ISSN:2821-1553 htpps://www.jasd.khadu.ir

فصل‌نامه علمی ((دفاع هوافضایی))
دوره ۲، شماره ۴ ، اسفند ۱۴۰۲

مقاله پژوهشی

رویکرد نوین در بهبود پدافند: الگوریتم‌های فراابتکاری برای ساختاردهی بهینه آرایه پوششی و تولید مجموعه‌ای کارآمد از آزمون‌ها

سجاد اسفندیاری[۱]،داور گیوکی[۱]، محمد فرخزادی[۲]

۱. استادیار، گروه مهندسی کامپیوتر، دانشکده فنی مهندسی، دانشگاه ملایر، ملایر، ایران
۲. گروه ریاضی، دانشگاه فنی و حرفه ای تهران

اطلاعات مقاله

تاریخ پذیرش: ۱۴۰۳/۰۲/۱۹

تاریخ دریافت: ۱۴۰۲/۰۸/۰۴

کلمات کلیدی:

الگوریتم‌های فراابتکاری، آزمون نرم‌افزار، آرایه پوششی.

نویسنده مسئول:
سجاد اسفندیاری

ایمیل:
esfandyari@malayeru.ac.ir

چکیده

عملیات آزمون نرم‌افزار، به‌عنوان یکی از روش‌های اساسی برای ارزیابی کیفیت محصولات نرم‌افزاری، از مراحل مختلف توسعه و نگهداری نرم‌افزار بهره می‌برد. با افزایش پیچیدگی و گستردگی نرم‌افزارها، نیازمندی به روش‌های آزمون کارآمد و موثر با تعادل مناسب بین کاهش زمان و افزایش دقت احساس می‌شود. در این مطالعه، روش‌های خودکار آزمون نرم‌افزار با استفاده از الگوریتم‌های بهینه‌سازی مبتنی بر جغرافیای زیستی و ترکیب آن با الگوریتم خفاش مورد بررسی قرار گرفته و آرایه پوششی با کارایی بالا تولید شده است. این نوآوری‌ها، به علاوه استفاده از ساختارهای داده بهینه، نه تنها به مدیریت پیچیدگی نرم‌افزار کمک کرده‌اند، بلکه نیازمندی‌های موفقیت پروژه‌های نرم‌افزاری را نیز مدنظر قرار داده‌اند. این پژوهش به منظور ارتقای کیفیت محصولات نرم‌افزاری، توصیه می‌شود و به تولید آرایه‌های پوشش با بهره‌وری بالا و افزایش کارایی آزمون‌های نرم‌افزار می‌پردازد.

استناد به مقاله: سجاد اسفندیاری، داور گیوکی، محمد فرخزادی، رویکرد نوین در بهبود پدافند: الگوریتم‌های فراابتکاری برای ساختاردهی بهینه آرایه پوششی و تولید مجموعه‌ای کارآمد از آزمون‌ها، مجله علمی پژوهشی دفاع هوافضایی دوره۲، شماره۴ اسفند۱۴۰۲.

ISSN:2821-1553 htpps://www.jasd.khadu.ir

Journal of Aerospace Defense
Vol. 2, No, 4, 2024

Research Paper

A Novel Approach to Enhancing Defense: Metaheuristic Algorithms for Optimal Structuring of Covering Arrays and Efficient Test Suite Generation

Sajad Esfandyari[1], Davar Giveki[1], Mohammad Farrokhzadi[2]

1. Department of Computer Engineering, Faculty of Engineering, Malayer University, Malayer, Iran
2. Department of Mathmatics, Technical and Vocational University (TVU), Tehran, Iran

Article Information

Accepted: 2024/05/08

Recceived:2023/10/26

Keywords:

Metaheuristic algorithm, software testing, covering array.

Abstract

Software testing operations, as one of the fundamental methods for evaluating the quality of software products, are employed throughout various stages of software development and maintenance. With the increasing complexity and ubiquity of software, the need for performance and efficiency testing methods with a suitable balance between reducing time and increasing accuracy is felt.

In this study, automated software testing methods using Biogeography-Based Optimization algorithms, coupled with the bat algorithm, have been investigated, resulting in the production of a high-performance covering array. These innovations, along with the utilization of optimized data structures, have not only aided in managing software complexity but also addressed the success requirements of software projects. This research is recommended for enhancing the quality of software products, focusing on producing covering arrays with high efficiency and improving the performance of software testing.

Corresponding anuthor:
Sajad Esfandyari
Email:
esfandyari@malayeru.ac.ir

HOW TO CITE: Sajad Esfandyari, Davar Giveki, Mohammad Farrokhzadi, A Novel Approach to Enhancing Defense: Metaheuristic Algorithms for Optimal Structuring of Covering Arrays and Efficient Test Suite Generation, Journal of Aerospace Defense, Vol. 2, No4. 2024.

۱-مقدمه

سامانه‌های نرم‌افزاری از قطعات مختلفی تشکیل شده‌اند که هر یک وظایف متفاوتی را انجام می‌دهند. این قطعات باید با پیاده‌سازی و تنظیماتی که با مجموعه خاصی از برنامه‌ها هماهنگ شده باشند، عمل کنند. این امر باعث می‌شود که هماهنگی و تعامل بین قطعات به حداکثر برسد. با توجه به انواع مختلف پیکربندی‌ها برای هر قطعه، کارهای تضمین کیفیت با فشار قابل توجهی مواجه شوند، به‌ویژه زمانی که هزینه‌های آزمون و فشار زمانی اهمیت زیادی دارند. از سوی دیگر، آزمون تمام حالت‌ها و ترکیب‌های ممکن به دلیل تعداد بسیار زیاد حالات، غیرممکن است. بنابراین، استفاده از یک روش دقیق برای نمونه‌برداری و پوشش تعامل پارامترهای آزمون، به‌ویژه برای بهبود کارایی و صحت، ضروری است [1]. نرم‌افزارها به عنوان یکی از ارکان اساسی فناوری اطلاعات و ارتباطات، به دستاوردهای مهمی در حوزه توسعه و پیشرفت دست یافته‌اند. از آنجایی که اهمیت نقش این نرم‌افزارها در زندگی روزمره به سرعت در حال افزایش است، امری که باعث شده تا بررسی بهره‌وری و کیفیت آن‌ها به عنوان یکی از مسائل حیاتی مورد توجه قرار گیرد. یکی از راهکارهایی که برای حفظ و بهبود کیفیت نرم‌افزارها ارائه می‌شود، انجام عملیات آزمون نرم‌افزار است. این عملیات نقش اساسی در ارزیابی عملکرد و کارایی نرم‌افزار، شناسایی خطاها و اشکالات ممکن، و بهبود کیفیت و قابلیت اطمینان نرم‌افزارها دارد. با توجه به پیچیدگی و گستردگی بیشترین نرم‌افزارها، نیاز به روش‌های آزمون هوشمند و کارآمد احساس می‌شود. از این رو، اصلاح و بهبود بهره‌وری و کارایی آزمون‌های نرم‌افزار از اهمیت چشمگیری برخوردار است. به همین دلیل، روش‌های خودکار آزمون نرم‌افزار برای این هدف به‌طور جدی مورد بررسی و ارزیابی قرار می‌گیرند [2].

