دانشکده مهندسی کامپیوتر
دانشگاه صنعتی شریف

# ژنومیک محاسباتی

**مطهری و فروغمند**
پاییز ۱۴۰۰

# درخت تبارزایی (۴)

جلسه هفتم: کار با چند درخت

**نگارنده: سهیل رستگار**

## ۱ مروری بر مباحث گذشته

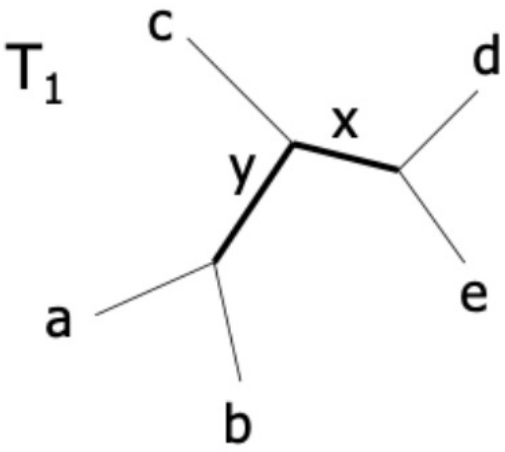
در جلسات گذشته‌ی حول این موضوع به درخت‌های تبارزایی[1] پرداختیم و روش‌های تولید آن‌ها را از روی داده‌های خام، یا ماتریس‌های فاصله بررسی کردیم. دیدیم که اگر ساختار درخت را داشته باشیم می‌توان با پیچیدگی محاسباتی کم(خطی) و با استفاده از برنامه‌نویسی پویا مقادیر میانی درخت تبارزایی را از روی گونه‌های موجود در حال حاضر تولید کرد. همینطور دیدیم که عموما ساختار درخت را نداریم و تولید یک درخت تبارزایی در این حالت بسیار پیچیده و در واقع یک الگوریتم np است.

در جلسه‌های بعد چند مدل احتمالاتی را برای ساخت درخت تبارزایی بررسی کردیم و در نهایت یک روش خوب که ساختن درخت بر اساس ماتریس فواصل بین گونه‌های مختلف بود را معرفی کرده و چند الگوریتم برای حالات مختلف ماتریس فاصله ارائه دادیم. در این جلسه قصد داریم که روش‌هایی را برای مقایسه ی درخت‌های مختلف ارائه دهیم تا بتوانیم مکانیزمی برای مقایسه‌ی و ارزیابی درخت‌های تولید شده و درخت‌های موجود داشته باشیم. همینطور در ادامه روش‌هایی را برای ساختن درخت‌های توافقی و اجماعی از روی چند درخت ارائه می‌دهیم.

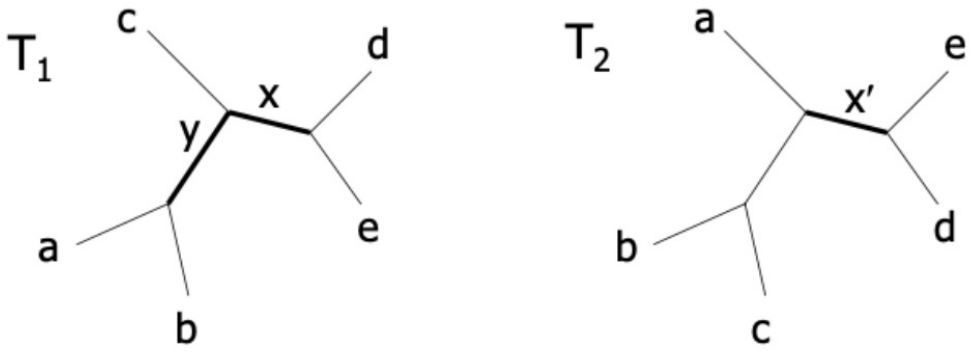
## ۲ مقایسه‌ی درخت‌ها

در این بخش قصد داریم روشی را برای محاسبه‌ی فاصله‌ی دو درخت ارائه دهیم تا بتوان میان درخت‌های مختلف مقایسه انجام داد و شباهت و تفاوت آن‌ها را ارزیابی کرد.

