مجله مدل‌سازی پیشرفته ریاضی
سال ۱۴۰۰، دوره ۱۱، شماره ۲، ص ۱۹۵-۲۰۹

شاپا چاپی: ۸۰۸۸-۲۲۵۱
DOI: 10.22055/jamm.2021.35124.1860

ارائه الگوریتمی نوین برای حل مساله ساخت درخت فیلوژنتیک ریشه‌دار براساس سه‌تایی‌های ریشه‌دار ورودی

هادی پورمحمدی[۱,*]، محسن سرداری زارچی [۱]، محمدعلی صحتی [۱]، محمد میرابی [۲]، سید حسن مرتضوی زارچ [۱]

[۱] استادیار گروه مهندسی کامپیوتر، دانشکده فنی ومهندسی، دانشگاه میبد، میبد، ایران
[۲] استادیار گروه مهندسی صنایع، دانشکده فنی و مهندسی، دانشگاه میبد، میبد، ایران

تاریخ دریافت: ۱۳۹۹/۶/۳۱ تاریخ پذیرش: ۱۳۹۹/۱۰/۴
دبیر مسئول: محمد رضا درفشه

چکیده: فیلوژنتیک شاخه‌ای از دانش بیوانفورماتیک است که تاریخچه روابط تکاملی بین گونه‌های موجودات زنده را بررسی می‌کند و مدل‌هایی را ارائه می‌دهد و گونه‌ها را دسته‌بندی نیز می‌کند. فیلوژنتیک از شبکه‌ها برای بیان روابط تکاملی در یک مدل جامع استفاده می‌کند. ساده‌ترین مدل شبکه ای، یک درخت است که درخت فیلوژنتیک نامیده می‌شود. درختهای فیلوژنتیک به دو زیرمجموعه ریشه‌دار و بدون ریشه تقسیم می‌شوند. در این پژوهش علاقه ما، درخت‌های فیلوژنتیک ریشه‌دار است. ورودی‌های مختلفی برای ساخت درخت‌های فیلوژنتیک ریشه‌دار موجود است. سه‌تایی‌های ریشه‌دار یکی از انواع مهم ورودی‌ها در تحلیل روابط بین گونه‌ها و ساخت درخت فیلوژنتیک ریشه‌دارند که در این پژوهش از آن استفاده می‌کنیم. سه‌تایی ریشه‌دار یک درخت دودویی ریشه‌دار با سه برگ است. سه‌تایی ریشه‌دار ساده‌ترین مدل درختی ریشه‌دار است که حاوی اطلاعات است. مساله ساخت درخت فیلوژنتیک ریشه‌داری که دربرگیرنده بیش‌ترین سه‌تایی ورودی باشد، یک مساله بهینه سازی است و به مساله MRTC مشهور است. چالش مهم در فیلوژنتیک، NP-سخت بودن مساله MRTC است. از این‌رو در این مقاله، یک الگوریتم نوین برای مساله MRTC با هدف افزایش میزان سازگاری سه‌تایی‌های ریشه‌دار ورودی با درخت نهایی، معرفی می‌شود. با هدف بیان کارایی الگوریتم نوین، روش معرفی‌شده را با روش TRH و بر روی داده‌های واقعی مقایسه می‌کنیم. الگوریتم TRH یکی از بهترین روش‌ها برای حل مساله MRTC و بر روی سه‌تایی‌های ریشه‌داری است که از روی داده‌های واقعی به‌دست آمده‌اند. نتایج آزمایش‌ها نشان می‌دهد که با در نظر گرفتن زمان اجرا و سازگاری سه‌تایی‌ها با درخت نهایی، الگوریتم ما نتایج TRH را بهبود می‌بخشد.

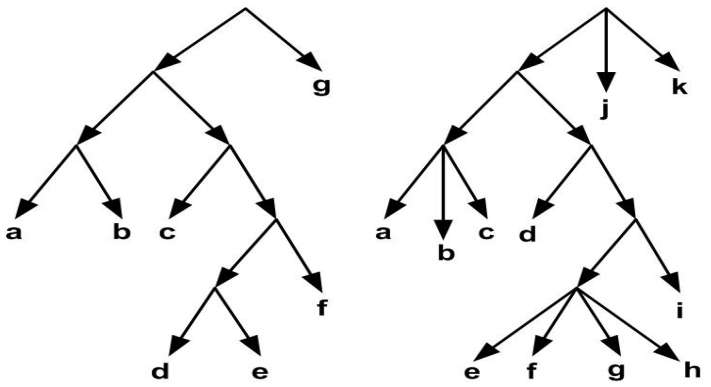
واژه‌های کلیدی: بیوانفورماتیک، توالی زیستی، مساله MRTC ، درخت فیلوژنتیک ریشه‌دار، سه‌تایی ریشه‌دار.

رده‌بندی ریاضی: 05C20, 92B05, 90C12, 05C12

* نویسنده مسئول مقاله: هادی پورمحمدی
رایانامه: pormohamadi@meybod.ac.ir

۱ مقدمه

توالی‌های زیستی مهم‌ترین ورودی برای تحلیل روابط بین موجوداتاند [۱۰، ۱۱، ۱۳]. برای یک مجموعه دلخواه داده‌شده از گونه‌ها، می‌توان با داشتن یک توالی خاص از ژنوم ، که در همه آنها مشترک است، نوع ارتباط بین گونه‌های مجموعه را بر اساس توالی‌های داده‌شده به‌دست آورد. فیلوژنتیک شاخه‌ای از علم بیوانفورماتیک است که روابط تکاملی بین گونه‌های موجودات زنده را بررسی می‌کند و مدل‌هایی را برای بیان روابط معرفی می‌کند [۲، ۸، ۱۳]. در فرایند تکامل و در مدل‌های پایه، فرض بر این است که گونه‌های یک مجموعه دارای یک جد مشترک در گذشته‌ای دور یا نزدیک بوده‌اند و تمام آن گونه‌ها از آن جد مشترک پدید آمده‌اند و تفاوت‌های ژنتیکی بین گونه‌ها که از یک جد مشترک پدید آمده‌اند، به دلیل تاثیرات محیطی و رخداد جهش در توالی‌های مربوط به گونه‌های آن دسته در طول قرن‌ها است. از این‌رو به‌صورت استاندارد از مدل‌های درختی برای تحلیل نوع ارتباط بین گونه‌ای استفاده می‌شود [۸، ۱۳]. در واقع درخت‌ها ساده‌ترین ساختارها برای نمایش مجموعه‌ای از روابط بین گونه‌ای در یک مدل جامع‌اند. بر اساس نیاز و نوع ورودی مساله، انواع مختلفی از ساختارهای درختی موجود است. در حالت کلی ساختارهای درختی به دو نوع ریشه‌دار و بدون ریشه تقسیم می‌شوند. در مدل‌های ریشه‌دار یک جد مشترک برای تمامی گونه‌ها در نظر گرفته می‌شود که ریشه نامیده می‌شود و تمامی راس‌های دیگر درخت نشان‌دهنده گونه‌هایی‌اند که در طول قرن‌ها و بر مبنای پدیده‌های زیستی و طبیعی مثل جهش، حذف، درج و غیره به‌وجود می‌آیند . بجز ریشه راس‌های درخت به دو مجموعه راس‌های درونی و برگ‌ها افراز می‌شوند. راس‌های درونی در یک مدل ریشه‌دار نشان‌دهنده گونه‌هایی‌اند که در گذشته می‌زیسته‌اند [۸]. برگ‌ها نشان‌دهنده گونه‌هایی‌اند که در حال حاضر موجودند و هدف تحلیل نوع روابط بین گونه‌ای برای همین داده‌های موجود است که به‌عنوان برگ‌های درخت معرفی می‌شوند. درخت‌های ریشه‌دار به‌عنوان ساختارهای گرافی جهت‌دار نیز شناخته می‌شوند. با این نگاه، ریشه درخت تنها راسی است که ورودی آن ۰ است. ورودی راس‌های درونی درخت عدد ۱ و خروجی آنها بزرگ‌تر از ۱ است. همچنین ورودی برگ‌های درخت عدد ۱ و خروجی آنها ۰ است. همان‌طور که گفته شد در یک مدل درختی برگ‌ها توسط گونه‌های ورودی به‌صورت مجزا برچسب‌گذاری می‌شوند. یعنی برای تحلیل یک مجموعه از داده‌ها که متناظر با یک مجموعه n تایی از گونه‌ها است به‌دنبال یک ساختار درختی با n برگ هستیم به‌طوری‌که برگ‌ها به‌صورت مجزا توسط این گونه‌ها برچسب‌گذاری شده‌اند [۸]. به‌صورت دقیق، درخت فیلوژنتیک ریشه‌دار به‌صورت زیر تعریف می‌شود. فرض کنید X مجموعه‌ای از گونه‌ها باشد. یک درخت فیلوژنتیک ریشه‌دار روی X، درختی ریشه‌دار است که یال‌ها جهت‌دارند. ریشه درجه ورودی ۰ و خروجی بزرگ‌تر از ۱ دارد. برگ‌ها درجه ورودی ۱ و درجه خروجی ۰ دارند. برگ‌ها مرتب نیستند و توسط اعضای مجموعه X به‌صورت مجزا برچسب‌گذاری شده‌اند و هر برگ دارای برچسب است و هر رأسی که برگ نیست درجه ورودی ۱ و درجه خروجی بزرگتر از ۱ دارد [۸، ۱۳]. برای سادگی درخت فیلوژنتیک ریشه‌دار را یک درخت می‌نامیم (شکل ۱آ). یک درخت دودویی ریشه‌دار یک درخت است با این تفاوت که درجه خروجی هر راس بجز برگ‌ها، دقیقا ۲ است [۸، ۱۳] (شکل ۱ب). با توجه به تعریف مشخص است که هر درخت دودویی ریشه‌دار یک درخت فیلوژنتیک ریشه‌دار هم است ولی عکس این مطلب لزوما درست نیست.