برای احقاق این هدف از آرایه پوشش که به‌عنوان یک ساختار ترکیباتی مورد استفاده قرار می‌گیرد، تولید آرایه پوشش کمینه که یک مسئله بهینه‌سازی NP-سخت است، مورد نیاز است [3]. این حوزه تحقیقاتی یکی از پرطرفدارترین حوزه‌های پژوهش در زمینه تست‌های ترکیبی می‌باشد. آرایه پوشش، یک روش مهم در آزمون نرم‌افزار است که برای افزایش کیفیت و کارایی آزمون‌ها استفاده می‌شود. این روش بر اساس مجموعه‌ای از داده‌های ورودی (موردهای آزمون) تولید می‌شود که پوشش کاملی از کد منبع خارجی نرم‌افزار را فراهم می‌کند. آرایه پوشش به وسیله انتخاب و ترکیب مثال‌های آزمون فراهم می‌آید که با استفاده از الگوریتم‌ها و ساختارهای مناسب، تضمین می‌کند که تمامی مسیرهای ممکن در کد نرم‌افزار بر اساس یک معیار خاص، اجرا شوند و همچنین خطاها و اشکالات ممکن در

نرم‌افزار شناسایی شوند. این روش می‌تواند به بهره‌وری و کارایی آزمون نرم‌افزار کمک زیادی نماید و در بهبود کیفیت نرم‌افزارها تأثیرگذار باشد [4]. در این پژوهش، یک روش نوآورانه مبتنی بر استفاده از الگوریتم بهینه‌سازی مبتنی بر جغرافیای زیستی [5] برای تولید موارد آزمون مورد بررسی قرار گرفته است. برای توجیه استفاده از این الگوریتم نیز می‌توان به مقاله [6] استناد کرد که در این مقاله الگوریتم جغرافیای زیستی را یکی مهمترین الگوریتم‌های فراابتکاری معرفی نموده است که هرگز در حوزه آرایه پوشش استفاده نشده است. همچنین، از یک ساختمان داده بهینه به‌منظور بهبود کارایی دنباله آزمون استفاده شده است. این روش‌ها نه تنها به بهبود بهره‌وری و کارایی آزمون‌های نرم‌افزار کمک می‌کنند، بلکه نیازهای موفقیت پروژه‌های نرم‌افزاری را نیز در نظر می‌گیرند و به افزایش کارایی و بهره‌وری آزمون‌ها منجر می‌شوند. از آن‌جا که اهمیت بهبود کیفیت و افزایش بهره‌وری نرم‌افزارها به عنوان یکی از چالش‌های اساسی صنعت نرم‌افزار ادامه دارد، این پژوهش به تولید آرایه پوشش با بهره‌وری بالا و افزایش کارایی آزمون‌های نرم‌افزار می‌پردازد و استفاده از این روش‌های پیشرفته را برای ارتقای کیفیت و بهره‌وری محصولات نرم‌افزاری توصیه می‌نماید.

۲-آرایه پوشش (CA[1])

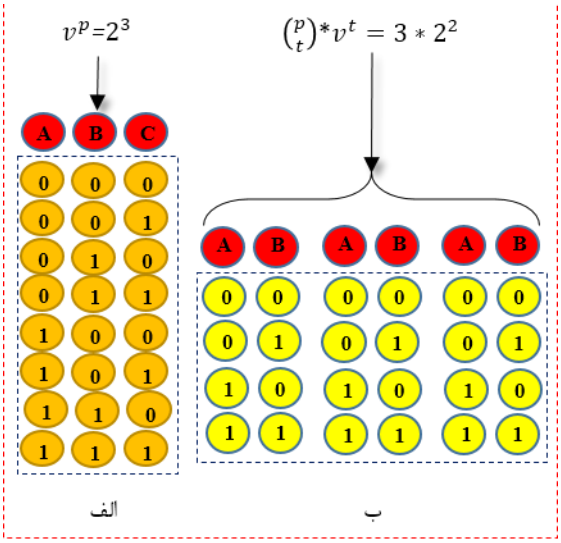
پیچیده شدن رفتار نرم‌افزارها و محیط اجرای آنها، باعث افزایش مجموعه شرایطی شده است که بر رفتار سیستم تأثیر می‌گذارد. از این رو، آزمون جامع سیستم به‌درستی امکان‌پذیر نیست. یک روش ریاضی برای کاهش تعداد نمونه‌های آزمون، استفاده از آرایه پوشش می‌باشد. در روش آزمون ترکیبی، آرایه پوشش با استفاده از یک جدولی که شامل سطرها و ستون‌هایی است که موارد آزمون را نشان می‌دهد، نمایش داده می‌شود. در هر سطر این جدول، یک نمونه آزمون و در ستون‌های آن، پارامترهای نرم‌افزار تحت آزمون قرار دارند. به‌طور معمول، آرایه پوشش شامل چهار پارامتر (N، t، p، v) است که به‌صورت CA(N; t; v^p) یا CA(N; t; p; v) نمایش داده می‌شود [6, 7]. در این رابطه، p تعداد پارامترهای ورودی و v تعداد مقادیری است که هر پارامتر می‌تواند به خود اختصاص دهد و t قدرت تعامل را نشان می‌دهد. با توجه به ویژگی‌هایی که ذکر شد، آرایه پوشش یک آرایه $N\times P$است که P تعداد پارامترهای سیستم و N تعداد موارد آزمون را نشان می‌دهد. این آرایه ویژگی‌های زیر را داراست [8, 9]:

[1] Covering Array

۱. هر ستون **i** که $1 \leq i \leq p$ شامل عضوهایی از مجموعه v_i است که تعداد این عناصر برابر با $|v_i| = v_i$ است.

۲. تعداد سطرهای هر زیر-آرایه N×t از آرایه پوشش، تمام ترکیبات $\times |v_{P_2}| \times |v_{Pt}| \times \ldots . \times |v_{p_1}|$ از t ستون را حداقل یک‌بار پوشش دهد، به این معنا که تمام مقادیر ممکن برای هر پارامتر در این ترکیبات حضور داشته باشد.

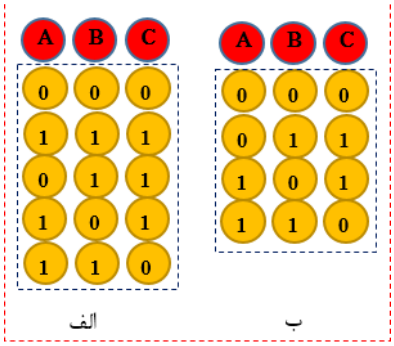
وقتی تعداد سطرهای آرایه پوشش به حداقل خود برسد، این آرایه به آرایه پوشش بهینه یا بهینه‌سازی شده معروف است که بهترین کارایی را در انجام آزمون‌های مختلف ارائه می‌دهد. برای تشریح کاملتر آرایه پوشش یک سیستم را در نظر بگیرید که این سیستم دارای سه پارامتر (A, B and C) باشد و فرض می کنیم هر پارامتر نیز دو مقدار ۰ و ۱ را به خود اختصاص می‌دهد. در حالت عادی برای آزمون این سیستم به هشت نمونه آزمون ($v^p = 2^3$) نیاز داریم (شکل ۱ (الف)). حال می خواهیم از این هشت نمونه آزمون تعدادی را انتخاب کنیم که نیاز آرایه پوشش را تامین کند. در این مثال از آنجایی که تعداد پارامترها فقط ۳ عدد است لذا قوه تعامل فقط می‌تواند ۲ باشد ($t = 2$). لذا در این روش بجای ترکیب کل پارامترها ترکیب دوتایی آن‌ها را در نظر می‌گیریم در مثال فوق ترکیب دوتایی شامل: AB، AC و BC است. به عبارتی باید نمونه آزمون‌هایی را انتخاب کنیم که برای هر ترکیب چهار حالت ۰۰، ۰۱، ۱۰ و ۱۱ را پوشش دهد. یعنی در حال کلی به ۱۲ ($Max_{Covering} = v^t * \binom{p}{t}$) نمونه برای پوشش لازم داریم (شکل ۱ (ب)).



