مقاله پژوهشی

تولید دنباله آزمون کمینه با ترکیب الگوریتم های جستجوی گرانشی و شاهین هریس

سجاد اسفندیاری، داور گیوکی، لیلا یوسف‌وند، عین‌الله پیرا

چکیده: در حال حاضر، استفاده از آزمون ترکیبی در نرم‌افزارهای مدرن امری ضروری برای توسعه نرم‌افزارها به شمار می‌رود. آزمون ترکیبی یا آزمون t-way به عنوان یک راهکار کارآمد برای کاهش حجم مجموعه آزمون شناخته شده است، به گونه‌ای که امکان آزمون نرم‌افزار با تعداد نمونه‌های کمتر فراهم می‌کند. پوشش آرایه یکی از جنبه‌های حیاتی در آزمون ترکیبی محسوب می‌شود که رویکردهای مختلفی برای تولید آن ارائه شده است. این روش‌ها اغلب دارای کارایی (سرعت) و بهره‌وری (اندازه آرایه) قابل قبولی هستند، اما کمبود یک روش که هر دو خصوصیت کارایی و بهره‌وری را به طور همزمان داشته باشد، مشهود است. در این پژوهش، با استفاده از الگوریتم جستجوی گرانشی و ترکیب آن با الگوریتم شاهین هریس، مجموعه آزمون بهینه با حداقل تعداد نمونه آزمون و کارایی بالا پدید آمده است. علاوه بر این، این رویکرد قابلیت تولید مجموعه آزمون برای پیکربندی‌های با توان بالا را نیز دارا می‌باشد. ارزیابی نتایج نشان می‌دهد که روش پیشنهادی از نظر کارایی و بهره‌وری بسیار مناسب و قدرتمند است.

کلیدواژه: **آزمون ترکیبی، آرایه پوشش، الگوریتم جستجوی گرانشی، الگوریتم شاهین هریس.**

۱- مقدمه

آزمون ترکیبی یک روش متداول و حیاتی در فرآیند آزمون نرم‌افزار برای تشخیص خطاهای ناشی از تعامل زیرسیستم‌های یک سیستم است [۱]. آرایه پوشش یکی از جنبه‌های اصلی آزمون ترکیبی است که با نمونه‌برداری از تعداد کمی از نمونه‌های آزمون، امکان آزمون سیستم را فراهم می‌آورد. انتخاب نمونه‌های آزمون باید با دقت انجام شود تا اهداف آزمون برآورد شود و همچنین تعداد نمونه‌ها باید حداقل و سرعت تولید آنها مناسب باشد [۲]. کاربرد آزمون ترکیبی برای سیستم‌های پیچیده، به ویژه سیستم‌های با ابعاد بزرگ، بسیار چالشی است. برای مثال، برای آزمون یک سیستم با ۹ پارامتر، که هر پارامتر آن ۵ مقدار دارد، نیاز به بررسی ۵ به توان ۹ (۱،۹۵۳،۱۲۵) نمونه آزمون است که بسیار حجیم و کاهش آن امری ضروری است. روش‌های مختلفی برای تولید آرایه پوشش وجود دارد که به دو دسته کلی روش‌های ریاضی و محاسباتی تقسیم می‌شوند [۳].

در روش‌های ریاضی، استفاده از روش‌های جبری و ترکیبی به عنوان روش‌های معمول برای حل مسائل مطرح است، در حالی که در حوزه محاسبات، روش‌های حریصانه و فراابتکاری برای تولید آرایه‌های پوشش استفاده می‌شود. به دلیل بهره‌وری بالاتر، روش‌های محاسباتی بیشتر مورد توجه قرار می‌گیرند و استفاده از آن‌ها در سال‌های اخیر افزایش یافته است. انواع متعددی از راهکارها با استفاده از الگوریتم‌های هوش مصنوعی ارائه شده‌اند، از جمله شبیه‌سازی تبرید، الگوریتم تپه‌نوردی، الگوریتم ژنتیک، الگوریتم‌های چندفازی، الگوریتم کلونی مورچگان و الگوریتم توده ذرات. این راهکارها نتایج بسیار مطلوبی را ارائه می‌دهند، با این حال بررسی‌ها نشان می‌دهد که راهکار DPSO [۴] بهترین عملکرد را دارد در حالی که GS [۵] و ImpARO [۶] از نظر کارایی بسیار مناسب هستند. بنابراین، نیاز به تحقیقات بیشتر برای توسعه یک روش با کارایی و بهره‌وری هماهنگ به چشم می‌خورد.

در این پژوهش، با استفاده از الگوریتم جستجوی گرانشی و ترکیب آن با الگوریتم شاهین هریس و بهینه‌سازی آنها، مجموعه آزمون بهینه با تعداد نمونه آزمون کمینه و کارایی مناسب تولید می‌شود. از رویکردهای مختلفی برای افزایش سرعت و دقت تولید مجموعه آزمون استفاده می‌شود، به علاوه از تابع کمینه‌ساز برای بهبود بهینگی مجموعه آزمون استفاده می‌شود. این روش توانایی غلبه بر مشکلات بهینه‌سازی محلی را دارد و با تولید نمونه‌های آزمون تصادفی، به بهبود سرعت و دقت کمک می‌کند.

ادامه مقاله به شرح ذیل است: فصل دوم مفاهیم پایه را تشریح می‌کند، فصل سوم به جزئیات پیاده‌سازی الگوریتم جستجوی گرانشی و شاهین هریس برای تولید آرایه پوشش می‌پردازد. فصل چهارم ارزیابی روش پیشنهادی را شرح می‌دهد و فصل پنجم به نتیجه‌گیری و پیشنهادها برای کارهای آینده اختصاص داده شده است.

۲- ادبیات موضوعی

۲-۱ آرایه پوشش

یکی از مجموعه‌های آزمونی پرکاربرد که در آزمون‌های ترکیبی استفاده می‌شود، آرایه پوشش[1] (CA) است. آرایه پوشش یا آزمون t-way (که t تعداد تعاملات آزمون است)، یک آرایه با ابعاد $N \times p$ است

این مقاله در تاریخ ۳ فروردین ماه ۱۴۰۳ دریافت و در تاریخ ۲۳ آبان ماه ۱۴۰۳ بازنگری شد.

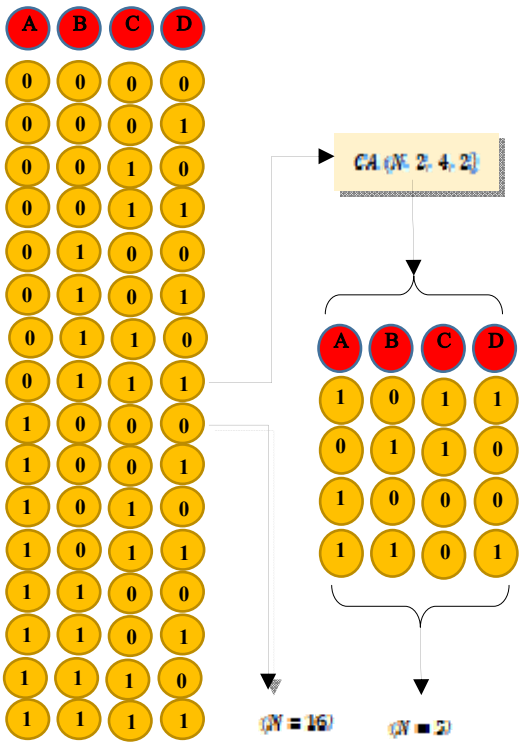
سجاد اسفندیاری، (نویسنده مسئول)، گروه مهندسی کامپیوتر، دانشکده فنی و مهندسی، دانشگاه ملایر، ملایر، ایران، (email: esfandyari@malayeru.ac.ir).

داور گیوکی، گروه مهندسی کامپیوتر، دانشکده فنی و مهندسی، دانشگاه ملایر، ملایر، ایران، (email: davar.giveki@malayeru.ac.ir).

لیلا یوسفوند، گروه مهندسی کامپیوتر، دانشکده فنی و مهندسی، دانشگاه لرستان، خرم آباد، ایران، (email: l.yousofvand@gmail.com).

عین‌الله پیرا، دانشکده فناوری اطلاعات و مهندسی کامپیوتر، دانشگاه شهید مدنی آذربایجان، تبریز، ایران، (email: pira@azaruniv.ac.ir).

1. Covering Array



شکل ۱: : ساختار آرایه پوشش.

است که در آن N تعداد نمونه‌های آزمون است و دارای دو ویژگی زیر است [۷]:

- هر ستون i شامل اعضایی از مجموعه V_i است.
- تعداد سطرهای هر زیرآرایه $N\times t$ نمایی از آن، تمام ترکیبات $|V_{p_1}|\times|V_{p_2}|\times...\times|V_{pt}|$ از t ستون را به طور حداقل یک بار پوشش می‌دهد.

آرایه پوشش با نماد $CA(N;t,p,x_1,x_2,...,x_p)$ و در حالت $x_1=x_2=...=x_p=x$ با نماد $CA(N;t,p,v)$ یا $CA(N;t,p,v^p)$ نمایش داده می‌شود. در اکثر منابع، اگر xها با یکدیگر برابر نباشند، به آن MCA[1] و اگر t متغیر باشد، آن را آرایه پوشش با قدرت متغیر[2] (VSCA) می‌نامند [۸].

