فصلنامه سیستم‌های پردازشی و ارتباطی چندرسانه‌ای هوشمند
Intelligent Multimedia Processing and Communication Systems (IMPCS)

Spring 2023, 4 (1), 23-33
DOR: 20.1001.1.27832570.1402.4.1.3.8

Received: 12 Feb 2023
Accepted: 11 Apr 2023

مقاله پژوهشی

Constraint Search based Test Data Generation to cover Prime Paths in Structural Testing

Ebrahim Fazli [1*], Mojtaba Aajami[2]

1.Assistant professor, Faculty of Electrical and Computer Engineering, Zanjan Branch, Islamic Azad University, Zanjan, *(Corresponding Author)*, efazli@znu.ac.ir

2.Assistant professor, Faculty of Electrical and Computer Engineering, Zanjan Branch, Islamic Azad University, Zanjan,

Abstract: This paper presents a novel scalable Constraint Search Based method for Test Data Generation, called CSBTDG, used in structural testing. CSBTDG outperforms existing methods for test data generation in several orders of magnitude, both in time and constraint efficiency. Search-based software testing is a powerful automated method to generate test inputs for software under test. Its goal is to reach a branch or a statement in a program. One major limitation of this approach is an insufficiently informed fitness function to start and guide search toward a test target within nested predicates (constraints). Another important limitation of this approach about equality constraints. To address these problems we propose a new search method based on integrating constraint programming using choco constraint solver and the Gray Wolf heuristic search algorithm. Preliminary experiments promise efficiency and effectiveness for the new constraint search based test data generation approach using multiple fitness functions.

Test data generation is done by solving constraints related to test paths extracted from the program under test [3][2]. In general, the constraint solving problem of the test paths is a sub problem of a wider set of problems called constraint satisfaction problem (CSP) [4]. Formally, a constraint satisfaction problem is defined as a triple (X, D, C). X represents a set of variables, D is a set of range of values for variables and C is a set of constraints. Constraint satisfaction is the process of finding solutions for a set of constraints that satisfy the conditions imposed on the variables. Evaluation of variables is a function of a subset of variables to a specific set of values in the corresponding domains. The proposed solution is called consistent evaluation if none of the constraints are violated. An evaluation is a complete evaluation if it includes all variables. If the assessment is consistent and complete, it is a solution. It is said that such evaluation is a solution to the constraint satisfaction problem. Therefore, a solution is a set of values for the variables that satisfy all the constraints, that is, a point in the feasible region.

The main innovation of this article is to present a cooperation model to exchange information between the solver and the searcher. In this model of data generation, the test solver is an initial setup to assign values to some variables to start searching for the meta-heuristic algorithm.

فصلنامه سیستم‌های پردازشی و ارتباطی چندرسانه‌ای هوشمند
Intelligent Multimedia Processing and Communication Systems (IMPCS)
انجمن علمی تجارت الکترونیکی ایران
واحد زنجان

تولید داده آزمون مبتنی بر جستجوی مقید جهت پوشش مسیرهای اولیه در آزمون ساختاری

تاریخ دریافت: ۱۴۰۱/۱۱/۲۳
تاریخ پذیرش:۱۴۰۲/۰۱/۰۶

دورهٔ چهارم، بهار ۱۴۰۲
شمارهٔ اول، صص: ۲۳-۳۳

ابراهیم فضلی [1*]، مجتبی اعجمی [2]

۱. استادیار، دانشکده مهندسی برق و کامپیوتر، واحد زنجان، دانشگاه آزاد اسلامی، زنجان، ایران. (نویسنده مسئول)

۲. استادیار، دانشکده مهندسی برق و کامپیوتر، واحد زنجان، دانشگاه آزاد اسلامی، زنجان، ایران.

چکیده: این مقاله به مسئله تولید داده آزمون جهت پوشش مسیرهای آزمون پوشش دهنده مسیرهای اولیه [1]می‌پردازد. روش ارائه شده جهت حل معادله مسیر ترکیبی از روش مبتنی بر جستجو و مبتنی بر قید می‌باشد. در بخش مبتنی بر قید از حل کننده ای بنام choco[2]جهت حل قیدهای مبتنی بر تساوی استفاده شده است. در بخش مبتنی بر جستجو یک روش فرا ابتکاری مبتنی بر الگوریتم بهینه سازی گرگ خاکستری[1] استفاده شده است. این روش صرفا با مقداردهی دو پارامتر، عمل جستجوی مقادیر را آغاز می‌کند. این روش قابلیت ارتقا به فضای چند بعدی بدون نیاز به پارامترهای اضافی را دارد. استفاده از برنامه نویسی مقید با استفاده از حل کننده می‌تواند یکی از نقاط ضعف اساسی روشهای مبتنی بر جستجو که حل شرط‌های تساوی می‌باشد را برطرف کند. نقطه قوت دیگر الگوریتم جستجو به روش گرگ خاکستری، استفاده از عامل‌های جستجوی مختلف و راه‌حل‌های چندگانه می‌باشد که از به دام افتادن فرآیند جستجو در بهینه‌های محلی جلوگیری می‌کند. نتایج اجرای روش پیشنهادی بر روی دسته ای محدود از برنامه‌های محک حاکی از بهبود سرعت و همچنین درصد موفقیت بالا در تولید داده آزمون نسبت به دیگر الگوریتم‌های جستجوی فرا ابتکاری مانند الگوریتم ژنتیک، می‌باشد.