شکل ۱: تعداد نمونه آزمون اولیه سیستم و تعداد حالات پوشش

حال قصد داریم با ساختار آرایه پوشش زیر مجموعه کمینه از هشت نمونه آزمون را انتخاب کنیم به عبارت دیگر برای پیکربندی CA (N; t, p, v) مقدار N را بدست آوریم (که در مثال فوق الذکر داریم: CA (N; 2, 3, 2) هر چه مقدار N کوچکتر باشد راهکار قدرت بیشتری را دارد. حال به انتخاب نمونه آزمون ها به صورت دستی خواهیم پرداخت، ابتدا نمونه آزمون ۰۰۰ را انتخاب می‌کنیم با انتخاب این نمونه آزمون سه وضعیت ۰۰ در AB، AB و BC را پوشش می دهیم (شکل۱ (ب)). در گام دوم ۱۱۱ را انتخاب و مجدد سه وضعیت ۱۱ را در سه ترکیب مختلف پوشش داده می شود.

در مرحله سوم نمونه آزمون ۰۱۱ را انتخاب می‌کنیم در این وضیعت فقط دو نمونه آزمون (AB = 01 و AC = 01) را پوشش می‌دهد از آنجایی که BC = 11 قبلا پوشش داده شده است دیگر نیازی به پوشش ندارد. گام بعدی نمونه آزمون ۱۰۱ را انتخاب می‌کنیم که این نمونه آزمون نیز دو حالت را پوشش می‌دهد و در گام آخر ۱۱۰ را انتخاب می کنیم که این نمونه آزمون نیز دو حالت را پوشش می‌دهد و در مجموع ۱۲ حالت پوشش داده می شود لذا با این ساختار توانستیم تعداد نمونه آزمون را به پنج نمونه کاهش دهیم (شکل۲ (الف)). اگر ترتیب انتخاب نمونه آزمون‌ها را تغییر دهیم ممکن است تعداد نمونه آزمون‌های کلی کاهش یابد برای مثال اگر ترتیب انتخاب نمونه آزمون‌هابه شکل ۰۰۰، ۰۱۱، ۱۰۱ و ۱۱۰ باشد (شکل۲ (ب)) ما می توانیم پوشش را فقط با ۴ نمونه آزمون تکمیل کنیم (دلیل این کاهش چه می‌تواند باشد؟ در ادامه بحث خواهد شد.).



شکل۲: کاهش نمونه آزمون با آرایه پوشش

۳-الگوریتم بهینه‌سازی مبتنی بر جغرافیای زیستی

الگوریتم BBO[۲] [5] یک الگوریتم تکاملی بر پایه جمعیت است که از پدیده مهاجرت حیوانات و پرندگان در زیستگاه‌ها الهام گرفته است. این الگوریتم بر اساس مطالعه توزیع

[2] Biogeography-Based Optimization

جغرافیایی زیستگاه‌ها توسعه یافته است. زیستگاه‌هایی که برای گونه‌های جانوری به عنوان محل اسکان مناسب شناخته می‌شوند، شاخص صلاحیت (HSI)[3] بالایی دارند. ویژگی‌هایی که HSI را تعیین می‌کنند شامل عواملی مانند بارندگی، تنوع گیاهی، ویژگی‌های نقشه‌برداری، خصوصیات خاکی منطقه و دما هستند. زیستگاه‌های با HSI بالا دارای تعداد زیادی گونه‌ها هستند که به زیستگاه‌های مجاور مهاجرت می‌کنند. زیستگاه‌های با HSI بالا دارای نرخ مهاجرت به داخل کمی هستند، زیرا پر شده و نمی‌توانند گونه‌های جدیدی را جذب کنند. از سوی دیگر، جزایر با HSI پایین دارای نرخ مهاجرت به داخل بالا هستند، زیرا جای خالی و برای مهاجرت به آنها فرصت مناسبی فراهم است. مهاجرت به داخل یا خارج گونه‌های جدید به مکان‌هایی با HSI پایین ممکن است باعث افزایش HSI این منطقه شود، زیرا تنوع جغرافیایی مکان متناسب با آن است.

استفاده از جغرافیای زیستی برای بهینه‌سازی الگوریتم BBO بر اساس استفاده از فرآیندی طبیعی برای حل مسائل بهینه‌سازی اتفاق افتاده است. عملگرهای مهاجرت و جهش در این الگوریتم نیز مشابه سایر الگوریتم‌های تکاملی همچون GA مطرح هستند و باعث ایجاد تغییرات مطلوب در جمعیت نسل‌ها می‌شوند. به واسطه مزایای سادگی، انعطاف‌پذیری، و کارایی محاسباتی، الگوریتم BBO یکی از سریعترین الگوریتم‌های الهام‌گرفته از طبیعت برای حل مسائل بهینه‌سازی محسوب می‌شود.

۳-۱کارهای پیشین

در این مطالعه، ارائه روش‌های محاسباتی در زمینه آزمون نرم‌افزارها مورد بررسی قرار گرفته است. به طور خلاصه، ابتدا ترکیب‌های ممکن بر اساس ورودی‌های مشخص ایجاد شده و سپس نمونه‌های آزمون برای پوشش این ترکیب‌ها تولید می‌شوند. این مطالعه به مقایسه دو رویکرد اصلی در ایجاد نمونه‌های آزمون، شامل روش یک-پارامتر-در-یک-زمان و روش یک-آزمون-در-یک-زمان، می‌پردازد [10]. استراتژی‌هایی از جمله IPOG-s، [11] IPOG، IPOG-D[12] ، ParaOrder، IPOG-F که بر اساس این روش تکامل یافته‌اند نیز مورد بررسی قرار گرفته‌اند. علاوه بر این، استراتژی‌هایی مانند ITCH [13]، Jenny [14]، TVG [15]، GTWay [16] و AETG [17] ایجاد شده‌اند. که در ادامه به بررسی مزایا و معایب این روش‌ها و استراتژی‌ها خواهیم پرداخت.

روش AETG، اولین استراتژی استفاده شده از رویکرد یک-آزمون-در-یک-زمان بوده است.

[3] habitat suitability index

در این روش، آزمونی که می‌تواند تعداد بیشتری از ترکیب‌ها را پوشش دهد انتخاب می‌شود و سپس این استراتژی بهبود یافته و استراتژی‌های mAETG و mAETG-sat به وجود آمدند. به عنوان یکی از قدرتمندترین استراتژی‌های محاسباتی، استراتژی GTWay اشاره شده که دنباله‌های آزمونی براساس رویکرد یک-آزمون-در-یک-زمان تولید می‌کند. این استراتژی با ذخیره تمام ترکیب‌ها به صورت ساختار بیتی و استفاده از جدول ایندکس، نتایج موثری در ارتباط با اندازه آرایه پوشش و سرعت اجرای الگوریتم ارائه می‌دهد و قابلیت تعامل تا قدرت t = 12 را دارد. از دیگر استراتژی‌های یک-آزمون-در-یک-زمان، استراتژی Jenny نیز به عنوان یک استراتژی با سرعت مناسب و نتایج قابل قبول از نظر اندازه آرایه پوشش شناخته شده است. این الگوریتم در ایجاد تمام تعاملات یک-ستونی آغاز می‌کند و سپس به تعاملات دو-ستونی می‌پردازد تا کل تعاملات t-ستونی پوشش داده شود.

