Journal of Soft Computing and Information Technology (JSCIT)
Babol Noshirvani University of Technology, Babol, Iran
Journal Homepage: jscit.nit.ac.ir
Volume 9, Number 3, Fall 2020, pp. 67-82
Received: 07/22/2019, Revised: 01/28/2020, Accepted: 03/06/2020

A non-dominated genetic algorithm for constructing balanced spanning tree

Marzieh Hajizadeh Tahan[1], Mohammad Ghasemzadeh[2*] and Ali Mohammad Latif[3]

1, 2*, 3 - Computer Engineering Department, Yazd University, Yazd, Yazd, Iran.

[1]mh.tahan@stu.yazd.ac.ir, [2*]m.ghasemzadeh@yazd.ac.ir, and [3]alatif@yazd.ac.ir

Corresponding author address: Mohammad Ghasemzadeh, Computer Engineering Department, Yazd University, Yazd, Iran, Post Code: 89168 – 69511.

Abstract- This research shows how we can find Balanced Spanning Trees, without the need to determine the relevant parameters, by using an evolutionary multi-objective algorithm. This problem belongs to the complexity class NP-complete; therefore, for big graphs, we cannot use exact algorithms, which often rely on exhaustive search. The existing approximate algorithms are all limited to obtaining the related parameter(s) and considering them separately during their computations; this usually leads to finding a solution with lower quality than desired. On the other hand, the existing methods solve the problem of the balanced spanning tree as a single objective; which loss search space information and finding only one solution, while in real-world problems, decision-makers often need several solutions to make a better decision. Overcoming the above-mentioned shortcomings leads to finding a balanced spanning tree with better characteristics; which in turn, yields more benefits in relevant applications, such as communication networks and high-performance computing projects. In this research, in order to overcome the above challenge, the use of evolutionary multi-objective optimization via Non-dominated Sorting Genetic Algorithm - NSGA is recommended. The proposed solution uses two objective functions to simultaneously optimize the weight of the spanning tree and the shortest path. The first objective function attempts to minimize the distance between each vertex and the root of the tree. The second objective function is selected to minimize the weight of the obtained spanning tree. The proposed method was implemented and run in a Python environment by specialized functions, on a seven-core microcomputer. In order to evaluate the performance of the proposed algorithm, a set of random graphs by Erdos and Renyi approaches were used. The proposed algorithm was compared with the state of the art methods in the problem of finding the balanced spanning tree. Experimental results show that the proposed algorithm is often able to find better responses than the other algorithms compared. The experimental results show that the proposed algorithm was always able to find highly ranked Balanced Spanning Trees. Meanwhile, often the optimal solutions, which can only be obtained through exact and very costly algorithms, were found, with teeny computations.

Keywords- Balanced Spanning Tree, Shortest Path Tree, Minimum Spanning Tree, Evolutionary Multi-Objective Problems.

مجله علمی پژوهشی رایانش نرم و فناوری اطلاعات
دانشگاه صنعتی نوشیروانی بابل
صفحه مجله: jscit.nit.ac.ir
جلد ۹، شماره ۳، پائیز ۱۳۹۹ ، صفحه ۶۸-۷۲
دریافت: ۱۳۹۸/۰۴/۳۱، بازنگری: ۱۳۹۸/۱۱/۰۸، پذیرش: ۱۳۹۸/۱۲/۱۶

الگوریتم ژنتیک با مرتب‌سازی نامغلوب برای ساخت درخت پوشای متوازن

مرضیه حاجی‌زاده طحان[۱]، محمد قاسم‌زاده*[۲]، علی‌محمد لطیف[۳]

۱،*۲،۳- گروه مهندسی کامپیوتر، دانشکده فنی و مهندسی، دانشگاه یزد، یزد، ایران.

[1]mh.tahan@stu.yazd.ac.ir, [2*]m.ghasemzadeh@yazd.ac.ir, and [3]alatif@yazd.ac.ir

* نشانی نویسنده مسئول: محمد قاسم‌زاده، یزد، خیابان پژوهش، دانشگاه یزد، ساختمان فنی۱، گروه مهندسی کامپیوتر، کد پستی: ۸۹۱۶۸-۶۹۵۱۱

چکیده- این پژوهش نشان می‌دهد که در رابطه با یافتن درخت پوشای متوازن، چگونه می‌توان با بهره‌گیری از الگوریتم چند هدفه تکاملی بدون نیاز به تعیین پارامترهای مربوطه به نمونه‌های با شرایط مطلوب دست یافت. این مسئله متعلق به کلاس پیچیدگی اِن-پی-کامل است، لذا برای ورودی‌های قدری بزرگ نمی‌توان از یک الگوریتم دقیق که غالباً متکی بر جستجوی همه‌جانبه است، بهره برد. الگوریتم‌های تقریبی موجود از یک سو همگی محدود به دریافت پارامتر مرتبط و لحاظ نمودن آن‌ها به‌طور مجزا می‌باشند؛ این امر موجب می‌شود که به‌طورمعمول پاسخ‌هایی باکیفیت پایین‌تر از مطلوب را بیابند. از سویی دیگر روش‌های موجود، مسئله درخت پوشای متوازن را به صورت تک هدفه حل می‌کند؛ که این امر منجر به از دست رفتن اطلاعات فضای جستجوی مسئله و یافتن تنها یک راه‌حل می‌گردد، درحالی‌که در مسائل واقعی، غالباً تصمیم‌گیرندگان برای تصمیم‌گیری بهتر، به چندین نمونه‌ی راه‌حل نیاز دارند. رفع کاستی‌های فوق می‌تواند منجر به یافتن درخت پوشای متوازن با ویژگی‌های برتری گردد، که به‌تبع آن منافع بیشتری در کاربردهای مربوطه مانند شبکه‌های ارتباطی و محاسبات با کارایی بالا، حاصل آید. در این پژوهش، برای غلبه بر چالش‌های فوق، به‌کارگیری بهینه‌سازی چندهدفه تکاملی از طریق الگوریتم ژنتیک با مرتب‌سازی نامغلوب توصیه می‌گردد. راه‌حل پیشنهادی از دو تابع هدف برای بهینه‌سازی هم‌زمان وزن درخت پوشا و کوتاه‌ترین مسیر بهره می‌برد. تابع هدف اول، سعی در حداقل‌سازی فاصله هر رأس تا ریشه درخت دارد، تابع هدف دوم به‌گونه‌ای انتخاب شده تا وزن درخت پوشای به‌دست‌آمده کمینه باشد. روش پیشنهادی توسط توابع تخصصی و در محیط پایتون، بر روی یک میکروکامپیوتر هفت هسته‌ای پیاده‌سازی و اجرا گردید. به‌منظور ارزیابی عملکرد الگوریتم پیشنهادی، مجموعه‌ای از گراف‌های تصادفی با رویکرد اردوس و رنی بکار گرفته شدند. الگوریتم پیشنهادی با روش‌های مطرح در حوزه یافتن درخت پوشای متوازن مورد مقایسه قرار گرفت. نتایج آزمایشی نشان می‌دهند که الگوریتم پیشنهادی، در غالب موارد قادر به یافتن پاسخ‌های بهتری نسبت به سایر الگوریتم‌های مورد مقایسه می‌باشد. ضمناً غالباً پاسخ‌های بهینه که به‌طورمعمول تنها از طریق الگوریتم‌های دقیق و بسیار پرهزینه قابل حصول‌اند، را با صرف توان محاسباتی ناچیزی می‌یابد.

واژه‌های کلیدی: درخت پوشای متوازن، درخت کوتاه‌ترین مسیر، درخت پوشای کمینه، مسائل چندهدفه تکاملی

۱- مقدمه

الگوریتم‌های تکاملی به‌طور موفقیت‌آمیزی در یافتن سریع پاسخ بهینه و یا شبه‌بهینه بسیاری از مسائل بکار گرفته شده‌اند [۱]. در این راستا، یافتن درخت پوشای متوازن[۱]، در علوم و مهندسی از اهمیت ویژه‌ای برخوردار است. یکی از کاربردهای آن افزایش سرعت و کاهش هزینه‌ها در سیستم‌های انتقال اطلاعات مانند شبکه‌های کامپیوتری و نیز شبکه‌های اجتماعی می‌باشد. به‌عنوان‌مثال در یک شبکه ارتباطی، مسئله «یافتن مسیر مناسب به منظور کاهش زمان و هزینه ارسال اطلاعات» را می‌توان به مسئله «یافتن درخت پوشای متوازن» سرایت داد [۲-۵].

درخت پوشای متوازن در واقع درخت پوشای ریشه‌داری است که از یک طرف وزن درخت به وزن درخت پوشای کمینه نزدیک باشد و از طرف دیگر فاصله رئوس تا ریشه درخت تا حد ممکن حداقل باشد. درواقع درخت پوشای متوازن، محصولِ مصالحه‌ای بین یافتن درخت پوشای کمینه و درخت کوتاه‌ترین مسیر می‌باشد.

اساساً مسئله‌ی یافتن درخت پوشای متوازنِ بهینه یک مسئله‌ی ان پی-کامل است [۵]. لذا هر الگوریتم دقیقی که برای آن ارائه شود، در حالت کلی، نمایی خواهد بود و به‌تبع آن، برای گراف‌های بزرگ و حتی قدری بزرگ ناکارآمد خواهد بود. بنابراین برای اینکه بتوان با هزینه قابل قبولی به پاسخ‌های خوبی دست یافت، از الگوریتم‌های تقریبی بهره می‌بریم. این الگوریتم‌ها غالباً پاسخ بهینه و یا نزدیک به بهینه را با صرف هزینه ناچیزی می‌یابند.

یکی از جدیدترین و بهترین روش‌های ارائه‌شده برای حل این مسئله، مبتنی بر الگوریتم‌های تکاملی است [۵]. این روش ضمن دریافت پارامترهای مرتبط، یک نمونه مناسب را می‌یابد. درواقع، محدودیت و کاستی این روش از یک طرف متوجه آن است که مسئله درخت پوشای متوازن را به صورت تک هدفه حل می‌کند؛ که این امر منجر به یافتن تنها یک راه‌حل می‌گردد، درحالی‌که در مسائل واقعی، غالباً تصمیم‌گیرندگان برای تصمیم‌گیری بهتر، به چندین نمونه‌ی راه‌حل نیاز دارند [۶]. از طرف دیگر، نیاز به تعیین پارامترها، ملزم می‌دارد که کاربر دارای دانش قبلی در مورد کیفیت و اهمیت نسبی هر کدام از آن‌ها و نیز اهداف مربوطه باشد [۷]. در این خصوص باید بیان نمود که اگرچه تعیین پارامترها در ابتدای کار حل مسئله را آسان‌تر می‌کند اما نماینده خوبی برای مسائل واقعی نیستند. بسیاری از پارامترها در مسائل واقعی ماهیت غیرقطعی و تصادفی دارند و با قطعی در نظر گرفتن داده‌های مسئله ممکن است جواب بهینه مسئله در شرایط واقعی موجه نباشد.

