱۹۸	۲۱۴۸	۲۱۴۰
۱۴	۲۹۹۱	۲۴۳۶	۲۳۵۹	۲۳۰۱	۲۲۷۴	۲۱۵۳
۱۵	۳۳۳۷	۲۵۳۸	۲۴۸۰	۲۴۱۵	۲۳۹۷	۲۲۶۴

ISSN: 2745-3673
http://www.Rcsj.ir



شکل (۴): نمایش قوه تعامل استراتژی‌ها

ارزیابی آخر مربوط به توان استراتژی‌ها در پوشش قوه تعامل (t) است بدیهی است هرچه استراتژی قوه تعامل بالاتری را پوشش دهد قوی‌تر است. همانطور که در شکل (۴) مشاهده میگردد استراتژی پیشنهادی در این ارزیابی باپوشش قوه تعامل ۲۲ (t = 22) از تمامی استراتژی‌های موجود قوی‌تر است.

نتیجه گیری و کارهای آتی

تست نرم افزار بخش مهمی از فرایند توسعه سیستم به حساب می‌آید به همین علت یک فعالیت مهم در تصمیم‌گیری کیفیت نرم افزار است. با پیشرفت سریع نرم افزارها نیاز به انجام مراحل تست به شکل خودکار به امری قابل توجه برای توسعه‌دهندگان سیستم‌ها تبدیل شده است.

در این پژوهش راهکار جدیدی برای تولید خودکار دنباله آزمون در حوزه تست نرم‌افزار پیشنهاد شده است. این روش با رویکرد تست جعبه سیاه، دنباله آزمون را استخراج می‌کند. در این راهکار از ترکیب الگوریتم جستجوی PSO و الگوریتم جستجوی SA استفاده شده است و با ترکیب این دو الگوریتم ساده موفق به ارائه راهکاری نو و کارآمد در تولید دنبال آزمون کمینه از لحاظ تعداد سطرهای آرایه شده ایم. با توجه به ارزیابی انجام شده، این راهکار برخلاف سایر الگوریتم‌های فرامکاشفه‌ای با ساختار نه چندان پیچیده، نتایج خوبی استخراج می‌کند.

راه‌های متعددی برای گسترش این کار تحقیقاتی وجود دارد. ترکیب سایر الگوریتم‌های فرامکاشفه‌ای ممکن است نتایج بهتر و با سرعت بالاتری ارائه دهد. استراتژی‌های هوش مصنوعی به دلیل ساختار پیچیده قالبا" تا قوه‌ی ۶ قادر به تولید دنباله آزمون خواهند بود. تغییر این الگوریتم‌ها و چشم پوشی از برخی توابع پیچیده‌ی آن‌ها می‌تواند دست مایه‌ای برای کارهای آتی باشد.

ISSN: 2745-3673
http://www.Rcsj.ir

منابع و مراجع

[1] S. Esfandyari and V. Rafe, "*A tuned version of genetic algorithm for efficient test suite generation in interactive t-way testing strategy*", Information and Software Technology, vol. 94, pp. 165– 185, 2018.

[2] B. S. Ahmed, K. Z. Zamli and C. P. Lim, "*Application of Particle Swarm Optimization to uniform and variable strength covering array construction*", Applied Soft Computing Vol 12, pp. 1330-134, 2012.

[3] A. A. Alsewari, K. Z. Zamli, "*Design and implementation of a harmony-search-based variable-strength t-way testing strategy with constraints support*", Information and Software Technology Vol. 54 pp. 553-568, 2012.

[4] B. S. Ahmed, T. S. Abdulsamad and M. Y. Potrus, "*Achievement of minimized combinatorial test suite for configuration-aware software functional testing using the Cuckoo Search algorithm*", Information and Software Technology Vol 66, pp. 13-29, 2015.

[5] M.B. Cohen, "*Designing Test Suites for Software Interaction Testing*", PhD Thesis, Department of Computer Science, University of Auckland, New Zealand, 2004.

[6] J. Stardom, "*Metaheuristic and the Search for Covering and Packing Array*", Master's thesis, Department of Mathematics, Simon Fraser University, Canada, 2001.

[۷] سجاد اسفندیاری، وحید رافع، "راهکار نوین جهت تولید دنباله آزمون کمینه در آزمون نرم افزار با ترکیب الگوریتم های جستجوی تپه نوردی و جستجوی خفاش"، مجله مهندسی برق دانشگاه تبریز، جلد ۴۶ ، شماره ۳، پائیز ۹۵.

[۸] زهرا عباسی، سجاد اسفندیاری، وحید رافع، "ساخت آرایه پوشش با استفاده از الگوریتم بهینه‌سازی مبتنی برآموزش و یادگیری "، مجله مهندسی برق دانشگاه تبریز، جلد ۴۸، شماره ۱، بهار ۱۳۹۵.

[9] B. S. Ahmed, K. Z. Zamli, "*A variable strength interaction test suites generation strategy using particle swarm optimization*", Journal of Systems and Software, Vol 84 pp. 2171-2185, 2011.