در استراتژی TVG، که از روش یک-آزمون-در-یک-زمان استفاده می‌کند، از الگوریتم‌های T-Reduce، plus-one و Random sets برای تولید دنباله‌های آزمون استفاده می‌کند. از این الگوریتم‌ها، T-Reduce به صورت کلی نتایج بهتری نسبت به دو الگوریتم دیگر دارد. استراتژی CTE-XL نیز دنباله‌های آزمون را بر اساس درخت طبقه‌بندی تولید می‌کند. در این روش، دامنه ورودی بر اساس ویژگی‌های خود به زیرمجموعه‌هایی تقسیم می‌شود و سپس موارد آزمون برای این زیرمجموعه‌ها تولید و به دنباله‌های آزمون افزوده می‌شوند، این استراتژی نیز تا t=3 توانایی تولید دنباله‌های آزمون را دارد.

در مورد ITCH نیز اشاره شده است که از جمله کندترین استراتژی‌های محاسباتی است که با جستجوی جامع دنباله‌های آزمون تولید می‌کند و معمولاً قادر به تولید دنباله‌های آزمون تا t = 4 است. PICT نیز یک روش حریصانه است که از استراتژی یک-آزمون-در-یک-زمان بهره می‌برد و ابتدا تمام ترکیبات پوشش داده شده را تولید کرده و سپس موارد آزمون را به صورت تصادفی تولید می‌کند، اما نتایج این الگوریتم مطلوب نیست زیرا به صورت تصادفی عمل می‌کند [18].

در این روند، استفاده از استراتژی‌های مبتنی بر هوش مصنوعی نیز به آگاهی از مجموعه تصادفی از راه حل‌ها آغاز می‌شود، سپس تغییرات بر روی آن راه حل‌های اولیه صورت می‌گیرد و در نهایت بهترین راه حل انتخاب می‌شود تا زمانی که تمام ترکیبات پوشش داده شوند. برخی از الگوریتم‌های استفاده شده تاکنون شامل SA، GA، PSO، BA، ACA، CS، TS، HS، H-B و HPTS است.

با ابتدا استاردم [19] با پیاده‌سازی الگوریتم‌های SA، GA و TS جهت تعامل دودویی (t = 2) که ساختار پیچیده‌ای داشت، نتایج مطلوبی حاصل نشد. نتایج نشان داد که SA بهتر از TS و TS بهتر از GA عمل می‌کند. پس از آن کوهن [20] الگوریتم SA را به گونه‌ای توسعه داد که قادر به پشتیبانی از تعامل تا t = 3 باشد. GA و ACA نیز به روز رسانی شده و قابلیت پشتیبانی از t = 3 را بدست آوردند، اما نتایج نشان داد که استراتژی SA قوی‌تر از GA و ACA است.

دیگر استراتژی PSTG [21] است که بر اساس الگوریتم فرامکاشفه‌ای PSO پیاده سازی شد، اولین استراتژی مبتنی بر هوش مصنوعی است که می‌تواند تا تعامل t = 6 را پشتیبانی کند. این استراتژی رویکردی برای محاسبه وزن مورد آزمون ارائه می‌دهد که نیازمند نگه‌داری ساختمان داده‌های بزرگ برای محاسبه وزن موردهای آزمون است.

یکی از نقاط ضعف استراتژی مبتنی بر PSO در تولید دنباله‌های آزمون این است که با اعمال معادله سرعت، لزوماً مقدار جدید بهبود نخواهد یافت. به‌منظور حل این مشکل، استراتژی DPSO [22] می‌تواند نتایج مطلوبی از نظر اندازه آرایه کسب کند.

الگوریتم FSAPSO [23] یک الگوریتم دیگر بر پایه PSO است که از تکنیک فازی برای محاسبه پارامترهای PSO استفاده می‌کند. این الگوریتم قادر است تا نتایجی تا تعامل t = 4 تولید کند و اندازه آرایه پوشش تولید شده از روش DPSO و CS در اکثر موارد بهتر است. اما به دلیل استفاده از محاسبات فازی، سرعت این استراتژی کمتر است.

در الگوریتم GA، جمعیت اولیه به صورت کاملا تصادفی انتخاب می‌شود و هر مورد آزمون به عنوان یک کروموزوم در نظر گرفته می‌شود. سپس توسط توابع ادغام و جهش، تغییرات بر روی آن‌ها اعمال می‌شود و وزن کروموزوم توسط تابع برازندگی محاسبه می‌شود. محاسبه وزن برای تمام اعضای جمعیت در یک حلقه انجام می‌شود، بهترین کروموزوم انتخاب شده و به دنباله آزمون افزوده می‌شود، و این مراحل تا پوشش کامل ادامه می‌یابد تا تمام ترکیبات پوشش یابند.

استراتژی CS [24] دنباله آزمون را تا تعامل t = 6 ایجاد می‌کند. هدف اصلی این استراتژی کاهش فضای جستجو برای موارد آزمون با استفاده از الگوریتم فاخته است که نتایج مشابهی را با الگوریتم PSTG با سرعت بسیار بالاتر تولید می‌کند.

استراتژی HHH [25] از چهار الگوریتم فرامکاشفه‌ای TLBO, GNA, PSO, CS استفاده می‌کند. این استراتژی از الگوریتم TS (الگوریتم جستجوی ممنوعه) برای انتخاب بهترین

عملکرد این چهارالگوریتم بهره می‌گیرد و براساس سه اپراتور تولید، تنوع، و افزون‌شدگی، در هر مرحله یکی از چهار الگوریتم را برای تولید موارد آزمون انتخاب می‌کند. این استراتژی نتایج خوبی را از نظر اندازه نهایی آرایه پوشش دارد.

استراتژی HSS [26] مبتنی بر الگوریتم مبتنی بر جستجوی هارمونی است. این الگوریتم از رفتار موسیقی‌دانان برای ساخت بهترین آهنگ الهام گرفته شده است. در این استراتژی، در هر تکرار یک مورد آزمون به دنباله آزمون نهایی اضافه می‌شود و الگوریتم تا پوشش کامل دنباله آزمون نهایی ادامه می‌یابد.

۴-راهکار پیشنهادی

امروزه، سیستم های نرم افزاری نیازمند قابلیت اعتماد مناسب برای بهبود عملکرد و کیفیت هستند. به منظور اطمینان از اعتماد سیستم‌ها، نرم‌افزارها را با استفاده از نمونه‌های آزمون مناسب مورد آزمون قرار می‌دهیم. آزمون یک فاز اساسی در توسعه نرم‌افزار است، اما انجام آزمون تعداد زیادی نمونه زمان‌بر است. با حذف نمونه‌های آزمون اضافی، می توان به دنباله آزمونی رسید که پوشش کامل را فراهم می کند. آزمون کننده برای برآورد اهداف خود به تعداد معینی از نمونه های آزمون نیاز دارد که به عنوان دنباله آزمون شناخته می شود. دنباله آزمون تولید شده باید مجموعه ای از دنباله های با ارزش بالا باشد که اهداف آزمون را برآورده کند. روش پیشنهادی برای تولید دنباله آزمون با استفاده از آرایه پوشش الگوریتم بهینه‌سازی مبتنی بر جغرافیای زیستی است. با استفاده از این الگوریتم می توان دنباله آزمون کمینه‌ای را تولید کرد. در این گفتار، مراحل انجام این الگوریتم پیشنهادی را به طور جامع توضیح خواهیم داد.