بسیاری از مسائل دنیای واقعی، مسائل بهینه‌سازی چند هدفه هستند و الگوریتم‌های تکاملی در چنین مسائلی کاملاً موفق هستند. تابع برازندگی با بیش از یک هدف، حاوی اطلاعات بیشتری است که در اصل می‌تواند جستجوی الگوریتم‌های تکاملی را هدایت کند. مسئله یافتن درخت پوشای متوازن، جزء مسائل بهینه‌سازی چندهدفه محسوب می‌شود. تبدیل این مسئله به یک مسئله تک هدفه برای مثال از طریق مجموع وزن‌دار مقادیر اهداف، منجر به از دست دادن اطلاعات موجود در جبهه پارتو و نقاط جستجوی مربوطه می‌شود [۸]. با بررسی الگوریتم‌های ابتکاری چندهدفه، الگوریتم ژنتیک با مرتب‌سازی نامغلوب دو[۲] (NSGA-II) برای این مسئله توصیه می‌گردد. مزیت روش NSGA-II نسبت به سایر روش‌های تکاملی چندهدفه، عدم نیاز به تعیین وزن اهداف است. از طرفی NSGA-II یک الگوریتم مبتنی بر ژنتیک است. جذاب‌ترین ویژگی الگوریتم های مبتنی بر ژنتیک انعطاف‌پذیری آن ها در انجام انواع مختلفی از عملکردهای هدف است. دلایل اصلی این موفقیت به شرح زیر است. این الگوریتم ها قادرند مسائل دشوار را به‌سرعت و اطمینان حل کنند. همچنین با مدل‌ها و شبیه‌سازی‌های موجود بسیار آسان ارتباط برقرار می‌کنند. علاوه بر این، به راحتی قابل توسعه و قابل ترکیب هستند [۹]. به طور کلی NSGA-II از سه اصل شامل مفهوم غلبه، حفظ تنوع و نخبه‌گرایی پیروی می‌کند که موجب می‌گردد که یک رویه چندهدفه تکاملی منعطف و قوی[۳] برای حل مسائل بهینه‌سازی چندهدفه باشد [۱۰].

در این پژوهش یک الگوریتم تقریبی و سریع بر اساس الگوریتم چندهدفه تکاملی NSGA-II ارائه‌شده است. راه‌حل پیشنهادی از دو تابع هدف برای بهینه‌سازی هم‌زمان وزن درخت پوشا و کوتاه‌ترین مسیر بهره می‌برد. تابع هدفِ اول، سعی در حداقل‌سازی فاصله هر رأس تا ریشه درخت دارد و تابع هدفِ دوم به‌گونه‌ای انتخاب شده است که وزن درخت پوشای به‌دست‌آمده کمینه باشد. کمینه‌سازی همزمان این دو هدف منجر به یافتن درخت پوشای متوازن با کیفیت بالاتری می‌گردد.

در الگوریتم پیشنهادی، جمعیت اولیه، اضافه بر مقداردهی تصادفی (روش بکار رفته در الگوریتم‌های قبلی موجود)، از طریق اکتشافی نیز مقدار می‌گیرند. این نوع از مقداردهی موجب می‌شود که فرآیند کشف پاسخ مسئله تسریع گردد. روش پیشنهادی بدون نیاز به تعیین پارامتر اولیه، چندین نمونه راه‌حل ارائه می‌نماید که انعطاف‌پذیری در تصمیم‌گیری و انتخاب راه‌حل مناسب را به همراه دارد. برای پیاده‌سازی روش پیشنهادی ضمن به‌کارگیری توابع

کتابخانه‌ای مرتبط، رویه‌های مورد نیاز در محیط پایتون طراحی و اجرا گردیدند. در ادامه، عملکرد آن بر روی مجموعه‌ای از گراف‌های تصادفی با رویکرد اردوس و رنی [۱۱] مورد ارزیابی قرار گرفت.

الگوریتم پیشنهادی با روش‌های مطرح در این حوزه شامل روش ارائه‌شده توسط محرم و مرسی [۵] و روش خولر و همکاران [۱۲] مورد مقایسه قرار گرفت. نتایج آزمایشی نشان می‌دهند که الگوریتم پیشنهادی با در نظر گرفتن اهداف مسئله به‌طور هم‌زمان قادر به یافتن پاسخ‌هایی با کیفیت بالاتر نسبت به سایر الگوریتم‌های مورد مقایسه می‌باشد. همچنین قادر است در غالب موارد پاسخ‌های بهینه را بیابد که به‌طورمعمول از طریق جستجوی همه‌جانبه قابل دستیابی است.

ساختار مقاله به شرح زیر است: در بخش دوم مفاهیم پایه مربوطه بیان می‌گردد. بخش سوم به بیان ادبیات و سابقه تحقیق می‌پردازد. در بخش چهارم الگوریتم پیشنهادی شرح داده می‌شود. نتایج آزمایش‌ها و ارزیابی آن‌ها در بخش پنجم ارائه‌شده است و درنهایت در بخش آخر نتیجه‌گیری انجام شده است.

۱- دانش پس‌زمینه

در این بخش به تشریح مفاهیم پایه‌ای مرتبط با مبحث پژوهشی این مقاله می‌پردازیم. در این رابطه، ابتدا درخت پوشای کمینه و درخت کوتاه‌ترین مسیر، معرفی می‌شوند. در ادامه، درخت پوشای متوازن به‌طور مبسوط تشریح می‌گردد. در دنباله آن، الگوریتم ژنتیک با مرتب‌سازی نامغلوب که نقش محوری در الگوریتم پیشنهادی دارد معرفی و مورد بحث قرار می‌گیرد.

۲-۱- درخت پوشای کمینه

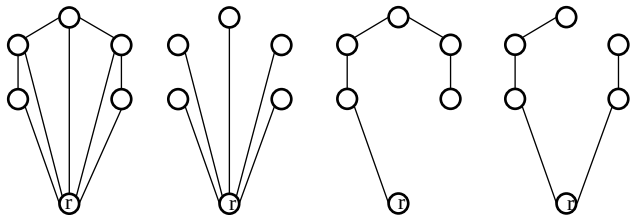
در رابطه با گراف وزن‌دار $G = (V, E)$، درخت پوشا زیرمجموعه‌ای از گراف G است که تمامی رئوس آن با کمترین مقدار یال‌های ممکن پوشش یافته است. بنابراین یک درخت پوشا دور ندارد و هیچ رأس ناهمبندی در آن دیده نمی‌شود. در گراف G درخت پوشای کمینه، درخت پوشایی است که کمترین وزن را نسبت به دیگر درخت‌های پوشای گراف داشته باشد. کروسکال و پرایم، دو الگوریتم کارآمد شناخته‌شده‌ای هستند که درخت پوشای کمینه یک گراف وزن‌دار را در زمان کثیرالجمله محاسبه می‌کنند [۵، ۱۲].

۲-۲- درخت کوتاه‌ترین مسیر

در گراف وزن‌دار $G = (V, E)$، درخت کوتاه‌ترین مسیر با ریشه r، یک درخت پوشا است به طوری که برای هر رأس v فاصله بین r و v برابر کوتاه‌ترین فاصله بین دو رأس در گراف باشد. الگوریتم دایجکسترا و بلمن فورد از الگوریتم‌های شناخته‌شده برای محاسبه درخت کوتاه‌ترین مسیر در گراف وزن‌دار می‌باشند [۵، ۱۲].

۲-۳- درخت پوشای متوازن

در گراف وزن‌دار $G = (V, E)$ با مجموعه رأس V و مجموعه یال E داده‌شده است. به ازای هر مقدار $\alpha, \beta \geq 1$ یک درخت پوشای ریشه‌دار T در G وجود دارد که (α, β)- درخت پوشا متوازن نامیده می‌شود اگر ۱) برای هر رأس v فاصله بین رأس v و ریشه در درخت T حداکثر α برابر کوتاه‌ترین فاصله بین دو رأس در G باشد. ۲) مجموع وزن T حداکثر β برابر حداقل وزن درخت در G باشد. در برخی از گراف‌های وزن‌دار، وزن کلی درخت کوتاه‌ترین مسیر ممکن است بسیار بیش‌تر از وزن درخت پوشای کمینه باشد. این در حالتی رخ می‌دهد که رأس‌های نزدیک به ریشه تعیین شده، در درخت پوشای کمینه از ریشه دور باشند [۵]. شکل (۱) نشان می‌دهد که وزن درخت کوتاه‌ترین مسیر بسیار بیشتر از وزن درخت پوشای کمینه است. فاصله رئوس تا ریشه در درخت پوشای کمینه بسیار بیشتر از فاصله رئوس تا ریشه در درخت کوتاه‌ترین مسیر است. هدف از مسئله درخت پوشای متوازن به دست آوردن درختی شبیه به درخت نشان داده‌شده در شکل (۱-د) است که بین این دو درخت توازن برقرار کند.



(الف) (ب) (ج) (د)

شکل ۱: یک مثال برای درخت پوشای متوازن. (الف) گراف G. (ب) درخت کوتاه‌ترین مسیر مربوط به گراف G. (ج) درخت پوشای کمینه مربوط به گراف G. (د) درخت پوشای متوازن مربوط به گراف G.

۲-۴- الگوریتم ژنتیک با مرتب‌سازی نامغلوب دو

الگوریتم ژنتیک با مرتب‌سازی نامغلوب دو، یکی از مهم‌ترین و پرکاربردترین الگوریتم‌های بهینه‌سازی چندهدفه برای حل مسائل بهینه‌سازی است. این الگوریتم با اضافه کردن دو عملگر مرتب‌سازی نامغلوب و فاصله ازدحام به الگوریتم ژنتیک تک هدفه بدست آمده است. مهم‌ترین مزیت NSGA-II نسبت به الگوریتم ژنتیک و سایر الگوریتم‌های چندهدفه، ایجاد یک جبهه پارتو مبتنی بر یک روش

مرتب‌سازی نامغلوب برای هر فرد و استفاده از یک روش اختصاص فاصله ازدحام برای پیاده‌سازی تخمین چگالی است. هم‌چنین در این الگوریتم نیاز به تعیین وزن برای اهداف مسئله وجود ندارد[۱۳]. این موجب می‌گردد که یک الگوریتم منعطف و قوی برای حل مسائل بهینه‌سازی چندهدفه باشد [۱۰].

مرتب‌سازی راه‌حل‌ها در الگوریتم NSGA-II بر اساس مفهوم غلبه صورت می‌گیرد. به این صورت که راه‌حل‌ها دو به دو با یکدیگر مقایسه می‌شوند و راه‌حلی که بر دیگری غلبه کند به عنوان راه‌حل نامغلوب در نظر گرفته می‌شود. مرتب‌سازی نامغلوب بر اساس رتبه‌بندی پارتو و معیار ازدحام صورت می‌گیرد. ابتدا رتبه‌های مختلف بر اساس مفهوم غلبه به هر یک از راه‌حل‌ها اختصاص داده می‌شود. راه‌حل‌های با رتبه کوچک‌تر بهتر از راه‌حل‌هایی هستند که رتبه بزرگ‌تر دارند. در میان راه‌حل‌های با رتبه‌های مشابه، یک معیار اضافی به نام معیار ازدحام در نظر گرفته می‌شود. معیار ازدحام برای یک راه‌حل، محاسبه فاصله بین راه‌حل‌های مجاور آن با رتبه‌ی مشابه در فضای هدف است. راه‌حل‌ها با ازدحام کمتر بهتر از راه‌حل‌هایی با ازدحام بیشتر هستند.

یکی از ویژگی‌های اصلی در الگوریتم NSGA-II نخبه‌گرایی است؛ به این صورت که هنگامی‌که جمعیت جدید با استفاده از ترکیب و جهش ایجاد می‌شود، جمعیت فعلی و فرزندان به یک جمعیت ادغام‌شده ترکیب می‌شوند. جمعیت بعدی با انتخاب یک شماره مشخص (به عنوان مثال، اندازه جمعیت) از بهترین راه‌حل‌های جمعیت ادغام‌شده ایجاد می‌شود [۱۴].

۳- کارهای مرتبط

در سال‌های اخیر الگوریتم‌های مختلفی برای حل مسئله ساختارهای درختی متوازن نظیر مسئله t-spanner [۱۵, ۱۶] و درخت پوشا با درجه متوازن [۸, ۱۷] و درخت پوشای متوازن [۵] ارائه شده است. در این بخش به بررسی روش‌های موجود در زمینه حل مسئله درخت پوشای متوازن می‌پردازیم. جدول۱، مهم‌ترین روش‌های ارائه‌شده در این زمینه، را نشان می‌دهد.