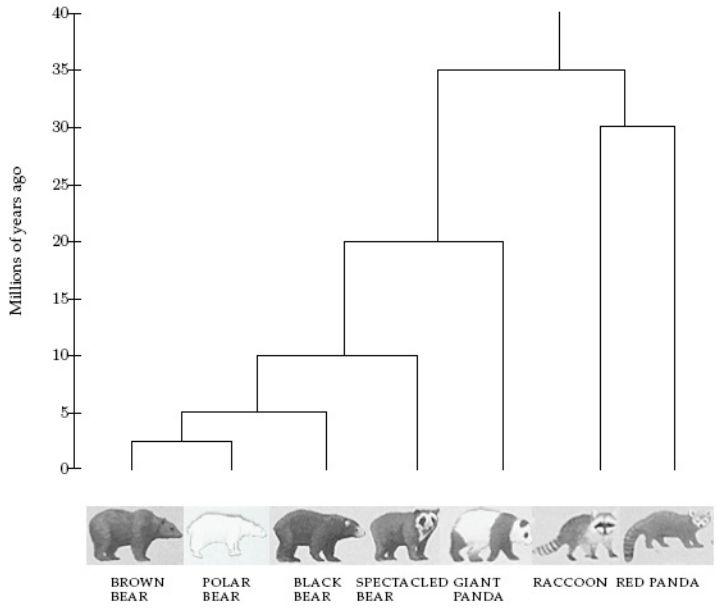


(آ) درخت فیلوژنتیک ریشه‌دار (ب) درخت دودویی ریشه‌دار

شکل ۱

اگر ساختار مورد نظر یک درخت بدون ریشه باشد در تعداد برگ‌های درخت نهایی تغییری ایجاد نمی‌شود. فقط این نکته مهم را باید در نظر گرفت که ساختارهای درختی بدون ریشه را نمی‌توان به‌عنوان گراف‌های جهت‌دار در نظر گرفت و گراف‌های ساده بدون جهت‌اند یعنی برای هر راس صرفا یک درجه داریم و دیگر درجه ورودی و خروجی معنایی ندارد. در این نوع ساختارها، برگ‌ها راس‌هایی با درجه ۱اند. توجه کنیم که گاهی بر اساس نیاز مساله هدف این است که یک ساختار درختی بدون ریشه به یک ساختار ریشه‌دار تبدیل شود. در این حالت یک روش استاندارد این است که در وسط یک از راس‌های درخت، یک راس جدید ایجاد شود و آن راس جدید به‌عنوان ریشه در نظر گرفته شود و سپس با توجه به ریشه جدید، بقیه ساختار درخت به‌صورت ریشه‌دار در نظر گرفته می‌شود. دقت کنیم که در این حالت به ساختار کلی درخت خللی وارد نمی‌شود. در این حالت نکته مهم این است که کدام راس به عنوان راسی در نظر گرفته شود که قرار است روی آن ریشه ایجاد شود. این کار بر اساس اطلاعات زیستی امکان‌پذیر است و تصادفی نیست و نیاز به تحلیل دقیق داده‌ها دارد [۸، ۱۳]. در این پژوهش هدف

ما بررسی مدل‌ها و ساختارهای ریشه‌دار و تحلیل این نوع مدل‌ها بر اساس داده ورودی است. در واقع هدف ساخت یک درخت (فیلوژنتیک ریشه‌دار) بر مبنای داده‌های ورودی است. در این مدل‌ها برگ‌ها توسط گونه‌های مورد نظر برچسب‌گذاری می‌شوند و رأس‌های داخلی ،درخت نماینده گونه‌هایی‌اند که در گذشته می‌زیسته‌اند (شکل ۲) [۸]. توجه کنیم که گاهی برای یک مجموعه داده‌شده از داده‌های ورودی، می‌توان مدل ریشه‌دار یا بدون ریشه در نظر گرفت ولی برای برخی داده‌ها صرفا مدل‌های درختی ریشه‌دار و برای برخی داده‌ها صرفا مدل‌های درختی بدون ریشه با معنی است. از آنجا که در یک ساختار درختی می‌توان فرایند تکامل را برای یک مجموعه داده‌شده از داده‌ها تحلیل کرد، از این‌رو این نوع درخت‌ها را درخت‌های تکاملی نیز می‌نامند [۸، ۱۳].

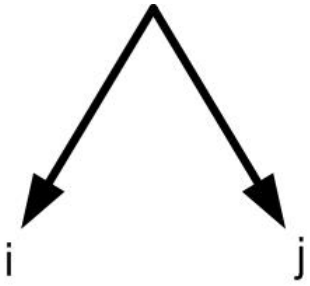


شکل ۲: درخت مرتبط با گونه‌های خرس.

در ساده‌ترین حالت، ورودی این مدل‌های درختی، توالی‌های مربوط به گونه‌ها بود و بر اساس توالی‌های ورودی روش‌هایی برای به‌دست آوردن درخت تکاملی پیشنهاد شد. در این نوع مدل‌ها توالی‌های زیستی ورودی، به‌صورت مستقیم و بدون واسطه برای تحلیل مورد استفاده قرار می‌گیرند. در این نوع مدل‌ها در ابتدا توالی‌های زیستی ورودی با استفاده از روش‌های معتبر موجود برای هم‌ردیف کردن توالی‌ها، هم‌ردیف می‌شوند. با انجام این کار در نهایت همه توالی‌ها هم‌طول می‌شوند و ورودی مساله تبدیل به یک مجموعه nتایی از توالی‌ها می‌شود که طول هر توالی m است. مساله هم‌ردیف کردن بیش از دو توالی یک مساله مهم در حوزه زیست‌شناسی است و از نظر پیچیدگی محاسباتی یک مساله NP– سخت است [۸، ۱۳]. در ادامه روش‌هایی معرفی شد که از روی اطلاعات توالی‌های مربوط به گونه‌ها و برخی پیش‌فرض‌ها، یک ماتریس فاصله به‌دست آمده و از روی ماتریس فاصله، درخت تکاملی ساخته می‌شود [۸]. در این گونه روش‌ها، در واقع از اطلاعات توالی‌ها به‌صورت مستقیم استفاده نمی‌شود و در ابتدا با استفاده از روش‌های استاندارد، اطلاعات توالی‌ها تبدیل به ماتریس فاصله می‌شود. در این حالت درخت تکاملی یک درخت وزن‌دار است که سعی بر این است که مجموع وزن یال‌ها در مسیر یکتای بین هر دو برگ، نزدیک به عدد متناظر در ماتریس فاصله باشد. بهترین حالت زمانی رخ می‌دهد که این دو عدد یکی شوند. روش‌هایی وجود دارد که درختی را بازسازی می‌کند که این دو عدد یکی می‌شوند. چون اطلاعات توالی‌ها تبدیل به ماتریس فاصله می‌شود، پیش‌فرض‌های زیادی در نظر گرفته می‌شود که در صورت تغییر هر کدام از آنها ممکن است ماتریس فاصله تغییر کند. در این شیوه نیز در ابتدا توالی‌ها هم‌ردیف می‌شوند. سپس برای هر دو توالی هم‌ردیف یک عدد به‌دست می‌آید که آن عدد نشان‌دهنده میزان فاصله زیستی بین دو توالی است. با توجه به در نظر گرفتن پیش‌فرض‌های زیستی زیاد در بدست آوردن ماتریس فاصله از روی توالی‌های زیستی و همچنین علاقه به یافتن شیوه‌های نوین برای تبدیل اطلاعات توالی‌های زیستی به داده ورودی، روش‌های دیگری برای تبدیل اطلاعات توالی‌ها به ورودی‌ها برای رسم درخت معرفی شد [۸].

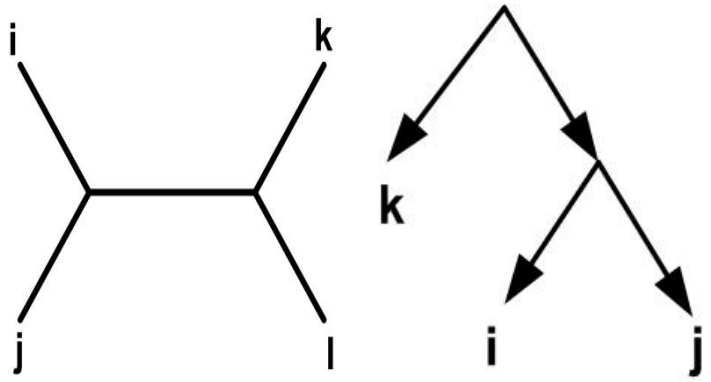
هر چه تعداد توالی‌های ورودی کم‌تر باشد درخت نهایی دقیق‌تر است. به عنوان مثال اگر داده ورودی تنها شامل دو توالی باشد بصورت دقیق می‌توان درخت نهایی را رسم کرد؛ زیرا در این حالت تنها یک ساختار درختی یکتا موجود است (شکل ۳).

هر چه تعداد توالی‌های ورودی بیش‌تر شود طبیعتا پیچیدگی حل مساله بیش‌تر می‌شود؛ زیرا تعداد حالت‌های رسم درخت نهایی با افزایش توالی‌های ورودی که منجر به افزایش تعداد برگ‌های درخت نهایی می‌شود، افزایش می‌یابد. با این دیدگاه می‌توان نتیجه گرفت که می‌توان مساله را در ابتدا به زیرمساله‌ها تبدیل کرد و سپس با استفاده از زیرمساله‌ها مدل نهایی را ارائه کرد و مساله را حل کرد. در این دیدگاه و در مدل‌های جدید، با اطمینان بالا، درخت تکاملی بین هر سه یا چهار گونه به‌دست می‌آید و سپس هدف یافتن درخت و یا شبکه تکاملی کلی است که دربرگیرنده همه اطلاعات مربوط به این درخت‌های کوچک است. تبدیل اطلاعات توالی‌ها به درخت‌های روی سه برگ نسبت به درخت‌های روی چهار برگ، با خطای کم‌تری همراه است و در واقع تمام اطلاعات مربوط به درخت‌های روی چهار برگ در درخت‌های روی



شکل ۳: ساختار درختی یکتا برای دو گونه.