برای بهترین فهم از آزمون ترکیبی، به یک سیستم با ۴ پارامتر که هر کدام دو مقدار (۰ و ۱) را به خود اختصاص می‌دهند، توجه کنید (شکل ۱). این سیستم دارای ۲ به توان ۴ نمونه آزمون (v^p) است، بنابراین برای آزمودن این سیستم نیاز به ۱۶ نمونه آزمون است [۹]. همچنین، اگر این سیستم را با تنظیم CA با تعامل ۲ مورد آزمون قرار دهیم، تعداد نمونه آزمون‌ها به ۵ نمونه آزمون کاهش می‌یابد. در این پیکربندی، نمونه‌های آزمون باید تمامی ترکیبات سیستم را پوشش دهند. برای این سیستم، شش ترکیب موجود است (AB, AC, AD, BC, BD, CD) که در هر نمونه آزمون برای هر ترکیب باید چهار مقدار ۰۰، ۰۱، ۱۰ و ۱۱ وجود داشته باشد تا تمام حالات پوشش داده شود. تعداد کل پوشش‌ها بر اساس (۱) و (۲) محاسبه می‌شود، در صورتی که تمامی پارامترها مقدار یکسانی داشته باشند از (۱) و در غیر این صورت از (۲) استفاده می‌شود [۱۰].

$$Max\,Coverage=\binom{p}{t}v^t \qquad (۱)$$

1. Mixed Coverage Array
2. Variable Strength Covering Array

جدول ۱: گزینه‌های آزانس مسافرتی.

مشتری	آژانس مسافرتی	خط هوایی	هتل	بانک
رافع	رخش	زاگرس	پارسیان	سپه
اسفندیاری	پاسارگاد	کاسپین	گلستان	پاسارگاد
-	-	آسمان	-	ملت

جدول۲: مجموعه آزمون برای پیکربندی ($2^3,3^2$) CA.

مشتری	آژانس مسافرتی	خط هوایی	هتل	بانک
رافع	رخش	کاسپین	پارسیان	ملت
اسفندیاری	پاسارگاد	زاگرس	گلستان	ملت
رافع	پاسارگاد	آسمان	پارسیان	سپه
اسفندیاری	رخش	آسمان	گلستان	پاسارگاد
رافع	رخش	زاگرس	گلستان	سپه
اسفندیاری	پاسارگاد	کاسپین	پارسیان	پاسارگاد
رافع	رخش	زاگرس	پارسیان	پاسارگاد
اسفندیاری	پاسارگاد	کاسپین	گلستان	سپه
اسفندیاری	رخش	آسمان	گلستان	ملت

$$Max\,Coverage=\binom{p}{t}*|V_1||V_2|...|V_p| \qquad (۲)$$

برای درک بهتر مفهوم آرایه پوشش، از مثال آژانس مسافرتی در مرجع [۵] استفاده می‌کنیم. این سیستم دارای ۵ پارامتر مستقل از هم است: هتل، آژانس مسافرتی، کاربر، خط هوایی و بانک. هر یک از این پارامترها می‌توانند مقادیر مختلفی به خود بگیرند (جدول ۱). آزمون کامل سیستم برای تشخیص همه خطاها، نیازمند بررسی همه حالات ممکن سیستم است. در این مثال، تعداد حالت‌های ممکن برابر با ۷۲ حالت است. اما آزمون ترکیبی به جای بررسی همه حالت‌ها، تنها بخشی از آن‌ها را انتخاب می‌کند. فرض کنید در این سیستم، مشتریان با نام خانوادگی "اسفندیاری" مجاز به استفاده از خدمات بانکی بانک ملت نیستند. بنابراین، هر ترکیبی که شامل "ملت" برای بانک و "اسفندیاری" برای نام خانوادگی باشد، منجر به خطا می‌شود. در آزمون ترکیبی، هدف یافتن حداقل مجموعه‌ای از حالات است که همه خطاهای احتمالی سیستم را پوشش دهد. در مثال ما، این مجموعه باید حداقل یک بار ترکیبی مانند (ملت, -,-,-, اسفندیاری) را شامل شود تا خطای مربوط به مشتریان "اسفندیاری" و "بانک ملت" شناسایی شود. همانطور که در جدول ۲ مشاهده می‌شود برای آزمون آژانس مسافرتی با آزمون ترکیبی بجای ۷۲ نمونه آزمون فقط به ۹ نمونه آزمون نیاز است.

۲-۲ الگوریتم جستجوی گرانشی

در الگوریتم جستجوی گرانشی[3] (GSA) [۱۱]، بهینه‌یابی با استفاده از قوانین گرانشی و حرکت در یک سیستم مصنوعی با زمان گسسته انجام می‌شود. در این الگوریتم، به کمک قانون گرانش، هر جرم قادر است موقعیت و وضعیت سایر اجرام را از طریق قوانین جاذبه گرانشی درک کند. این نیروی گرانشی می‌تواند به عنوان یک ابزار برای تبادل اطلاعات

3. Gravitational Search Algorithm

۱. تعیین محیط سیستم و مقدار دهی اولیه
۲. جایابی اولیه اجرام
۳. ارزیابی اجرام
۴. به‌روزرسانی پارامترها
۵. محاسبه نیروی وارد بر هر جرم
۶. محاسبه شتاب و سرعت هر جرم
۷. به‌روزرسانی موقعیت اجرام
۸. در صورتی که شرط توقف برآورده نشده به مرحله ۳ برو در غیر این صورت بهترین جواب دیده‌شده تاکنون به خروجی را در خروجی چاپ و الگوریتم را متوقف کن

شکل ۲: مراحل الگوریتم GSA.

استفاده شود. بهینه‌سازی در این الگوریتم به صورت موقعیت‌ها در فضا تعریف شده و شباهت آن‌ها با سایر موقعیت‌ها به صورت یک فاصله قابل بیان است. میزان اجرام بر اساس تابع هدف تعیین می‌شوند و با استفاده از این تابع، بهینه‌سازی انجام می‌شود. مراحل انجام الگوریتم در شکل ۲ آورده شده است.

۲-۳ الگوریتم شاهین هریس

الگوریتم شاهین هریس[1] یک الگوریتم بهینه‌سازی است که توسط حیدری در سال ۲۰۱۹ [۱۲] ارائه شد. این الگوریتم از الگوی رفتار شکارگری و پرندگی پرها برای حل مسائل بهینه‌سازی الهام گرفته شده است. در الگوریتم شاهین هریس، یک جمعیت از افراد به عنوان جستجوگران (شاهین‌ها) و یک مقصد (شکار) وجود دارد. شکار بهره‌ای است که باید توسط شاهین‌ها پیدا شود. این الگوریتم از روش‌های تصادفی و تغییرات تصادفی برای جستجوی بهینه‌ی مسئله استفاده می‌کند. شاهین‌ها از قوانین جستجوگری و شکارگری الهام گرفته شده از طبیعت برای حرکت و جستجو در فضای جستجو استفاده می‌کنند. آن‌ها با استفاده از عوامل چندگانه محیط را بررسی کرده و به دنبال شکار حرکت می‌کنند.

الگوریتم شاهین هریس شامل مراحل مختلفی است که باید به ترتیب اجرا شوند. این مراحل شامل مراحل شکار و جستجوی شاهین‌ها در فضای جستجو می‌باشد. مراحل اصلی این الگوریتم در شکل ۳ آورده شده است.

۳- روش‌های پیشین

تحقیقات فراوانی برای حل آرایه‌های پوشش‌دهی وجود دارد که می‌توان آن‌ها را به دو گروه اصلی تقسیم کرد [۱۳] و [۱۴]:

۱ -روش‌های ریاضی.
۲ -روش‌های محاسباتی.

روش‌های ریاضی برای تولید آرایه‌های پوشش‌دهی بر اساس ساختارهای ترکیبی مانند آرایه‌های متعامد وابسته هستند. در مقابل، روش‌های محاسباتی معمولاً از راهکارهای حریصانه یا تکنیک‌های فراابتکاری برای بررسی فضای جستجو استفاده می‌کنند و آرایه‌های پوشش‌دهی تولید می‌کنند. در حالی که روش‌های ریاضی به تنظیمات خاص محدود می‌شوند، روش‌های محاسباتی می‌توانند به یک دامنه گسترده‌تر از سناریوها پاسخ دهند. روش‌های موجود در روش‌های ریاضی

۱. مسئله بهینه‌سازی و شرایط آن تعریف می‌شود. همچنین، پارامترهای مورد نیاز برای اجرای الگوریتم مشخص می‌شود.
۲. جمعیت اولیه از شاهین‌ها با موقعیت تصادفی در فضای جستجو ایجاد می‌شود.
۳. هر شاهین بر اساس تابع هدف، ارزیابی می‌شود و عملکرد آن محاسبه می‌شود.
۴. شاهین‌ها بر اساس قوانین جستجوگری جابجایی می‌شوند تا بهبود در عملکرد و یافتن شکار داشته باشند.
۵. با ترکیب و جابجایی شاهین‌ها، فرزندان جدید ایجاد می‌شوند که می‌توانند برای جستجو در فضای جستجو استفاده شوند.
۶. جمعیت جدید ایجاد شده توسط فرزندان و شاهین‌های موجود تکامل می‌یابد تا به تحسین در عملکرد و بهبود وضعیت برسد.
۷. مراحل ۴ تا ۶ تا زمانی که شرایط خاتمه مردودیت اجرای الگوریتم تکامل یافته کن برآورده شود تکرار می‌شوند.
۸. پس از بهبود عملکرد و یافتن شکار بهینه، الگوریتم به پایان می‌رسد و جواب بهینه به عنوان خروجی ارائه می‌شود.

شکل۳: مراحل الگوریتم شاهین هریس.

مانند خدمات آزمون ترکیبی و TConfig بر اساس آرایه‌های متعامد[2] (OA) می‌باشند و برای تنظیمات خاص موثر هستند. با این حال، راه‌حل‌های مبتنی بر آرایه‌های تعامل، محدودیت‌هایی در تولید آرایه‌های پوششی برای تنظیمات کوچک و خاص دارند که نیازمند به اعتماد بیشتر به روش‌های محاسباتی دارد. در روش‌های محاسباتی، رویکرد معمول شامل تولید تمام ترکیبات ممکن بر اساس مشخصات ورودی است و سپس ایجاد مورد آزمون برای پوشش این ترکیبات است. تفاوت اصلی بین روش‌های مختلف در روش‌های محاسباتی در نحوه ساخت موردهای آزمون وجود دارد. دو رویکرد اصلی برای تولید موردهای آزمون در روش‌های محاسباتی وجود دارد، که هر کدام مزایا و معایب متمایزی دارند [۱۵].