یکی از جدیدترین و بهترین روش‌های ارائه‌شده برای حل مسئله درخت پوشای متوازن، روش ارائه‌شده توسط محرم و مرسی در سال ۲۰۱۷ است [۵]. در روش پیشنهادی توسط محرم و مرسی، مسئله ساختارهای درخت پوشای متوازن در قالب سه زیر مسئله درخت پوشای کمینه محدود شده[۴] (CMST)، درخت کوتاه‌ترین مسیر محدود شده[۵] (CSPT) و مسئله درخت پوشا با حداقل حداکثر کشش[۶] (MMST) مورد بررسی قرار گرفته است. در مسئله CMST، هدف پیدا کردن درخت پوشایی است که حداکثر فاصله رئوس تا ریشه درخت کمینه باشد به طوری که وزن درخت بدست آمده حداکثر C (C مقدار ثابت) برابر وزن درخت پوشای کمینه باشد. مسئله CSPT، به دنبال درخت پوشایی با وزن کمینه است به طوری که حداکثر فاصله رئوس تا ریشه درخت بدست آمده حداکثر C (C مقدار ثابت) برابر حداکثر فاصله رئوس تا ریشه درخت کوتاه‌ترین مسیر باشد. مسئله MMST به دنبال درخت پوشا با حداقل فاکتور کشش است.

در روش ارائه شده توسط محرم و مرسی [۵] ضمن دریافت پارامترهای مرتبط، یکی از اهداف مسئله درخت پوشای متوازن را ثابت در نظر می‌گیرد و با استفاده از الگوریتم ژنتیک به دنبال بهینه کردن هدف دیگر است. درواقع، محدودیت و کاستی این روش از یک‌طرف متوجه آن است که مسئله درخت پوشای متوازن را به صورت محدود شده و تک هدفه حل می‌کند. در روش‌های تبدیل مسئله به یک مسئله تک‌هدفه، به اجبار برخی از اطلاعات فضای جستجو از دست می‌رود که برای حل این مشکل باید مسئله چندین بار حل شود که بسیار وقت‌گیر است. ولی روش‌های چندهدفه چنین مشکلی ندارند و بسیار سریع می‌باشند. از طرف دیگر این روش تنها یک راه‌حل را به همراه دارد که امکان تصمیم‌گیری برای کاربر وجود ندارد، درحالی‌که در مسائل واقعی، غالباً تصمیم‌گیرندگان برای تصمیم‌گیری بهتر، به چندین نمونه‌ی راه‌حل نیاز دارند[۶].

اگر چه الگوریتم‌های تاکنون ارائه‌شده‌اند سعی در کاهش ضرایب α و β دارند، اما نیاز به تعریف پارامتر اولیه و دانش قبلی در مورد تعیین اهمیت نسبی هر یک از ضرایب α و β دارند. این تعریف پارامتر در نتایج مسئله تأثیرگذار است. می‌توان با حل مسئله به روش چندهدفه این امکان را فراهم کرد که نه تنها چندین جواب را به همراه داشته باشد بلکه جواب‌های مناسب‌تری بدون نیاز به تعریف پارامتر انتخاب گردد. الگوریتم‌های چندهدفه این امکان را فراهم می‌سازند که به طور هم‌زمان تمامی اهداف مسئله بهینه شود و نیازی به تعریف هیچ‌گونه پارامتر اولیه نباشد [۱۸, ۱۹].

۴- الگوریتم ژنتیک با مرتب‌سازی نامغلوب برای ساخت درخت پوشا متوازن

الگوریتم پیشنهادی در این مقاله یک الگوریتم چندهدفه تکاملی مبتنی بر الگوریتم NSGA-II برای مسئله درخت پوشای متوازن است. این الگوریتم می‌تواند به طور هم‌زمان وزن درخت پوشا در گراف را به حداقل برساند، هم‌چنین مجموع فواصل رئوس تا ریشه را در درخت کوتاه‌ترین مسیر کاهش دهد. به طور کلی روند الگوریتم

جدول ۱: طبقه‌بندی روش‌های ارائه‌شده برای مسئله درخت پوشای متوازن

سال	نویسندگان	روش حل مسئله	معیارهای مدنظر	استفاده از رویکرد تکاملی	نیاز به تعیین پارامتر اولیه
۱۹۸۳	بهارات-کومار و ژافه* [۲۰]	استفاده از فرمول تقریبی: اگر $\beta\alpha \geq \theta(n)$ ، نسبت مجموع فاصله ریشه تا تمامی رئوس درخت به مجموع فاصله رئوس تا ریشه در درخت کوتاه‌ترین مسیر $= c\sqrt{m}$	فاصله	×	✓
۱۹۹۱	کونگ و همکاران** [۲۱]	استفاده از فرمول تقریبی: پیدا کردن درخت پوشای کمینه با شعاع حداکثر $(1+\epsilon)\cdot R$.	وزن و شعاع	×	✓
۱۹۹۲	کونگ و همکاران**[۲۲]	استفاده از فرمول تقریبی: پیدا کردن درخت پوشا با شعاع حداکثر $(1+\epsilon)\cdot R$، $\beta = 1 + 2/\epsilon$	وزن و شعاع	×	✓
۱۹۹۲	آورباچ و همکاران [۲۳]	استفاده از فرمول تقریبی: $\beta = 1 + 4/(\alpha - 1)$	وزن و فاصله	×	✓
۱۹۹۵	خولر و همکاران [۱۲]	استفاده از فرمول تقریبی: $\beta = 1 + 2/(\alpha - 1)$	وزن و فاصله	×	✓
۲۰۱۰	آووخ و میرجلیلی [۲۴]	مبتنی بر روش‌های کلاسیک برای حل مسائل چندهدفه	فاصله، وزن گره و انرژی باقیمانده	×	✓
۲۰۱۵	محرم و مرسی [۲۵]	مبتنی بر الگوریتم ژنتیک به صورت تک هدفه	وزن و فاصله	✓	✓
۲۰۱۷	محرم و مرسی [۵]	مبتنی بر الگوریتم ژنتیک به صورت تک هدفه	وزن و فاصله	✓	✓
۲۰۱۸	آلفرت و همکاران [۲۶]	مبتنی بر روش‌های کلاسیک برای حل مسائل چندهدفه (مجموع وزن‌دار)	وزن و فاصله	×	✓

* c مقدار ثابت و $m = |V| - 1$.

** R حداقل مقدار شعاع درخت، $\epsilon > 0$

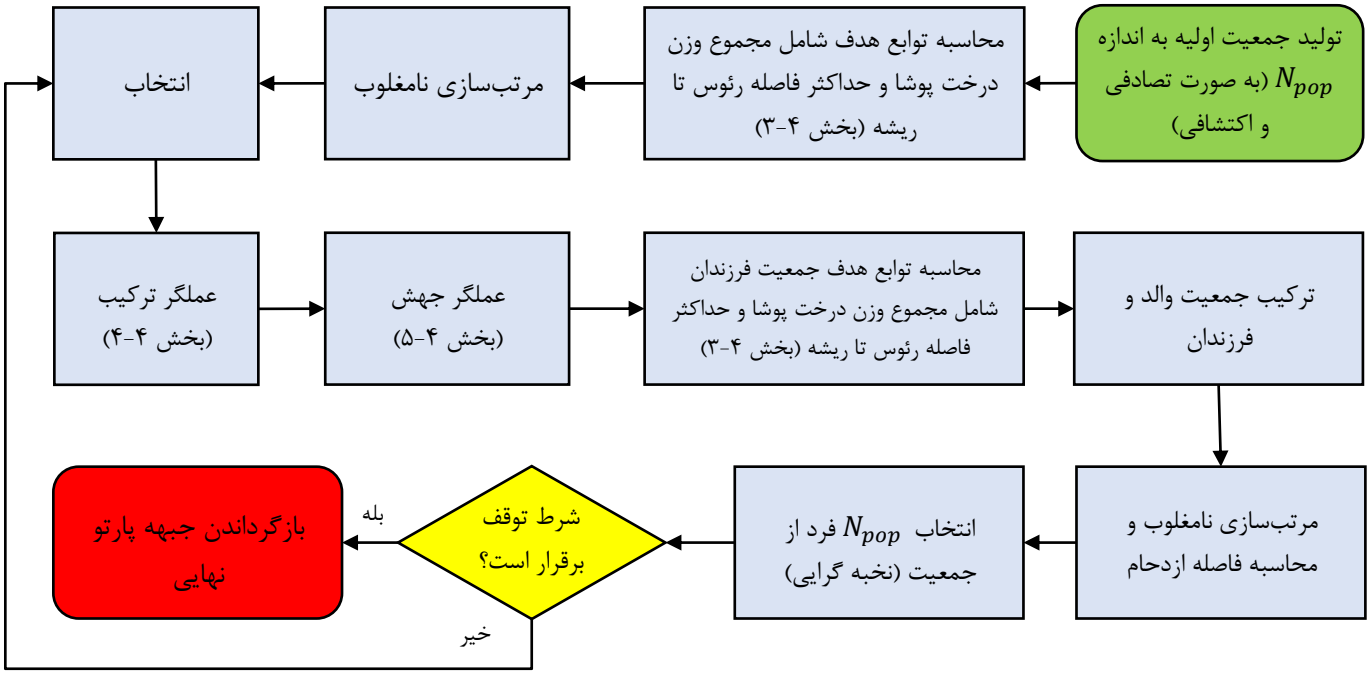
پیشنهادی به این ترتیب است که ابتدا جمعیت اولیه، از طریق مقداردهی تصادفی و اکتشافی مقدار می‌گیرند. سپس جمعیت اولیه بر اساس توابع هدف تعیین شده ارزیابی می‌گردند. در هر مرحله، با اعمال عملگرهای انتخاب، ترکیب و جهش جمعیت جدید ایجاد می‌شود. سپس مرتب‌سازی و انتخاب راه‌حل‌ها برای نسل بعدی بر اساس مفهوم غلبه و فاصله ازدحام انجام می‌شود. فلوچارت مربوط به الگوریتم پیشنهادی در شکل (۲) نشان داده‌شده است. در ادامه مراحل اصلی انجام الگوریتم با جزئیات بیشتر بیان می‌گردد.

۴-۱- کد کردن کروموزوم

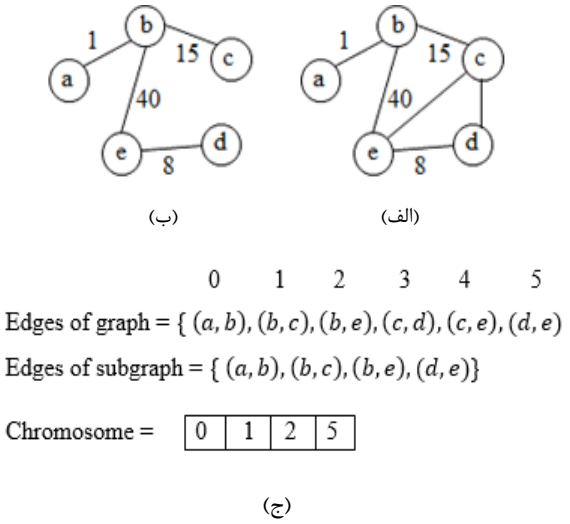
کد کردن کروموزوم اولین قدم در هنگام استفاده از الگوریتم‌های تکاملی است. سه روش رایج برای کدگذاری درختان پوشا در الگوریتم‌های ژنتیکی استفاده می‌شود شامل کدگذاری مبتنی بر بردارهای مشخص [۲۷, ۲۸]، کدگذاری مبتنی بر گره [۲۹]، و کدگذاری بر اساس یال [۵, ۳۰]. در بسیاری از مطالعات نشان داده‌شده است که کدگذاری مبتنی بر یال مناسب‌ترین روش برای نشان دادن درختان پوشا برای مسائل ناشی از وزن یال است [۵]. از این رو در این مطالعه از روش کدگذاری مبتنی بر یال استفاده شده است به طوری که هر یک کروموزوم‌ها بیانگر یک درخت پوشا می‌باشد که شامل $V - 1$ یال است. هرکدام از ژن‌های مربوط به کروموزوم به یک یال از گراف اختصاص دارند. شکل (۳-الف) گراف G با ۵ رأس و ۶ یال را نشان می‌دهد. شکل (۳-ب) درخت پوشای گراف G با ۴ یال را نشان می‌دهد. در شکل (۳-ج) نحوه کدگذاری بر اساس یال برای زیرگراف را نشان می‌دهد.