سه برگ نهفته است و از این‌رو، این نوع درخت‌ها، قابل اعتمادترند. به درخت‌های ریشه‌دار روی سه برگ، سه‌تایی ریشه‌دار و به درخت‌های بدون ریشه روی چهار برگ، چهارتایی می‌گویند. به‌بیان دقیق یک سه‌تایی ریشه‌دار (سه‌تایی به اختصار) یک درخت ریشه‌دار دودویی غیر مرتب همراه با سه برگ است که در آن برگ‌ها به‌صورت مجزا برچسب‌گذاری شده‌اند. از نماد $ij|k$ برای نمایش یک سه‌تایی استفاده می‌شود که در آن برگ‌های i و j در یک سمت ریشه و برگ k در سمت دیگر ریشه است (شکل ۴أ). [۸، ۱۳]. همچنین به‌صورت دقیق یک چهارتایی یک درخت دودویی بدون ریشه روی چهار برگ است که یک یال داخلی و دو راس داخلی دارد و درجه راس‌های داخلی ۳ است. از نماد $ij|kl$ برای نمایش یک چهارتایی استفاده می‌شود که در آن برگ‌های i و j به یک راس داخلی و برگ‌های k و l به راس دیگر داخلی متصل‌اند [۷] (شکل ۴ب).



(أ) سه‌تایی ریشه‌دار روی برگ‌های i,j,k (ب) چهارتایی روی برگ‌های i,j,k,l

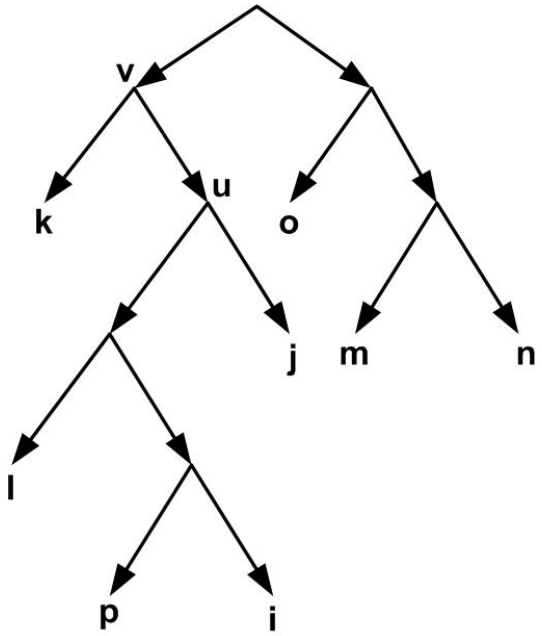
شکل ۴

روش‌های استانداردی برای تبدیل اطلاعات توالی‌ها به سه‌تایی‌های(ریشه‌دار) وجود دارد. روش‌های بزرگ‌ترین درست‌نمایی (ML) و بیش‌ترین خست (MP) دو روش استاندارد موجود برای تولید سه‌تایی‌هایند. به‌طور کلی این دو روش، روش‌های استانداردی برای ساخت درخت از روی توالی‌های زیستی‌اند [۵، ۲۸]. هم‌چنین در چند سال اخیر روشی جدید برای ساخت سه‌تایی‌ها با استفاده از ماتریس فاصله، ارائه شده است [۲۱]. در برخی از مسالهها دادههایی که از روش‌های آزمایشگاهی به‌دست می‌آیند به شکل سه‌تایی‌اند [۱۹].

یک مساله پایه و اساسی فیلوژنتیک در حوزه درخت‌ها و سه‌تایی‌ها، ساخت درختی ریشه‌دار است که برگ‌ها توسط گونه‌ها برچسب‌گذاری شده‌اند و هر سه‌تایی یک زیرگراف القایی این درخت ریشه‌دار است. در این حالت سه‌تایی با درخت مورد نظر سازگار است [۸]. به بیان دقیق سه‌تایی $ij|k$ با درخت داده‌شده T سازگار است و یا به‌طور معادل T با $ij|k$ سازگار است هرگاه مجموعه برگ‌های $\mathrm{ij|k}$ زیرمجموعه‌ای از مجموعه برگ‌های T باشد و T شامل راس‌های متمایزی مانند u و v و مسیرهای جهت‌دار $v \to k$ و $u \to i, u \to j, v \to u$ باشد به‌طوری‌که این چهار مسیر حداکثر در رأس‌های ابتدایی و انتهایی اشتراک داشته باشند و هیچ یال مشترکی نداشته باشند (شکل ۵).

فرض‌کنید τ مجموعه‌ای از سه‌تایی‌ها باشد. τ با درخت T سازگار است، هرگاه هر عضو τ با T سازگار باشد. به‌عنوان مثال $\tau = \{pi|j, il|k, mn|k, ij|k, on|p, lj|m, pl|j\}$با درخت شکل ۵ سازگار است.

در ادامه و در بخش ۲ مساله ساخت درختی سازگار با مجموعه داده‌شده از سه‌تایی‌های ورودی را به‌صورت دقیق بررسی می‌کنیم و الگوریتم‌های معتبر موجود برای این مساله را معرفی می‌کنیم. در بخش ۳ در ابتدا تعاریف و مفاهیم پایه ارائه می‌شود. سپس الگوریتم‌های پایه که در روش پیشنهادی مورد استفاده قرار می‌گیرد، توضیح داده‌می‌شود. در بخش ۴ روش پیشنهادی معرفی می‌شود. در بخش ۵ نتایج آزمایش‌ها بیان می‌شود. در نهایت و در بخش ۶ یک نتیجه‌گیری کلی در مورد کارایی روش معرفی‌شده و از طریق تحلیل نتایج بخش ۵ ارائه می‌شود.



شکل ۵: مجموعه $\tau = \{pi|j, il|k, mn|k, ij|k, on|p, lj|m, pl|j\}$ با درخت داده‌شده سازگار است.

۲ مرور سوابق

برای مجموعه داده‌شده τ از سه‌تایی‌های ورودی، ایهو و همکاران مساله ساخت درختی ریشه‌دار و سازگار با τ را بررسی و الگوریتم BUILD را معرفی کردند [۱]. این الگوریتم در زمان چندجمله‌ای تصمیم‌گیری می‌کند که آیا برای مجموعه دلخواه داده‌شده از سه‌تایی‌ها درختی ریشه‌دار و سازگار با همه سه‌تایی‌های ورودی، وجود دارد یا خیر و اگر چنین درخت ریشه‌داری وجود داشته باشد در زمان چندجمله‌ای درخت ریشه‌دار مورد نظر را به‌صورت یکتا می‌سازد. در ادامه دو روش جهت بهبود زمان اجرای الگوریتم BUILD معرفی شد [۱۲، ۱۶]. اگر درخت ریشه‌داری که دربرگیرنده تمام سه‌تایی‌های ریشه‌دار ورودی باشد وجود نداشته باشد آنگاه الگوریتم BUILD در مرحله‌ای متوقف می‌شود. در این حالت مساله مهم ساخت درختی است که با بیش‌ترین سه‌تایی ورودی سازگار باشد. در برخی موارد می‌دانیم که برای ورودی‌های به‌شکل سه‌تایی، درختی که با تمام سه‌تایی‌های ورودی سازگار باشد وجود دارد اما به‌دلیل بروز خطا در به‌دست آوردن ورودی‌ها، یافتن چنین درختی ممکن نیست. از این‌رو هدف ساخت درختی است که دربرگیرنده بیش‌ترین سه‌تایی ورودی باشد. هدف از این کار، حذف کم‌ترین تعداد اطلاعات سه‌تایی‌ها به‌منظور به‌دست آوردن جواب نهایی است. این مساله، بیش‌ترین سازگاری سه‌تایی ریشه‌دار با درخت (MRTC) نامیده می‌شود و از نظر پیچیدگی محاسباتی ، NP– سخت است [۳، ۴، ۱۵]. همچنین شیوه‌هایی وجود دارد که بر اساس سه‌تایی‌های ورودی و هنگامی‌که الگوریتم BUILD قادر به ساخت یک درخت نیست، هدف ساخت یک شبکه بهینه است به‌طوری‌که شامل تمام سه‌تایی‌های ورودی باشد [۱۷، ۱۸، ۲۲–۲۵، ۲۷]. مساله ساخت شبکه‌ای بهینه سازگار با تمام سه‌تایی‌های ورودی نیز یک مساله NP– سخت است [۲۲، ۲۴] . در حالت کلی همیشه یک شبکه سازگار با تمام سه‌تایی‌های ورودی موجود است. از این‌رو در مساله ساخت شبکه به‌دنبال شبکه بهینه‌ایم در حالی‌که در مساله MRTC درخت بهینه نداریم و صرفا هدف ساخت درختی سازگار با بیش‌ترین سه‌تایی ورودی است. در این مقاله ساخت شبکه‌های بهینه بر مبنای سه‌تایی‌های ورودی، مورد بررسی قرار نمی‌گیرد و هدف بررسی مساله MRTC است.

باتوجه به الگوریتم BUILD، «گاسینیک» و همکاران دو الگوریتم را معرفی کردند. این دو الگوریتم به‌ترتیب One–Leaf–Split و Min–Cut–Split نامیده می‌شوند [۹]. الگوریتم One–Leaf–Split تضمین می‌کند که درخت ساخته شده، حداقل دربرگیرنده یک‌سوم از سه‌تایی‌های ورودی است [۹]. کران پایین یک‌سوم عدد قابل ملاحظه‌ای نیست و صرفا هدف این روش ارائه یک پاسخ برای مساله MRTC است. از این‌رو در عمل این الگوریتم کارایی چندانی ندارد. الگوریتم دوم از روند کلی الگوریتم BUILD پیروی می‌کند و در هر مرحله که BUILD متوقف می‌شود، با استفاده از یک شیوه پیشنهادی و حذف برخی از اطلاعات، الگوریتم BUILD را ادامه می‌دهد و در نهایت منجر به ساخت یک درخت می‌شود. در حالت کلی نتایج الگوریتم دوم نیز فاصله زیادی تا جواب بهینه دارد و صرفا به‌عنوان روشی برای پیشبرد الگوریتم BUILD معرفی شده است [۹]. «وو» یک الگوریتم دقیق و نمایی از مرتبه $O(m + n^2)3^n$ و فضای $O(2^n)$ برای مساله MRTC معرفی کرده است که m تعداد سه‌تایی‌ها و n تعداد گونه‌های موجود یا همان تعداد برگ‌ها است [۲۹]. این الگوریتم دقیق و نمایی بهترین جواب ممکن را می‌دهد اما بسیار زمان‌بر است و با افزایش تعداد گونه‌ها، زمان اجرای آن به‌صورت نمایی رشد پیدا می‌کند و در عمل در زمان مناسب و قابل قبول خروجی نمی‌دهد. همچنین «وو» یک الگوریتم تقریبی ساخت درخت بر مبنای روش ساخت از پایین به بالا معرفی کرد. این الگوریتم به اختصارBPMF نامیده می‌شود. زمان اجرای این الگوریتم $O(mn^3)$ است [۲۹]. نتایج الگوریتم BPMF نشان می‌دهد که به‌طور متوسط کارایی این الگوریتم بر روی داده‌هایی که به‌صورت تصادفی تولید شده‌اند، خوب و مناسب است. همچنین

به‌طور متوسط زمان اجرای این الگوریتم قابل قبول است. این الگوریتم به‌طور متوسط نسبت به تمامی روش‌های قبلی کاراتر است. «مامورا» و همکاران الگوریتمBPMR را معرفی کردند. الگوریتم BPMR بهبود یافته الگوریتم BPMF است. زمان اجرای این الگوریتم همانند الگوریتم BPMF از مرتبه $O(mn^{۳})$ است [۲۰]. همچنین «جهانگیری» و همکاران دو روش تقریبی ساخت درخت با نام‌های FastTree و BPMTR معرفی کردند [۱۴]. الگوریتم FastTree زمان اجرای بسیار پایینی دارد و یکی از سریع‌ترین روش‌های موجود برای حل مساله است. الگوریتم BPMTR بهبودیافته روش BPMR است و نسبت به BPMR نتایج بهتری می‌دهد [۱۴].