۱- رویکرد یک‌آزمون–در–زمان[3] (OTAT)، یک مورد آزمون به منظور پوشش یک ترکیب خاص از مقادیر ورودی تولید می‌شود. استراتژی‌های برجسته مبتنی بر این روش عبارتند: از AETG [۱۶]، PICT، TVG [۱۷]، Jenny [۱۸]، GTWay [۱۹] و ITCH [۲۰].

۲- رویکرد یک‌پارامتر–در–زمان[4] (OPAT) شامل تولید موردهای آزمون با متغیر کردن تعداد پارامترها در هر مرحله می‌باشد، برای پوشش ترکیب‌های مختلف مقادیر ورودی برای هر پارامتر انجام می‌شود تا زمانی که پوشش کامل صورت پذیرد در ادامه یک پارامتر دیگر اضافه می‌شود و مجددا نمونه آزمون‌هایی به سطر مورد نظر پیوست می‌شود تا پوشش ککامل شود و این روند تا تکمیل پارامترها و پوشش کامل ادامه دار خواهد بود. استراتژی‌ها بر اساس این روش شامل IPOG [۲۱]، IPOG-s [۲۲] و IPOG-D [۲۳] می‌باشند.

راهبرد اولیه برای تولید دنباله‌های آزمون با استفاده از رویکرد یک‌آزمون–در–زمان، AETG بود. این راهبرد یک تعدادی مورد آزمون را در هر چرخه انتخاب کرده و به طور حریصانه یک مورد آزمون به دنباله آزمون اضافه می‌کند تا پوشش تعامل‌ها را بیشینه کند. در ادامه، این راهبرد به عناوین mAETG و mAETG-sat در زمینه آرایه پوشش

1. Harris Hawks Optimization

2. Orthogonal Arrays
3. One Test at Time
4. One Parameter at Time

بهبود یافتند.

یک راه‌حل دیگر قابل توجهی که بر اساس رویکرد یک‌آزمون-در-زمان ارائه شد، استراتژی آزمون ترکیبی به نام PICT است. در این استراتژی یکی از مورد آزمون‌های تولید شده به صورت تصادفی انتخاب می‌شود. هرچند ویژگی انتخاب تصادفی PICT ممکن است کارآیی آن را نسبت به راهبردهای دیگر محدود کند، اما در ارائه عملکرد مطلوب و تولید مجموعه‌های آزمون تا توان ۱۰ موفق عمل می‌کند. به علاوه، PICT از تنظیمات آرایه‌های پوشش توان متغیر (VSCA) پشتیبانی می‌کند.

استراتژی GTWay یک استراتژی محاسباتی قوی است که بر پایه رویکرد یک‌آزمون-در-هر-زمان بنیان گذاشته شده است در این روش با بکار گیری از ساختار بیتی و افزایش سرعت، امکان تولید موردهای آزمون تا توان ۱۲ را فراهم می‌کند. GTWay در هر دو پارامتر عملکرد و کارایی نتایج مناسبی را تولید می‌کند.

یک دسته دیگر از روش‌های بنیادی بر الگوریتم‌های محاسباتی، الگوریتم‌های مبتنی بر هوش مصنوعی هستند. این الگوریتم‌ها بر اساس رویکرد یک‌آزمون-در-زمان بهبود یافته‌اند. با افزایش استفاده از مهندسی نرم‌افزار مبتنی بر جستجو در حل چالش‌های بهینه‌سازی نرم‌افزار، استفاده از الگوریتم‌های متامحور برای تولید آرایه‌های پوششی محبوبیت زیادی پیدا کرده امد. الگوریتم‌های متامحور معمولاً با یک مجموعه تصادفی از راه حل‌ها شروع می‌شوند و پس از تغییرات مکرر برای بهبود این راه‌حل‌های اولیه، در نهایت بهترین راه‌حل را انتخاب می‌کنند. این فرآیند تا زمان پوشش تمام ترکیب‌ها تکرار می‌شود. برخی از الگوریتم‌های متامحور مورد استفاده در این زمینه شامل SA، TS، GA، PSO، CS، H-B، ImpARO هستند.

در سال ۲۰۰۱، استاردم یک مجموعه آزمون با توان تعاملی ۲ با استفاده از سه الگوریتم شبیه‌سازی تبرید[1] (SA)، الگوریتم ژنتیک[2] (GA) و جست‌وجوی ممنوعه[3] (TS) ایجاد کرد [۲۴]. از این سه الگوریتم، SA عملکرد بهتری را نشان داد. سپس، کوهن SA را بهبود داد تا امکان پذیر کردن توان تعاملی ۳ را فراهم کند [۲۵]. با این حال، زمان اجرای این استراتژی به دلیل ساختار پیچیده و الگوریتم فشرده‌سازی که به کار می‌برد، بسیار پایین بود.

در استراتژی بهینه‌سازی ازدحام ذرّات[4] PSO، هر مورد آزمون مانند یک پرنده با موقعیت و سرعت خاص در فضای جستجو است. سرعت یک ذره معادل وزن مورد آزمون است، که سرعت بیشتر نشان‌دهنده وزن بالاتر است [۲۶]. مفهوم اساسی PSO بر این است که تمام ذرّات سعی می‌کنند که به سمت ذرّه‌ای حرکت کنند که به جواب بهینه نزدیک‌تر است. با گسترش PSO، استراتژی‌های مختلفی توسعه یافته‌اند، از جمله PSO [۱۴]، DPSO [۴] و BAPSO [۸]. این استراتژی‌ها تعامل‌های بالاتری (DPSO: تا $t = ۱۲$، BAPSO: تا $t = ۱۶$) را نسبت به PSO سنتی را پشتیبانی می‌کنند [۲۷]. با این حال، آن‌ها تعامل‌های توان متغیر یا محدودیت‌ها را تولید نمی‌کنند. علاوه بر این، این استراتژی‌ها محدودیت‌هایی مانند پیکربندی پارامترها و چالش‌های مرتبط با همگرایی زود هنگام دارند. به طور کلی، در حالی که PSO و مشتقات آن در تولید آرایه‌های پوششی با توان‌های افزایشی کارآمد هستند، اما ممکن است برای حالت‌هایی که تعامل‌های توان متغیر و محدودیت‌ها دخیل هستند، مناسب نباشند. توجه دقیق به تنظیمات پارامتر و مسائل مربوط به همگرایی در طول اجرای این استراتژی‌ها ضروری است. در یک مطالعه دیگر [۱۳]، الگوریتم فاخته برای تولید مورد آزمون تا توان شش استفاده شده است و نتایج کارآیی و عملکرد بهتری نسبت به PSTG [۱۴] ارائه داده است. با این حال، الگوریتم فاخته دارای قابلیت تولید VSCA نیست.

الگوریتم ژنتیک یک الگوریتم فرامکاشفه‌ای است که الهام‌گرفته از اصل انتخاب طبیعی و بقا گواهی به اصول داروین است. در آزمون نرم‌افزار، GA مورد آزمون‌ها را بر اساس تابع برازندگی (وزن) آن‌ها انتخاب می‌کند، از کراس آور و جهش برای تولید مورد آزمون‌های جدید استفاده می‌کند و جمعیت بعدی را از مورد آزمون‌های فعلی و تولید شده تا زمان تکمیل پوشش ادامه می‌یابد. یکی از پیشرفت‌های این حوزه استراتژی ژنتیک[5] (GS) [۵] است که یک روش نوآورانه را معرفی می‌کند که شامل اعمال مکرر تابع جدید کراس‌آور و جهش بر روی هر مورد آزمون است. نتایج مطالعات تجربی در انواع پیکربندی‌ها نشان داده است که GS در کارآیی بهتری را از راهبردهای جایگزین دارد. علاوه بر این، GS قابلیت تولید آرایه‌های پوششی با توان تا $t = ۲۰$ را دارد. با این حال، مهم است بدانیم که در حالی که GS می‌تواند تعامل‌های توان متغیر را مدیریت کند، اما از محدودیت‌ها در مراحل بهینه‌سازی پشتیبانی نمی‌کند.

در یک مطالعه دیگر [۶]، نسخه بهبود یافته‌ای از الگوریتم بهینه‌سازی خرگوش‌های مصنوعی[6] (ARO) برای تولید مجموعه‌های آزمون ترکیبی t-way در آزمون سیستم نرم‌افزاری بررسی شده است. این پژوهش تغییرات مهمی را در الگوریتم ARO معرفی کرده است که به بهبود نتایج منتهی شده است وآن را ImpARO نامیده اند. یافته‌های تجربی در انواع پیکربندی‌ها نشان می‌دهد که کارایی استراتژی ImpARO به طور قابل توجهی بهبود یافته است. ImpARO نه تنها عملکرد الگوریتم اصلی ARO را بیش‌تر کرده است، بلکه از استراتژی‌های ریاضی، حریصانه و فرامکاشفه‌ای مانند DPSO، GS، BAPSO، GALP، WOA و GSTG بهتر است. علاوه بر این، ImpARO سرعت همگرایی سریع‌تری نسبت به ARO از خود نشان می‌دهد که آن را به عنوان یک رویکرد مناسب برای ایجاد مجموعه‌های آزمون ترکیبی قرار می‌دهد.

٤- راهکار پیشنهادی

در این بخش، توضیح داده شده است که چگونه می‌توان از الگوریتم‌های جستجوی گرانشی و شاهین هریس برای تولید آرایه پوشش استفاده کرد. در ابتدا، الگوریتم‌های مذکور تعدادی نمونه آزمون تولید می‌کنند و سپس تابع برازندگی بهترین نمونه آزمون را انتخاب می‌کند. تابع برازندگی، معیاری است که تعداد ترکیباتی که هر نمونه آزمون می‌تواند پوشش دهد را اندازه‌گیری می‌کند. برای محاسبه برازندگی هر نمونه آزمون، ابتدا باید تمام ترکیبات را تولید کرده و سپس سازوکاری برای ارزیابی تابع برازندگی از ترکیبات تولید شده طراحی کرد. به طور خلاصه، این راهکار از دو مرحله اصلی تشکیل شده است: تولید ماتریس‌پوشش و تولید دنباله‌آزمون با استفاده از الگوریتم‌های جستجوی گرانشی و شاهین هریس.