۴-۲- مقداردهی اولیه جمعیت

دو روش اصلی برای مقداردهی اولیه یک جمعیت در الگوریتم‌های تکاملی روش‌های تصادفی و اکتشافی می‌باشد. در روش مقداردهی اولیه تصادفی، جمعیت اولیه با راه‌حل‌های کاملاً تصادفی مقداردهی می‌گردد. در روش مقداردهی اولیه اکتشافی (ابتکاری)، جمعیت اولیه با استفاده از روش‌های اکتشافی متناسب با مسئله مقداردهی می‌گردد.



شکل ۲: فلوچارت الگوریتم پیشنهادی



شکل ۳: نحوه کد کردن کروموزوم. (الف) گراف G را نشان می‌دهد. (ب) زیرگراف مربوط به درخت پوشا را نشان می‌دهد. (ج) یال‌ها و نحوه کدگذاری مربوط به آن را نمایش می‌دهد.

در این مقاله از روش اکتشافی به همراه تصادفی استفاده شده است. از آن‌جا که هر کروموزوم نشان دهنده یک درخت پوشا می‌باشد، بنابراین در این مقاله از یک ورژن تصادفی از الگوریتم کروسکال و یک درخت کوتاه‌ترین مسیر بدست آمده از الگوریتم دایجکسترا برای ایجاد راه‌حل‌های اولیه استفاده شده است. این نوع مقداردهی منجر به تسریع اجرای الگوریتم می‌گردد. بقیه اعضای جمعیت به صورت تصادفی ایجاد می‌شوند. به این صورت که گره u به صورت تصادفی انتخاب می‌شود و از بین گره‌های همسایه آن گره، گره v به تصادف انتخاب می‌شود به طوری که آن گره در بین مجموعه گره‌هایی که تاکنون انتخاب شده‌اند نباشد ($v \notin V(T)$). سپس یال مربوط به آن به مجموعه یال‌ها زیرگراف اضافه می‌گردد. یال‌ها تا زمانی اضافه می‌گردد که تعداد آن‌ها کم‌تر از $V-1$ باشد. می‌توان نشان داد که زیرگراف حاصل (T) یک درخت با $V-1$ گره است. در الگوریتم فرض شده است ریشه درخت کوتاه‌ترین مسیر اولین نود درگراف است.

۴-۳- توابع هدف

الگوریتم پیشنهادی از دو تابع هدف شامل مجموع وزن درخت پوشا و حداکثر فاصله رئوس تا ریشه r در درخت پوشای انتخاب شده در گراف برای دستیابی به درخت پوشای متوازن استفاده می‌کند. در ادامه هر یک از اهداف با جزئیات بیشتر بیان می‌گردد.

تابع هدف اول (مجموع وزن درخت پوشا): به منظور دستیابی به درخت پوشای کمینه، مجموع وزن یال‌های انتخاب شده توسط هر

کروموزوم به عنوان تابع هدف اول در نظر گرفته می‌شود. الگوریتم سعی در حداقل‌سازی این تابع هدف دارد. رابطه (۱) تابع هدف مجموع وزن درخت پوشا را نشان می‌دهد که E_T مجموعه یال‌های درخت پوشای مرتبط با هر راه‌حل است. $weight(e_T)$ نیز وزن هر یال در درخت را نشان می‌دهد.

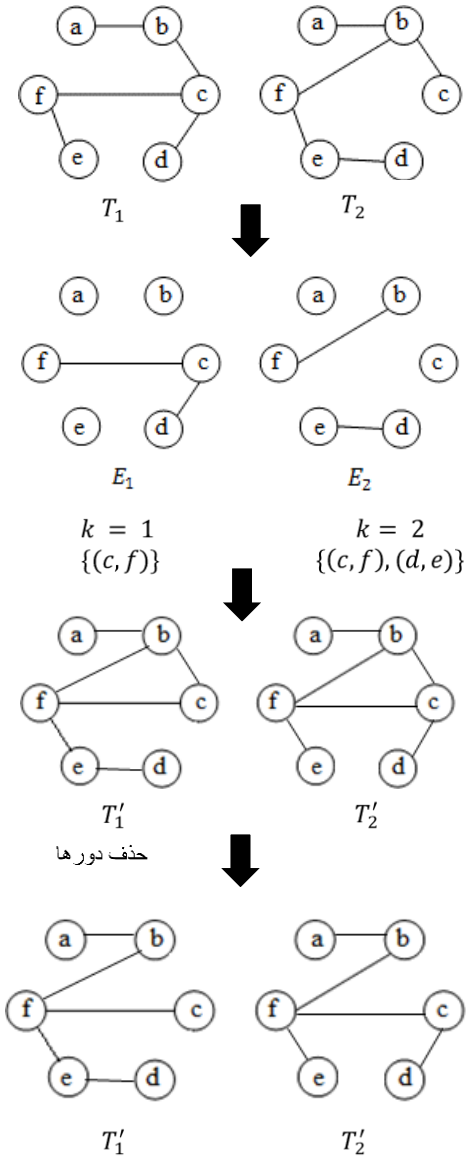
$$Of_1 = \sum_{e_T \in E_T} weight(e_T) \qquad (۱)$$

تابع هدف دوم (حداکثر فاصله رئوس تا ریشه r در درخت پوشا): هدف دوم به منظور دستیابی به درخت پوشای متوازن به دست آوردن درخت کوتاه‌ترین مسیر از ریشه r است. بدین منظور فاصله هر رأس تا ریشه r محاسبه می‌شود. الگوریتم سعی در حداقل‌سازی این تابع هدف دارد. در رابطه (۲) نحوه به دست آوردن تابع هدف سوم آمده است. در این رابطه، V مجموعه رأس‌های گراف می‌باشد. $Distance(v,r)$ فاصله هر رأس v تا ریشه است. ریشه درخت کوتاه‌ترین مسیر می‌تواند توسط کاربر تعیین گردد ولی در این‌جا اولین نود وارد شده به عنوان ریشه فرض می‌شود.

$$Of_2 = max_{v \in V}\, Distance(v,r) \qquad (۲)$$

۴-۴- ترکیب

در الگوریتم پیشنهادی انتخاب والدین برای انجام عمل ترکیب به صورت تصادفی صورت گرفته است. به طور کلی روش‌های مختلفی برای انجام ترکیب وجود دارد. در این الگوریتم از رویکرد ترکیب چند نقطه‌ای مشابه روش ارائه‌شده در مرجع محرم و مرسی [۵] استفاده شده است. به این صورت که ابتدا دو کروموزوم T_1 و T_2 به صورت تصادفی انتخاب می‌شوند و بر اساس یال‌های مربوط به هر درخت دو مجموعه $E_1 = E(T_1) - E(T_2)$ و $E_2 = E(T_2) -$ $E(T_1)$ تعریف می‌شود. فرض کنید $t = |E_1| = |E_2|$ ، یک عدد تصادفی k در بازه $[1,t]$ تولید می‌شود. یک زیر مجموعه k تایی به نام E_1' از E_1 انتخاب می‌شود و به T_2 اضافه می‌شود و زیرگراف T_1' به دست می‌آید. واضح است که T_1' شامل k دور است که یکی از یال‌های E_1' نیز در آن می‌باشد. برای حذف دورها، به ازای هر دور یکی از یال‌های موجود در دور (به صورت تصادفی) به غیر از یال جدید اضافه شده حذف می‌گردد. این رویه برای به دست آوردن فرزند دیگر نیز تکرار می‌شود به این صورت که نقش T_1 و T_2 عوض می‌شود [۵]. در شکل (۴) روند انجام فرآیند ترکیب قابل مشاهده است. در مرحله اول دو درخت T_1 و T_2 که برای ترکیب انتخاب شده‌اند نشان داده‌شده است. در مرحله بعد یال‌هایی که در دو درخت باهم متفاوت است نمایش داده‌شده است. سپس از بین یال‌ها چند مورد به تصادف انتخاب می‌شود و به درخت‌ها اضافه می‌گردد. در نهایت دور به وجود آمده در درخت‌ها با روش بیان شده حذف می‌گردد.



شکل ۴: نحوه اعمال عملگر ترکیب بر روی والدهای T_1 و T_2.

۴-۵- جهش

عملگر جهش با انتخاب یک والد و تغییر دادن یک یا چندین ژن آن، سعی در جستجو در فضای کشف نشده راه‌حل‌ها دارد. در الگوریتم پیشنهادی عملگر جهش با یک احتمال از قبل تعیین شده p_m بر روی جمعیت اعمال می‌گردد. روش کار به این صورت است که یکی از راه‌حل‌ها به صورت تصادفی انتخاب می‌گردد. سپس یک عدد تصادفی تولید می‌شود، درصورتی که عدد تصادفی از p_m کم‌تر بود عملگر جهش بر روی راه‌حل T اعمال می‌شود. اعمال عملگر

جهش به این صورت است که یک یال از بین مجموعه یال‌هایی که در درخت پوشای مربوط به راه‌حل وجود ندارد ($E - E_T$) انتخاب می‌شود و به درخت پوشا اضافه می‌گردد و فرزند جدید T' ایجاد می‌شود. بدیهی است که یک دور در زیرگراف T' ایجاد می‌گردد سپس برای حذف دور یکی از یال‌های موجود در دور به غیر از یالی که به تازگی اضافه شده به تصادف از زیرگراف T' حذف می‌گردد.

۵- نتایج ارزیابی

برای پیاده‌سازی الگوریتم پیشنهادی از نرم‌افزار پایتون ۳/۷ و کتابخانه networkx استفاده شده است. الگوریتم پیشنهادی بر روی مجموعه گراف‌های تصادفی مورد آزمایش قرار گرفته است. در این بخش ابتدا به معرفی نمونه گراف‌های مورد استفاده پرداخته می‌شود. سپس نحوه تنظیم پارامترهای الگوریتم‌های مورد مقایسه و الگوریتم پیشنهادی برای دستیابی به نتایج مطلوب بیان می‌گردد. در زیر بخش سوم نحوه تنظیم پارامترهای اندازه جمعیت و حداکثر تعداد ارزیابی توابع هدف مورد بررسی قرار می‌گیرد. در زیر بخش چهارم عملکرد الگوریتم مورد بررسی قرار می‌گیرد. درنهایت راه‌حل‌های به‌دست‌آمده توسط الگوریتم پیشنهادی با الگوریتم CMST در مرجع محرم و مرسی [۵]، الگوریتم ارائه شده توسط خولر و همکاران [۱۲] و راه‌حل بهینه (راه‌حلی که توسط جستجوی کامل درختان پوشای گراف به دست می‌آید) مورد مقایسه قرار می‌گیرد.