۴-۱ گام اول: تولید جمعیت اولیه

در گام اول این کارآمدی، مقداردهی اولیه پارامترهای مرتبط با الگوریتم بهینه‌سازی جغرافیایی زیستی (BBO) انجام می شود. هر فرد در الگوریتم جستجوی جغرافیایی زیستی به اندازه تعداد پارامترهای یک تنظیم دارای عناصری است که هرکدام از این عناصر مقداری بین ۱ تا v-1 را در خود جای داده‌اند. به عنوان مثال، برای تنظیم CA (N; 2, 6, 3) که در آن تعداد پارامترها ۶ می باشد و هر پارامتر ۳ مقدار به خود اختصاص می‌دهد، یکی از جمعیت‌های آن ممکن است به شکل زیر باشد:

۴-۲ گام دوم: تولید ماتریس پوشش

در گام دوم، یک ساختمان داده جدید برای نگهداری حالت های پوشش داده نشده، بر اساس الگوریتم بهینه‌سازی جغرافیایی زیستی(BBO) ، ارائه شده است. این ساختمان داده امکان بهبود سرعت محاسبه وزن نمونه‌های آزمون و در نهایت تولید سریع‌تر کل مجموعه آزمون را فراهم می‌کند. از پیکربندی CA (N; 3, 42) برای این منظور استفاده می‌شود.

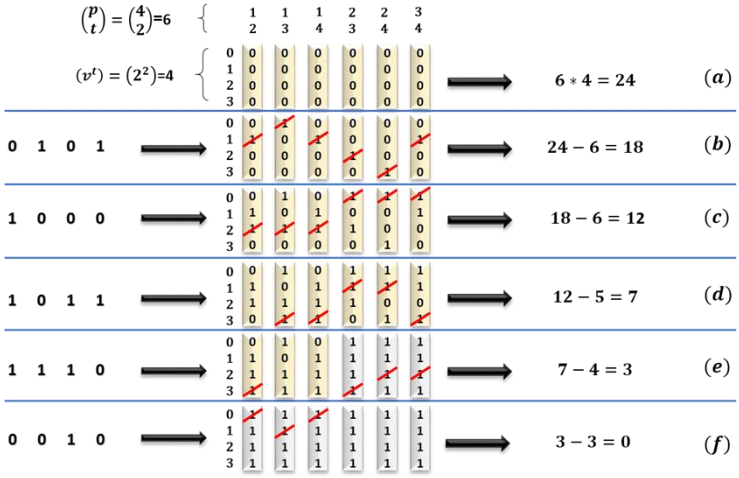
در این پیکربندی، برای ذخیره سازی، نیاز به ۳۲ سطر در حافظه وجود دارد. تعداد سطرها در ساختمان داده، برابر با $\binom{p}{t}$ است و ستون‌ها به v+1 بخش تقسیم می‌شوند. بخش اول ترکیبات هر سطر را نگهداری می‌کند و v بخش دیگر هرکدام شامل (Max_Coverage)/v بیت است. در این مدل ذخیره‌سازی، جستجو در نصف ماتریس پوشش در هر مرحله انجام می‌شود که بر سرعت محاسبه وزن نمونه آزمون تأثیرگذار است. همچنین، تعداد نمونه های آزمون پوشش داده نشده در هر سطر در ستون آخر ساختمان داده ذخیره می‌شود.

نحوه محاسبه وزن یک نمونه آزمون که در این مدل با عبارت (۱۰۱۰) نمایش داده شده است، نشان داده شده است. با بررسی چهار ترکیب مختلف، مقادیر متناظر با آنها بررسی می‌شوند و در صورت عدم پوشش، مقادیر در ماتریس به یک تبدیل می‌شوند. اگر همه مقادیر در حالت عدم پوشش باشند، وزن نمونه آزمون محاسبه شده و تعداد حالات پوشش داده نشده به‌روز شده و عملیات به همین منوال ادامه می یابد تا تعداد پوشش داده نشده به صفر برسد.

۴-۳ گام سوم: محاسبه وزن یک نمونه آزمون

محاسبه وزن یک نمونه آزمون صورت می پذیرد. ابتدا معادل دهدهی نمونه آزمون محاسبه می شود و در صورتی که سلول متناظر با آن صفر باشد، مقدار آن به یک (پوشش) تغییر می‌یابد و یک واحد به وزن نمونه آزمون اضافه می‌شود. در مثال ارائه شده، با ۶ ترکیب مختلف روبرو هستیم که هر کدام شامل ۴ مقدار ۰۰، ۰۱، ۱۰ و ۱۱ هستند که در مجموع ۲۴ حالت پوشش داده نشده را ایجاد می کنند. ابتدا، نمونه آزمون ۰۱۰۱ انتخاب می شود. در گام اول، ترکیب ۱ و ۲ انتخاب می شود که با توجه به نمونه آزمون ۰۱۰۱ به حالت **۰۱ منجر می‌شود. سپس، سلول اول و دوم بررسی می‌شود و اگر مقدار آنها صفر باشد، به یک تغییر می‌کند و یک واحد به وزن نمونه آزمون اضافه می‌شود. این فرایند برای تمام ترکیب‌ها ادامه پیدا می‌کند. در پایان این مراحل، وزن نمونه آزمون با توجه به حالت‌های پوشش داده نشده کم شده و نمونه آزمون بعدی مورد بررسی قرار می‌گیرد. این مراحل تا زمانی ادامه پیدا می کنند که تمامی صفرهای ماتریس به یک تبدیل شوند. در این روند، الگوریتم BBO

مسئول انتخاب نمونه آزمون و محاسبه وزن آن است و این وزن از تعداد حالات پوشش داده نشده کم می‌شود و روند محاسبه تا رسیدن به وزن صفر ادامه می‌یابد (شکل ۳).



شکل ۳: محاسبه وزن یک نمونه آزمون

۴-۴ گام چهارم: تولید دنباله آزمون با الگوریتم *BBO* و *BAT*

در گام چهارم این راهکار، دنباله آزمون با استفاده از الگوریتم بهینه‌سازی جغرافیایی زیستی (BBO) و الگوریتم خفاش تولید می‌شود. در این الگوریتم برای تولید پیکربندی CA(N: t, p, v)، هر فرد شامل داده و وزن است که داده‌ها از یک آرایه p عنصری تشکیل شده است که مقادیر عناصر آن بین ۰ تا v-1 قرار دارند و وزن از تابع برازندگی بدست می‌آید. در ابتدا، نمونه آزمون اول به صورت کاملا تصادفی انتخاب می‌شود و وزن آن برابر با $\binom{P}{t}$ است. سپس، در یک حلقه تعداد کل تکرار مشخص می‌شود که پس از آن به دنباله آزمون نهایی می‌رسیم. در هر تکرار، یک نمونه آزمون انتخاب شده و به دنباله آزمون اضافه می‌شود. تکرار بعدی با بهترین مقدار (best) به صورت تصادفی انتخاب می‌شود و این مراحل تا رسیدن به دنباله آزمون نهایی ادامه می‌یابد. در این حلقه اصلی، الگوریتم BBO-BAT برای انتخاب نمونه آزمون بعدی اجرا می‌شود و تغییراتی با توجه به تابع‌های کراس آور و جهش اعمال می‌شود. تغییرات در جمعیت براساس ارزش‌های آنها اعمال می‌شود و جمعیت‌های با ارزش بالاتر کمتر مورد تغییر قرار می‌گیرند. همچنین، تابع جهش نیز بر اساس یک مقدار تصادفی برای هر عضو جمعیت، ممکن است تغییر کند. این روند تا وقتی ادامه پیدا می‌کند که تمامی صفرهای ماتریس به یک تبدیل شوند و دنباله آزمون نهایی به‌دست آید. این روال تولید دنباله آزمون با الگوریتم BBO-BAT با توجه به پارامترهای ورودی CA، با تعداد

تکرارها بین ۵۰ تا ۲۰۰ متغیر است. این مراحل تا رسیدن به دنباله آزمون نهایی ادامه خواهند یافت.