واژه‌های کلیدی: آزمون ساختاری، مسیر اولیه، مسیر آزمون پوشش دهنده مسیر اصلی

[1] Prime Path
[2] www.choco-solver.org

۱. مقدمه

تولید داده آزمون بوسیله حل قیدهای مربوط به مسیرهای آزمون استخراج شده از برنامه تحت آزمون صورت می‌گیرد[2][3]. به‌طور کلی مسئله حل قیدهای مسیرهای آزمون در ذیل یک مجموعه وسیعتر مسائل بنام مسئله ارضا محدودیت (CSP)[1] قرار می‌گیرند[4] . به‌صورت صوری یک مسئله ارضا محدودیت به‌صورت یک سه گانه (X,D,C) تعریف می‌شود. X معرف یک مجموعه از متغیرها، D یک مجموعه دامنه مقادیر برای متغیرها و C یک مجموعه از محدودیت‌ها می‌باشد. ارضا محدودیت فرایند یافتن راه حل برای مجموعه محدودیت‌ها است که شرایط تحمیل شده بر روی متغیرها را برآورده می‌سازند. ارزیابی متغیرها تابعی از زیر مجموعه متغیرها به مجموعه خاصی از مقادیر در زیر مجموعه مربوطه از دامنه‌ها می‌باشد. راه حل ارائه شده در صورت عدم نقض هیچ یک از محدودیت‌ها، ارزیابی سازگار نامیده می‌شود. ارزیابی اگر تمام متغیرها را شامل شود، یک ارزیابی کامل است. اگر ارزیابی سازگار و کامل باشد، یک راه حل است. گفته می‌شود چنین ارزیابی یک راه حل برای مسئله ارضا محدودیت می‌باشد. بنابراین، یک راه حل مجموعه ای از مقادیر برای متغیرهایی است که همه محدودیت‌ها را برآورده می‌کند، یعنی یک نقطه در منطقه امکان پذیر است. مسئله تولید داده آزمون جهت پوشش مسیرهای آزمون مبتنی بر مسیرهای اولیه به‌صورت زیر فرموله شده است:

مسئله ۱ (تولید داده آزمون جهت پوشش مسیرهای آزمون مبتنی بر مسیرهای اولیه):

- ورودی: یک CSP به‌صورت (X,D,C)
- خروجی: یک راه حل (مجموعه ای از مقادیر برای متغیرها که تمام محدودیت‌ها را برآورده می‌کند).

نوآوری اصلی این مقاله ارائه یک مدل همکاری جهت تبادل اطلاعات بین حل کننده و جستجوگر می‌باشد. در این مدل از تولید داده آزمون حل کننده یک راه انداز اولیه جهت مقدار دهی به برخی متغیرها جهت شروع جستجوی الگوریتم فرا ابتکاری می‌باشد.

۲. حل کننده choco

Choco یک نرم افزار متن باز رایگان می‌باشد که برای برنامه نویسی مقید ابداع شده است[5] . این نرم افزار به زبان جاوا و تحت مجوز BSD توسعه یافته است. کاربرد اصلی آن توصیف مسائل ترکیبیاتی دنیای واقعی به شکل یک مسئله ارضا محدودیت و حل آنها بوسیله تکنیک‌های برنامه نویسی مقید می‌باشد. Choco یکی از سریعترین حل کننده‌ها می‌باشد و بعنوان یک کتابخانه جاوا ۸ شامل موارد زیر می‌باشد:

- انواع مختلف متغیرها (صحیح، بولی، مجموعه و گویا)
- برخی قیدها شامل تمایز، تعداد و ...
- جستجوی قابل تنظیم (جستجوی سفارشی، مبتنی بر فعالیت، همسایگی‌های بزرگ و ...)
- عبور از تصادم (عقبگرد مبتنی بر تصادم، عقبگرد پویا، اصلاح مسیر و ...)

این نرم افزار همچنین شامل امکاناتی برای تعامل با فرآیند جستجو، ابزارهایی برای کمک به مدل سازی و بسیاری از نمونه‌ها می‌باشد. Choco Solver افزونه‌های بسیاری نیز از جمله تجزیه کننده

[1] Constraint Satisfaction Problem

FlatZinc[1] برای حل موارد MiniZinc[2] و یک ماژول برای حل بهتر مسائل مبتنی بر گراف مانند فروشنده دوره گرد، دارا می‌باشد.

می‌توان از MiniZinc برای مدل سازی مسائل ارضا محدودیت و بهینه‌سازی به‌صورت سطح بالا و مستقل از حل کننده استفاده کرد که دارای کتابخانه بزرگی از محدودیتهای از پیش تعریف شده می‌باشد. سپس مدل ایجاد شده به FlatZinc ترجمه می‌شود که یک زبان ورودی حل کننده می‌باشد و توسط طیف گسترده ای از حل کننده‌ها درک می‌شود.