۵-۱- نمونه‌های گراف

به منظور ارزیابی الگوریتم پیشنهادی، از چند نمونه گراف تصادفی استفاده شده است. این نمونه‌ها با استفاده از رویکرد اردوس و رنی [۱۱] که هر یال بین دو گره به صورت مستقل با احتمالی بین $p \in [0,1]$ ایجاد می‌شوند. برای یافتن درخت پوشا در یک گراف که تمامی گره‌ها در آن وجود داشته باشد، لازم است گراف‌ها همبند باشد. برای اطمینان از این که گراف‌ها همبند هستند p طوری انتخاب می‌شود که $p \gg \frac{\log n}{n}$ باشد. برای ایجاد یک گراف ابتدا یک گراف خالی ایجاد می‌شود. سپس هر یک از جفت گره با احتمال p به هم متصل می‌شود [۳۱]. در این مقاله گراف‌های تصادفی با اندازه‌های ۱۰، ۵۰، ۱۰۰، ۱۵۰ و ۲۰۰ تولید و وزن یال‌ها به صورت تصادفی در محدوده ۱ تا ۱۰۰ تعیین می‌شود.

در بخش ۵-۵، یافتن راه‌حل بهینه برای گراف‌ها با اندازه‌های بزرگ به دلیل رشد تعداد درختان پوشا با افزایش تعداد گره‌ها امکان‌پذیر نیست. بدین منظور از گراف‌ها با اندازه‌های کوچک‌تر شامل گراف با تعداد گره‌های ۶، ۱۰، ۱۵ و ۲۰ استفاده شده است.

جدول ۲، مشخصات نمونه گراف‌های تصادفی شامل تعداد گره، یال، وزن درخت پوشای کمینه و حداکثر فاصله رئوس تا ریشه مربوط به هر یک را نشان می‌دهد.

جدول ۲: مشخصات نمونه‌های گراف.

	تعداد گره	تعداد یال	وزن درخت پوشای کمینه گراف	حداکثر فاصله رئوس تا ریشه درخت کوتاه‌ترین مسیر گراف
۱	۶	۱۲	۱۳۷	۶۸
۲	۱۰	۳۶	۱۳۱	۴۷
۳	۱۵	۷۷	۱۹۵	۷۱
۴	۲۰	۱۵۹	۱۳۴	۵۲
۵	۵۰	۶۴۲	۳۳۸	۴۷
۶	۱۰۰	۱۰۶۵	۵۴۹	۴۱
۷	۱۵۰	۲۵۱۳	۶۳۶	۳۴
۸	۲۰۰	۴۴۱۴	۵۹۸	۲۶

۵-۲- تنظیم پارامترها

تعیین پارامترها ممکن است در نحوه رسیدن الگوریتم به جواب مناسب تأثیر داشته باشند. ترکیب‌های مختلف از پارامترها می‌تواند باعث به‌وجود آمدن تنوع مختلفی از راه‌حل‌ها شود. جدول۳، پارامترهای مربوط به الگوریتم پیشنهادی و سایر الگوریتم‌های مورد مقایسه در این تحقیق را نشان می‌دهد. پارامترهای الگوریتم پیشنهادی از طریق آزمون و خطا و با تأسی از مرجع محرم و مرسی[۵]، تنظیم شده‌اند. در ادامه نحوه انتخاب پارامترهای مناسب برای تعداد جمعیت و حداکثر تعداد ارزیابی توابع هدف شرح داده می‌شود.

جدول۳: پارمترهای الگوریتم‌های مورد مقایسه

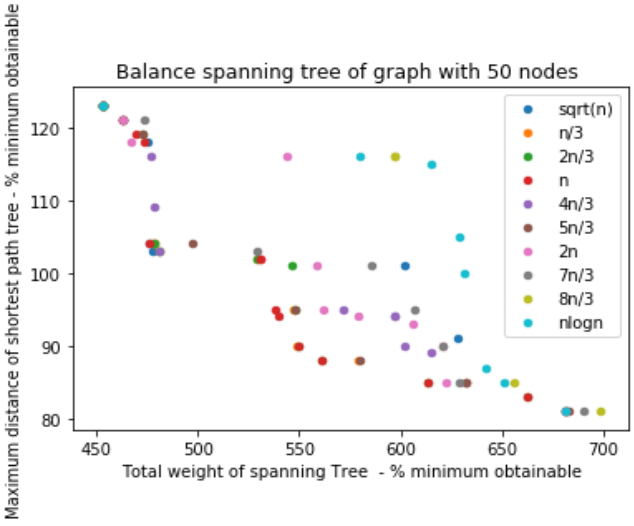
الگوریتم	خصوصیات و پارامترها
روش خولر و همکاران	$\beta = 1 + 2/(\alpha - 1)$
روش محرم و مرسی (CMST)	تعداد نسل‌ها = حداکثر ۳۰۰، تعداد جمعیت = n نرخ ترکیب = ۰/۹، نرخ جهش = ۰/۲ تعداد ژن‌ها = $V - 1$، ارزش ژن ها = در بازه $[0, E]$
روش پیشنهادی	معیار خاتمه= حداکثر تعداد ارزیابی توابع هدف تعداد جمعیت = n ، نرخ ترکیب = ۰/۹، نرخ جهش = ۰/۲ تعداد ژن ها = $V - 1$، ارزش ژن ها = در بازه $[0, E]$

۳-۵- تأثیر اندازه جمعیت و حداکثر تعداد ارزیابی توابع هدف

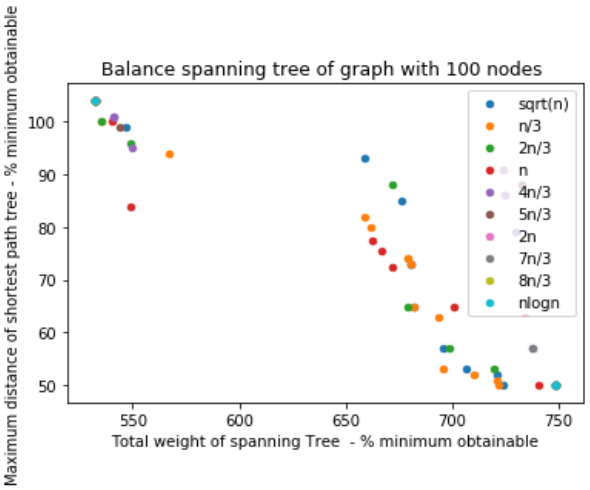
در این زیر بخش تأثیر اندازه جمعیت و حداکثر تعداد ارزیابی توابع هدف بر روی جبهه پارتو بدست آمده توسط الگوریتم پیشنهادی مورد بررسی قرار می‌گیرد. به منظور تعیین اندازه مناسب جمعیت، الگوریتم پیشنهادی با اندازه جمعیت $\sqrt{n}$، $n/3$، $2n/3$، $4n/3$، $5n/3$، n، $2n$، $7n/3$، $8n/3$، $nlogn$ مورد ارزیابی قرار گرفت که در آن n اندازه گراف است. برای تعیین مقدار پارامتر حداکثر تعداد ارزیابی توابع هدف، الگوریتم با مقادیر 10^4، 10^5 و 10^6 مورد ارزیابی قرار گرفت. ارزیابی‌های انجام شده با جمعیت‌ها و تعداد ارزیابی‌های مختلف بر روی گراف با اندازه‌های ۵۰ و ۱۰۰ اعمال گردید (شکل ۵ تا ۸).

شکل ۵ و ۶ جبهه‌های پارتو بدست آمده توسط الگوریتم پیشنهادی در اجراها مختلف را نشان می‌دهد که جمعیت با اندازه n (جبهه پارتو قرمز رنگ) نسبت به سایرین بهتر عمل کرده است و راه‌حل‌هایی که توسط الگوریتم با تعداد جمعیت n بدست آمده تقریباً بر سایر راه‌حل‌ها غلبه می‌کند. بنابراین مقدار n به عنوان پارامتر تعداد جمعیت درنظر گرفته می‌شود.

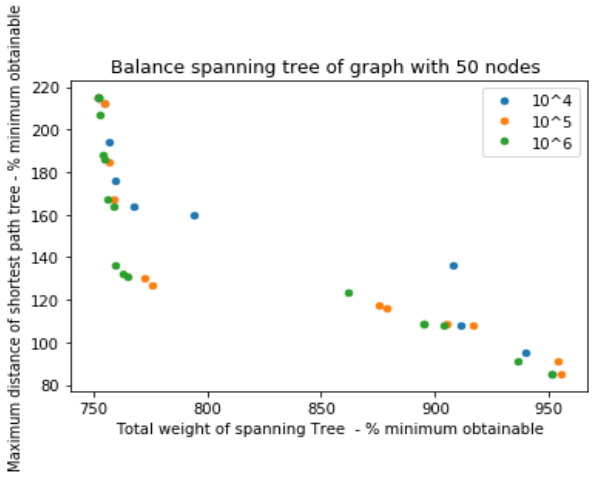
شکل ۷ و ۸ به ترتیب جبهه‌های پارتو مربوط به گراف با اندازه‌های ۵۰ و ۱۰۰ را با معیار خاتمه با مقادیر مختلف نشان می‌دهد. نتایج حاکی از آن است که جبهه پارتو بدست آمده با مقادیر 10^5 و 10^6 تفاوت چندانی با یکدیگر ندارد. از آن جایی‌که معیار خاتمه با حداکثر ارزیابی تابع هدف 10^5 به زمان اجرای کمتری نیاز دارد، مقدار 10^5 به عنوان پارامتر مورد نظر درنظر گرفته می‌شود.



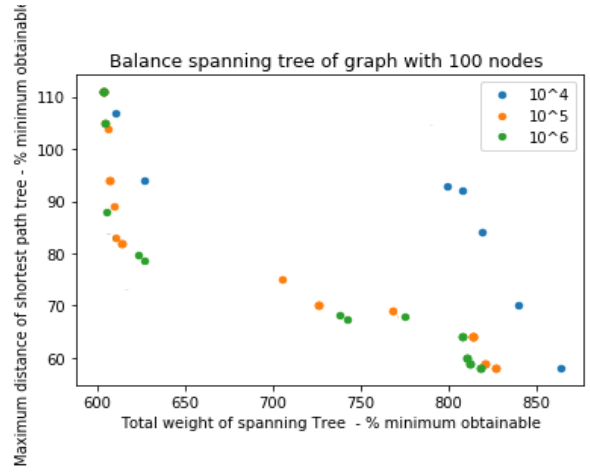
شکل ۵: تأثیر اندازه جمعیت بر نتایج گراف با ۵۰ گره.



شکل ۶: تأثیر اندازه جمعیت بر نتایج گراف با ۱۰۰ گره.