1. Simulated Annealing
2. Genetic Algorithm
3. Tabu Search
4. Particle Swarm Optimization
5. Genetic Strategy
6. Artificial Rabbits Optimization

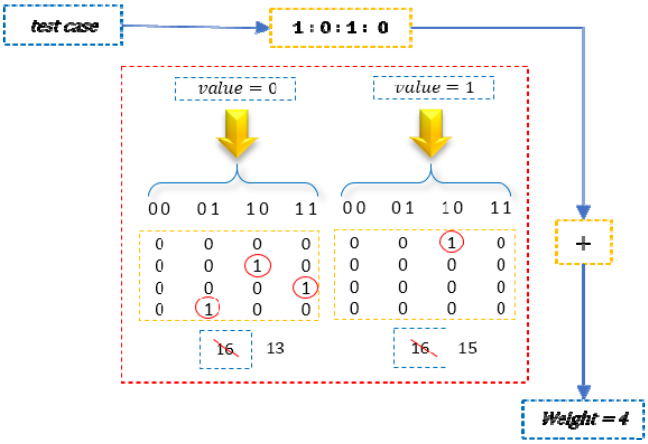
۴-۱ تولید ماتریس پوشش

در این ارائه راهکار جدید برای تولید آرایه‌پوشش با استفاده از الگوریتم‌های جستجوی گرانشی و شاهین هریس به‌گونه‌ای طراحی شده است که مزایای مختلفی دارد. ابتدا، ماتریس پوشش به گونه‌ای ایجاد شده است که نیاز به حافظه کمتری برای ذخیره‌سازی دارد، این باعث کاهش چشمگیری در میزان حافظه مصرفی می‌شود. علاوه بر این، مکانیسم جستجوی عناصر-t تایی در ماتریس به گونه‌ای طراحی شده که با سرعت بسیار بالا و با توانایی پشتیبانی تا $t=۲۰$ و پیچیدگی‌های با ابعاد بزرگ انجام می‌شود.

برای ذخیره‌سازی حالت‌های پوشش داده نشده، از ساختار بیتی استفاده شده در [۲۶] و [۲۷] بهره می‌گیریم. یعنی برای هر نمونه آزمون پوشش داده نشده، یک بیت تعیین می‌شود. بر خلاف روش‌های دیگر مانند CS و PSTG که برای v^t نمونه آزمون یک سطر در نظر گرفته می‌شود، در این روش برای تعداد ترکیبات $\binom{p}{t}$ سطر ایجاد می‌شود. هر ترکیب در سطر مربوطه ذخیره می‌شود که این کار از سوئیچ کردن بین ماتریس‌ها مستقل است و با داشتن اعداد مربوط به هر ترکیب می‌توان به راحتی به سلول مورد نظر دسترسی پیدا کرد.

در الگوریتم مورد بحث، ابتدا مشخصات آرایه پوشش $CA(N;t,p,v)$ ورودی گرفته می‌شود. سپس یک ماتریس با ابعاد $\binom{k}{t}\times(v^t+t)$ ساخته می‌شود. در این ماتریس، سطرها به تعداد $\binom{k}{t}$ در حالت اولیه قرار دارند که نشان دهنده تعداد ترکیبات t-تایی ممکن در بین k پارامتر است. در t ستون اول ماتریس، اندیس‌های ترکیبات مربوط به هر سطر ذخیره می‌شوند. همچنین، v^t ستون بعدی ماتریس نماینده مقادیر ممکن دهدهی معادل ترکیبات-t تایی است. تا این مرحله، کل عناصر t-تایی که باید توسط الگوریتم پوشش داده شوند تولید شده است. در این مرحله، درایه‌های ماتریس پوشش (CM) مقدار صفر می‌گیرند که به معنای عدم پوشش این عناصر است. به عبارت دیگر، در این مرحله هیچ ترکیب t-تایی مورد پوشش قرار نگرفته و به واسطه صفر شدن درایه‌های متناظر با آن‌ها در ماتریس CM، نشان داده می‌شود که این عناصر هنوز پوشش داده نشده است. این مرحله اولیه از الگوریتم مهم است زیرا مشخص می‌کند که کدام عناصر از آرایه پوشش باید پوشش داده شوند و کدام‌ها هنوز پوشش داده نشده‌اند.

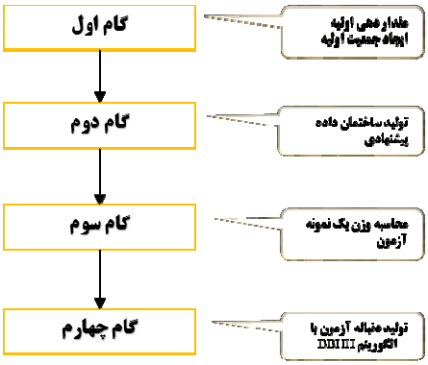
در این راهکار، برای دسترسی سریع‌تر به مقادیر پوشش داده نشده، ستون‌های ماتریس را به v دسته تقسیم می‌کنیم. این بیان‌گر این است که ستون‌های ماتریس متناظر با مقادیر مختلف دهدهی (که تعداد آن‌ها v است) به v دسته تقسیم شده‌اند. این تقسیم برای افزایش سرعت دسترسی و کارایی در جستجوی عناصر پوشش داده نشده بسیار مهم است. با این روش، وقتی که الگوریتم نیاز دارد به مقادیر پوشش داده نشده دسترسی پیدا کند، به جای جستجو در همه ستون‌ها، می‌توان به سرعت به یکی از v دسته از ستون‌ها دسترسی پیدا کرد. این تقسیم‌بندی بهبود قابل توجهی در سرعت عملکرد الگوریتم و کاهش تلاش‌های لازم برای جستجو و پردازش اطلاعات مربوطه ایجاد می‌کند. به این ترتیب، تقسیم ستون‌ها به v دسته، باعث افزایش کارایی و بهبود عملکرد الگوریتم در جستجوی و پردازش مقادیر پوشش داده نشده می‌شود و از منابع زمان و حافظه به بهترین شکل استفاده می‌کند.



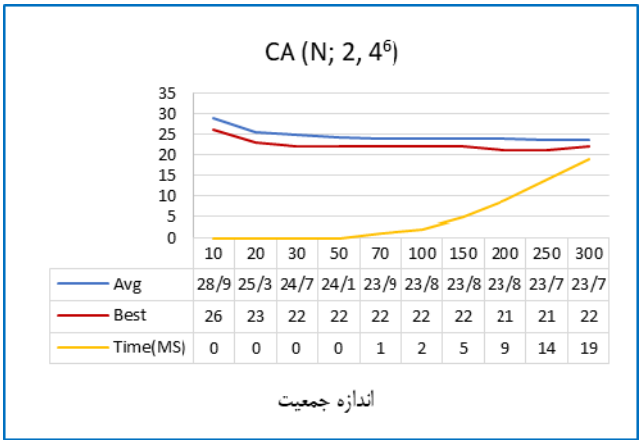
شکل ۴: محاسبه وزن یک نمونه آزمون.

در پیکربندی $CA(N;۳,۴^۲)$، که $v=۲$ است، اگر از رابطه‌ای مشابه (۱) استفاده کنیم، تعداد کل نمونه‌های آزمون در این مورد ۳۲ بیت است. در راهکار استفاده شده، ستون‌های ماتریس به دو بخش تقسیم شده و هر بخش به ۱۶ بیت تقسیم می‌شود (شکل ۴). این تقسیم‌بندی باعث می‌شود که در هر مرحله، از نصف ماتریس پوشش برای جستجو استفاده کنیم. این روش ذخیره‌سازی و تقسیم‌بندی ستون‌ها باعث می‌شود که جستجو و پردازش اطلاعات در ماتریس پوشش به شکل بهتری انجام شود. با این روش، تعداد عملیات مورد نیاز برای هر مرحله جستجو و محاسبه وزن یک نمونه آزمون کاهش می‌یابد، که به طور قابل توجهی به بهبود سرعت محاسبات و کارایی کلی راهکار کمک می‌کند. بنابراین، تقسیم ستون‌های ماتریس به دو بخش و هر بخش را به ۱۶ بیت باعث بهبود عملکرد و کارایی در جستجو و پردازش اطلاعات ماتریس پوشش می‌شود، به‌خصوص در مواردی که تعداد ستون‌ها و مقادیر v به این شکل مشخص شده باشد.

در این مثال، می‌خواهیم وزن نمونه آزمون (۱۰۱۰) را محاسبه کنیم. برای این کار، باید ترکیب‌های ترکیبات t-تایی مورد نظر (۳،۲،۱)، (۴،۲،۱)، (۴،۳،۱) و (۴،۳،۲) را بررسی کنیم که به ترتیب مقادیر (۰۱۰)، (۱۱۰)، (۱۰۰) و (۱۰۱) را نمایندگی می‌کنند. اولین مقادیر (۰۱۰) در بخش اول ($v=۰$) و دومین مقدار (۱) در بخش دوم ($v=۱$) جستجو می‌شود. اگر ترکیب (۳،۲،۱) یا (۰۱۰) در ماتریس پوشش با مقدار ۰ (عدم پوشش) وجود داشته باشد، آن را به مقدار ۱ (پوشش داده شده) تغییر می‌دهیم. سپس مقدار دوم (۱۱۰) در بخش اول و مقدار سوم (۰) در بخش دوم جستجو می‌شود. اگر ترکیب (۴،۲،۱) یا (۱۱۰) در ماتریس پوشش با مقدار ۰ (عدم پوشش) وجود داشته باشد، آن را به مقدار ۱ (پوشش داده شده) تغییر می‌دهیم. با ادامه همین روند برای ترکیب‌های (۴،۳،۱) و (۴،۳،۲) نیز به همین شکل عمل کنیم و در صورت عدم پوشش مقادیر ۰ (عدم پوشش) را به ۱ (پوشش داده شده) تغییر دهیم. با انجام این محاسبات و تغییرات در ماتریس پوشش، وزن نمونه آزمون (۱۰۱۰) محاسبه می‌شود. این روند ادامه پیدا می‌کند تا همه ترکیب‌های مورد نیاز برای پوشش دادن نمونه آزمون محاسبه و اعمال شود. مزیت این روش این است که برای مراحل بعدی الگوریتم سعی خواهیم کردن که تغییرات الگوریتم فرامکاشفه‌ای را در قسمتی از ساختمان داده‌ها که تعداد نمونه آزمون‌های پوشش داده نشده آن بیشتر را افزایش دهد که با این کار امید تولید نمونه آزمونی که بتواند تعداد بیشتری را پوشش دهد افزایش می‌یابد.