در سال ۲۰۱۲ الگوریتم TRH بر پایه ایده تابع ارتفاع برای مساله MRTC معرفی شد [۲۱]. به‌طور متوسط الگوریتم TRH بر روی داده‌های واقعی عملکرد بهتری نسبت به الگوریتم BPMR دارد. همچنین در حالت کلی زمان اجرای TRH نسبت به BPMR بهتر است [۲۱]. در واقع هدف از معرفی الگوریتم TRH ارائه یک روش موثر برای حل مساله MRTC و بر روی داده‌های واقعی و در زمان قابل قبول است که تا پیش از این مورد توجه قرار نگرفته است. همچنین با توجه به مزیت نسبی زمان اجرای الگوریتم TRH نسبت به BPMR و نزدیک بودن تقریبی پاسخ‌های دو الگوریتم، در حالت کلی TRH یک روش کاراتر برای حل مساله MRTC نسبت به روش‌های موجود است. در الگوریتم TRH یک ضعف عمده وجود دارد. این الگوریتم از روش تصادفی و غیر هوشمندانه برای ساخت درخت دودویی نهایی استفاده می‌کند. از این‌رو معرفی شیوه‌ای نوین و کارامد برای رفع این نقص در الگوریتم TRH ضروری است. یک شیوه کارامد بدین صورت است که با رفع این نقص هم زمان اجرای الگوریتم TRH افزایش نیابد و هم درخت نهایی در برگیرنده تعداد بیشتری سه‌تایی ورودی نسبت به قبل باشد. همچنین در برخی قسمت‌های دیگر الگوریتم TRH از شیوه‌های نوین استفاده می‌کنیم. از این‌رو در این مقاله روشی نوین برای مساله MRTC معرفی می‌شود که ضعف الگوریتم TRH در بازسازی درخت دودویی نهایی را مرتفع می‌سازد. همچنین برخی قسمت‌های الگوریتم TRH بهبود پیدا می‌کند و در حالت کلی زمان اجرای شیوه نوین همانند الگوریتم TRH است.

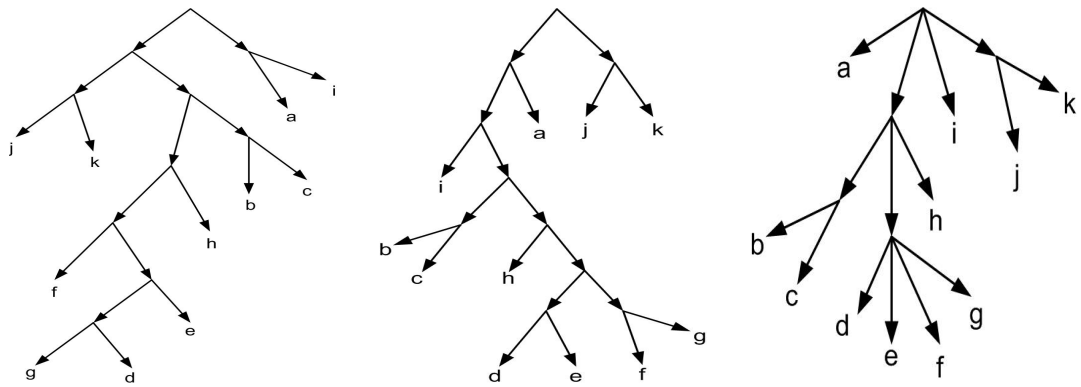
۳ تعاریف و الگوریتم‌های پایه

در این بخش در ابتدا تعاریف مورد نیاز گراف و درخت بیان می‌شود و سپس الگوریتم‌های پایه که در روش پیشنهادی مورد استفاده قرار گرفته‌اند، توضیح داده‌می‌شود.

۱.۳ تعاریف

تعریف ۱.۳. برای درخت داده‌شده T فرض کنید $\tau(T)$ و L_T به‌ترتیب نمایانگر مجموعه همه سه‌تایی‌های سازگار با T و مجموعه همه برچسب‌های برگ‌های T باشد. برای مجموعه سه‌تایی‌های τ، فرض کنید $L(\tau) = \cup_{t\in\tau} L_t$. برای مجموعه داده‌شده از سه‌تایی‌های τ، τ مجموعه‌ای از سه‌تایی‌ها روی مجموعه X است هرگاه $L(\tau) = x$ [۲۲، ۲۴]. به عنوان مثال برای $\tau = \{mj|i, kl|i, im|k, ij|k\}$، τ یک مجموعه از سه‌تایی‌ها روی $X = \{i, j, k, l, m\}$ است.

تعریف ۲.۳. فرض‌کنید T یک درخت باشد. درخت T' یک دودویی‌سازی از T است هرگاه T' یک درخت ریشه‌دار دودیی باشد، مجموعه برگ‌های دو درخت یکی باشد، و هر سه‌تایی که با T سازگار است با T' نیز سازگار باشد (شکل ۶). برای هر درخت T یک دودویی‌سازی دلخواه وجود دارد [۲۲].



شکل ۶: دو درخت دودویی، دو دودویی‌سازی دلخواه از درخت غیر دودویی‌اند.

تعریف ۳.۳. G_τ گراف جهت‌دار وابسته به τ گرافی است که $V(G_\tau) = \{\{i,j\} : i,j \in L(\tau), i \neq j\}$ (برای سادگی $\{i,j\}$ را با ij نمایش می‌دهیم) و $E(G_\tau) = \{(ij,ik) : ij|k \in \tau\} \cup \{(ij,jk) : ij|k \in \tau\}$ [۲۲].

تعریف ۴.۳. فرض‌کنید $\begin{pmatrix} X \\ ۲ \end{pmatrix}$ نمایانگر مجموعه همه زیرمجموعه‌های دو عضوی از مجموعه متناهی X باشد. تابع دلخواه $h : \begin{pmatrix} X \\ ۲ \end{pmatrix} \to N$ یک تابع ارتفاع روی X نامیده می‌شود [۲۲].

فرض‌کنید τ مجموعه‌ای از سه‌تایی‌ها باشد، G_τ یک گراف جهت‌دار بدون دور باشد، و l_{G_τ} نشان‌دهنده طول بزرگ‌ترین مسیر جهت‌دار در G_τ باشد. چون G_τ یک گراف جهت‌دار بدون دور است، دارای رأس‌هایی با درجه خروجی صفر است. به رأس‌های با درجه خروجی صفر در G_τ عدد ۱ + l_{G_τ} نسبت داده می‌شود و از G_τ حذف می‌شوند. در گراف جدید به رأس‌های با درجه خروجی صفر عدد l_{G_τ} نسبت داده می‌شود و این کار تا آنجا ادامه می‌یابد تا همه رأس‌ها حذف شوند. با این کار رأس‌های G_τ با اعداد بین ۱ و ۱ + l_{G_τ} برچسب‌گذاری می‌شوند.

تعریف ۵.۳. به‌ازای هر دو رأس متمایز $i,j \in L(\tau)$، $h_{G_\tau}(i,j)$ مقداری است که در فرایند بالا به رأس ij نسبت داده شده است و تابع ارتفاع وابسته به G_τ نام دارد. اگر τ مجموعه‌ای از سه‌تایی‌ها باشد که با درختی سازگار است، آنگاه G_τ یک گراف جهت‌دار بدون دور است و h_{G_τ} خوش‌تعریف است [۲۲].

۲.۳ الگوریتم‌های پایه

فرض‌کنید X مجموعه‌ای از گونه‌ها و τ مجموعه‌ای از سه‌تایی‌ها روی X باشد. گراف ایهو $AG(\tau) = (V,E)$ وابسته به τ گرافی است که در آن V=X و دو رأس دلخواه i و j از V توسط یالی در E به‌یکدیگر متصل‌اند اگر و تنها اگر برای یک مقدار دلخواه k، سه‌تایی ij|k در τ باشد [۱].

فرض کنید τ مجموعه‌ای از سه‌تایی‌ها باشد که با درختی سازگار است. الگوریتم BUILD یک الگوریتم ساخت درخت از بالا به پایین است که درختی سازگار با τ را می‌سازد. به‌بیان دیگر این الگوریتم در ابتدا ریشه را می‌سازد و با شروع از ریشه و ساخت یال‌ها و رأس‌های داخلی درخت، در نهایت به برگ‌ها می‌رسد و برگ‌ها را می‌سازد. با این روند کل درخت ساخته می‌شود. این الگوریتم ساخت، از گراف ایهو به‌عنوان یک راهنما استفاده می‌کند [۱].