۴-۵ ارزیابی

در این پژوهش، ارزیابی به سه دسته بهره‌وری (اندازه آرایه)، کارایی (زمان تولید) و قوه تعامل تقسیم شده است. بهره‌وری تنها به مراحل اجرای استراتژی مربوط است، در حالی که کارایی مرتبط با بستر سخت‌افزاری و نرم‌افزاری است. تعداد زیادی استراتژی تولید آرایه پوشش وجود دارد، اما برای مقایسه عادلانه ما تعدادی از استراتژی‌ها را مورد ارزیابی قرار داده‌ایم. در بین استراتژی‌های هوش مصنوعی، CPSO، DPSOو GS و در بین استراتژی‌های غیر هوش مصنوعی IPOG انتخاب شده‌اند. مشخصات بستر سخت‌افزاری شامل Windows 7، GHz core™ i7QM CPU۲.۲۰ و RAM 6GB می‌باشد و کد نویسی در محیط (eclipse (jdk 1.8 انجام شده است.

استراتژی‌های غیر هوش مصنوعی به طور عموم قطعی هستند و اجرای چندباره آن‌ها تاثیری در نتیجه نخواهد داشت؛ اما استراتژی‌های مبتنی بر هوش مصنوعی غیرقطعی هستند و با اجرای مجدد نتیجه تغییر خواهد کرد. بنابراین، هر پیکربندی صد بار اجرا شده و مقدار Best که بهترین اجرا برای هر پیکربندی است را نشان می‌دهند، ثبت شده است. مقادیر بهره‌وری برای استراتژی‌های GS، CPSO و DPSO از مقاله‌ها مربوطه گرفته شده‌اند، در غیر این صورت تعداد برابر با الگوریتم پیشنهادی اجرا و نتیجه ثبت شده است. در نهایت، برای تعامل‌های بالا تعداد دفعات تکرار کاهش یافته است. پارامترهای راهکار پیشنهادی در جدول ۱ نشان داده شده‌اند.

جدول ۱: پارامترهای راهکار پیشنهادی

الگوریتم	پارامتر	مقدار
BBO-BAT	Population	10-80
	Mutation rate	0.1
	Emigration rate	0-1
	alpha	0.9

در این پژوهش، معیار مهم برای ارزیابی قدرت یک استراتژی در تولید آرایه پوشش، بهره‌وری آن استراتژی است. ارزیابی ابتدایی به t = 2 اختصاص داده شده است و در جدول ۲ نتایج ارزیابی مرتبط با این پیکربندی نشان داده شده است. در این ارزیابی، ۱۳ پیکربندی در نظر گرفته شده و برای مقایسه قدرت استراتژی‌ها از روش آزمون آماری Wilcoxon

signed rank sum استفاده شده است که در نرم‌افزار SPSS موجود است.

در نتایج آزمون Wilcoxon که در جدول ۳، مقایسه‌ای بین راهکار پیشنهادی و سایر استراتژی‌ها انجام شده است. در یکی از مقایسه‌ها، راهکار پیشنهادی (BBO-BAT) با استراتژی Jenny مقایسه شده است. در بخش ranks سه عدد ۱۲، ۰ و ۱ مشاهده می‌شود که به ترتیب نشان‌دهنده تعداد نمونه‌هایی است که راهکارBBO-BAT از Jenny قوی‌تر، ضعیف‌تر و برابر است. همچنین مقدار Asymp. Sig. (2-tailed) کمتر از ۰.۰۵ بوده که نشان‌دهنده وجود اختلاف معنایی بین دو استراتژی می‌باشد و از این‌رو می‌توان اظهار نمود که قدرت استراتژیBBO-BAT نسبت بهJenny، به شکل معناداری بیشتر است. در مقایسه با دو استراتژی GS و DPSO، راهکار پیشنهادی تفاوت معنایی ندارد، اما با توجه به مقادیر ranks، می‌توان نتیجه گرفت که راهکار پیشنهادی برتری کمی نسبت به این دو استراتژی دارد. در ارزیابی بعدی که به t = 3 اختصاص داده شده است، ۱۴ پیکربندی در نظر گرفته شده است و نتایج ارزیابی در جدول ۵ نشان داده شده است. در این ارزیابی، به طور کلی IPOG در یک مورد، CPSO در سه مورد، DPSO در نه مورد، GS در چهار مورد و BBO-BAT در ۱۳ مورد بهترین نتایج را تولید کرده‌اند.

جدول ۲: مقایسه راهکار پیشنهادی در t = 2

	PICT N	IPOG N	CPSO N.Best	DPSO N.Best	GS N.Best	BBO-BAT N.Best
CA (N; 2, 2^7)	۷	۷	۷	۷	٦	٦
CA (N; 2, 3^3)	۱۰	۹	۹	۹	۹	۹
CA (N; 2, 3^4)	۱۳	۹	۹	۹	۹	۹
CA (N; 2, 3^5)	۱۳	۱۵	۱۱	۱۱	۱۱	۱۱
CA (N; 2, 3^6)	۱٤	۱۵	۱٤	۱٤	۱۳	۱۳
CA (N; 2, 3^7)	۱٦	۱۵	۱۵	۱۵	۱٤	۱٤
CA (N; 2, 3^8)	۱٦	۱۵	۱۵	۱۵	۱۵	۱۵
CA (N; 2, 3^9)	۱۷	۱۵	۱٦	۱۵	۱۵	۱۵
CA (N; 2, 3^{10})	۱۸	۱۵	۱٦	۱٦	۱٦	۱٦
CA (N; 2, 3^{11})	۱۸	۱۷	۱٦	۱۷	۱٦	۱٦
CA (N; 2, 3^{12})	۱۹	۲۱	۱۷	۱٦	۱٦	۱٦
CA (N; 2, 4^7)	۲۷	۲۹	۲۵	۲٤	۲٤	۲٤
CA (N; 2, 5^7)	٤۰	٤۵	۳٦	۳٤	۳۸	۳٤

2.20GHz core™ i7QM CPU, RAM 6GB, Windows 7

نتایج آزمون Wilcoxon برای جدول ۵ در جدول ۶ نشان داده شده است. در این ارزیابی، استراتژی پیشنهادی در تمامی چهارده پیکربندی نتایج بهتری نسبت به Jenny، TConfig و PICT داشته و در مقایسه با دیگر استراتژی‌ها اختلاف معنایی دارد. به عنوان مثال،

BBO-BAT در ۱۳ پیکربندی از IPOG بهتر بوده و در یک مورد نتیجه برابر با آن بوده است. CPSO در یازده مورد ضعیف‌تر از راهکار پیشنهادی بوده و در سه مورد نتایج برابر داشته است. GS نیز در ۱۰-مورد از BBO-BAT ضعیف‌تر بوده و در چهار مورد نتایج برابر داشته است. استراتژی پیشنهادی با تمام استراتژی‌های مذکور اختلاف معنایی داشته و مقدار ((Asymp. Sig. (2-tailed) صحت این موضوع را تایید کرده است. همچنین، در مقایسه با DPSO، استراتژی پیشنهادی تفاوت معنایی نداشته اما با توجه به مقادیر ranks، می‌توان نتیجه گرفت که استراتژی پیشنهادی قدرت بیشتری دارد.