شکل ۵: مراحل راهکار پیشنهادی.



شکل ۶: ارزیابی تاثیر جمعیت بر پیکربندی $CA(N; 2,6,4)$.

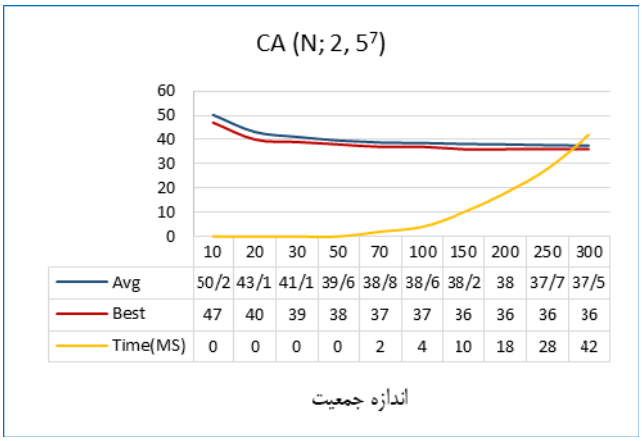
۴-۲ تولید با الگوریتم جستجوی گرانشی و شاهین هریس

در این بخش، با تغییراتی در الگوریتم جستجوی گرانشی، به دنبال تولید مجموعه آزمون بهینه‌ای هستیم. در این الگوریتم، هر نمونه آزمون به عنوان یک جسم در نظر گرفته می‌شود که شامل تعداد p پارامتر است. هر پارامتر مقادیری بین ۰ و $v-1$ را به خود اختصاص می‌دهد و وزن هر جسم بر اساس محاسبات خاصی، مانند توضیحات شکل ۵، محاسبه می‌شود. جسمی که بیشترین وزن را دارد، به عنوان $best$ تعیین می‌شود. در این الگوریتم، یکی از عوامل حیاتی انتخاب $Kbest$ است. در این پژوهش، به جای استفاده از استخراج $Kbest$ به شیوه معمولی، k نمونه آزمون جدید از $best$ با استفاده از الگوریتم شاهین هریس ایجاد می‌شود. این k نمونه که با استفاده از الگوریتم شاهین هرسین تولید می‌شود غالباً نمونه‌هایی با بیشترین پوشش هستند و این امر باعث تولید مجموعه آزمون بهینه می‌شود و همچنین سرعت تولید را افزایش می‌دهد. از این پس الگوریتم پیشنهادی را GSHH می‌نامیم.

۴-۳ انتخاب پارامترهای الگوریتم GSHH

در این پژوهش، یک الگوریتم هیبرید متشکل از GSA و HHO برای بهینه سازی آرایه پوشش استفاده شده است. ارزیابی دقیق پارامترهای این الگوریتم از اهمیت بالایی برخوردار است.

پارامترهای کلیدی HHO شامل تعداد نسل‌ها و تعداد شاهین‌ها ($popsize$) هستند و در الگوریتم GSA جرم اجسام، فاصله بین اجسام، ثابت گرانش و نپایر آن است. تاثیر تک تک این پارامترهاخارج از بحث این مقاله است. پارامتر اصلی تاثیرگذار بر عملکرد GSHH در این مسئله، اندازه جمعیت ($popsize$) است. با افزایش اندازه پیکربندی، اندازه جمعیت



شکل ۷: ارزیابی تاثیر جمعیت بر پیکربندی $CA(N; 2,7,5)$.

جدول ۳: پارامترهای راهکار پیشنهادی.

نام پارامتر	مقدار پارامتر
تعداد عوامل (Agents)	۱۰-۵۰
تعداد شاهین‌ها (Hawks)	۱۰-۵۰
حداکثر نیروی جاذبه (G0)	۱۰۰
ثابت جاذبه (Alpha)	۲۰-۳۰
نرخ افت جاذبه (Gravity Damping Rate)	۰/۹۹
پارامتر ضربه (Leap Strength)	۱/۵
حداکثر تکرارها	با توجه به پیکربندی بین ۱۰۰-۱۰۰۰۰۰۰

نیز باید افزایش یابد. برای ارزیابی تاثیر افزایش جمعیت بر بهبود کارایی و بهره‌وری، $CA(N; 2,6,4)$ را مورد بررسی قرار می‌دهیم. نتایج شکل ۶ نشان می دهد که بهترین مقدار برای اندازه جمعیت در این پیکربندی، ۱۰۰ است. با این حال، میانگین بهبود تا جمعیت ۳۰۰ ادامه دارد اما افزایش جمعیت منجر به افزایش زمان محاسبات می‌شود.

در ادامه، پیکربندی $CA(N; 2,7,5)$ را مورد بررسی قرار می‌دهیم. در این پیکربندی نیز تاثیر افزایش اندازه جمعیت (popsize) بر عملکرد الگوریتم را ارزیابی می‌کنیم. اندازه جمعیت در بازه ۱۰ تا ۳۰۰ تغییر می‌کند. نتایج در شکل ۷ نشان می‌دهد که افزایش اندازه جمعیت، زمان اجرای الگوریتم را به طور قابل توجهی افزایش می‌دهد. بهترین عملکرد الگوریتم در اندازه جمعیت ۱۵۰ حاصل شده است، اما به طور کلی با افزایش اندازه جمعیت، میانگین عملکرد بهبود می‌یابد.

۵- ارزیابی

در این قسمت، ما به بررسی نتایج حاصل از پیاده‌سازی استراتژی جستجوی گرانشی و شاهین هریس می‌پردازیم. دو معیار مهم که در آرایه پوشش مورد بررسی قرار می‌گیرند، دقت الگوریتم و سرعت الگوریتم هستند. دقت الگوریتم به میزان صحت یا دقت راهکار مورد بررسی اشاره دارد. به عبارت دیگر، هر چه تولید دنباله‌های آزمون کمتر باشد، دقت الگوریتم بالاتر خواهد بود. از سوی دیگر، سرعت الگوریتم نشان دهنده قدرت الگوریتم در تولید آرایه پوشش برای پیکربندی‌های با ابعاد بزرگتر است. این ابعاد شامل مقادیر بالای t، v و p هستند، که مهم‌ترین آن‌ها t می‌باشد. در این ارزیابی‌ها، توانمندی راهکار ما با اجرای الگوریتم بر روی چندین پیکربندی مختلف بررسی می‌شود و نتایج با روش‌های کارآمد دیگر و استراتژی‌های موجود مقایسه می‌شود. بهترین نتایج برای هر پیکربندی به صورت تکرار الگوریتم در ۱۰۰ بار مشخص شده است.

جدول ۴: اندازه دنباله آزمون (N) برای $(۳ \le p \le ۱۲)$ $CA(N; ۲, p, ۳)$.

p	GSHH N.Best	GSHH N.Avg	GALP N.Best	GS N.Best	BAPSO N.Best	DPSO N.Best	WOA N.Best	TConfig N.Best	PICT N.Best	IPOG N.Best
۳	۹	۹٫۷	۹	۹	۹	۹	۹	۱۰	۱۰	۹
٤	۹	۹٫۹	۹	۹	۱۰	۹	۹	۱۰	۱۳	۹
٥	۱۱	۱۱٫٦	۱۱	۱۱	۱۱	۱۱	۱۱	۱۴	۱۳	۱٥
٦	۱۳	۱۳٫٥	۱۳	۱۳	۱٤	۱۴	۱۴	۱٥	۱۴	۱٥
۷	۱٤	۱٤٫۷	۱٥	۱٤	۱٥	۱٥	۱٤	۱٥	۱٦	۱٥
۸	۱٥	۱٥٫۲	۱٥	۱٥	۱٥	۱٥	۱٥	۱۷	۱٦	۱٥
۹	۱٥	۱٥٫٤	۱۷	۱٥	۱٦	۱٥	۱٦	۱۷	۱۷	۱٥
۱۰	۱٥	۱٥٫۹	۱۸	۱٦	۱۷	۱٦	۱٦	۱۷	۱۸	۱٥
۱۱	۱٦	۱۷٫۳	۱۹	۱٦	۱۷	۱۷	۱۷	۲۰	۱۸	۱۷
۱۲	۱٦	۱۷٫۱	۱۹	۱٦	۱۷	۱٦	۱۷	۲۰	۱۹	۲۱

جدول ۵: اندازه دنباله آزمون (N) برای $(۳ \le v \le ۲, ۱۲ \le p \le ۷)$ $CA(N; ۲, p, v)$.