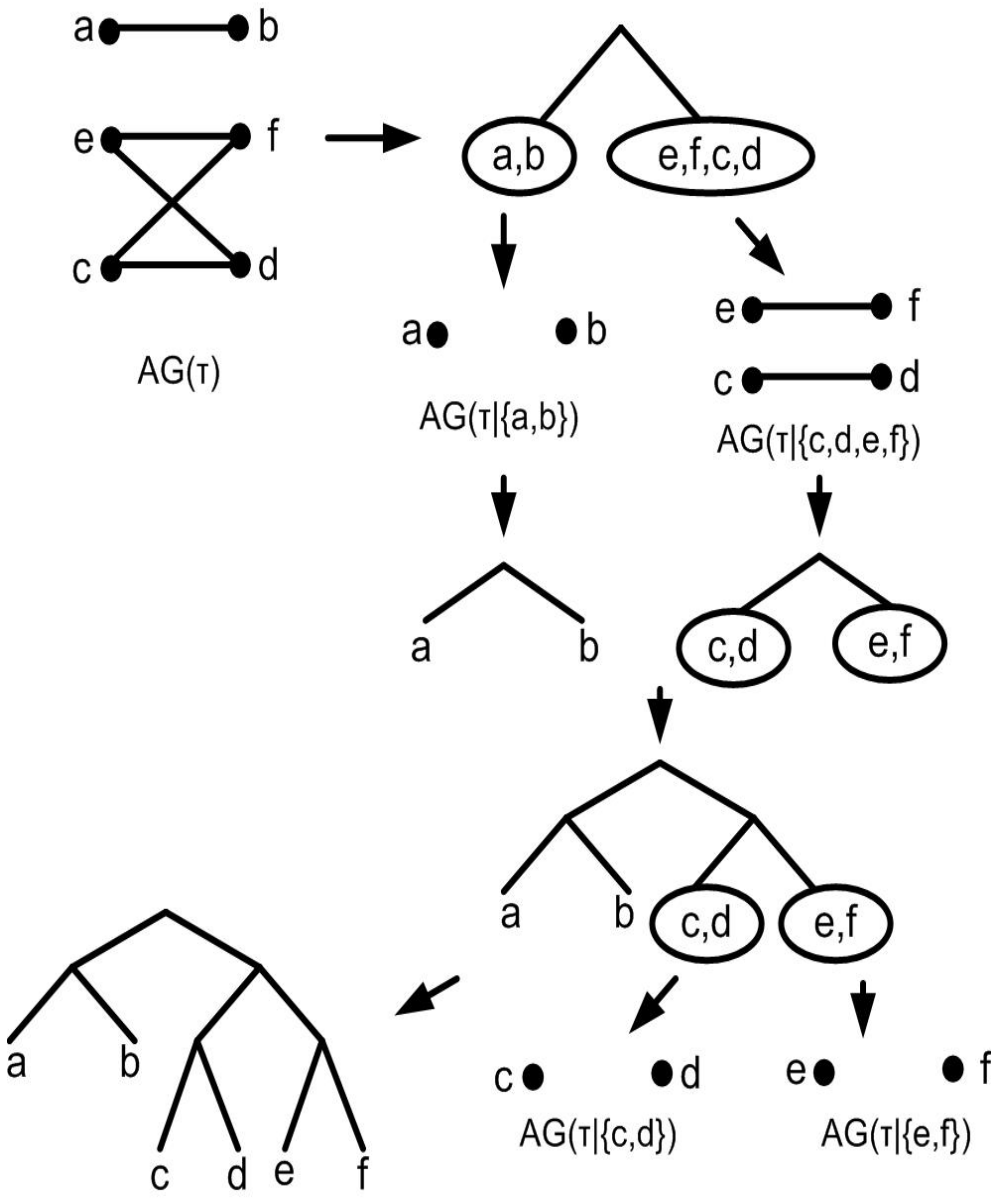
الگوریتم BUILD: فرض کنید X مجموعه‌ای از گونه‌ها و τ مجموعه‌ای از سه‌تایی‌ها روی X باشد. هدف ساخت درختی ریشه‌دار و سازگار با τ همانند T است، اگر چنین درختی وجود داشته باشد. اگر X شامل حداکثر دو رأس باشد، آنگاه الگوریتم BUILD درختی ریشه‌دار و با حداکثر دو برگ را برمی‌گرداند. در غیر این‌صورت الگوریتم BUILD گراف ایهو $AG(\tau)$ را محاسبه می‌کند. اگر $AG(\tau)$ تنها شامل یک مؤلفه همبندی باشد، الگوریتم BUILD متوقف می‌شود و درختی سازگار با τ وجود ندارد. در غیر این‌صورت برای هر یک از مجموعه رأس‌های موجود در یک مؤلفه همبندی مانند U از گراف $AG(\tau)$، مجموعه $\tau|_U$ محاسبه می‌شود.

این مجموعه نشان‌دهنده تمام سه‌تایی‌های τ است که تمام برگ‌های آن در U وجود دارند. سپس به‌صورت بازگشتی زیردرخت ریشه‌دار $T(\tau|_U)$ و سازگار با $\tau|_U$ توسط الگوریتم BUILD ساخته می‌شود. در نهایت یک رأس به نام r به‌عنوان ریشه درخت نهایی در نظر گرفته می‌شود و برای هر زیردرخت، r توسط یک یال به ریشه آن زیردرخت متصل می‌شود. درخت نهایی ساخته‌شده همان درخت T است که با τ سازگار است [۱].

به عنوان مثال برای مجموعه داده‌شده X={a،b،c،d،e،f} و مجموعه داده‌شده از سه‌تایی‌های $\{cd|a, cd|b, ef|a, ef|d, cd|e, cf|a, ab|c, ab|f, ed|b\}$ روی X که با درختی سازگارند، فرایند ساخت درخت توسط الگوریتم BUILD در شکل ۷ بیان شده است.

با استفاده از مفهوم تابع ارتفاع، الگوریتمی شبیه به الگوریتم BUILD به‌نام HBUILD معرفی شد [۲۲].

الگوریتم HBUILD: فرض کنید h یک تابع ارتفاع روی X باشد. گراف کامل وزن‌دار (G,h) که در آن V(G)=X و یال دلخواه $\{i,j\}$ دارای وزن h(i،j) است، تعریف می‌شود. یال‌های با وزن ماکسیمم از G حذف می‌شوند. اگر با حذف یال‌های با وزن ماکسیمم از گراف G، گراف باقیمانده ناهمبند نشد آنگاه الگوریتم متوقف می‌شود. در غیر این‌صورت فرایند حذف یال‌های با وزن ماکسیمم روی هر مؤلفه همبندی نیز انجام می‌شود و این کار تا آنجا ادامه می‌یابد که هر مؤلفه همبندی تنها شامل یک رأس باشد. در پایان این فرایند می‌توان با عکس کردن مراحل الگوریتم، درختی یکتا را بازسازی کرد که در آن برگ‌ها توسط اعضای مجموعه X برچسب‌گذاری شده‌اند. الگوریتم HBUILD در زمان چندجمله‌ای تصمیم‌گیری می‌کند که آیا درختی برای تابع ارتفاع داده‌شده h وجود دارد یا خیر. از این‌رو اگر τ مجموعه‌ای از سه‌تایی‌ها باشد که با درختی سازگار است آنگاه G_τ یک گراف جهت‌دار بدون دور است و الگوریتم HBUILD درختی را بازسازی می‌کند که با τ سازگار است. این درخت همان درخت الگوریتم BUILD است (شکل ۸) [۲۲].

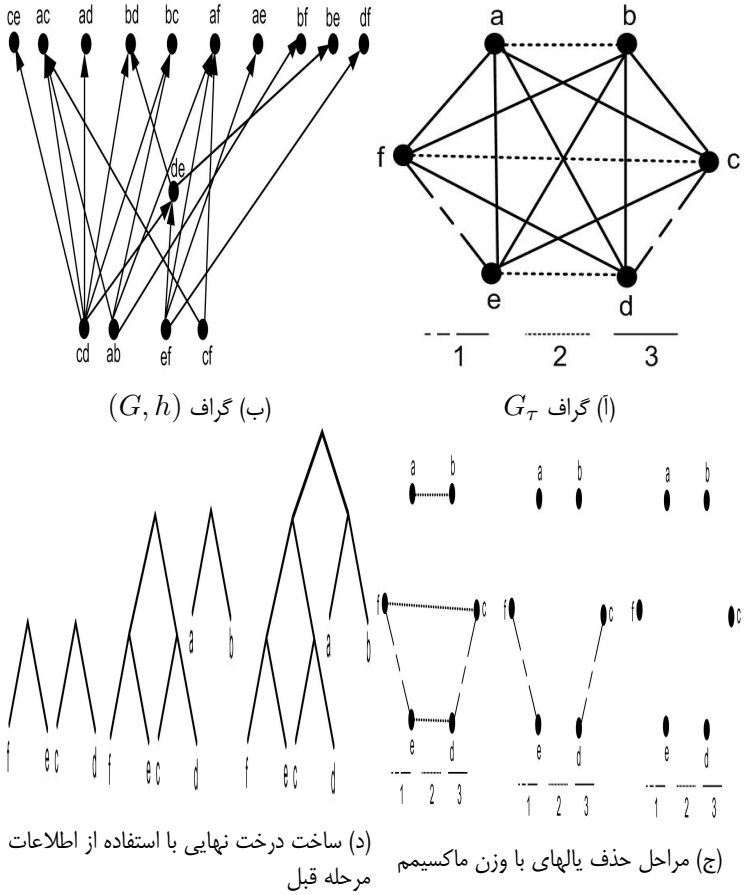


شکل ۷: مثالی از BUILD برای مجموعه $\{cd|a, cd|b, ef|a, ef|d, cd|e, cf|a, ab|c, ab|f, ed|b\}$ از سه‌تایی‌ها.

۴ روش پیشنهادی

همان‌طور که در بخش ۲ بیان شد نقص عمده روش TRH در حل مساله MRTC، به‌کارگیری روش تصادفی و غیر هوشمندانه برای ساخت درخت دودویی نهایی است. از این‌رو معرفی شیوه‌ای نوین و کارامد برای رفع این نقص در الگوریتم TRH ضروری است. بنابراین روش نوینی برای حل مساله MRTC پیشنهاد می‌گردد که نسبت به TRH کارایی بهتری دارد و نقص آن را رفع می‌کند. ورودی الگوریتم نوین مجموعه سه‌تایی‌های τ است. اگر درختی سازگار با τ وجود داشته باشد، آنگاه این الگوریتم، یک دودویی‌سازی از درختی را که توسط الگوریتم BUILD ساخته می‌شود، می‌سازد.