شکل ۷: تأثیر تعداد ارزیابی های توابع هدف مختلف بر نتایج گراف با ۵۰ گره



شکل ۸: تأثیر تعداد ارزیابی های توابع هدف مختلف بر نتایج گراف با ۱۰۰ گره

۵-۴- بررسی عملکرد الگوریتم پیشنهادی

در این بخش عملکرد الگوریتم پیشنهادی بر روی نمونه گراف‌های مورد استفاده نشان داده می‌شود. شکل (۹) جبهه پارتو به‌دست‌آمده برای گراف‌های تصادفی با اندازه‌های ۱۰، ۵۰، ۱۰۰، ۱۵۰ و ۲۰۰ را نشان می‌دهد. نقاط آبی رنگ در شکل مربوط به جبهه پارتو به‌دست‌آمده توسط الگوریتم پیشنهادی است. ستاره قرمز رنگ مختصات نقطه‌ای که حداکثر فاصله رئوس تا ریشه در درخت کوتاه‌ترین مسیر است و درخت پوشای کمینه مرتبط با آن را نشان می‌دهد. طول و عرض مربع سیاه رنگ به ترتیب مجموع وزن درخت پوشای کمینه و حداکثر فاصله رئوس تا ریشه مرتبط با درخت کوتاه‌ترین مسیر در درخت پوشا به‌دست‌آمده را نشان می‌دهد. مشاهده می‌شود الگوریتم توانسته است برای هریک از گراف‌ها چندین راه‌حل به دست آورد؛ این امر موجب می‌گردد که کاربر بر اساس نیازش (که کوتاه‌ترین مسیر از اهمیت بیشتر برخوردار است یا درخت پوشا کمینه) یکی از راه‌حل‌ها را انتخاب نماید. از طرفی استفاده از الگوریتم NSGA-II به دلیل بهره‌گیری از ویژگی فاصله ازدحام منجر به حفظ تنوع راه‌حل‌های واقع بر جبهه پارتو شده است. راه‌حل‌های به‌دست‌آمده توسط الگوریتم پیشنهادی با راه‌حل حاصل از درخت کوتاه‌ترین مسیر و درخت پوشا کمینه مقایسه شده است. مقایسه‌ها نشان می‌دهد که راه‌حل‌های بدست آمده توانسته‌اند مصالحه خوبی را بین درخت پوشای کمینه و درخت کوتاه‌ترین مسیر بدست آوردند. جداول۴ تا ۸، نسبت حداکثر فاصله رئوس تا ریشه درخت پوشای منتخب به حداکثر فاصله رئوس تا ریشه درخت کوتاه‌ترین مسیر (α) و نسبت مجموع وزن درخت پوشای منتخب به مجموع وزن درخت پوشای کمینه (β) متناسب با هر کدام از راه‌حل‌های به‌دست‌آمده توسط الگوریتم پیشنهادی را نشان می‌دهد. مقادیر«وزن» و «فاصله» به ترتیب مجموع وزن درخت پوشا و حداکثر فاصله رئوس تا ریشه آن را برای درخت پوشا منتخب نشان می‌دهد.

جدول ۴: مقادیر توابع هدف برای راه‌حل‌های به دست آمده توسط الگوریتم پیشنهادی برای گراف با ۱۰ گره

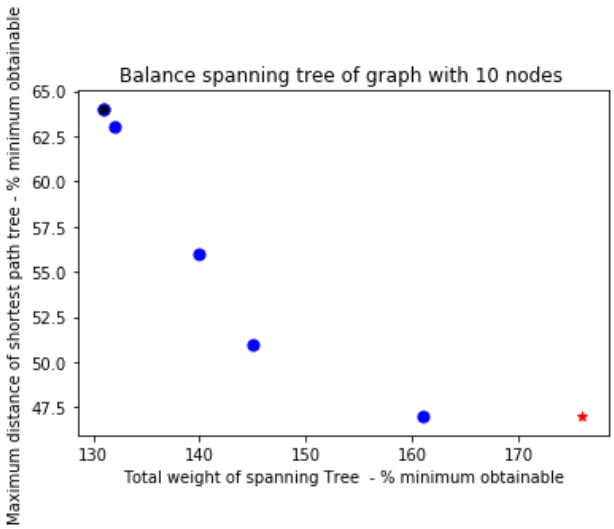
وزن	فاصله	تابع هدف اول (α)	تابع هدف دوم (β)
۱۶۱	۴۷	۱	۱/۲۲۹
۱۴۵	۵۱	۱/۰۸۵	۱/۱۰۷
۱۴۰	۵۶	۱/۱۹۱	۱/۰۶۹
۱۳۲	۶۳	۱/۳۴	۱/۰۰۸
۱۳۱	۶۴	۱/۳۶۲	۱

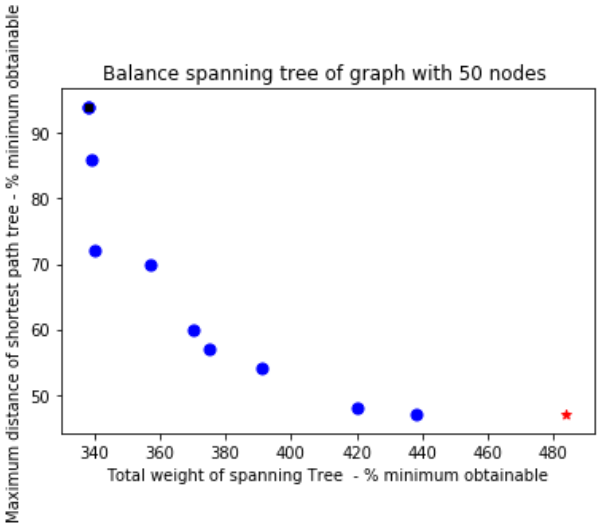
جدول ۵: مقادیر توابع هدف برای راه‌حل‌های به دست آمده توسط الگوریتم پیشنهادی برای گراف با ۵۰ گره

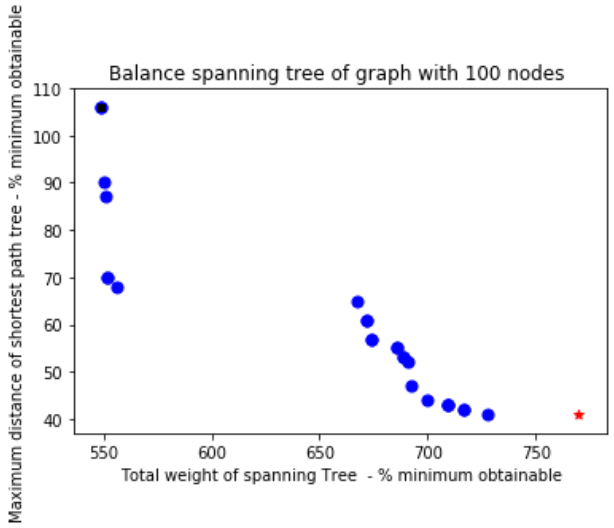
وزن	فاصله	تابع هدف اول (α)	تابع هدف دوم (β)
۴۳۸	۴۷	۱	۱/۲۹۶
۴۲۰	۴۸	۱/۰۲۱	۱/۲۴۳
۳۹۱	۵۴	۱/۱۴۹	۱/۱۵۷
۳۷۵	۵۷	۱/۲۱۳	۱/۱۰۹
۳۷۰	۶۰	۱/۲۷۷	۱/۰۹۵
۳۵۷	۷۰	۱/۴۸۹	۱/۰۵۶
۳۴۰	۷۲	۱/۵۳۲	۱/۰۰۶
۳۳۹	۸۶	۱/۸۳	۱/۰۰۳
۳۳۸	۹۴	۲	۱

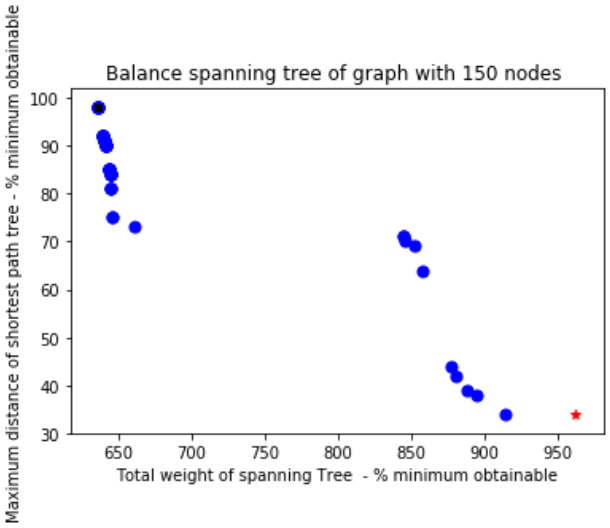
جدول ۶: مقادیر توابع هدف برای راه‌حل‌های به دست آمده توسط الگوریتم پیشنهادی برای گراف با ۱۰۰ گره

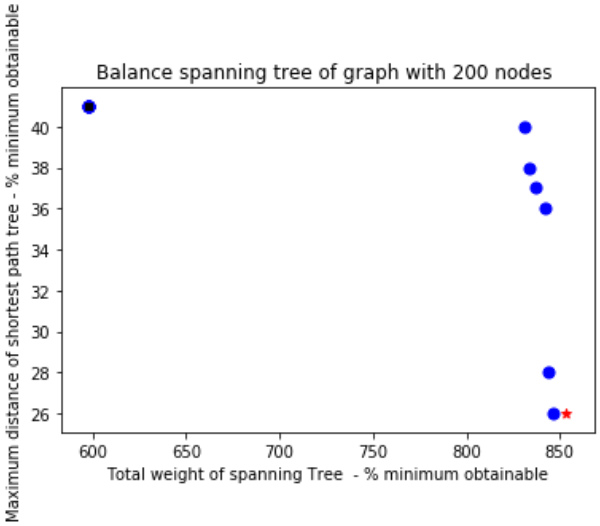
وزن	فاصله	تابع هدف اول (α)	تابع هدف دوم (β)
۷۲۸	۴۱	۱	۱/۳۲۶
۷۱۷	۴۲	۱/۰۲۴	۱/۳۰۶
۷۰۹	۴۳	۱/۰۴۹	۱/۲۹۱
۷۰۰	۴۴	۱/۰۷۳	۱/۲۷۵
۶۹۲	۴۷	۱/۱۲۲	۱/۲۶
۶۹۱	۵۲	۱/۱۴۶	۱/۲۵۵
۶۸۹	۵۳	۱/۱۹۵	۱/۲۵۰
۶۸۶	۵۵	۱/۲۶۸	۱/۲۲۴
۶۷۴	۵۷	۱/۳۴۱	۱/۲۱۵
۶۷۲	۶۱	۱/۳۹	۱/۰۱۳
۶۶۷	۶۵	۱/۴۸۸	۱/۰۰۵
۵۵۶	۶۸	۱/۵۸۵	۱/۰۰۴
۵۵۲	۷۰	۱/۶۵۹	۱/۰۰۲
۵۵۰	۹۰	۱/۷۰۷	۱











شکل ۹: جبهه پارتو به‌دست‌آمده توسط الگوریتم پیشنهادی برای نمونه گراف با اندازه‌های ۱۰، ۵۰، ۱۰۰، ۱۵۰ و ۲۰۰ گره

جدول ۷: مقادیر توابع هدف برای راه‌حل‌های به دست آمده توسط الگوریتم پیشنهادی برای گراف با ۱۵۰ گره

وزن	فاصله	تابع هدف اول (α)	تابع هدف دوم (β)
۹۱۴	۳۴	۱	۱/۴۳۷
۸۹۴	۳۸	۱/۱۱۸	۱/۴۰۶
۸۸۸	۳۹	۱/۱۴۷	۱/۳۹۶
۸۸۰	۴۲	۱/۲۳۵	۱/۳۸۴
۸۷۷	۴۴	۱/۲۹۴	۱/۳۷۹
۸۵۷	۶۹	۱/۳۸۲	۱/۳۴۷
۸۵۲	۷۱	۲/۰۲۹	۱/۳۴۰
۶۶۱	۷۳	۲/۰۵۹	۱/۳۲۷
۶۴۶	۷۵	۲/۰۸۸	۱/۰۱۶
۶۴۵	۸۱	۲/۲۰۶	۱/۰۱۴
۶۴۴	۸۴	۲/۴۷۱	۱/۰۱۳
۶۴۳	۸۵	۲/۵۰۰	۱/۰۱۱
۶۴۲	۹۰	۲/۶۴۷	۱/۰۰۸
۶۴۱	۹۱	۲/۶۷۶	۱/۰۰۶
۶۳۹	۹۲	۲/۷۰۶	۱/۰۰۵
۶۳۶	۹۸	۲/۸۸۲	۱

جدول ۸: مقادیر توابع هدف برای راه‌حل‌های به دست آمده توسط الگوریتم پیشنهادی برای گراف با ۲۰۰ گره