v, p	GSHH N.Best	GSHH N.Avg	GALP N.Best	GS N.Best	BAPSO N.Best	DPSO N.Best	WOA N.Best	TConfig N.Best	PICT N.Best	IPOG N.Best
۲, ۷	۱۲	۱۲٫٦	۱۲	۱۲	۱۲	۱٥	۱۲	۱٦	۱٥	۱٦
۲, ۸	۱۲	۱۲٫۹	۱۲	۱۴	۱۴	۱٦	۱۴	۱۸	۱۷	۱۸
۲, ۹	۱٥	۱٦٫۲	۱٦	۱٦	۱٥	۱٦	۱٥	۲۰	۱۷	۲۰
۲, ۱۰	۱٥	۱٦٫٥	۱٦	۱٦	۱٦	۱٦	۱٦	۲۰	۱۸	۲۰
۳, ٤	۲۷	۲۸٫۳	۲۷	۲۷	**۲۷۲۷**	۲۸	**۲۷۲۷**	۳۲	۳۴	۳۲
۳, ٥	۳۷	۳۸٫٦	۳۷	۳۸	۳۸	۴۱	۳۸	۴۰	۴۳	۴۱
۳, ٦	٤۳	٤٤٫۱	٤۰	۴۳	۴۳	۳۳	۴۲	۴۸	۴۸	۴۶
۳, ۷	٤۸	٥۰٫۱	٤۸	۴۹	٤۸	٤۸	٤۸	٥٥	٥۱	٥٥
۳, ۸	٥۲	٥٤٫۲	٥۲	٥۴	٥۲	٥۲	٥۳	٥۸	٥۹	٥۶
۳, ۹	٥٦	٥۷٫۸	٥٦	٥۸	٥۷	٥۶	٥۷	۶۴	۶۳	۶۳
۳, ۱۰	٥۹	٦۱٫٦	٥۹	۶۱	٥۹	٥۹	٥۹	۶۸	۶٥	۶۶
۳, ۱۱	٦۲	٦۳٫۹	٦۲	۶۳	۶۳	۶۳	۶۴	۷۲	۷۰	۷۰
۳, ۱۲	٦٦	٦۸٫۰	٦٥	۶۷	٦٥	٦٥	٦٥	۷۷	۷۲	۷۳

جدول ۶: اندازه دنباله آزمون (N) برای $CA(N; ۴, p, v)$ $(۳ \le v \le ۲, ۱۲ \le p \le ۷)$.

v, p	GSHH N.Best	GSHH N.Avg	GALP N.Best	GS N.Best	BAPSO N.Best	DPSO N.Best	WOA N.Best	TConfig N.Best	PICT N.Best	IPOG N.Best
۲, ۷	۲٤	۲٥٫۷	۲٤	۲۷	۲٥	۳۱	**۲٤۲٤**	۳٦	۳۲	۳٥
۲, ۸	۲۷	۲۸٫۳	۲۹	۳۰	۲۷	۳۲	۳۱	۳۸	۳٥	۳۹
۲, ۹	۲٥	۲٦٫۲	۲٥	۳۳	۲۹	۳۴	۲۶	۴۱	۴۱	۴۱
۲, ۱۰	۲٥	۲۹٫۳	۲٦	**۲٥**	**۲٥۲٥**	۳۴	۲۷	۴٥	۴۳	۴۶
۳, ٥	۸٥	۸۸٫٦	۸۸	۸۸	۸۲	**۸۱۸۱**	۹۰	۹۷	۱۰۰	۹۷
۳, ٦	۱۲۹	۱۳۱٫۳	۱۲۹	۱۲۹	۱۳۰	۱۳۱	**۱۲۹۱۲۹**	۱۴۰	۱۴۲	۱۴۱
۳, ۷	۱٥۰	۱٥۲٫۸	۱٥۲	۱٥۲	۱٥۲	۱٥۰	۱٥۴	۱۶۹	۱۶۸	۱۶۷
۳, ۸	۱۷۱	۱۷۲٫۹	۱۷۱	۱۷۱	۱۷۱	۱۷۱	۱۷۱	۱۸۷	۱۸۹	۱۹۲
۳, ۹	۱۸۸	۱۹۰٫۹	۱۸۹	۱۸۷	۱۹۰	۱۸۷	۱۸۷	۲۰۶	۲۱۱	۲۱۰
۳, ۱۰	۲۰٦	۲۱۰٫٤	۲۰٦	۲۰٦	۲۰۷	۲۰٦	۲۰٦	۲۲۱	۲۳۱	۲۳۳
۳, ۱۱	۲۲۱	۲۲٥٫٦	۲۲۱	۲۲۳	۲۲۱	۲۲۱	۲۲۱	۲۳۶	۲۴۹	۲٥۱
۳, ۱۲	۲۳٦	۲٤۰٫۳	۲۳۷	۲۳٦	۲۳٦	۲۳۷	۲۳۸	۲٥۲	۲۶۹	۲۷۲

محیط مورد استفاده برای این ارزیابی‌ها شامل سیستم عامل Windows ۷، پردازنده Core i۷QM ۲.۲۰ GHz و حافظه RAM ۶ GB می‌باشد. پیاده‌سازی و کدنویسی استراتژی جستجوی گرانشی و شاهین هریس به زبان جاوا انجام شده است. در جداول مورد استفاده ما، NA به معنای عدم وجود نتایج معتبر یا نتایجی که برای تولید آن‌ها بیش از یک روز زمان نیاز است، و NS به معنای عدم توانایی راهکار در تولید آرایه پوشش برای پیکربندی‌های مورد نظر استفاده شده است. اعدادی که بصورت برجسته نمایش داده شده‌اند، بهترین نتایج برای هر پیکربندی مربوطه را نشان می‌دهند. در جدول ۳ پارامترهای مربوط به الگوریتم GSHH آورده شده است.

در این بخش ارزیابی شامل چهار جدول (جداول ۴ تا ۷) انجام می‌شود ابتدا به صورت کلی این چهار جدول مورد ارزیابی قرار می‌گیرند و سپس هر یک از جداول نقد و بررسی جامع می‌شوند. به صورت کلی راهکار پیشنهادی در ۳۸ پیکربندی مختلف بهترین نتایج را تولید می‌کند به

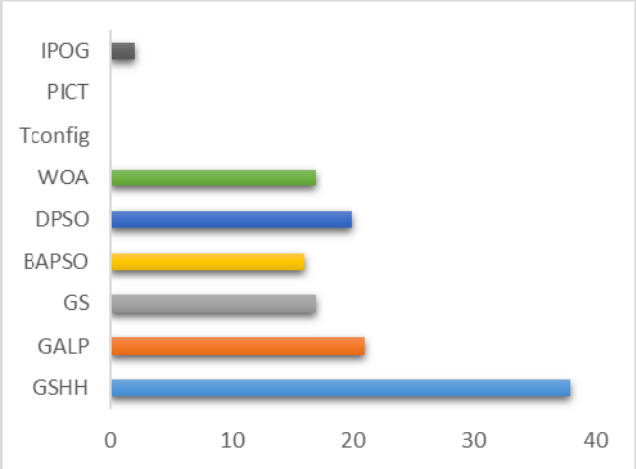
جدول ۷: اندازه دنباله آزمون (N) برای $CA(N;t,p,۳)$ $(t+۱\le p\le ۱۲, ۷\le v\le ۱۲)$.

t,p	GSHH N.Best	GSHH N.Avg	GALP N.Best	GS N.Best	BAPSO N.Best	DPSO N.Best	WOA N.Best	TConfig N.Best	PICT N.Best	IPOG N.Best
۸, ۷	۲۸۸۸	۲۸۹۸.۲	**۳۰۲۹**	۳۰۳۲	۳۰۲۹	**۲۲۴۱**	۳۰۳۱	> day	۳۱۴۳	NA
۹, ۷	٤٤۲٥	٤٤۳۱.۲	۴۴۲۵	۴۴۳۷	**٤٤۲٥**	۴۴۲۷	**٤٤۲٥**	> day	۴۶۱۸	NA
۱۰, ۷	٥٤٤۹	٥٤٥۸.٦	۵۴۵۹	۵۴۶۴	۵۴۷۲	۵۴۳۴	۵۴۷۴	> day	۵۸۸۴	NA
۱۱, ۷	٦٤٤۷	٦٤٦٤.۸	۶۴۶۸	۶۵۳۳	۶۵۱۵	**٦٤۱۳**	۶۵۳۱	> day	۷۱۱۶	NA
۱۲, ۷	۷٥٦۱	۷٥۸٤.۱	۷۵۹۳	۷۶۰۳	۷۶۱۴	> day	۷۶۱۰	> day	۸۳۱۴	NA
۹, ۸	۹۲٥۸	٥۲٦۷.۹	**۹۲۵۸**	۹۲۶۶	۹۲۷۳	۹۲۸۴	**۹۲۵۸**	> day	۹۷۶۳	NA
۱۰, ۸	۱۳۹۰۰	۱۳۹۱۱.۳	۱۳۹۰۳	۱۳۹۰۷	۱۳۹۱۲	۱۳۹۳۹	۱۳۹۱۲	> day	۱۴۵۹۹	NA
۱۱,۸	۱۷٥۷٤	۱۷٥۹۸.٦	۱۷۶۳۹	۱۷۷۹۲	۱۷۸۳۱	> day	۱۷۶۴۱	> day	۸۵۹,۱۸	NA
۱۲, ۸	۲۰٤٦٤	۲۰٤۹۷.٤	۲۰۹۶۳	۲۱۶۷۰	۲۱۶۳۱	> day	۲۱۳۱۰	> day	۲۳۱۱۲	NA
۱۰, ۹	۲۲٤۹۸	۲۲٥۱۰.۷	۲۸۳۱۲	۲۸۶۲۹	۲۷۸۱۳	**۲۱٤۳۳**	۲۸۳۱۹	> day	۳۰۱۸۱	NA
۱۱, ۹	٤۳٤۲۷	٤۳٤۳۹.۱	۴۳۵۴۳	۴۳۸۰۹	۴۳۹۳۱	> day	۴۳۵۹۲	> day	۴۵۵۲۱	NA
۱۲, ۹	٥٤۹٤۱	٥٤۹۸۹.۹	۵۶۲۱۹	۵۶۴۸۱	۵۷۹۳۱	> day	۵۵۹۳۱	> day	۵۹۹۶۶	NA
۱۱, ۱۰	۸۷٦۰۱	۸۷٦٥۷.۲	۸۷۷۱۲	۸۷۷۶۶	۸۸۳۵۱	> day	۸۷۷۲۱	> day	۹۲۴۳۵	NA
۱۲, ۱۰	۱۳٥٤٤۸	۱۳٥٤۸٦.۳	۱۳۵۹۶۲	۱۳۶۰۹۶	۱۴۰۳۱۵	> day	۱۳۶۸۴۱	> day	۱۴۱۹۹۰	NA
۱۲, ۱۱	۲٥۳۱۹٦	۲٥۳٤۳۲.٥	۲۶۷۰۸۵	۲۶۷۱۳۱	۲۷۴۱۰۲	> day	۲۵۹۳۴۱	> day	۲۷۸۹۹۳	NA

عبارتی دیگر در ۳۸ مورد دنباله آزمون کمینه تولید می‌شود و بعد از آن الگوریتم GALP با ۲۱ مورد در جایگاه دوم قرار دارد، سایر راهکارهای مبتنی بر الگوریتم‌های فرامکاشفه‌ای نیز نتایج مشابهی را با الگوریتم GALP دارند اما همانطور که در شکل ۸ نشان داده شده است راهکارهای مبتنی بر محاسبات توان رقابت با الگوریتم‌های هوش مصنوعی را ندارند و تنها الگوریتم IPOG توانسته است در ۲ مورد نتایج مطلوبی را ارائه دهد.