اگر درختی سازگار با سه‌تایی‌های داده‌شده τ وجود نداشته باشد، آنگاه ممکن است که G_τ یک گراف جهت‌دار بدون دور نباشد. در این حالت هدف حذف کم‌ترین یال از G_τ است تا گراف باقیمانده یک گراف جهت‌دار بدون دور شود. مساله حذف کم‌ترین یال از یک گراف جهت‌دار و با هدف تبدیل آن به یک گراف جهت‌دار بدون دور، به اختصار MFAS نامیده می‌شود. مساله MFAS یک مساله بهینه‌سازی است و از نظر پیچیدگی محاسباتی یک مساله NP– سخت است [۶]. روش‌های مختلفی برای حل مساله MFAS وجود دارد. یکی از این روش‌ها، روش GR است. این الگوریتم نسبت به بقیه روش‌های موجود برای حل مساله MFAS زمان اجرای کم‌تری دارد و نیاز ما را مرتفع می‌سازد. از این‌رو از الگوریتم GR استفاده شده و سعی می‌شود تا حد امکان کم‌ترین یال از G_τ حذف شود تا گراف G'_τ که از حذف برخی یال‌های G_τ به‌وجود آمده است، یک گراف جهت‌دار بدون دور شود [۶]. با این کار سعی می‌شود کم‌ترین اطلاعات ممکن که وابسته به



(ب) گراف (G,h) (آ) گراف G_τ

(د) ساخت درخت نهایی با استفاده از اطلاعات مرحله قبل (ج) مراحل حذف یالهای با وزن ماکسیمم

شکل ۸: ساخت درخت با استفاده از الگوریتمHBUILD سازگار با سه‌تایی‌های شکل ۷.

سه‌تایی‌هایند حذف شود و گراف باقیمانده یک گراف جهت‌دار بدون دور شود. برای سادگی گراف G'_τ همان G_τ نام دارد.

حال تمام یال‌های با وزن ماکسیمم از گراف (G_τ, h_{G_τ}) حذف می‌شود. اگر در مرحله‌ای و با حذف یال‌های با وزن ماکسیمم گراف مورد نظر ناهمبند نشود، آنگاه مجموعه سه‌تایی‌های داده شده با درختی سازگار نیست. اگر در مرحله‌ای و با حذف تمام یال‌های با وزن ماکسیمم از یک مؤلفه همبندی همانند C، گراف باقیمانده که C' نامیده می‌شود دوباره همبند بود، از روش زیر برای ناهمبند کردن C' استفاده می‌شود.

در ابتدا سه شیوه برای ناهمبند کردن C' در نظر گرفته می‌شود. ۱- یال‌های با وزن ماکسیمم از C' حذف می‌شوند و این کار تا آنجا ادامه می‌یابد که گراف باقیمانده ناهمبند شود. ۲- یال‌ها با شیوه‌ای حذف می‌شوند که گراف باقیمانده به دو مولفه همبندی تبدیل شود و مجموع وزن یال‌های حذف شده کم‌ترین مقدار ممکن باشد. این شیوه، الگوریتم Min-Cut نام دارد و خروجی آن یک گراف دوبخشی است و حذف یال‌ها از گراف همبند برای تبدیل به گراف دوهمبند به‌گونه‌ای است که مجموع وزن یال‌های حذف‌شده کم‌ترین باشد [۲۶]. ۳- فرض کنید که در C' وزن یال با وزن ماکسیمم m باشد. برای هر یال دلخواه در C' با وزن $w \leq m$، وزن آن تبدیل به $m-w+1$ می‌شود. سپس حذف یال‌ها با کم‌ترین مجموع وزن با هدف تبدیل شدن گراف باقیمانده به دو مولفه همبندی انجام می‌شود. پس از انجام این سه شیوه، به‌صورت موضعی سازگاری درخت ساخته‌شده تا این مرحله با سه‌تایی‌های موجود را بررسی می‌کنیم. از بین سه شیوه، روشی را انتخاب می‌کنیم که باعث سازگاری تعداد بیش‌تری سه‌تایی با درخت می‌شود. این شیوه را تا آنجا ادامه می‌دهیم تا هر مولفه همبندی تنها شامل یک راس شود. در ادامه با عکس کردن مراحل الگوریتم پیشنهادی، یک درخت به‌دست می‌آید که لزوما دودویی نیست.

در ادامه از شیوه‌ای منطقی و خلاقانه برای دودویی‌سازی درخت ساخته‌شده استفاده می‌شود. بدین منظور با استفاده از سه پارمتر p, w و t که در ادامه تعاریف آنها ارائه می‌شود، از یک تابع امتیازدهی استفاده می‌کنیم. این تابع امتیازدهی یک ابزار موثر برای دودویی‌سازی هوشمند درخت غیردودویی ساخته‌شده است. برای مجموعه τ داده‌شده از سه‌تایی‌ها فرض کنید V_i و V_j دو زیرمجموعه دلخواه از $L(\tau)$ باشند به‌طوری‌که $V_i \cap V_j = \emptyset$. سه مجموعه T, P, W به‌صورت زیر تعریف می‌شود. فرض کنید:

$$W(V_i,V_j) = \{v_iv_j | v \in \tau \ : \ v_i \in V_i, v_j \in V_j \ and \ v \notin V_i \cup V_j\}$$
$$P(V_i,V_j) = \{v_iv | v_j \in \tau \ or \ v_jv \ : \ v_i \in \tau | v_i \in V_i, v_j \in V_j \ and \ v \notin V_i \cup V_j\}$$
$$T(V_i,V_j) = \{v_iv_j | v \in \tau \ : \ v_i \in V_i, v_j \in V_j\}.$$

با توجه به سه مجموعه بالا، سه پارامتر t, p, w تعریف می‌شود که $w(V_i,V_j)$، $p(V_i,V_j)$ و $t(V_i,V_j)$ به‌ترتیب تعداد اعضای سه مجموعه $W(V_i,V_j)$، $P(V_i,V_j)$، و $T(V_i,V_j)$ باشند. با توجه به این سه پارامتر در این پژوهش چهار تابع امتیازدهی

$f_۱, f_۲, f_۳, f_۴$ برای ورودی‌های V_i و V_j (که دو زیرمجموعه دلخواه از $L(\tau)$ هستند) به‌صورت زیر تعریف می‌شود [۲۹].

$$f_۱ = w(V_i, V_j)$$

$$f_۲ = (w-p)(V_i, V_j) = w(V_i, V_j) - p(V_i, V_j)$$

$$f_۳ = \frac{w-p}{t}(V_i, V_j) = \frac{w(V_i,V_j)-p(V_i,V_j)}{t(V_i,V_j)}$$

$$f_۴ = \frac{w}{w+p}(V_i, V_j) = \frac{w(V_i,V_j)}{w(V_i,V_j)+p(V_i,V_j)}$$

فرض کنید درخت به‌دست‌آمده قبل از دودویی‌سازی تصادفی، درخت $T_۱$ باشد. فرض کنید در این درخت، راسی همانند x دارای $k > ۲$ فرزند همانند $x_۱, x_۲, \ldots, x_k$ باشد و برای هر $۱ \le i \le k$، X_i مجموعه برگ‌هایی باشند که x_i جد آنها است. واضح است که برای $۱ \le i, j \le k, i \ne j$، داریم $X_i \cap X_j = \emptyset$. برای هر زیرمجموعه دوعضوی از مجموعه $\{X_۱, X_۲, \ldots, X_k\}$ تابع امتیازدهی محاسبه می‌شود و دو زیرمجموعه با بالاترین مقدار به‌دست‌آمده وابسته به تابع امتیازدهی، ادغام می‌شوند.

این کار برای k مجموعه تا آنجا ادامه می‌یابد که زیردرخت وابسته به راس x یک درخت دودویی شود. این کار در هر مرحله و برای هر راس که تعداد فرزندان آن بیش‌تر از ۲ است انجام می‌شود تا در نهایت یک درخت دودویی همانند T به‌دست آید.

در الگوریتم نوین برای درخت به‌دست‌آمده غیردودویی که روش ساخت آن توضیح داده‌شد و با استفاده از تابع امتیازدهی، ۴ دودویی‌سازی به‌دست می‌آید. یعنی در نهایت ۴ درخت دودویی داریم. بهترین درخت از بین ۴ درخت ممکن به عنوان خروجی الگوریتم ما گزارش می‌شود.

توجه کنیم که اگر درخت ساخته‌شده توسط TRH یک درخت دودویی باشد، نتایج روش ما و روش TRH یکی است. حتی در حالتی‌که درخت ساخته‌شده توسط TRH یک درخت دودویی نباشد نیز ممکن است پس از فرایند تصادفی دودویی‌سازی و یا فرایند هوشمندانه معرفی‌شده، جواب‌های یکسانی به‌دست آید.

همچنین ممکن است که دودویی‌سازی تصادفی صورت‌گرفته در الگوریتم TRH نسبت به دودویی‌سازی هوشمندانه معرفی‌شده توسط ما در برخی موارد نتایج بهتری را ارئه دهد. هرچند این پیش‌فرض محتمل است، اما با توجه به روش هوشمند دودویی‌سازی و ساخت درخت نهایی، انتظار داریم که تقریبا در تمامی موارد نتایج ما به‌خوبی نتایج TRH باشد و در موارد قابل قبولی نتایج ما بهتر از نتایج TRH باشد.

اما نکته مهم در روش نوین معرفی‌شده این است که در بسیاری از موارد بهترین درخت گزارش‌شده توسط الگوریتم TRH یک درخت دودویی نیست و فرایند دودویی‌سازی تصادفی نمی‌تواند یک شیوه کارامد برای دودویی‌سازی و بهبود نتایج باشد. همین امر اهمیت روش ما رو دوچندان می‌کند.