وزن	فاصله	تابع هدف اول (α)	تابع هدف دوم (β)
۸۴۶	۲۶	۱	۱/۴۱۵
۸۴۴	۲۸	۱/۰۷۷	۱/۴۱۱
۸۴۲	۳۶	۱/۳۸۵	۱/۴۰۸
۸۳۷	۳۷	۱/۴۲۳	۱/۴۰۰
۸۳۴	۳۸	۱/۴۶۲	۱/۳۹۵
۸۳۱	۴۰	۱/۵۳۸	۱/۳۹۰
٥٩٨	٤١	۱/۵۷۷	۱

۵-۵- مقایسه الگوریتم پیشنهادی با سایر الگوریتم‌ها

به منظور نمایش کیفیت الگوریتم پیشنهادی، الگوریتم مورد نظر با راه‌حل بهینه، الگوریتم CMST در مرجع محرم و مرسی [۵] و الگوریتم ارائه شده توسط خولر و همکاران [۱۲] مورد مقایسه قرار گرفت. در این بخش هدف بدست آوردن مقدار β نزدیک به مقدار β بهینه برای یک مقدار ثابت α است. از آن جایی که الگوریتم پیشنهادی دارای چندین جواب است، هر بار یکی از راه‌حل‌های بدست آمده انتخاب می‌گردد و با سایر روش‌ها مقایسه می‌شود. راه‌حل بهینه می‌تواند از طریق بررسی تمام درخت‌های پوشای گراف (جستجوی کامل) به دست آید [۵]. جستجوی کامل زمانی قابل استفاده است که گراف بسیار کوچک باشد. زیرا تعداد درخت‌های پوشا به سرعت با افزایش تعداد گره‌ها افزایش می‌یابد. برای مثال در یک گراف کامل با n گره، به تعداد n^{n-2} درخت پوشا وجود دارد [۳۲]. بنابراین از گراف‌ها با تعداد گره‌های کم‌تر با اندازه‌های ۶، ۱۰، ۱۵ و ۲۰ گره برای این مقایسه استفاده شده است. جداول ۹ تا ۱۲ نتایج حاصل از این مقایسات را نشان می‌دهد. برای به دست آوردن مقادیر α و β برای سایر روش‌ها، مقدار α برابر هر یک از مقادیر α به‌دست‌آمده برای الگوریتم پیشنهادی در نظر گرفته می‌شود. سپس مقدار β محاسبه می‌شود. مشاهده می‌شود نتایج به‌دست‌آمده توسط الگوریتم پیشنهادی در غالب موارد بهتر از سایر روش‌ها می‌باشد و به راه‌حل بهینه نزدیک‌تر است. در واقع در بیشتر موارد β بدست آمده توسط الگوریتم پیشنهادی با راه‌حل بهینه برابر است؛ در سایر موارد نیز راه‌حل بدست آمده به β راه‌حل بهینه بسیار نزدیک است.

تمامی مراحلی که برای بدست آوردن β انجام شد، می‌تواند برای بدست آوردن و مقایسه α نیز صورت گیرد؛ با این تفاوت که جای α و β تغییر می‌کند که در نتایج مقایسه تفاوتی نخواهد داشت.

جدول۹: مقایسه راه‌حل‌های به دست آمده توسط الگوریتم پیشنهادی و سایر الگوریتم‌ها برای ۶ گره

α	β – بهینه	β – روش خولر و همکاران	β – *CMST*	β –روش پیشنهادی
۱	**۱/۷۵۹**	-	۱/۹	**۱/۷۵۹**
۱/۰۱۵	**۱/۳۶۵**	۱۳۴/۳۳	**۱/۳۶۵**	**۱/۳۶۵**
۱/۲۶۵	**۱/۰۸۸**	۸/۵۴۷	۱/۳۶۵	**۱/۰۸۸**
۱/۷۶۵	**۱**	۶/۵۲۵	۱/۰۸۸	**۱**

جدول۱۰: مقایسه راه‌حل‌های به دست آمده توسط الگوریتم پیشنهادی و سایر الگوریتم‌ها برای ۱۰ گره

α	β – بهینه	β – روش خولر و همکاران	β – *CMST*	β –روش پیشنهادی
۱	**۱/۲۲۹**	-	**۱/۲۲۹**	**۱/۲۲۹**
۱/۰۸۵	**۱/۱۰۷**	۲۴/۵۹۱	۱/۲۶	۱/۲۶
۱/۱۹۱	**۱/۰۶۹**	۱۱/۴۷۱	۱/۱۸۹	**۱/۰۶۹**
۱/۳۴	**۱/۰۰۸**	۶/۸۸۲	۱/۱۲	**۱/۰۰۸**
۱/۳۶۲	**۱**	۶/۵۲۵	**۱**	**۱**

جدول۱۱: مقایسه راه‌حل‌های به دست آمده توسط الگوریتم پیشنهادی و سایر الگوریتم‌ها برای ۱۵ گره

α	β – بهینه	β – روش خولر و همکاران	β – *CMST*	β – روش پیشنهادی
۱	۱/۳۴۴	-	۱/۳۴۴	۱/۳۴۴
۱/۰۸۶	۱/۱۲۵	۲۴/۲۵۶	۱/۱۳۰	۱/۱۳۰
۱/۱۵۵	۱	۲۳/۷۲۷	۱/۱۳۰	۱

جدول۱۲: مقایسه راه‌حل‌های به دست آمده توسط الگوریتم پیشنهادی و سایر الگوریتم‌ها برای ۲۰ گره

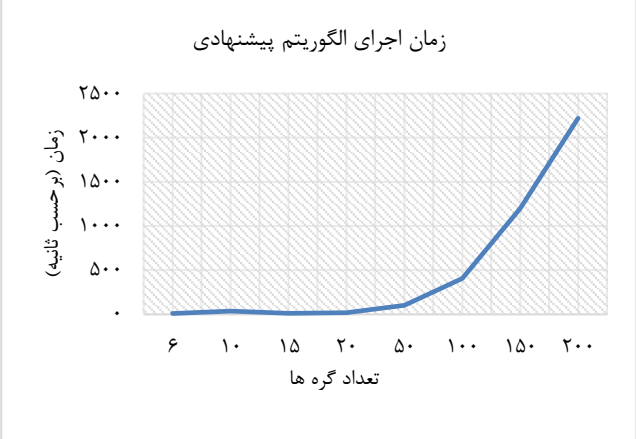
α	β – بهینه	β – روش خولر و همکاران	β – *CMST*	β – روش پیشنهادی
۱	۱/۳۵۱	-	۱/۴	۱/۳۵۱
۱/۰۱۹	۱/۲۶۹	۱۰۶/۲۶۳	۱/۳۵	۱/۲۶۹
۱/۰۹۷	۱/۱۱۴	۲۱/۶۱۹	۱/۲۳	۱/۱۱۴
۱/۲۱۲	۱	۱۰/۴۳۴	۱/۰۲	۱

۶- زمان اجرای الگوریتم پیشنهادی

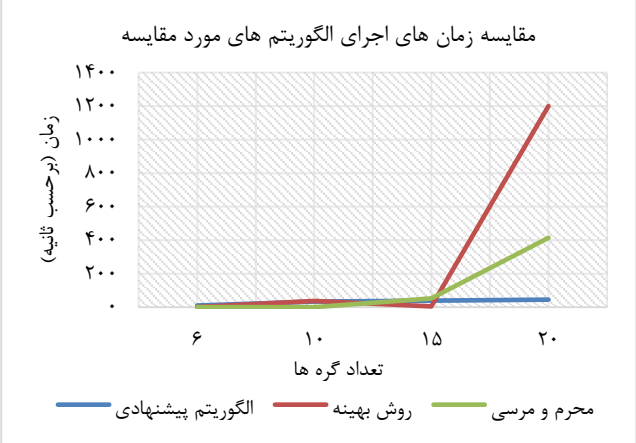
در این بخش ابتدا به مقایسه زمان اجرای الگوریتم پیشنهادی و سایر الگوریتم مورد مقایسه برای گراف‌های مورد بررسی پرداخته می‌شود. در زیر بخش دوم پیچیدگی زمانی الگوریتم پیشنهادی مورد ارزیابی قرار می‌گیرد.

۶-۱- زمان اجرا بر روی گراف‌های مورد بررسی

شکل (۱۰) زمان اجرای الگوریتم پیشنهادی بر روی گراف‌های تصادفی با اندازه‌های مختلف را نشان می‌دهد. همان‌طور که در شکل (۱۰) مشاهده می‌شود، به طور کلی زمان اجرای الگوریتم پیشنهادی با افزایش تعداد گره‌ها و یال‌های گراف افزایش می‌یابد. در شکل (۱۱) زمان اجرای الگوریتم پیشنهادی و سایر الگوریتم‌ها بر روی گراف‌های تصادفی با اندازه‌های ۶، ۱۰، ۱۵ و ۲۰ مورد مقایسه قرار می‌گیرد. همان‌طور که در شکل (۱۱) مشاهده می‌شود الگوریتم پیشنهادی قادر است در زمان به نسبت کمتری جواب‌های نزدیک به بهینه و حتی بهینه بدست آورد. هر چه اندازه گراف بزرگ‌تر شود زمان اجرا الگوریتم دقیق برای بدست آوردن راه‌حل بهینه به صورت نمایی افزایش می‌یابد و تفاوت در زمان اجرای الگوریتم‌ها ملموس‌تر خواهد بود. زمان اجرای الگوریتم مبتنی بر ژنتیک (محرم و مرسی) نسبت به الگوریتم پیشنهادی بیشتر است؛ این مسئله به این دلیل است که در این روش در هر مرحله تنها کروموزوم‌هایی که شرایط مسئله را ارضا می‌کند به جمعیت بعدی اضافه می‌شود که بررسی امکان پذیر بودن تمامی کروموزوم‌های جمعیت فرزندان کاری به نسبت زمان‌بر است.



شکل ۱۰: زمان اجرای الگوریتم پیشنهادی



شکل ۱۱: مقایسه زمان اجرای الگوریتم پیشنهادی و سایر الگوریتم‌ها

۶-۲- تحلیل پیچیدگی زمانی

به طور کلی پیچیدگی زمانی الگوریتم پیشنهادی به تعداد اجرا (T)، تعداد مقایسه‌های توابع هدف و بدترین پیچیدگی زمانی برای

عملگرهایی که در الگوریتم NSGA-II به‌کاررفته، وابسته است. پیچیدگی هر کدام از بخش‌های الگوریتم به شرح زیر است:

۱) پیچیدگی زمانی مرتب‌سازی نامغلوب در الگوریتم NSGA-II برای هر تکرار $O(mN^2)$ است [۱۴]. m تعداد اهداف و N اندازه جمعیت می‌باشد که ما در این مسئله آن را برابر تعداد گره‌ها یعنی V درنظر گرفته‌ایم. در این مسئله $m = 2$ است. بنابراین پیچیدگی این مرحله برابر $O(2|V|^2)$ خواهد بود.

۲) عملگر ترکیب دو کروموزوم شامل دو مرحله اصلی زیر است:

- انتخاب k ($k \leq |E_1|$) یال از مجموعه E_1 به صورت تصادفی و اضافه کردن آن به T_2: $O(|E_1|)$. چون E_1 تفاوت بین یال‌های دو درخت را نشان می‌دهد بنابراین $|E_1|$ بین صفر تا $|V| - 1$. بنابراین پیچیدگی زمان این بخش از مرتبه $O(|V|)$ است.
- حذف حداکثر k دور از T_2: پیچیدگی این مرحله از مرتبه $O((|V| + |E(T_2)|)(k + 1))$ است [۳۳]. از آن جا که درخت ایجاد شده k یال بیشتر از درخت پوشا دارد و بنابراین $|E(T_2)| = |V| - 1 + k$ و می‌دانیم مقدار k حداکثر برابر $|V| - 1$ خواهد بود. هم‌چنین به ازای هر عملگر ترکیب این فرآیند دو بار تکرار می شود. بنابراین پیچیدگی زمانی برای یک عملگر ترکیب برابر $O(6|V|^2 - 4|V|)$ است.