اولین ارزیابی مربوط به پیکربندی $CA(N;۲,p,۳)$ است که در آن p مقادیر بین ۳ تا ۱۲ را به خود اختصاص می‌دهد نتایج این ارزیابی در جدول ۴ آورده شده است همانطور که ملاحظه می شود در این ارزیابی راهکار پیشنهادی در تمامی ده پیکربندی بهترین خروجی را دارد، الگوریتم GS در ۹ مورد، DPSO در نه مورد و الگوریتم‌های GSTG، BAPSO، WOA در پنج مورد، الگوریتم IPOG در چهار مورد بهترین نتایج را تولید می‌کنندو دو الگوریتم TConfig و PICT در هیچ یک از پیکربندی‌ها نتیجه مناسبی را تولید نمی‌کنند. در این ارزیابی مشخص است که راهکارهای مبتنی بر الگوریتم‌های فرامکاشفه ای نتایج بسیار بهتری را نسبت به راهکارهای مبتنی بر محاسبات ارائه می‌دهند، همچنین نتایج نشان می‌دهد که راهکار پیشنهادی از سایر راهکارها قوی‌تر است. با توجه به اینکه دامنه فضای جستجو بسیار بزرگ است و با توجه به ماهیت جستجوی هیوریستیکی الگوریتم‌های فرامکاشفه‌ای یک الگوریتم ممکن است در یک پیکربندی نتایج مناسبی داشته باشد اما در پیکربندی مشابه نتیجه مطلوبی را نسبت به رقبای خود نداشته باشد.

دومین ارزیابی مربوط به پیکربندی $CA(N;۳,p,v)$ است که در آن p مقادیر بین ۷ تا ۱۲ و V مقادیر بین ۲ تا ۳ را به خود اختصاص می‌دهد نتایج این ارزیابی در جدول ۵ آورده شده است همانطور که ملاحظه می شود در این ارزیابی راهکار پیشنهادی در ده پیکربندی بهترین خروجی را دارد، الگوریتم GALP در ده مورد، الگوریتم GS در دو مورد، DPSO و WOA در شش مورد و الگوریتم BAPSO، در پنج مورد، و سه الگوریتم IPOG، TConfig و PICT در هیچ یک از پیکربندی‌ها نتیجه مناسبی را تولید نمی‌کنند.



شکل ۸: ارزیابی کلی راهکارپیشنهادی با سایر روش‌های موجود.

جدول ۶ نیز پیکربندی دیگر را با ۱۲ سطر ارائه داده است در این پیکربندی‌ها راهکار پیشنهادی در ۶ مورد، BAPSO، GALP و WOA در پنج مورد، GS و DPSO در ۴ مورد و راهکارهای مبتنی بر محاسبات در هیچ یک از موارد بهترین نتایج را تولید کرده اند. همانطوری که در این ارزیابی نمایان است راهکارهای مبتنی بر الگوریتم‌های فرامکاشفه‌ای نتایج به مراتب بهتری را نسب به راهکارهای محاسباتی ارایه داده اند که در این بین راهکار پیشنهادی سرآمد است.

درجدول ۷ نیز پیکربندی با ۱۵ سطر مختلف را مورد ارزیابی قرار می‌دهد در این ارزیابی راهکار پیشنهادی در تمامی ۱۵ مورد از رقبای خود بهتر است تنها الگوریتم GALP در ۲ مورد با راهکار پیشنهادی رقابت دارد، همانطور که ملاحظه شد هرچه راهکار پیشنهادی در پیکربندی‌های سنگین نتایج بسیار قوی‌تری را نسبت به سایر رقبای خود تولید می‌کند.

برای سنجش آماری راهکارهای موجود از روش آزمون ویلکاکسون[1] استفاده می‌شود. این ابزار که در SPSS نهفته شده است. برای داده‌های ورودی بین دو راهکار دو خروجی دارد: Test Statistics و ranks که برای دو راهکار i و j، ranks سه مقدار دارد: تعداد نمونه‌های $i=j$ و

1. Wilcoxon Signed Rank Sum

جدول ۸: نتایج آزمون ویلکاکسون برای جدول ۴.

	Ranks			Test Statistics	
	GSHH<	GSHH>	GSHH=	Z	A
GALP- GSHH	۵	۰	۵	−۰/۹۷	۰/۰۲۸
GS- GSHH	۱	۰	۹	−۱/۳۴	۰/۳۸۵
BAPSO- GSHH	۱۱	۰	۰	−۲/۹۳	۰/۰۰۳
DPSO- GSHH	۵	۰	۵	−۰/۹۷	۰/۰۲۸
WOA- GSHH	۵	۰	۵	−۰/۹۷	۰/۰۲۸
TCon- GSHH	۱۰	۰	۰	−۳/۲۰	۰/۰۰۱
PICT- GSHH	۱۰	۰	۰	−۳/۲۰	۰/۰۰۱
IPOG- GSHH	۸	۰	۲	−۲/۹۷	۰/۰۰۴

جدول ۹: نتایج آزمون ویلکاکسون برای جدول ۵.

	Ranks			Test Statistics	
	GSHH<	GSHH>	GSHH=	Z	A
GALP- GSHH	۱	۱	۱۰	−۱/۹۷	۰/۳۲۸
GS- GSHH	۲	۰	۱۰	−۱/۳۴	۰/۳۸۵
BAPSO- GSHH	۱۰	۰	۲	−۳/۱۳	۰/۰۰۹
DPSO- GSHH	۷	۰	۵	−۱/۲۴	۰/۰۳۸
WOA- GSHH	۷	۰	۵	−۱/۲۴	۰/۰۳۸
TCon- GSHH	۱۲	۰	۰	−۳/۰۵	۰/۰۰۲
PICT- GSHH	۱۲	۰	۰	−۳/۰۵	۰/۰۰۲
IPOG- GSHH	۱۲	۰	۰	−۳/۰۵	۰/۰۰۲

جدول ۱۰: نتایج آزمون ویلکاکسون برای جدول ۶

	Ranks			Test Statistics	
	GSHH<	GSHH>	GSHH=	Z	A
GALP- GSHH	۸	۰	۴	−۱/۹۷	۰/۳۲۸
GS- GSHH	۹	۰	۳	−۱/۳۴	۰/۳۸۵
BAPSO- GSHH	۸	۰	۴	−۳/۱۳	۰/۰۰۹
DPSO- GSHH	۴	۲	۴	−۱/۲۴	۰/۰۳۸
WOA- GSHH	۷	۰	۵	−۱/۲۴	۰/۰۳۸
TCon- GSHH	۱۲	۰	۰	−۳/۰۵	۰/۰۰۲
PICT- GSHH	۱۲	۰	۰	−۳/۰۵	۰/۰۰۲
IPOG- GSHH	۱۲	۰	۰	−۳/۰۵	۰/۰۰۲

جدول ۱۱: نتایج آزمون ویلکاکسون برای جدول ۷.

	Ranks			Test Statistics	
	GSHH<	GSHH>	GSHH=	Z	A
GALP- GSHH	۱۳	۰	۲	−۲/۸۲	۰/۰۰۲
GS- GSHH	۱۲	۰	۳	−۲/۲۵	۰/۰۰۳
BAPSO- GSHH	۱۴	۰	۱	−۲/۳۶	۰/۰۰۱
DPSO- GSHH	۱۲	۳	۰	−۲/۵۷	۰/۰۰۴
WOA- GSHH	۱۳	۲	۰	−۲/۳۴	۰/۰۰۳
TCon- GSHH	۱۵	۰	۰	−۳/۰۰	۰/۰۰۱
PICT- GSHH	۱۵	۰	۰	−۳/۰۰	۰/۰۰۱
IPOG- GSHH	۱۵	۰	۰	−۳/۰۰	۰/۰۰۱

$i < j, i > j$ و Test Statistics نیز دو خروجی دارد: z و جدول ۸: نتایج آزمون ویلکاکسون برای جدول ۴. Asymp. Sig. (۲-tailed). اهمیت z در این گفتار به چشم نمی‌خورد؛ مقدار Asymp. Sig. (۲-tailed) عددی بین ۰ و ۱ است که به زبان ساده اگر این مقدار کمتر از ۰/۰۵ باشد به معنای آن است که بین i و j اختلاف معنایی وجود دارد. نتایج آزمون ویلکاکسون در جدول‌های ۸ تا ۱۱، نشان داده شده است.