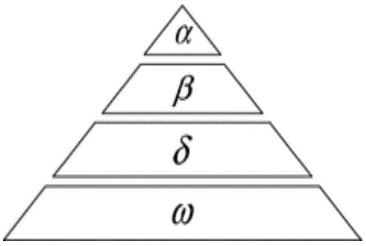
۳. روش جستجو مبتنی بر الگوریتم گرگ خاکستری

الگوریتم GWO[3] (Mirjalili et al, 2014) از یک سلسله مراتب رهبری و سازوکار شکار گرگ‌های خاکستری در طبیعت تقلید می‌کند. در این الگوریتم از چهار نوع گرگ خاکستری با نام‌های آلفا، بتا، دلتا و امگا برای شبیه‌سازی سلسله مراتب رهبری و فرآیند شکار استفاده شده است. علاوه بر این، سه مرحله اصلی شکار شامل جستجوی طعمه، محاصره و حمله به طعمه نیز شبیه‌سازی می‌شوند که در ادامه به شرح آنها می‌پردازیم.

۳,۱. سلسله مراتب و رفتار اجتماعی گرگ‌های خاکستری

در دنیای واقعی، گرگ‌های خاکستری از یک سلسله مراتب اجتماعی پیروی می‌کنند که در شکل ۱ نشان داده شده است.

سلسه مراتب مطرح شده در شکل ۱ به این صورت است که گروه آلفا (α)، حکم رهبران گروه را دارند که در واقع یک جنس نر یا ماده هستند. آلفا مسئولیت تصمیم‌گیری در مورد شکار، محل خواب و غیره را به عهده دارد. خصوصیت شاخص گرگ‌های آلفا در کیفیت مدیریت گروه آنها می‌باشد و لزوما قوی‌ترین اعضای گروه نیستند. این امر بیانگر این است که در اجتماع گرگ‌های خاکستری، نظم و سازمان‌دهی از اهمیت بالاتری نسبت به قدرت برخوردار است.



شکل۱- سلسله مراتب گرگ‌های خاکستری

در سطح دوم، گروه بتا قرار می‌گیرد که در تصمیم‌گیری‌ها یا دیگر فعالیت‌های گروه، به آلفا کمک می‌کنند. گرگ بتا می‌تواند نر یا ماده باشد. ضمن اینکه درصورت پیر و فرسوده شدن یا مرگ آلفا، گرگ‌های بتا بهترین نامزد برای جایگزینی آلفا، می‌باشند. با توجه به سلسه‌مراتب گروه، گرگ بتا از آلفا فرمان می‌گیرد و به سطوح پایینی خود فرمان می‌دهد و می‌توان گفت بتا نقش مشاور آلفا و سازمان‌دهنده گروه را بازی می‌کند. در نهایت، در پایین‌ترین سطح از گرگ‌های خاکستری، امگا قرار می‌گیرد که همیشه باید مطیع دیگر گرگ‌ها باشند. در این میان چنانچه گرگی در دسته آلفا، بتا و گاما قرار نگیرد، زیردست یا دلتا نامیده می‌شود. می‌توان گفت پیشاهنگان، نگهبانان، ارشدان ،شکارچیان و مراقبان به این دسته تعلق دارند که به ترتیب وظیفه مراقبت از مرزهای قلمرو، حفظ و تضمین امنیت گروه، ارائه تجربیات (در واقع ارشدان، گرگ‌های با تجربه‌ای هستند که قبلاً عنوان آلفا یا بتا را داشته‌اند)، شکار طعمه و تأمین مواد غذایی، مراقبت از گرگ‌های ضعیف را به عهده دارند.

علاوه بر سلسله مراتب اجتماعی گرگ‌ها، از دیگر رفتارهای جالب گرگ‌های خاکستری، شکار گروهی آنها است که روند آن به شرح زیر می‌باشد:

[1] FlatZinc is a low-level modelling language for constraint problems

[2] MiniZinc is a free and open-source constraint modeling language

[3] Gray Wolf Optimizer

ردیابی، تعقیب و نزدیک شدن به طعمه

تعقیب، محاصره و آزار طعمه تا زمانی که از حرکت باز ایستد

حمله به سمت طعمه و شکار

۳,۲. مدل ریاضی رفتار گرگ‌های خاکستری

مدل‌های ریاضی مربوط به سلسله مراتب اجتماعی، ردیابی، محاصره و حمله به طعمه به صورت زیر می‌باشند:

۳,۳. سلسله مراتب اجتماعی

همانطور که ذکر شد، در دنیای واقعی، گرگ‌ها دارای یک سلسه مراتب هستند که در راس آنها آلفا قرار می‌گیرد. بنابراین در هنگام طراحی ریاضی الگوریتم نیز این امر مد نظر قرار می‌گیرد و به‌صورت یک فرآینده تکرار شونده در هر دور الگوریتم مناسب‌ترین راه‌حل به عنوان آلفا (α) در نظر گرفته می‌شود و به همین ترتیب، دومین و سومین بهترین راه-حل‌ها به بتا (β) و دلتا (δ) تعلق می‌گیرد و در نهایت مابقی راه‌حل‌های کاندید شده امگا (ω) نامیده می‌شوند. در واقع گرگ‌های آلفا، بتا و دلتا بنابر تسلط آنها در یافتن بهترین پاسخ و قرارگیری در بهترین موقعیت تعریف می‌شوند. در شکل ۸-۲ می‌توان نحوه تعریف گرگ‌های آلفا، بتا، دلتا و امگا را در الگوریتم GWO مشاهده نمود. ضمن اینکه در الگوریتم GWO، α، β و δ موظفند فرآیند شکار (بهینه‌سازی) را هدایت نمایند و گروه ω نیز باید از سه گروه بیان‌شده پیروی کنند.