جدول ۳: مقایسه راهکار پیشنهادی در t = 3

	PICTN	IPOGN	CPSO N.Best	DPSO N.Best	GS N.Best	BBO-BAT N.Best
CA (N; 3, 2^7)	۱۵	۱۶	۱۲	۱۵	۱۲	۱۲
CA (N; 3, 2^8)	۱۷	۱۸	۱۶	۱۶	۱۴	۱۲
CA (N; 3, 2^9)	۱۷	۲۰	۱۶	۱۶	۱۶	۱۶
CA (N; 3, 3^4)	۳۴	۳۲	۳۰	۲۸	۲۷	۲۷
CA (N; 3, 3^5)	۴۳	۴۱	۳۸	۴۱	۳۸	۳۷
CA (N; 3, 2^{10})	۱۸	۲۰	۱۶	۱۶	۱۶	۱۶
CA (N; 3, 3^6)	۴۸	۴۸	۴۲	۳۳	۴۳	۳۹
CA (N; 3, 3^7)	۵۱	۵۵	۴۹	۴۸	۴۹	۴۸
CA (N; 3, 3^8)	۵۹	۵۸	۵۳	۵۲	۵۴	۵۲
CA (N; 3, 3^9)	۶۳	۶۳	۵۸	۵۶	۵۸	۵۶
CA (N; 3, 3^{10})	۶۵	۶۶	۶۱	۵۹	۶۱	۵۹
CA (N; 3, 3^{11})	۷۰	۷۰	۶۳	۶۳	۶۳	۶۱
CA (N; 3, 3^{12})	۷۲	۷۳	۶۸	۶۵	۶۷	۶۵
CA (N; 3, 4^7)	۱۲۴	۱۱۲	۱۱۵	۱۱۲	۱۱۶	۱۱۲

2.20GHz core™ i7QM CPU, RAM 6GB, Windows 7

در ارزیابی آخرین مرحله که برای t > 4 انجام شده است، نتایج آن در جدول ۴ نمایش داده شده است. در این مقایسه، استراتژی TConfig تنها در دو پیکربندی (CA (N; 5, 3^7 و CA (N; 6, 3^8) قادر به تولید دنباله آزمون در زمان کمتر از ۲۴ ساعت بوده و در سایر موارد نیازمند زمان بیشتری است که با علامت ">day" مشخص شده است. همچنین، استراتژی IPOG توانایی تولید تا t = 6 را دارد، اما مقادیر بزرگ‌تر از ۶ را پشتیبانی نمی‌کند. استراتژی DPSO نیز در پنج مورد نیاز به زمان بیشتر از یک روز داشته است، اما سایر استراتژی‌ها تعاملات بالا را پشتیبانی می‌کنند، و در این بین استراتژی پیشنهادی از دیگر

استراتژی‌ها قوی‌تر عمل کرده است. نتایج آزمون Wilcoxon برای این ارزیابی در جدول ۷ نشان داده شده است و این نتایج نشان می‌دهد که در مقایسه با سایر استراتژی‌ها، استراتژی پیشنهادی قدرت بیشتری دارد. این نتایج نشان می‌دهند که استراتژی پیشنهادی بهبود قابل توجهی را در بهره‌وری نسبت به دیگر روش‌ها ارائه داده است و به عنوان یک انتخاب قوی برای تولید آرایه پوشش در موارد پیچیده‌تر مورد توجه قرار می‌گیرد.

جدول ۴: مقایسه راهکار پیشنهادی در t >4

	PICTN	IPOGN	CPSO N.Best	DPSO N.Best	GS N.Best	BBO-BAT N.Best
CA (N; 5, 3^7)	۴۵۲	۴۶۶	۴۴۴	۴۲۸	۴۳۱	۴۳۰
CA (N; 6, 3^8)	۱۴۵۵	۱۴۰۹	۱۴۰۲	۱۴۰۹	۱۳۹۸	۱۳۹۲
CA (N; 7, 3^9)	۴۶۱۶	NS	۴۴۳۳	۴۴۵۱	۴۴۳۷	۴۴۲۲
CA (N; 8, 3^{10})	۱۴۵۹۹	NS	۱۳۹۴۴	۱۳۹۳۳	۱۳۹۰۷	۱۳۹۰۳
CA (N; 9, 3^{11})	۴۵۵۲۱	NS	۴۳۵۹۱	>day	۴۳۸۰۸	۴۳۵۴۹
CA (N; 10, 3^{12})	۱۴۱۹۹۰	NS	۱۳۵۴۹۸	>day	۱۳۶۰۹۶	۱۳۵۳۹۲
CA (N; 11, 3^{12})	۲۷۸۹۹۳	NS	۲۶۸۱۷۳	>day	۲۶۷۶۳۰	۲۶۷۸۰۸
CA (N; 12, 2^{14})	۹۱۱۲	NS	۸۸۸۲	۸۹۷۲	۸۸۹۰	۸۸۸۹
CA (N; 13, 2^{14})	۱۲۴۴۱	NS	۱۱۳۷۱	>day	۱۰۵۲۱	۱۰۴۷۴
CA (N; 14, 2^{15})	۲۵۰۳۶	NS	۲۳۸۸۹	>day	۲۳۳۷۷	۲۲۶۳۹
CA (N; 15, 2^{16})	۵۱۱۲۷	NS	۴۶۴۲۳	>day	۴۶۵۷۵	۴۵۷۹۲
CA (N; 16, 2^{17})	۱۰۰۲۶۶	NS	۹۶۰۰۵	>day	۹۵۶۷۵	۹۴۹۹۷
CA (N; 17, 2^{18})	>day	NS	>day	>day	۱۷۹۵۴۰	۱۷۹۱۴۷
CA (N; 18, 2^{19})	>day	NS	>day	>day	۳۳۰۳۹۱	۳۳۰۲۳۵
CA (N; 19, 2^{20})	>day	NS	>day	>day	۶۲۴۹۴۰	۶۲۴۷۲۱
CA (N; 20, 2^{20})	>day	NS	>day	>day	۱۰۴۸۵۷۶	۱۰۴۸۳۵۸

2.20GHz core™ i7QM CPU, RAM 6GB, Windows 7

جدول ۵: نتایج آزمون ویکاکسون برای جدول ۲

	Ranks			Test Statistics
	BBO<	BBO>	BBO=	Asymp. Sig. (2-tailed)
PICT - BBO-BAT	۱۳	۰	۰	۰.۰۰۱
IPOG - BBO-BAT	۸	۱	۴	۰.۰۱۶
CPSO - BBO-BAT	۷	۰	۶	۰.۰۰۸
DPSO - BBO-BAT	۴	۱	۸	۰.۱۷۹
GS - BBO-BAT	۱	۰	۱۲	۰.۳۱۷

جدول ۶: نتایج آزمون ویکاکسون برای جدول ۳

	Ranks			Test Statistics
	BBO<	BBO>	BBO=	Asymp. Sig. (2-tailed)

PICT - BBO-BAT	۱۴	۰	۰	۰.۰۰۹۴
IPOG - BBO-BAT	۱۳	۰	۰	۰.۰۰۱۳
CPSO - BBO-BAT	۱۱	۰	۳	۰.۰۰۳۸
DPSO - BBO-BAT	۵	۱	۸	۰.۳۵
GS - BBO-BAT	۱۰	۰	۴	۰.۰۰۴۴

جدول ۷: نتایج آزمون ویکاکسون برای جدول ۴

	Ranks			Test Statistics
	BBO<	BBO>	BBO=	Asymp. Sig. (2-tailed)
PICT - BBO-BAT	۱۲	۰	۰	۰.۰۰۹۴
IPOG - BBO-BAT	۲	۰	۰	۰.۰۰۱۳
CPSO - BBO-BAT	۱۲	۰	۰	۰.۰۰۳۸
DPSO - BBO-BAT	۴	۱	۰	۰.۰۰۴
GS - BBO-BAT	۱۵	۰	۱	۰.۰۰۴۴