[1] phylogenetic tree



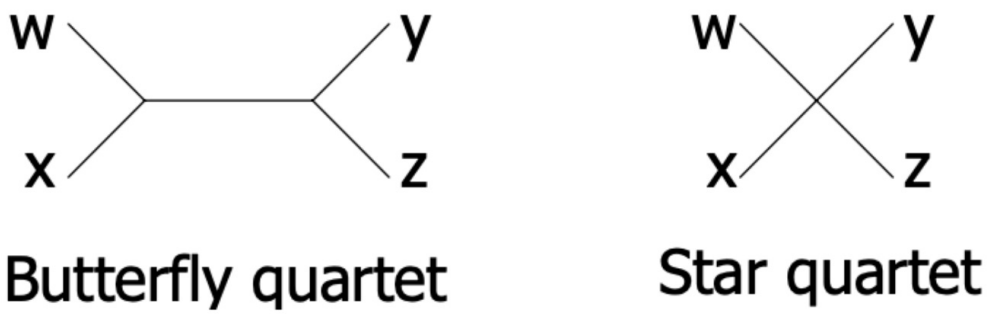

شکل ۱: یک نمونه درخت




شکل ۲: دو درخت دلخواه

## ۱.۲ تقسیم (Split)

می‌دانیم که درخت گراف بدون دور است. پس اگر یک یال آن را قطع کنیم به دو بخش مجزا می‌رسیم که یالی بین هم‌دیگر ندارند. از همین قضیه برای تعریف تقسیم روی درخت استفاده می‌کنیم. دو مجموعه‌ی مجزا از برگ‌های درخت را یک تقسیم روی یال x می‌نامیم، اگر با حذف یال x از درخت اصلی، برگ‌های درخت به دو مجموعه‌ی مجزای داده شده افراز شوند. برای مثال در شکل ۱ در صورتی که تقسیم روی یال x انجام گیرد، به دو مجموعه مجزای $\{abc \mid de\}$ می‌رسیم و اگر تقسیم روی یال y انجام شود به دو بخش $\{ab \mid cde\}$.




شکل ۳: حالت‌های بخش‌بندی

## ۲.۲ فاصله‌ی Robinson-Foulds

برای محاسبه‌ی این معیار فاصله بین دو درخت، ابتدا تمام تقسیم‌های درخت اول و درخت دوم را به صورت جدا حساب می‌کنیم، در واقع به دو مجموعه از تقسیم‌های مختلف می‌رسیم. سپس عناصر غیرمشترک بین این دو مجموعه، یا به عبارت دیگر اجتماع دو مجموعه منهای اشتراک دو مجموعه را حساب کرده و تقسیم بر ۲ می‌کنیم. عدد بدست آمده، فاصله‌ی Robinson-Foulds بین دو درخت است.

برای مثال جهت محاسبه‌ی فاصله‌ی بین دو درخت در شکل ۲ داریم:

$$Splits(T۱) = \{a|bcde, b|acde, c|abde, d|abce, e|abcd, ab|cde, abc|de\}$$

$$Splits(T۲) = \{a|bcde, b|acde, c|abde, d|abce, e|abcd, bc|ade, abc|de\}$$

$$D = Splits(T۱) \bigcup Splits(T۲) - Splits(T۱) \cap Splits(T۲) = \{ab|cde, bc|ade\}$$

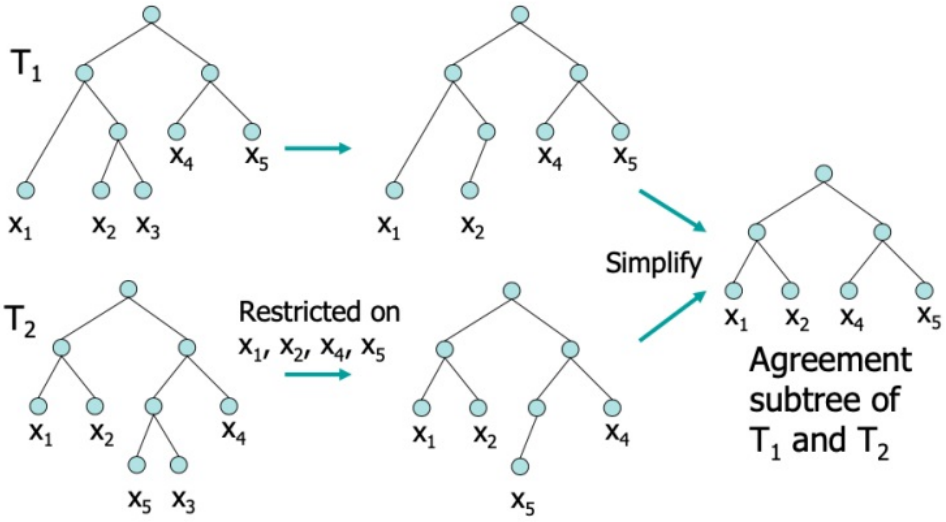
$$\text{Robinson-Foulds}(T۱, T۲) = \frac{|D|}{۲} = ۱$$

از تعریف بعضی از خواص فاصله‌ی Robinson-Foulds مشخص می‌شود. برای مثال فاصله‌ی هر درخت با خودش ۰ است و فاصله‌ی درخت a با درخت b برابر فاصله‌ی درخت b تا درخت a است.

## ۳.۲ فاصله‌ی چهارتایی (Quartet Distance)

فرض می‌کنیم ۴ برگ از یک درخت را به نام‌های w، x، y، z داشته باشیم. این ۴ برگ بر اساس بخش‌بندی ۴ حالت دارند، اینکه x با w جد مشترک نزدیک‌تری داشته باشند و y با z، اینکه x با y و w با z و اینکه x با z و w با y، سه حالت را می‌سازند که در این سه حالت یعنی یالی وجود دارد که در صورت قطع شدن آن، برگ‌های که جد مشترک نزدیکتر دارند در یک درخت و دو برگ دیگر که باز جد مشترک نزدیکتر دارند در درخت دیگر قرار می‌گیرند. به این ۳ حالت، Butterfly quartet گفته می‌شود. یک حالت نیز وجود دارد که جد مشترک هر ۴ برگ یکسان باشد که به آن Star quartet گفته می‌شود (شکل ۳). پس در کل ۴ حالت بخش‌بندی وجود دارد.