۳,٤. محاصره طعمه

از جمله رفتارهای گرگ‌های خاکستری، محاصره طعمه است. حال آنکه برای مدل‌سازی ریاضی این اقدام از دو رابطه استفاده می‌شود که در رابطه‌های (۱) و (۲) ارائه شده‌اند.

$$D = |C.X_P(t) - X(t)| \qquad (1)$$

$$X(t+1) = X_P(t) - A.D \qquad (2)$$

که t بیانگر تعداد تکرار (مرحله اجرا)، A و C، بیانگر بردارهای ضریب، $\vec{X}_p$، بیانگر بردار موقعیت شکار و $\vec{X}$ مبین بردار موقعیت گرگ خاکستری (عامل جستجو) و $\vec{D}$ فاصله از شکار می‌باشند.

همانطورکه مطرح‌شد، بردارهای A و C بردارهای ضریب می‌باشند که به ترتیب از طریق رابطه‌های (۳) و (۴) بدست می‌آیند [1]. برای بدست آوردن بردارهای ضریب، از متغیرهای a و r استفاده شده است که در طول تکرار، a به‌صورت خطی از ۲ تا ۰ کاهش می‌یابد و r1 و r2 بیان‌کننده بردارهای تصادفی هستند. برای بیان نحوه عملکرد مطالب ذکرشده می‌توان شکل ۸-۳ را در نظر گرفت. رابطه‌ای که به‌منظور به-روزرسانی مقدار a مورد استفاده قرار می‌گیرد به صورت رابطه (۵) می-باشد [1].

$$A = 2a.r1 - a \qquad (3)$$

$$C = 2.r2 \qquad (4)$$

همانطور که مطرح شد، t بیان‌کننده دور فعلی و T مبین حداکثر تعداد تکرار می‌باشد.

$$\vec{a} = 2 - t\left(\frac{2}{T}\right) \qquad (5)$$

شایان ذکر است گرگ از طریق بردارهای تصادفی r1 و r2 می‌تواند در موقعیت‌های مابین نقاط نشان‌داده شده، قرار گیرد. بنابراین گرگ می-تواند با استفاده از رابطه‌های (۶) و (۷) موقعیت خود را در فضایی که طعمه را در برگرفته، تغییر دهد. تمامی این مفاهیم قابل تعمیم به یک فضای n-بعدی نیز می‌باشد.

۳,۵. شکار

روند شکار توسط آلفا هدایت می‌شود و این در حالی است که گرگ‌های بتا و دلتا نیز گاها در شکار شرکت می‌کنند. به‌منظور شبیه‌سازی ریاضی رفتار گرگ‌ها، فرض این است که آلفا بهترین راه‌حل موجود را می‌یابد و بعد از آن بتا و دلتا در مورد مکان بالقوه طعمه آگاهی بهتری دارند. حال

با ذخیره کردن سه مورد از بهترین راه‌حل‌های بدست آمده، باید دیگر عوامل جستجو (از جمله گرگ‌های امگا) موقعیت خود را با توجه به موقعیت بهترین عوامل جستجو، به‌روزرسانی کنند. در این رابطه می‌توان رابطه‌های (۶)، (۷) و (۸) را در نظر گرفت [1].

$$\vec{D}_{\alpha} = |C_1.\vec{X}_{\alpha} - X|, \vec{D}_{\beta} = |C_2.\vec{X}_{\beta} - X|, \vec{D}_{\delta} = |C_3.\vec{X}_{\delta} - X| \quad (6)$$

$$X_1 = \vec{X}_{\alpha} - A_1.\vec{D}_{\alpha}, X_2 = \vec{X}_{\beta} - A_2.\vec{D}_{\beta}, X_3 = \vec{X}_{\delta} - A_3.\vec{D}_{\delta} \quad (7)$$

$$X(t+1) = \frac{X_1 + X_2 + X_3}{3} \quad (8)$$

۳,٦.حمله به طعمه (بهره برداری)

حمله زمانی صورت می‌گیرد که شکار متوقف شود. بنابراین برای مدل سازی ابتدا برای نزدیک شدن به طعمه، مقدار a را کاهش می‌دهیم. شایان ذکر است محدوده نوسان بردار A نیز توسط a کاهش می‌یابد. در حقیقت مولفه‌های بردار A مقداری تصادفی در بازه [-2a,2a] است، در حالی که a در طول تکرارها از مقدار ۲ تا ۰ کاهش می‌یابد. زمانی که مقادیر تصادفی A در بازه [-1,1] قرار دارند، موقعیت بعدی یک عامل جستجو می‌تواند در هر موقعیتی مابین موقعیت فعلی آن و موقعیت طعمه باشد. شکل ۸-۵ نشان می‌دهد اگر مقدار |A|<1 باشد گرگ‌ها مجبور به حمله به سمت طعمه خواهند شد.

۳,۷.جستجو برای طعمه

همانطور که مطرح‌شد گرگ‌ها با توجه به موقعیت آلفا، بتا و دلتا به فرآیند جستجو می‌پردازند. آنها برای جستجوی شکار از یکدیگر فاصله گرفته و برای حمله به آن، به یکدیگر نزدیک می‌شوند. برای مدل کردن این واگرایی از بردار A با مقدار تصادفی بزرگتر از 1 یا کوچکتر از -1 استفاده می‌کنیم تا عامل جستجو را ملزم به واگرایی و فاصله از طعمه نمائیم. در صورتیکه مقدار |A|>1 باشد گرگ‌ها وادار به واگرایی از طعمه و یافتن شکاری مناسب‌تر می‌گردند. در صورتیکه مقدار |A|<1 باشد گرگ‌ها اقدام به حمله خواهند نمود.