روش استفاده از تابع نوین امتیازدهی بدین صورت است که در شروع و برای مجموعه τ داده‌شده از سه‌تایی‌ها همراه با $|L(\tau)| = n$، زیرمجموعه تک عضوی از گونه‌ها داریم. در هر مرحله، دو زیرمجموعه ادغام می‌شوند و با این کار در هر مرحله یکی از تعداد زیرمجموعه‌ها کم می‌شود. این کار ادامه پیدا می‌کند تا در نهایت تنها یک زیرمجموعه n عضوی باقی بماند. در واقع هرکدام از این زیرمجموعه‌ها یک زیردرخت از درخت نهایی است. به‌عبارت دیگر در ابتدا n درخت تک عضوی داریم که این درخت‌های تک عضوی همان برگ‌هایند. در هر مرحله، ریشه دو درخت انتخاب‌شده را به یک راس جدید که ریشه درخت جدید است متصل می‌کنیم. سوال مهم این است که در هر مرحله کدام دو زیرمجموعه یا به‌عبارت معادل کدام دو زیردرخت باید انتخاب شده و ادغام شوند. بدین منظور از تابع امتیازدهی استفاده می‌شود که نشان‌دهنده امتیاز و ارزش ادغام شدن دو زیرمجموعه همانند $V_۱$ و $V_۲$ است. با این کار در هر مرحله دو زیرمجموعه‌ای انتخاب می‌شوند که بیش‌ترین امتیاز ادغام را داشته باشند. به‌صورت منطقی، دو زیرمجموعه‌ای همانند $V_۱$ و $V_۲$ ادغام می‌شوند که مقدار $w(V_۱, V_۲)$ بیش‌تری داشته باشند. یعنی در هر مرحله به‌صورت حریصانه دو زیرمجموعه انتخاب می‌شوند با این شرط که ادغامشان تعداد سه‌تایی‌های بیش‌تری را ارضا کند. ادغام دو زیرمجموعه نه تنها باعث ارضا شدن و سازگاری برخی از سه‌تایی‌ها می‌شود بلکه باعث ناسازگاری برخی از سه‌تایی‌ها با درخت ساخته‌شده هم می‌شود. به‌بیان دقیق اگر برای دو مجموعه همانند $V_۱$ و $V_۲$ داشته باشیم $v_۱ \in V_۱$ و $v_۲ \in V_۲$ و x عضو $V_۱ \cup V_۲$ نباشد آنگاه باعث سازگاری سه‌تایی‌هایی به‌شکل $v_۱v_۲|x$ و ناسازگاری سه‌تایی‌هایی به‌شکل $v_۱x|v_۲$ و $v_۲x|v_۱$ خواهد بود. از این‌رو $p(V_۱, V_۲)$ تعداد سه‌تایی‌های ناسازگار با ادغام دو مجموعه $V_۱$ و $V_۲$ است.

با توجه به‌این‌که توزیع سه‌تایی‌ها ممکن است یکنواخت نباشد، انتخاب دو زیرمجموعه ممکن است به‌صورت نادرستی تحت تاثیر سه‌تایی‌های ارضا شده قرار گیرد. بنابراین بهتر است از پارامتر نسبی $t(V_۱, V_۲)$ نیز برای انتخاب دو زیرمجموعه استفاده شود. در نتیجه همان‌طور که در بالا بیان شد، ۴ حالت برای تابع امتیازدهی داریم و از آنها برای دودویی‌سازی هوشمند درخت غیردودویی استفاده می‌کنیم.

فرض کنید برای τ داده شده، $|L(\tau)| = n$ و $|\tau| = m$ است. گراف G_τ در زمان $O(m)$ ساخته می‌شود. اگر G_τ یک گراف جهت‌دار بدون‌دور نباشد الگوریتم GR در زمان $O(|edges|)$ که معادل $O(m)$ است اجرا می‌شود. برای ساخت گراف (G, h) نیاز به زمان $O(n^۲)$ داریم. بنابراین تا این مرحله زمان اجرا $O(m + n^۲)$ است. برای سه شیوه ناهمبندکردن گراف و در شیوه اول زمان $O(mn^۲)$ و برای شیوه دوم و سوم که از نظر زمان اجرا یکی‌اند، $O(mn^۲ + n^۳logn)$ زمان نیاز است. فرایند دودویی‌سازی هوشمند در زمان $O(mn^۲)$ اجرا می‌شود. از این‌رو زمان اجرای الگوریتم نوین، همانند TRH است و از مرتبه $O(mn^۲ + n^۳logn)$ است [۲۱].

۵ نتایج آزمایش‌ها

در حالت کلی، ورودی مساله MRTC سه‌تایی‌هایی است که از روی توالی‌های زیستی به‌دست آمده‌اند. بنابراین در گام اول سه‌تایی‌ها را همانند [۲۱، ۲۲] از روی اطلاعات توالی‌های زیستی به‌دست می‌آوریم. الگوریتم TRH به‌طور میانگین بهترین روش موجود برای حل مساله MRTC و بر روی داده‌های زیستی است. از این‌رو برای بیان کارایی روش ابتکاری معرفی‌شده، فرایند زیر را انجام می‌دهیم. از روش استاندارد ML برای تولید سه‌تایی از روی توالی‌های زیستی استفاده می‌کنیم [۲۸]. در ابتدا با استفاده از نرم‌افزار TREEVOLVE به تعداد ۲۰۰۰ مجموعه از توالی‌ها تولید می‌شود که هر یک شامل ۱۵ توالی است. TREEVOLVE یک نرم‌افزار استاندارد برای تولید توالی‌های زیستی است. این نرم‌افزار پارامترهای مختلفی دارد که در این پژوهش طول توالی‌ها ۴۰۰ انتخاب شده است. در واقع این نرم‌افزار با استفاده از پیش‌فرض‌های زیستی اقدام به تولید شبه تصادفی توالی‌های زیستی می‌کند. برای هر مجموعه از توالی‌ها با استفاده از روش ML مجموعه‌ای از سه‌تایی‌ها به‌دست می‌آید [۲۸]. سپس برای آستانه‌های مختلف [۲۲] نتایج روش ابتکاری خود را با نتایج TRH مقایسه می‌کنیم. نتایج در جدول ۱ نمایش داده شده است.

آستانه	۰	۲.۰	۴.۰	۶.۰	۸.۰	۹.۰
درخت‌هایی که TRH بهتر است	۸.۰ ٪	۷.۰ ٪	۷.۰ ٪	۷.۰ ٪	۶.۰ ٪	۵.۰ ٪
درخت‌هایی که روش نوین بهتر است	۲۳ ٪	۲۲ ٪	۲۰ ٪	۲۲ ٪	۲۰ ٪	۱۶ ٪
میانگین درصد سازگاری سه‌تایی‌ها برای TRH	۸۰ ٪	۸۲ ٪	۸۴ ٪	۸۶ ٪	۸۹ ٪	۹۰ ٪
میانگین درصد سازگاری سه‌تایی‌ها برای روش نوین	۸۲ ٪	۸۴ ٪	۸۵ ٪	۸۸ ٪	۵.۸۹ ٪	۵.۹۰ ٪

جدول ۱: مقایسه نتایج روش ابتکاری نوین و TRH برای داده‌های تولیدشده توسط روش استاندارد ML و برای آستانه‌های مختلف

همچنین بر روی داده‌های تصادفی نیز روش خود را با روش TRH مقایسه می‌کنیم. بدین منظور از دو پارامتر m و n استفاده می‌کنیم. فرض کنید n تعداد گونه‌ها و m تعداد سه‌تایی‌ها باشد. برای $n = ۱۵$ و ۳۰۰و ۲۰۰و ۱۰۰و $m = ۵۰$ به‌صورت تصادفی و برای هر یک از چهار حالت به تعداد ۲۰۰۰ نمونه تصادفی تولید می‌کنیم. به‌عنوان مثال، برای $n = ۱۵$ و $m = ۵۰$ هدف، تولید ۵۰ سه‌تایی از روی ۱۵ گونه است. بدین منظور تمام سه‌تایی‌های ممکن برای ۱۵ گونه را به فرم $ab|c$ تولید می‌کنیم. سپس با استفاده از توزیع یکنواخت، ۵۰ سه‌تایی از مجموعه کل، انتخاب می‌شود. نتایج در جدول ۲ بیان شده است.

تعداد سه‌تایی‌ها	۵۰	۱۰۰	۲۰۰	۳۰۰
درخت‌هایی که TRH بهتر است	۷.۰ ٪	۶.۰ ٪	۴.۰ ٪	۶.۰ ٪
درخت‌هایی که روش نوین بهتر است	۲۴ ٪	۲۱ ٪	۲۲ ٪	۱۶ ٪
میانگین درصد سازگاری سه‌تایی‌ها برای TRH	۶۰٪	۵۲ ٪	۴۴ ٪	۴۰٪
میانگین درصد سازگاری سه‌تایی‌ها برای روش نوین	۶۳ ٪	۵۴ ٪	۴۸ ٪	۴۲ ٪

جدول ۲: نتایج شبیه‌سازی و برای $n = ۱۵$ و ۳۰۰و ۲۰۰و ۱۰۰و $m = ۵۰$ و بر روی TRH و روش نوین

نتایج نشان می‌دهد که به‌طور میانگین روش نوین معرفی‌شده در تمامی حالت‌ها بهتر از TRH است.

۶ نتیجه‌گیری

در این مقاله روشی نوین برای حل مساله MRTC معرفی شد. هدف از معرفی الگوریتم نوین ابتکاری، رفع نقص عمده روش TRH در ساخت درخت دودویی نهایی است. بدین منظور با استفاده از ایده تابع امتیازدهی از یک روش هوشمندانه برای دودویی‌سازی درخت نهایی استفاده کردیم. سپس نتایج خود را بر روی سه‌تایی‌های به‌دست‌آمده از روی توالی‌های زیستی و سه‌تایی‌های به‌صورت تصادفی تولیدشده، با روش TRH مقایسه کردیم. نتایج آزمایش‌ها نشان می‌دهد که روش ما به‌خوبی قادر است نتایج TRH را بهبود ببخشد. به‌طور دقیق‌تر نتایج دو الگوریتم برای سه‌تایی‌های به‌دست‌آمده از روی توالی‌های زیستی نشان می‌دهد که برای آستانه‌های مختلف و در بهترین حالت، حداکثر در ۰٫۸٪ موارد نتایج TRH بهتر از روش ما است. اما برای آستانه‌های مختلف در حداقل ۱۶٪ و حداکثر ۲۳٪ موارد نتایج ما بهتر از نتایج TRH است. همچنین در تمامی موارد درصد سازگاری سه‌تایی‌ها با درخت نهایی و برای روش ما بهتر از روش TRH است. از آنجا که تا پیش از این و به‌طور میانگین بر روی سه‌تایی‌هایی که از روی توالی‌های زیستی و با استفاده از روش استاندارد ML تولید شده‌اند، روش TRH

بهترین روش موجود بود و از طرفی چون زمان اجرای روش ما و TRH یکی است، می‌توان نتیجه گرفت که به‌طور متوسط الگوریتم ما بهترین روش موجود برای حل مساله MRTC و بر روی این نوع از داده‌ها است. همچنین بر روی سه‌تایی‌های تصادفی تولید شده نیز روش خود را با روش TRH مقایسه کردیم. بدین منظور برای $n = ۱۵$ و ۳۰۰و ۲۰۰و ۱۰۰و $m = ۵۰$ که n تعداد گونه‌ها و m تعداد سه‌تایی‌ها است، دو روش را مقایسه کردیم. نتایج نشان می‌دهد که در بهترین حالت، حداکثر در ۰/۷٪ موارد نتایج TRH بهتر از روش ما است. در صورتی‌که در بدترین حالت حداقل در ۱۶٪موارد نتایج ما بهتر از نتایج TRH است. همچنین در تمامی موارد درصد سازگاری سه‌تایی‌ها با درخت نهایی و برای روش ما بهتر از روش TRH است.