٦- نتیجه‌گیری

در این تحقیق، تولید آرایه پوشش به عنوان یکی از مسائل حیاتی و اساسی در زمینه آزمون‌های ترکیبی مورد بررسی قرار گرفته است. الگوریتم جستجوی گرانشی و شاهین هریس برای ساخت آرایه پوشش به کار گرفته شده است. با استفاده از یک روش طراحی خلاقانه برای بهینه‌سازی تابع برازندگی، فضای جستجو بهبود یافته و سرعت عمل این الگوریتم‌ها به طور چشم‌گیری افزایش یافته و این الگوریتم‌ها توانایی پشتیبانی از تا توان ۲۰ را دارند. علاوه بر افزایش سرعت، دقت الگوریتم نیز به شدت افزایش یافته و قادر به تولید دنباله‌های آزمون بهینه در اکثر شرایط می‌باشد. مقایسه این راهکار با روش‌های فرامکاشفه‌ای و الگوریتم‌های مبتنی بر محاسبات حاکی از این است که الگوریتم جستجوی گرانشی و شاهین هریس معمولاً بهترین نتایج را در اندازه آرایه تولید می‌دهد. از جهت سرعت عمل، این استراتژی نیز با دقت به شدت سریع‌تر از استراتژی‌های مبتنی بر هوش مصنوعی عمل می‌کند و توانایی تولید نتایج را در پیکربندی‌های بزرگ تا توان ۲۰ را دارد. در آینده، با بهبود روش‌های مختلف، قابلیت گسترش این تحقیق و ارائه سه روش پیشنهادی از جمله ترکیب الگوریتم جستجوی گرانشی و شاهین هریس با سایر الگوریتم‌های فرامکاشفه‌ای، استفاده از رویکرد یک–پارامتر–در– یک –زمان به جای یک–آزمون–در–یک–زمان، و ترکیب الگوریتم جستجوی گرانشی و شاهین هریس با روش‌های مبتنی بر محاسبات نیز مورد ارزیابی قرار خواهد گرفت.

مراجع

[1] Z. Abbasi, S. Esfandyari and V. Rafe, "Covering array generation using teaching learning base optimization algorithm," *Tabriz Journal of Electrical Engineering*, vol. 48, no. 1, pp. 161-171, May 2018.

[2] S. Esfandyari and V. Rafe, "A hybrid solution for software testing to minimum test suite generation using hill climbing and bat search algorithms," *Tabriz Journal of Electrical Engineering*, vol. 46, no. 3, pp. 25-35, Nov. 2016.

[3] E. Pira, V. Rafe and S. Esfandyari, "A three-phase approach to improve the functionality of t-way strategy," *Soft Computing*, vol. 28, pp. 415-435, 2023.

[4] H. Wu, C. Nie, F.-C. Kuo, H. Leung and C. J. Colbourn, "A discrete particle swarm optimization for covering array generation," *IEEE Trans. on Evolutionary Computation*, vol. 19, no. 4, pp. 575-591, Aug. 2015.

[5] 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, Feb. 2018.

[6] E. Pira and M. Khodizadeh-Nahari, "Combinatorial t-way test suite generation using an improved asexual reproduction optimization algorithm," *Applied Soft Computing*, vol. 150, Article ID: 111070, Jan. 2024.

[7] S. Esfandyari and V. Rafe, "Extracting combinatorial test parameters and their values using model checking and evolutionary algorithms," *Applied Soft Computing*, vol. 91, Article ID: 106219, Jun. 2020.

[۸] س. اسفندیاری و و. رافع، "استفاده از الگوریتم تودهٔ ذرات بهینه در تولید دنبالهٔ آزمون کمینه در آرایهٔ پوشش با قوهٔ ثابت،" مجله محاسبات نرم، جلد ۸، شماره ۲، صص. ۷۹-۶۶، پاییز و زمستان ۱۳۹۸.

[9] M. Farokhian, J. Shoaei and S. Esfandyari, "GWC: A tool for automatic web data extraction," in *Proc. 13th Symp. on Advances in Science and Technology: Sustainable Land of Computer and Information Technology*, pp.43831-43848, Mashhad, Iran, 18-18 Dec. 2018.

[10] E. Pira, V. Rafe and S. Esfandyari, "Minimum covering array generation using success-history and linear population size reduction based adaptive differential evolution algorithm," *Tabriz Journal of Electrical Engineering*, vol. 52, no. 2, pp. 77-89, Jul. 2022.
[11] E. Rashedi, H. Nezamabadi-pour and S. Saryazdi, "GSA: a gravitational search algorithm," *Information Sciences*, vol. 179, no. 13, pp. 2232-2248, 2009 .
[12] Ali Asghar Heidari, *et al.*, "Harris hawks optimization: Algorithm and applications," *Future Generation Computer Systems*, vol. 97, pp. 849-872, Aug. 2019.
[13] B. S. Ahmed, T. Sh. 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, p. 13-29, Oct. 2015.
[14] 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, no. 4, pp. 1330-1347, Apr. 2012.
[15] S. Esfandyari and V. Rafe, "GALP: a hybrid artificial intelligence algorithm for generating covering array," *Soft Computing*, vol. 25, pp. 7673-7689, 2021.
[16] D. M. Cohen, S. R. Dalal, M. L. Fredman and G. C. Patton, "The AETG system: an approach to testing based on combinatorial design," *IEEE Trans. on Software Engineering*, vol. 23, no. 7, pp. 437-444, Jul. 1997.
[17] J. Arshem, *Test Vector Generator*, accessed 2023. [Online]. Available: http://sourceforge.net/projects/tvg
[18] B. Jenkins, *Jenny: A Pairwise Testing Tool*, accessed 2023. [Online]. Available: http://burtleburtle.net/bob/math/jenny.html
[19] K. Z. Zamli, M. F. J. Klaib, M. I. Younis, N. A. M. Isa, and R. Abdullah, "Design and implementation of a t-way test data generation strategy with automated execution tool support," *Information Sciences*, vol. 181, no. 9, pp. 1741-1758, May 2011.
[20] A. Hartman, *IBM Intelligent Test Case Handler*, IBM Alphaworks, 2023. [Online]. Available: http://www.alphaworks.ibm.com/tech/whitch
[21] Y. Lei, R. Kacker, D. R. Kuhn, V. Okun and J. Lawrence, "IPOG: a general strategy for t-way software testing," in *Proc. 4th Annual IEEE Int. Conf. and Workshops on the Engineering of Computer-Based Systems*, pp. 549-556, Tucson, AZ, USA, 26-29 Mar 2007.
[22] A. Calvagna and A. Gargantini, "IPO-s: incremental generation of combinatorial interaction test data based on symmetries of covering arrays," *in Proc. Int. Conf. on Software Testing, Verification, and Validation Workshops*, pp. 2-10, Denver, CO, USA, 1-4 Apr. 2009.
[23] Y. Lei, R. Kacker, D. R. Kuhn, V. Okun and J. Lawrence, "IPOG/IPOG-D: efficient test generation for multi-way combinatorial testing, software testing," *Software Testing, Verification & Reliability*, vol. 18, no. 3, pp. 125-148, Sept. 2008.
[24] J. Stardom, *Metaheuristics and the Search for Covering and Packing Array*, M.Sc. Thesis, Simon Fraser University, 2001.
[25] M. B. Cohen, *Designing Test Suites for Software Interactions Testing*, Ph.D. Thesis, University of Auckland, Dept. of Computer Science, Auckland, 2004.
[26] L. Yousofvand, S. Soleimani, V. Rafe, and S. Esfandyari "Automatic program bug fixing by focusing on finding the shortest sequence of changes," *Artificial Intelligence Review*, vol. 57, Article ID: 39, 2024.
[27] S. Esfandyari and V. Rafe, "Correction to: GALP: a hybrid artificial intelligence algorithm for generating covering array," *Soft Computing*, vol. 25, p. page 7691, 2021.

سجاد اسفندیاری عضو هیأت علمی گروه مهندسی کامپیوتر دانشگاه ملایر است. وی در سال ۲۰۱۷ مدرک کارشناسی ارشد مهندسی کامپیوتر، گرایش نرم‌افزار، و در سال ۲۰۲۱ مدرک دکتری مهندسی کامپیوتر را از دانشگاه اراک دریافت کرده است. زمینه‌های تخصصی و پژوهشی وی شامل آزمون نرم‌افزار، بهینه‌سازی ترکیبیاتی، الگوریتم‌های فراابتکاری و هوش مصنوعی است.

داور گیوکی در سال ۲۰۰۶ مدرک کارشناسی مهندسی کامپیوتر را از دانشگاه سجاد مشهد، مشهد، ایران، و در سال ۲۰۰۸ مدرک کارشناسی ارشد علوم کامپیوتر را از دانشگاه تهران، ایران، دریافت کرد. وی همچنین در سال ۲۰۱۷ موفق به اخذ مدرک دکتری مهندسی کامپیوتر از پژوهشگاه علوم و فناوری اطلاعات ایران (ایرانداک)، تهران، ایران، شد. وی در حال حاضر دانشیار گروه مهندسی کامپیوتر دانشگاه ملایر است. زمینه‌های تحقیقاتی وی شامل هوش محاسباتی، بینایی ماشین و یادگیری عمیق است.

لیلا یوسفوند عضو هیأت علمی دانشگاه لرستان است. وی مدرک کارشناسی ارشد خود را از دانشگاه رازی و مدرک دکتری خود را از دانشگاه اراک دریافت کرده است. حوزه‌های تخصصی و پژوهشی وی عمدتاً بر وارسی مدل، روش‌های صوری در مهندسی نرم‌افزار، تحلیل و بررسی صحت سیستم‌های نرم‌افزاری و سخت‌افزاری، و تأیید خودکار ویژگی‌های سیستم‌ها متمرکز است. از دیگر زمینه‌های پژوهشی وی می‌توان به استفاده از روش‌های صوری و تکنیک‌های خودکار برای شناسایی خطاها، بررسی رفتار سیستم‌های پیچیده و افزایش قابلیت اطمینان و صحت سیستم‌های نرم‌افزاری اشاره کرد.

عین‌الله پیرا عضو هیأت علمی دانشگاه شهید مدنی آذربایجان است. وی مدرک کارشناسی ارشد خود را از دانشگاه صنعتی شریف و مدرک دکتری خود را از دانشگاه اراک دریافت کرده است. زمینه‌های تخصصی و پژوهشی وی در حوزه مهندسی کامپیوتر و علوم کامپیوتر قرار دارد و از زمینه‌های مورد علاقه پژوهشی ایشان می‌توان به هوش مصنوعی، الگوریتم‌های هوشمند، بهینه‌سازی محاسباتی و روش‌های محاسباتی برای حل مسائل پیچیده اشاره کرد. فعالیت‌های پژوهشی وی بر طراحی، توسعه و ارزیابی روش‌های محاسباتی و هوشمند و استفاده از آن‌ها در حل مسائل مختلف مهندسی و محاسباتی متمرکز است.