از دیگر مقادیر تاثیرگذار در فرآیند شناسایی، مقدار C است. با توجه به رابطه (۸)، بردار C مقادیری تصادفی در بازه [0,2] می‌گیرد. در حقیقت C، برای شکار وزن‌هایی تصادفی مشخص می‌کند تا تاثیر موقعیت طعمه را در تعیین فاصله در معادله (۶)، شدت (C>1) یا ضعف (C<1) بخشد. شایان ذکر است C نسبت به A به‌صورت خطی کاهش نمی‌یابد. ضمن اینکه در همه حال به C نیاز داریم تا مقادیری تصادفی فراهم کند و فرآیند شناسایی را نه تنها در تکرار اولیه، بلکه در تکرار نهایی اجرا نماید. این مؤلفه در جلوگیری از گیرکردن در بهینه‌های محلی، به خصوص در تکرار نهایی بسیار مفید است.

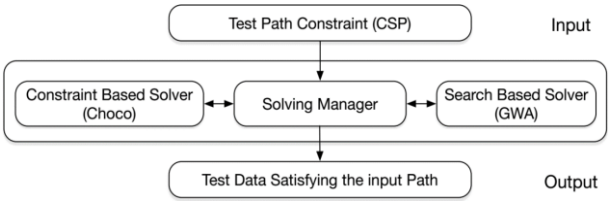
٤.مدل پیشنهادی

به‌طور کلی روشهای فرا ابتکاری جستجو مانند ژنتیک یا گرگ خاکستری در برخورد با قیدهای مقایسه ای که دارای شرط تساوی هستند دچار مشکل می‌شوند که باعث می‌شود در تکرارهای کمتر الگوریتم نمیتوان به مقادیر ارضا کننده کل قید مسیر دست یافت. از طرف دیگر حل کننده‌ها دارای توان محدودی در حل قیدها می‌باشند، برای مثال اگر قیدها دارای مقایسه‌های پیچیده غیر خطی باشند یا بخشی از کد در دسترس نباشد درباره مقادیر ارضا کننده قید نمیتواند تفسیری داشته باشد.

در مدل پیشنهادی در مرحله اول بخشهای از قید مسیر که دارای شرط تساوی هستند به حل کننده ارسال می‌شوند تا مقدار اولیه برای آنها انتخاب شوند. این فرآیند به تعداد جمعیت اولیه مورد نظر (عامل‌های جستجو) فراخوانی می‌شود تا مقادیری متنوع برای شروع جستجو تنظیم شود. در مرحله دوم فرآیند جستجو از طریق الگوریتم گرگ خاکستری شروع شده و تا رسیدن به شرط پایان ادامه می‌یابد. در صورتیکه راه حلی برای قید مسیر تولید نشده باشد، مقادیر ابتدایی توسط حل کننده تغییر یافته و الگوریتم جستجو دوباره فراخوانی خواهد شد. شکل ۱ شمایی کلی از روش حل قید مسیر را نمایش می‌دهد.

٤,١. پیاده سازی

بنابر توضحیات ارائه‌شده، شبه کد روش پیشنهادی که ترکیبی از حل کننده choco و الگوریتم GWO بشرح زیر خواهد بود. سه مورد از بهترین پاسخ‌ها را ذخیره می‌کند که این اولین، دومین و سومین بهترین پاسخ‌ها به ترتیب به گرگ‌های آلفا، بتا و دلتا تعلق می‌گیرند.



شکل ۱- روش حل قید مسیر جهت تولید داده آزمون

Algorithm 1 Constrained GW based test data generator

Input: $Path\ Constraint\ PC(V, D, L)$
Output: a solution for PC
Initialize: Maximum process's iteration: M_P, Maximum GW's iteration: M_G, p=1;

1: **while** (P < M_p) and ($F_\alpha > 0$) **do**
2: ChocoSolver.solve($v \in V, d \in D, l \in L$); ▷ alternate initial values
3: Gray wolf population $X_i (i = 1, ..., n)$, a, A and C;
4: Calculate the fitness of all search agents;
5: X_α = the first best search agent;
6: X_β = the second best search agent;
7: X_δ = the third best search agent;
8: F_α = fitness(X_α);
9: t = 1;
10: **while** (t < M_G) and ($F_\alpha > 0$) **do**
11: **for each** search agent **do**
12: Update the position of the search agent by equation 8;
13: Update a, A, and C;
14: Calculate the fitness of all search agents;
15: Update X_α, X_β, X_δ;
16: F_α = fitness(X_α);
17: t = t + 1;
18: p = p + 1;
19: return X_α;

الگوریتم۱- تولید داده آزمون به روش گرگ خاکستری مقید

نکته‌ای که در الگوریتم ۱ مشاهده می‌شود، مقداردهی اولیه به دو پارامتر A و C است. در حقیقت استفاده از پارامتر A منجر به اجتناب از گیر افتادن در بهینه محلی می‌شود. همان‌طور که پیش‌تر ذکر شد نیمی از تکرارها به مرحله اکتشاف |A|>=1 و بقیه به مرحله بهره-برداری |A|<1 اختصاص داده می‌شوند. بنابراین، این مکانیسم به الگوریتم GWO کمک می‌کند تا به‌صورت همزمان حداقل‌های محلی را شناسایی و از آنها اجتناب کند و بتواند به نتایج مطلوب و مناسبی دست یابد [1]. نکته دیگری که در الگوریتم ۱ مورد توجه قرار می‌گیرد، به‌روزرسانی موقعیت گرگ و قرارگیری در موقعیت X(t+1) می‌باشد که از طریق رابطه (۲) انجام می‌شود.