به طور کلی پیچیدگی زمانی عملگر ترکیب به ازای حداکثر $N = |V|$ بار عمل ترکیب از مرتبه $O(|V|^3)$ است.

۳) عملگر جهش شامل دو مرحله اصلی زیر است:

- انتخاب یک یال از $(E - E_T)$ و اضافه کردن به درخت پوشا E_T: $O(1)$.
- حذف دور ایجاد شده: در این حللت حداکثر یک دور ایجاد شده است بنابراین زمان اجرا برابر $O(2(|V| + |E_T|))$ است. از آن جا که درخت ایجاد شده یک یال بیشتر از درخت پوشا دارد و بنابراین $|E_T| = |V|$. بنابراین برای انجام عمل حذف دور به زمان $O(3|V|)$ نیاز است.

با توجه به موارد بالا، پیچیدگی زمانی عملگر جهش به ازای حداکثر $N = |V|$ بار عمل جهش (در حالتی که عملگر جهش بر روی تمامی اعضای جمعیت اعمال شود) از مرتبه $O(|V|^2)$ خواهد بود.

۴) پیچیدگی زمانی فاز ارزیابی:

در هر تکرار پیچیدگی زمانی فاز ارزیابی شامل محاسبه وزن درخت پوشا و محاسبه حداکثر فاصله بین رئوس و ریشه است. برای محاسبه وزن درخت پوشا تمامی وزن یال‌های انتخاب شده با یکدیگر جمع می‌گردند که برابر $O(E_T)$ می‌باشد. E_T برابر تعداد یال‌های درخت پوشا منتخب است. در الگوریتم پیشنهادی حداکثر فاصله رئوس تا ریشه توسط الگوریتم دایجکسترا محاسبه گردید. ازآنجایی‌که پیچیدگی زمانی الگوریتم دایجکسترا در صورت استفاده از هیپ فیبوناچی $O(|E_T| + |V|log|V|)$ است. بنابراین پیچیدگی زمانی محاسبه حداکثر فاصله بین رئوس و ریشه $O(|E_T| + |V|log|V|)$ خواهد بود. در هر تکرار حداکثر $2N = 2|V|$ تعداد کروموزوم ایجاد می‌شوند که لازم است ارزیابی گردند. درنتیجه پیچیدگی زمانی فاز ارزیابی از درجه $O(2|V|(|E_T| + |V|log|V|))$ است. از آن‌جا که E_T تعداد یال‌های درخت پوشا است و می‌دانیم برابر $|V| - 1$ است می‌توان آن را به صورت $O(2|V|^2 + 2|V|^2log|V| - 2|V|)$ نوشت.

با توجه به اینکه بدترین پیچیدگی زمانی متعلق به عملگر ترکیب است، الگوریتم پیشنهادی دارای زمان اجرا از مرتبه $O(\mathrm{T}|V|^3)$ خواهد بود.

۷- نتیجه‌گیری و کارهای آینده

در این پژوهش یک الگوریتم چندهدفه تکاملی برای مسئله درخت پوشا متوازن ارائه گردید. نتایج آزمایشی و ارزیابی‌های انجام شده بر روی مجموعه آزمون، حاکی از آن است که الگوریتم پیشنهادی قادر است با بهینه‌سازی هم‌زمان اهداف مسئله، بدون نیاز به تعیین پارامتر اولیه، پاسخ‌های مطلوبی را بیابد؛ همچنین این الگوریتم می‌تواند چندین نمونه راه‌حل (جبهه پارتو) را ارائه نماید. لذا به‌کارگیری روش پیشنهادی در کاربردهای مرتبط ارجح است. هر چند الگوریتم پیشنهادی توانسته در زمان اجرای پایین‌تری نسبت به سایر روش‌ها، نمونه راه‌حل مسئله را بیابد؛ ولیکن به طور کلی با افزایش اندازه گراف، زمان اجرای الگوریتم پیشنهادی افزایش می‌یابد. به عنوان ادامه پژوهش، می‌توان عملکرد تکنیک‌های پردازش موازی را به منظور کاهش زمان اجرا بررسی نمود. هم‌چنین می‌توان روش پیشنهادی را برای یک نمونه‌ی واقعی پیاده‌سازی و اجرا نمود تا نتایج تحقیق در کاربرد، ارزیابی گردد.

مراجع

[1] S. M. Ejabati, "Adaptive Increasing/Decreasing PSO for Solving Dynamic Optimization Problems," *Journal of Soft Computing and Information Technology*, vol. 7, no. 2, pp. 58-70, 2018.

[2] A. Mohammadi, S.-H. Zahiri, and S.-M. Razavi, "Performance of Intelligent Optimization Methods in IIR System Identification Problems," *Journal of Soft Computing and Information Technology*, vol. 6, no. 2, pp. 25-39, 2017.

[3] R. Campos and M. Ricardo, "A fast algorithm for computing minimum routing cost spanning trees ",*Computer Networks*, vol. 52, no. 17, pp. 3229-3247, 2008.

[4] C. Li, H. Zhang, B. Hao, and J. Li, "A survey on routing protocols for large-scale wireless sensor networks," *Sensors*, vol. 11, no. 4, pp. 3498-3526, 2011.

[5] R. Moharam and E. Morsy, "Genetic algorithms to balanced tree structures in graphs," *Swarm and Evolutionary Computation*, vol. 32, pp. 132-139, 2017.

[6] C. A. C. Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary algorithms for solving multi-objective problems*. Springer, 2007.

[7] P. Ngatchou, A. Zarei, and A. El-Sharkawi, "Pareto multi objective optimization," in *Proceedings of the 13th International Conference on, Intelligent Systems Application to Power Systems*, 2005, pp. 84-91: IEEE.

[8] W. Gao, M. Pourhassan, V. Roostapour ,and F. Neumann, "Runtime Analysis of Evolutionary Multi-objective Algorithms Optimising the Degree and Diameter of Spanning Trees," in *International Conference on Evolutionary Multi-Criterion Optimization*, 2019, pp. 504-515: Springer.

[9] S. Ronoud and S .Asadi, "An evolutionary deep belief network extreme learning-based for breast cancer diagnosis," *Soft Computing*, vol. 23, no. 24, pp. 13139-13159, 2019.

[10] K. Deb, "A robust evolutionary framework for multi-objective optimization," in *Proceedings of the 10th annual conference on Genetic and evolutionary computation*, 2008, pp. 633-640.

[11] P. ERDdS and A. wi, "On random graphs I," *Publ. Math. Debrecen*, vol. 6, pp. 290-297, 1959.

[12] S. Khuller, B. Raghavachari, and N. Young, "Balancing minimum spanning trees and shortest-path trees," *Algorithmica*, vol. 14, no. 4, pp. 305-321, 1995.

[13] M. Souier, M. Dahane, and F. Maliki, "An NSGA-II-based multiobjective approach for real-time routing selection in a flexible manufacturing system under uncertainty and reliability constraints," *The International Journal of Advanced Manufacturing Technology*, vol. 100, no. 9-12, pp. 2813-2829, 2019.

[14] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182-197, 2002.

[15] K. Singh and S. Sundar, "Artifical bee colony algorithm using problem-specific neighborhood strategies for the tree t-spanner problem," *Applied Soft Computing*, vol. 62, pp .110-118 ,2018.

[16] S. Sundar, "A Steady-State Genetic Algorithm for the Tree t-Spanner Problem," in *Soft Computing: Theories and Applications*: Springer, 2019, pp. 387-398.

[17] Y. Ran, Z. Chen, S. Tang, and Z. Zhang, "Primal dual based algorithm for degree-balanced spanning tree problem," *Applied Mathematics and Computation*, vol. 316, pp. 167-173, 2018.

[18] M. H. Tahan and S. Asadi, "MEMOD: a novel multivariate evolutionary multi-objective discretization," *Soft Computing*, vol. 22, no. 1, pp. 301-323, 2018.

[19] M. H. Tahan and S. Asadi, "EMDID: Evolutionary multi-objective discretization for imbalanced datasets," *Information Sciences*, vol. 432, pp. 442-461, 2018.

[20] K. Bharath-Kumar and J. Jaffe, "Routing to multiple destinations in computer networks ",*IEEE Transactions on communications*, vol. 31, no. 3, pp. 343-351, 1983.

[21] J. Cong, A. Kahng, G. Robins, M. Sarrafzadeh, and C. Wong, "Performance-driven global routing for cell based ics," in *[1991 Proceedings] IEEE International Conference on Computer Design: VLSI in Computers and Processors*, 1991, pp. 170-173: IEEE.

[22] J. Cong, A. B. Kahng, G. Robins, M. Sarrafzadeh, and C.-K. Wong, "Provably good performance-driven global routing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 11, no. 6, pp. 739-752, 1992.

[23] B. Awerbuch, A. Baratz, and D. Peleg, "Cost-sensitive analysis of communication protocols," Massachusetts Inst of Tech Cambridg Lab for Computer Science,1991.

[24] A. Avokh and G. Mirjalily, "Dynamic balanced spanning tree (DBST) for data aggregation in wireless sensor networks," in *2010 5th international symposium on telecommunications*, 2010, pp. 391-396: IEEE.

[25] R. Moharam, E. Morsy, and I. A. Ismail, "Genetic algorithms for balanced spanning tree problem," in *2015 Federated Conference on Computer Science and Information Systems (FedCSIS)*, 2015, pp. 537-545: IEEE.

[26] C. J. Alpert *et al.*, "Prim-Dijkstra Revisited: Achieving Superior Timing-driven Routing Trees," in *Proceedings of the 2018 International Symposium on Physical Design*, 2018, pp. 10-17: ACM.

[27] L. Davis, D. Orvosh, A. Cox, and Y. Qiu, "A genetic algorithm for survivable network design," in *Proceedings of the 5th International Conference on Genetic Algorithms*, 1993, pp. 408-415: Morgan Kaufmann Publishers Inc.

[28] P. Piggott and F. Suraweera, "Encoding graphs for genetic algorithms: An investigation using the minimum spanning tree problem," in *Progress in Evolutionary Computation*: Springer, 1993, pp. 305-314.

[29] R. Mathur, I. Khan, and V. Choudhary, "Genetic algorithm for dynamic capacitated minimum spanning tree," *International Journal of Computer Technology and Applications*, vol. 4, no. 3, p. 404, 2013.

[30] Q. P. Tan, "A genetic approach for solving minimum routing cost spanning tree problem," *International Journal of Machine Learning and Computing*, vol. 2, no. 4, p. 410, 2012.

[31] S. Balaji, V. Swaminathan, and K. Kannan, "Approximating maximum weighted independent set using vertex Support," *International Journal of Computational and Mathemtical Sciences1*, vol. 3, no. 8, pp. 406-411, 2009.

[32] A. Cayley, "A theorem on trees," *The Quarterly Journal of Mathematics*, vol. 23, pp. 376-378, 1889.

[33] D. B. Johnson, "Finding all the elementary circuits of a directed graph," *SIAM Journal on Computing*, vol. 4, no. 1, pp. 77-84, 1975.

پاورقی‌ها:

1 Balanced spanning tree

2 Non-dominated Genetic Algorithm-II

3 robust

4 Constrained Minimum Spanning Tree Problem (CMST)

5 Constrained Shortest Path Tree Problem (CSPT)

6 Minimum Maximum Stretch Spanning Tree Problem (MMST)