۵-نتیجه‌گیری

در این پژوهش، به بررسی تولید آرایه پوشش (دنباله آزمون) با استفاده از الگوریتم بهینه‌سازی مبتنی بر جغرافیای زیستی و خفاش پرداخته شده است، که این الگوریتم شامل استراتژی‌های اکتشاف و بهره‌برداری مبتنی بر مهاجرت است و از الگوریتم‌های الهام گرفته شده از طبیعت که یکی از سریع‌ترین الگوریتم‌ها است، استفاده می‌کند. ارزیابی استراتژی‌های تولید دنباله آزمون به دو دسته اندازه آرایه پوشش (بهره‌وری) و زمان تولید دنباله آزمون تقسیم شده است. ارزیابی اندازه آرایه مستقل از سخت‌افزار و سیستم عامل است، در حالی که ارزیابی زمانی وابسته به سخت‌افزار و سیستم عامل است. استراتژی‌های مختلفی اعم از DPSO، CPSO، IPOG، PICT، Tconfig و Jenny بر روی سیستم عامل اجرا شده و با استراتژی BBO-BAT مقایسه شده‌اند. استراتژی‌های مبتنی بر محاسبات محض مانند PICT، Jenny و IPOG توانایی تولید دنباله آزمون تا t>6 را دارند اما اغلب در بهره‌وری بهینه عمل نمی‌کنند، در حالی که از لحاظ سرعت اجرا عملکرد خوبی دارند. نتایج نشان دادند که استراتژی‌های مبتنی بر هوش مصنوعی مانند BBO-BAT نسبت به استراتژی‌های مبتنی بر محاسبات محض قدرتمندتر هستند و راهکار پیشنهادی در مقایسه با راهکارهای دیگر بسیار قوی‌تر عمل کرده است. این پژوهش به وضوح نشان داده که قدرت استراتژی‌های مبتنی بر هوش مصنوعی، از راهکارهای مبتنی بر محاسبات محض بیشتر است.

۶-تشکر و قدردانی

این اثر را با عشق و احترام، تقدیم می‌کنم به کسانی که با حمایت‌های بی‌دریغ و انگیزه‌های دائمی خود، مرا در این سفر علمی یاری رسانده‌اند. امیدوارم که این پژوهش بتواند بیانگر قدردانی عمیق من از همراهی‌های شان باشد. صمیمانه از جناب آقای دکتر وحید رافع، عضو محترم هیأت علمی دانشگاه اراک، بابت راهنمایی‌های ارزنده و نقطه نظرات سازنده‌اش در طول این پروژه تشکر می‌کنم. حمایت‌های ایشان نقش مهمی در موفقیت این پژوهش داشته است.

۷-تعارض منافع

نویسنده(گان) اعلام می‌دارند که در مورد انتشار این مقاله تضاد منافع وجود ندارد. علاوه بر این، موضوعات اخلاقی شامل سرقت ادبی، رضایت آگاهانه، سوء رفتار، جعل داده‌ها، انتشار و ارسال مجدد و مکرر توسط نویسندگان رعایت شده است.

۸-دسترسی آزاد

۹- منابع

[1] 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, 2016.

[2] S. Esfandyari and V. Rafe, "Extracting Combinatorial Test parameters and their values using model checking and evolutionary algorithms," *Applied Soft Computing*, vol. 91, pp. 1-19, 2020.

[3] S. Esfandyari and V. Rafe, "GALP: a hybrid artificial intelligence algorithm for generating covering array," *soft computing*, vol. 25, p. 7673–7689, 2021.

[4] M. K.-N. Einollah Pira, "Combinatorial t-way test suite generation using an improved asexual reproduction optimization algorithm," *Applied Soft Computing*, vol. 150, 2024.

[5] D. Simon, "Biogeography-based optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 6, pp. 702-713, 2008.

[6] A. A. Muazu, A. S. Hashim and A. Sarlan, "Review of Nature Inspired

Metaheuristic Algorithm Selection for Combinatorial t-Way Testing," *IEEE Access*, vol. 10, pp. 27404 - 27431, 2022.

[7] 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, 2022.

[8] E. Pira, V. Rafe and S. Esfandyari, "A three-phase approach to improve the functionality of t-way strategy," *Soft Computing*, pp. 1-21, 2023.

[9] S. Esfandyari and V. Rafe, "Using the Particle Swarm Optimization Algorithm to Generate the Minimum Test Suite in Covering Array with Uniform Strength," *Soft Computing Journal*, vol. 8, no. 2, pp. 66-79, 2021.

[10] S. Esfandyari and V. Rafe, "Correction to: GALP: a hybrid artificial intelligence algorithm for generating covering array," *Soft Computing*, 2021.

[11] 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, 2018.

[12] Y. Lei, R. Kacker, D. R. Kuhn, V. Okun and J. Lawrence, "IPOG: a general strategy for t-way software testing," in *4th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems, IEEE Computer Society*, Tucson, AZ, 2007.

[13] 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, 2008.

[14] A. Hartman, "IBM Intelligent Test Case Handler," IBM alphaworks, 2023. [Online]. Available: http://www.alphaworks.ibm.com/tech/whitch.

[15] B. Jenkins, "Jenny download web page," Bob Jenkins' Website, 2023. [Online]. Available: http://burtleburtle.net/bob/math/jenny.html.

[16] J. Arshem, "TVG download page," 2023. [Online]. Available: http://sourceforge.net/projects/tvg.

[17] 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, 2011.

[18] 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 Transactions on Software Engineering*, vol. 23, no. 7, pp. 437 - 444, 1997.

[19] S. S. V. R. S. E. Leila Yousofvand, "Automatic program bug fixing by focusing on finding the shortest sequence of changes," *Artificial Intelligence Review*, vol. 57, no. 2, p. 39, 2024.

[20] J. Stardom, "Metaheuristics and the Search for Covering and Packing

Array," *Thesis (M.Sc.), Simon Fraser University, 2001*, 2001.

[21] M. B. Cohen, "Designing Test Suites for Software Interactions Testing," PHD Thesis, University of Auckland,Department of Computer Science ,Auckland,, 2004.

[22] 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, p. 1330–1347, 2012.

[23] H. Wu, C. Nie, F.-C. Kuo, H. Leung and C. J. Colbourn, "A Discrete Particle Swarm Optimization for Covering Array Generation," *IEEE Transactions on Evolutionary Computation,* vol. 19, no. 4, pp. 575-591, 2015.

[24] K. Z. Zamli, B. S. Ahmed, T. Mahmoud and W. Afzal, "Fuzzy Adaptive Tuning of a Particle Swarm Optimization Algorithm for Variable-Strength Combinatorial Test Suite Generation," in *Swarm Intelligence,* vol. 3, Y. Tan, Ed., IET, 2018.

[25] 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, 2015.

[26] K. Z. Zamli, B. Y. Alkazemi and G. Kendall, "A Tabu Search hyper-heuristic strategy for t-way test suite generation," vol. 44, pp. 57-74, 2016.

[27] A. R. A. Alsewari and K. Z. Zamli, "Design and implementation of a harmony-search-based variable-strengtht-way testing strategy with constraints support," *Information and Software Technology,* vol. 54, no. 6, p. 553–568, 2012.

[28] Amirzdeh, M., Hosseini Moradi, S. A., & Ghobadi, N. (2023). Real Time Detection of Multi-Rotor Unmanned Aerial Vehicle Using YOLOv5 Optimized Algorithm. Journal of Advanced Defense Science & Technology, 14(1), 11-22.

[29] 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.