با توجه به این شکل می‌توان چگونگی به‌روزرسانی موقعیت گرگ‌ها را بر اساس رابطه ریاضی که در رابطه (۲) بیان شده است را درک نمود. در این رابطه، موقعیت بعدی گرگ، بر اساس موقعیت طعمه و با در نظر گرفتن فاصله میان گرگ و طعمه، مشخص می‌شود. بنابراین متناسب با جایگاه طعمه، گرگ می‌تواند در یک فضای دایره‌ای (در یک فضای دو بعدی)، در یک فضای کره‌ای (در یک فضای سه بعدی) یا یک فضای فراکروی (در یک فضای n بعدی) حول طعمه و با استفاده از رابطه (۲) حرکت کند. در نهایت مسئله دیگری که مطرح می‌شود تشخیص میزان خوب بودن موقعیت هر گرگ است که از طریق یک تابع برازندگی مشخص می‌شود.

٤,٢. تابع برازش مبتنی بر سختی انشعاب

روش‌های مبتنی بر جستجو به‌منظور تولید داده‌ی آزمون، از راهنمایی تابع برازش استفاده می‌کنند. در حقیقت می‌توان گفت روش‌های جستجوی بدون تابع برازش، همان روش‌های تصادفی می‌باشند. به عنوان یک تعریف ساده می‌توان گفت، تابع برازش تابعی است که راه‌حل کاندید برای یک مسئله را به عنوان ورودی دریافت می‌کند و یک خروجی را که مشخص‌کننده میزان خوب بودن راه‌حل مورد نظر را ارائه می‌کند. بنابراین در مسائل مطرح‌شده هرچه تابع برازش دقیق‌تر و عملگرهای تکاملی آگاهانه‌تر باشند، الگوریتم نیز دقیق‌تر عمل کرده و بیشتر با الگوریتم‌های تصادفی فاصله می‌گیرد.

به‌طور معمول کاندیدای داده آزمونی که انشعاب‌های سخت تر را برآورده می‌کند امیدوار کننده تر از آنهایی است که فقط شاخه‌های "آسان" را برآورده می‌کنند [6]. بنابراین دشواری برآوردن یک مجموعه محدودیت (به عنوان مثال یک گزاره شرطی در یک جمله شرطی یا یک انشعاب) به عنوان اطلاعات کلیدی برای سنجش برازندگی یک کاندیدا در نظر گرفته می‌شود. در این فصل از یک تابع برازش مبتنی بر سختی برای هدایت الگوریتم جستجوی گرگ خاکستری به سمت هدف آزمون استفاده کرده ایم. در این روش اهداف را با توجه به میزان سختی برآورده شدن اولویت بندی می‌کنیم. مشابه روش نمادین پیشرفته، علاوه بر تحلیل ایستای شاخه‌های غیر اجرایی یک مقدار درجه سختی نیز به تمام انشعاب‌ها اضافه می‌شود.

سختی برآوردن یک محدودیت c با تعدد[۱] و استحکام[۲] آن ارتباط دارد [7]. استحکام، نسبت تعداد (تقریبی) راه حلها نسبت به اندازه فضای جستجوی آن (یعنی ضرب دکارتی دامنه متغیرهای درگیر در c) می‌باشد. تعدد قید نیز، یعنی آزادی کمتر در انتخاب برخی از متغیرها برای تغییر کاندیدای آزمون را نشان می‌دهد. استحکام نزدیک به ۰ نشان از میزان بالا بودن دشواری و سختی برآوردن یک محدودیت می‌باشد. به همین ترتیب پارامترهای $\alpha(c) = 1/\text{arity}(c)$ و $1 = (c)$ $\beta-$ projection-tightness(c) که مقادیر بین ۰ و ۱ بوده و سختی را نشان می‌دهد.

٤,٣. ضریب سختی (DC)[۳]

DC یک عدد حقیقی بزرگتر از ۱ بوده و گویای سختی احتمالی یک شاخه می‌باشد. هر قید ضریب سختی خاص خود را دارد که با توجه به ساختار و استحکام آن تعیین می‌شود. ضریب سختی توسط فرمول زیر محاسبه می‌شود:

که $B>1$ بعنوان یک پارامتر تقویت می‌باشد.

برای بیان عملکرد تابع برازش مبتنی بر ضریب سختی، از DC به عنوان ضریب پنالتی برای شکستن یک قید استفاده می‌کنیم. ایده اصلی در این تابع برازش تعیین فاصله شاخه استاندارد و سپس مرتب کردن کاندیداها با توجه به فاصله شاخه استاندارد کل آنها است.