توجه کنیم که هرچه تعداد سه‌تایی‌ها نسبت به تعداد گونه‌ها کم‌تر باشد، احتمال دودیی شدن سه درخت به‌دست‌آمده توسط روش TRH هم پایین می‌آید. این بدان معنی است که استفاده هوشمندانه ما از یک ابزار مناسب برای دودیی‌سازی این سه درخت، یک راهکار بسیار مناسب برای بهبود نتایج الگوریتم TRH و به‌خصوص برای این نوع از داده‌ها است. توجه کنیم در حالتی‌که تعداد سه‌تایی ها خیلی کم باشند و اطلاعات بسیار ناقص باشد، ممکن است که الگوریتم پیشنهادی در موارد نادری دارای دقت کم‌تری نسبت به حالت تصادفی باشد که عدد ۰/۷٪ در بالا نتیجه این نوع داده‌ها است.

فهرست منابع

[۱] com- lowest from tree a Inferring (۱۹۸۱). J.D. Ullman، and T.G. Szymanski، Y. Sagiv، A.V. Aho، Jour- SIAM expressions، relational of optimization the to application an with ancestors mon ۴۰۵–۴۲۱. ۱۰(۳)، Computing، on nal

[2] Bouckaert, R. Lemey, P. Dunn, M. Greenhill, S.J. Alekseyenko, A.V. Drummond, A.J. Gray, R.D. Suchard, M.A. and Atkinson, Q.D. (2012). Mapping the origins and expansion of the Indo-European language family. Science, **337(6097)**, 957-960.

[3] Bryant, D. (1997). Building trees, hunting for trees, and comparing trees: theory and methods in phylogenetic analysis, University of Canterbury. Mathematics and Statistics.

[4] Byrka, J. Gawrychowski, P. Huber, K. T. and Kelk, S. (2010). Worst-case optimal approximation algorithms for maximizing triplet consistency within phylogenetic networks. Journal of Discrete Algorithms, **8(1)**, 65-75.

[5] Chor, B. Hendy, M. and Penny, D. (2007). Analytic solutions for three taxon ML trees with variable rates across sites, Discrete Applied Mathematics, **155(6-7)**, 750-758.

[6] Eades, P., Lin, X., and Smyth, W. F. (1993). A fast and effective heuristic for the feedback arc set problem, Information Processing Letters, **47(6)**, 319-323.

[7] Eslahchi, C. Hassanzadeh, R. Mottaghi, E. Habibi, M. Pezeshk, H. and Sadeghi, M. (2012). Constructing circular phylogenetic networks from weighted quartets using simulated annealing, Mathematical biosciences, **235(2)**, 123-127.

[8] Felsenstein, J. (2003). Inferring Phylogenies. Sunderland, Sinauer Ass. Inc. Publishers.

[9] Gasieniec, L. Jansson, J. Lingas, A. and ?stlin, A. (1999). On the complexity of constructing evolutionary trees. Journal of Combinatorial Optimization, **3(2-3)**, 183-197.

[10] Grassly, N. and Rambaut, A. (1997).Treevole: a program to simulate the evolution of DNA sequences under different population dynamic scenarios, Wellcome Centre for Infectious Disease, Department of Zoology.

[11] Hein J. (1990). Reconstructing evolution of sequences subject to recombination using parsimony, Mathematical biosciences, **98(2)**, 185-200.

[12] Henzinger, M.R. King, V. and Warnow, T. (1999). Constructing a tree from homeomorphic subtrees, with applications to computational evolutionary biology, Algorithmica, **24(1)**, 1-13.

[13] Huson, D.H. and Rupp, R. and Scornavacca, C. (2010). Phylogenetic networks: concepts, algorithms and applications, Cambridge University Press.

[14] Jahangiri, S. Hashemi, S. N. and Poormohammadi, H. (2013). New heuristics for rooted triplet consistency, Algorithms, **6(3)**, 396-406.

[15] Jansson, J. (2001). On the complexity of inferring rooted evolutionary trees. Electronic Notes in Discrete Mathematics, **7**, 1-4.

[16] Jansson, J. Ng, J.H.K. Sadakane, K. and Sung, W.K. (2005). Rooted maximum agreement supertrees, Algorithmica, **43(4)**, 293-307.

[17] Jansson, J. Nguyen, N.B. and Sung, W.K. (2006). Algorithms for combining rooted triplets into a galled phylogenetic network, SIAM Journal on Computing, **35(5)**, 1098-1121.

[18] Jansson, J. and Sung, W.K. (2006). Inferring a level-1 phylogenetic network from a dense set of rooted triplets, Theoretical Computer Science, **363(1)**, 60-68.

[19] Kannan, S.K. Lawler, E.L. and Warnow, T.J. (1996). Determining the evolutionary tree using experiments, Journal of Algorithms, **21(1)**, 26-50.

[20] Maemura, K. Jansson, J. Ono, H. Sadakane, K. and Yamashita, M. (2007). Approximation algorithms for constructing evolutionary trees from rooted triplets. In Proceedings of 10th Korea-Japan Joint Workshop on Algorithms and Computation, 56-63.

[21] Poormohammadi H. (2012). A New Heuristic Algorithm for MRTC Problem, Journal of Emerging Trends in Computing and Information Sciences, **3**, 1037-1041.

[22] Poormohammadi, H. Eslahchi, C., and Tusserkani, R. (2014). Tripnet: a method for constructing rooted phylogenetic networks from rooted triplets, PLoS One, **9(9)**, e106531.

[23] Poormohammadi, H. and Sardari Zarchi, M. (2020). NetCombin: An algorithm for optimal level-k network construction from triplets, Plos One, **15(9)**, e0227842.

[24] Poormohammadi, H. Sardari Zarchi, M. and Ghaneai, H. (2020). NCHB: A method for constructing rooted phylogenetic networks from rooted triplets based on height function and binarization, Journal of Theoretical Biology, **489**, 110-144.

[25] Reyhani, M.H. and Poormohammadi, H. (2017). RPNCH: A method for constructing rooted phylogenetic networks from rooted triplets based on height function. Archives of Advances in Biosciences, **8(4)**, 14-20.

[26] Stoer, M., and Wagner, F. (1997). A simple min-cut algorithm, Journal of the ACM (JACM), **44(4)**, 585-591.

[27] Van Iersel, L. Keijsper, J. Kelk, S. Stougie, L. Hagen, F. and Boekhout, T. (2008). Constructing level-2 phylogenetic networks from triplets. In Annual International Conference on Research in Computational Molecular Biology (pp. 450-462). Springer, Berlin, Heidelberg.

[28] Van Iersel, L. and Kelk, S. (2011). Constructing the simplest possible phylogenetic network from triplets, Algorithmica, **60(2)**, 207-235.

[29] Wu, B. Y. (2004). Constructing the maximum consensus tree from rooted triples. Journal of Combinatorial Optimization, **8(1)**, 29-39.

A novel Algorithm for constructing rooted phylogenetic trees based on rooted triplets

Hadi Poormohammadi[1] [†] **, Mohsen Sardari Zarchi**[1]**, Mohammad Ali Sehati**[1]**, Mohammad Mirabi**[2]**, Seyed Hasan Mortazavi Zarch**[1]

(1) Assistant Professor, Department of Computer Engineering, Meybod University, Meybod, Iran
(2) Assistant Professor, Department of Industrial Engineering, Meybod University, Meybod, Iran

Received: 2020/9/21 Accepted: 2020/12/24

Communicated by: Mohammad Reza Darafsheh

Abstract: Phylogenetics is a field in bioinformatics that categorizes and structures the evolutionary hisotiry of currently living species. Phylogenetics uses network tools to represent evolutionary histories in a comprehensive model. The simpleset possible network model is a tree model that is called a phylogenetic tree. Phylogenetic trees are divided into two categories, which are rooted and un-rooted. In this research, rooted phylogenetic trees are of interest. There are different types of input for constructing rooted phylogenetic trees. Rooted triplets are one of the important inputs for constructing rooted phylogenetic trees, which are used in this study. A rooted triplet is a rooted binary tree with three leaves. The rooted triplet is the most simplest model, which contains valuable information. The problem of building a rooted phylogenetic tree that includes the maximum number of a set of rooted triplets is an optimizataion problem, known as MRTC problem. The important challenge in phylogenetics is that MRTC is NP-hard. Therefore, the focus of this research is to introduce a novel algorithm for MRTC problem. The aim of the new algorithm is to improve the consistency of input rooted triplets with the final rooted phylogenetic tree. In order to indicate the performance of the new algorithm, it is compared with on of the best methods on biological data which is called TRH. The Experimental results on triplets that are obtained from biological data show that our new algorithm outperforms TRH base on time complexity and consistency.

Keywords: Bioinformatics, Biological sequence, MRTC problem, Rooted phylogenetic tree, Rooted triplet.

[†]Corresponding author.
E-mail addresses: pormohamadi@meybod.ac.ir, (Sardari Zarchi), sardari@meybod.ac.ir (Sehati), msehati@meybod.ac.ir, (Mirabi), mirabi@meybod.ac.ir, (Mortazavi Zarch), hassanmortazavi@meybod.ac.ir.