برای محاسبه مقدار برازش یک نامزد آزمون i در مجموعه ای از شاخه‌های (محدودیت‌ها) C فرمول ۱۰ را اعمال می‌کنیم[7][6]:

$$DC(c) = (B^2 \times \propto (c)) + (B \times \beta(c)) + 1 \quad (9)$$

$$F_{DC}(i,c) = \sum_{c\epsilon C} DC(c) \times \eta(i,c) \quad (10)$$

این تابع برازش میزان پنالتی محدودیت مشخص را به صورت نسبی مطابق با ضریب سختی و فاصله شاخه ای تعیین می‌کند. به این ترتیب ممکن است کاندیدایی را ترجیح دهیم که یک شاخه سخت را برآورده کند اما شاخه آسانتر را با یک فاصله نرمال بزرگ شکسته باشد. اما کاندیدایی که شرط سخت تر را با یک فاصله نرمال کوچکتر شکسته باشد ولی شاخه آسانتر را برآورده کرده باشد، مورد اقبال قرار نگرفته باشد.

٤,٤. ارزیابی الگوریتم پیشنهادی

هدف در این مطالعه تجربی تحلیل تأثیر تابع برازش پیشنهادی به‌همراه روش گرگ خاکستری مقید از منظر اثربخشی و کارآیی در فرآیند تولید داده آزمون می‌باشد. مدل پیشنهادی کارآیی بالاتری خواهد داشت اگر قادر به کاهش تعداد تکرار الگوریتم باشد. در صورتیکه مدل پیشنهادی بتواند اهداف بیشتری نسبت به مدل‌های جاری را پوشش دهد، اثربخش تر خواهد بود.

در این بخش روش ارائه‌شده از منظر درصد پوشش اهداف آزمون و تعداد تکرار مراحل ، مورد ارزیابی قرار گرفته است و نتایج حاصل را با دیگر الگوریتم‌ها و توابع برازش موجود از جمله f_{AL} و f_{SE} به‌همراه

[۱] Arity
[۲] Projection Tightness
[3] Difficulty Coefficient

الگوریتم ژنتیک مورد ارزیابی قرار گرفته است. شایان ذکر است که اطلاعات مندرج در جدول میانگین ۲۰ بار اجرای الگوریتم می‌باشد.

٤,٥. معیارهای ارزیابی

به‌منظور ارزیابی روش‌های مطرح‌شده، آزمایش بر روی ۵ برنامه با عنوان محک‌های استاندارد[1] انجام گرفته است. در حقیقت محک‌های استاندارد، برنامه‌های استاندارد و شناخته‌شده هستند که برای ارزیابی روش‌های تولید داده‌ی آزمون خودکار مورد استفاده قرار می‌گیرند.

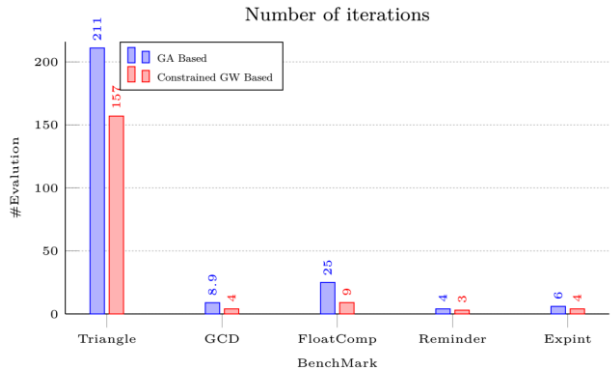
معیارهایی که در این بخش برای بررسی و مقایسه کارایی روش‌ها مورد استفاده قرار می‌گیرند عبارتند از درصد پوشش اهداف آزمون (پوشش انشعاب)، تعداد تکرار هر الگوریتم و همچنین زمان اتمام جستجو که در جدول ۱ نمایش داده شده است[8] .

BenchMark	CC	GA based TDG			Constrained GW based TDG		
		Evaluation	Time (ms)	Coverage	Evaluation	Time (ms)	Coverage
Triangle	4	211	29310	86.6%	157	16897	100%
GCD	5	9	2352	93.3%	4	845	100%
Floatcomp	5	25	1272	96.6%	9	736	100%
Reminder	4	4	427	100%	3	295	100%
Expint	12	6	563	100%	4	408	100%

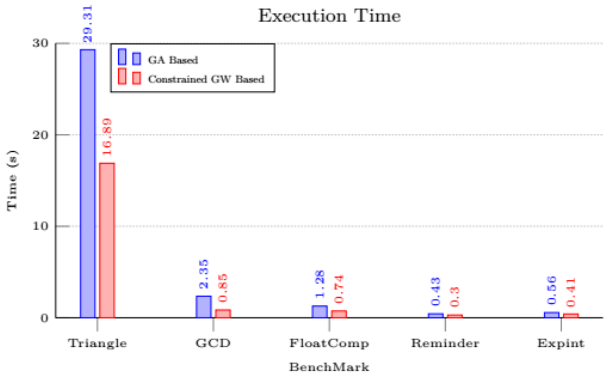
جدول ۱. مقایسه الگوریتم‌های ژنتیک و گرگ خاکستری مقید

نمودارهای ۱ و ۲ جهت مقایسه دقیقتر دو الگوریتم تولید داده آزمون ارائه شده اند. نمودار ۸-۱ نشان دهنده تعداد مقایسه‌های انجام شده جهت رسیدن به داده‌های آزمون مطلوب می‌باشد که میانگین ۲۰ بار اجرای هر الگوریتم بر روی هر یک از برنامه‌های محک می‌باشد. همانطور که مشاهده می‌شود الگوریتم پیشنهادی در تمام موارد دارای عملکرد مطلوب تری نسبت به الگوریتم ژنتیک موجود می‌باشد. در برنامه محک تعیین نوع مثلث به‌دلیل وجود عملگر تساوی در مجموعه شرطهای مسیرهای آزمون، تعداد مقایسه‌ها و در نتیجه آن زمان اجرای هر دو الگوریتم مقدار بالاتری نسبت به دیگر برنامه‌های محک دارا می‌باشد.

نمودار ۲ معرف میانگین زمان اجرای هریک از الگوریتم‌های جستجوی داده آزمون بر روی برنامه‌های محک می‌باشد. مقادیر ثبت شده نشان دهنده بهبود زمان اجرا در روش تولید داده آزمون مبتنی بر الگوریتم گرگ خاکستری نسبت به روش مبتنی بر الگوریتم ژنتیک می‌باشد.



نمودار ۱. تعداد ارزیابی‌های انجام شده جهت رسیدن به هدف جستجو



نمودار ۲. میانگین زمان اجرای هر الگوریتم

به‌طور کلی بر اساس نتایج حاصل از اجرای روش پیشنهادی که ترکیبی از روشهای مبتنی بر قید و مبتنی بر جستجو می‌باشد، نسبت به روش صرفا مبتنی بر جستجو به شرح زیر می‌باشد:

۱- روش ترکیبی اثر بخشی بیشتری نسبت به روش مبتنی بر جستجو دارد به‌طوریکه در تمام برنامه‌های محک ارزیابی شده به پوشش ۱۰۰درصد دست یافته است.

[1] Standard Benchmarks

۲- روش ترکیبی نسبت به روش مبتنی بر جستجو دارای بهره وری مناسب تری می‌باشد به‌طوریکه در تمام برنامه‌های محک به‌طور قابل ملاحظه ای دارای زمان اجرای پایین تری می‌باشد.

۳- در برنامه‌هایی که دارای شرط تساوی می‌باشند، روش مبتنی بر جستجو با وجود اجرای طولانی تر نتوانسته است به پوشش ۱۰۰ درصد دست یابد. اما روش ترکیبی در زمانی به مراتب کمتر به پوشش ۱۰۰ درصد دست یافته است.

۵. جمع بندی

این مقاله روشی موثر با کارآیی بالا از نظر زمان اجرا و توان حل قیدهای پیچیده جهت تولید داده‌های آزمون برنامه‌هایی با تعداد خطوط بالا و ساختار پیچیده پیشنهاد کرده است. به‌طور خاص یک الگوریتم ترکیبی مبتنی بر قید (با استفاده از حل کننده choco) و مبتنی بر جستجو (با استفاده از روش جستجوی گرگ خاکستری) ارائه کرده ایم که یک معادله مسیر آزمون هدف را به‌صورت (لیست متغیرها، دامنه متغیرها و محدودیت‌ها) را بعنوان ورودی دریافت کرده و مقادیر اختصاص داده شده به هریک از متغیرهای موجود در مسیر را بعنوان خروجی تولید می‌کند. الگوریتم پیشنهادی در مقابل مسیرهای دارای قید تساوی عملکرد بسیار مناسب تری داشته است در حالیکه الگوریتم‌های فرا ابتکاری در چنین قیدهایی دچار مشکل می‌شوند. دو بخش اصلی روش ترکیبی پیشنهاد شده به زبان جاوا پیاده سازی شده و نسبت به روشهای موجود مورد مقایسه قرار گرفته است. نتایج آزمایشی نشان می‌دهند که روش پیشنهاد شده به‌طور قابل ملاحظه ای عملکردی کارآتر و اثر بخش تری نسبت به روشهای موجود برای برنامه‌های دارای شرط تساوی داشته است.

مراجع

[1] Faris, H., Aljarah, I., Al-Betar, M. A., & Mirjalili, S. (2018). Grey wolf optimizer: a review of recent variants and applications. Neural computing and applications, 30, 413-435.

[2] P. Ammann and J. Offutt. Introduction to software testing. Cambridge University Press, 2016.

[3] E. Fazli and M. Afsharchi, "A Time and Space-Efficient Compositional Method for Prime and Test Paths Generation," in *IEEE Access*, vol. 7, pp. 134399-134410, 2019, doi: 10.1109/ACCESS.2019.2941429.

[4] P. M. S. Bueno and M. Jino. Automatic test data generation for program paths using genetic algorithms. International Journal of Software Engineering and Knowledge Engineering, 12(06):691–709, 2002.

[5] Jussien, N., Rochart, G., & Lorca, X. (2008). Choco: an open source java constraint programming library. In CPAIOR'08 Workshop on Open-Source Software for Integer and Contraint Programming (OSSICP'08) (pp. 1-10).

[6] A. Dwarakanath and A. Jankiti. Minimum number of test paths forprime path and other structural coverage criteria. In IFIP International Conference on Testing Software and Systems, pages 63–79. Springer, 2014.

[7] S. C. Ntafos and S. L. Hakimi. On path cover problems in digraphs and applications to program testing. IEEE Transactions on Software
Engineering, (5):520–529, 1979.

[8] B. A. Nejmeh. Npath: a measure of execution path complexity and its applications. Communications of the ACM, 31(2):188–200, 1988.