برای محاسبه‌ی فاصله‌ی چهارتایی، تمام ۴ تایی‌های ممکن را در نظر می‌گیریم و بخش‌بندی آن‌ها را حساب می‌کنیم. اگه بخش‌بندی این ۴ تایی در دو درخت یکسان نبود، آن حالت را به عنوان یک واحد فاصله‌ساز در نظر می‌گیریم. در نهایت تعداد ۴ تایی هایی که در دو درخت بخش‌بندی متفاوتی دارند را حساب کرده و به عنوان فاصله‌ی دو درخت در نظر می‌گیریم.




شکل ۴: مراحل رسیدن به زیردرخت توافقی

## ۳ درخت توافقی

بزرگترین زیردرخت توافقی بین دو درخت، درختی است که با حذف کمترین تعداد گونه از هر دو درخت و ساده‌سازی (یعنی حذف راس های درجه دو به جز ریشه، یعنی راس هایی که فرزند خود را به والد متصل می‌کنند، و اتصال مستقیم فرزند به والد) بتوان به آن رسید. شکل ۴ مثالی از زیردرخت توافقی را نشان می‌دهد. واضح است که با انتخاب دو گونه‌ی دلخواه یکسان از هر دو درخت و حذف بقیه‌ی برگ‌ها، به دو درخت یکسان که در آن این دو گونه به ریشه متصل‌اند می‌رسیم، پس در حالت حداقلی درختی به اندازه ۲ وجود دارد.

الگوریتم بدست آوردن بزرگترین زیردرخت توافقی از برنامه‌نویسی پویا کمک می‌گیرد. به این صورت که به ازای هر راس u از درخت اول و هر راس v از درخت دوم، امتیاز بزرگترین زیردرخت بین زیردرخت‌هایی که این رئوس ریشه‌ی آن هستند را حساب می‌کند، و برای اینکار حالات مختلف انتصاب زیردرخت هرکدام از فرزندان این راس‌ها به همدیگر را بررسی کرده و مقدار بیشینه را حساب می‌کند. در رابطه‌ی بازگشتی زیر، این موضوع نشان داده شده. با این نکته که $MAST(T_1^u, T_2^v)$ یعنی اندازه‌ی بزرگترین زیردرخت توافقی بین زیردرخت نوادگان u از $T_1$ و زیردرخت نوادگان v از $T_2$. حالت پایه نیز زمانی است که u و v جزو برگ‌ها باشند که اگر یکسان باشند جواب ۱ و در غیراین‌صورت جواب ۰ است.

$$MAST(T_1^u, T_2^v) = \max \begin{cases} MAST(T_1^{u_1}, T_2^{v_1}) + MAST(T_1^{u_2}, T_2^{v_2}) \\ MAST(T_1^{u_2}, T_2^{v_1}) + MAST(T_1^{u_1}, T_2^{v_2}) \\ MAST(T_1^{u_1}, T_2^{v}) \\ MAST(T_1^{u_2}, T_2^{v}) \\ MAST(T_1^{u}, T_2^{v_1}) \\ MAST(T_1^{u}, T_2^{v_2}) \end{cases}$$

# ۴ درخت اجماعی (Consensus Tree)

هدف مسئله‌ی درخت اجماعی این است که با دریافت مجموعه از درخت‌ها $\{T_1, T_2, ..., T_m\}$ با کمک گرفتن از مجموعه‌ی تقسیم‌های آنان $\{W(T_1), W(T_2), ..., W(T_m)\}$ یک درخت $T$ بسازد که مجموعه‌ی تقسیم‌های آن $W(T)$ باشد.

## ۱.۴ درخت اجماعی اکید

درختی که مجموعه‌ی تقسیم‌هایش برابر اشتراک مجموعه تقسیم‌های چند درخت باشد، درخت اجماعی اکید آن چند درخت نامیده می‌شود. به عبارت دیگر $W(T) = W(T_1) \cap W(T_2) \cap ... \cap W(T_m)$ . برای ساخت این درخت اجماعی از لم زیر استفاده می‌کنیم.

**لم ۱.** اگر یک درخت و مجموعه‌ی تقسیم‌های آن را داشته باشیم، می‌توانیم با حذف هرکدام از تقسیم‌ها، درخت جدیدی با مجموعه‌ی تقسیم جدید بسیازیم.

اثبات: کافی است در درخت اصلی، یالی که تقسیم حذف شده را تولید می‌کند منقبض کنیم، یعنی رئوس دو طرف آن را روی هم قرار داده و تبدیل به یک راس واحد کنیم.

طبق این لم، برای تولید درخت اجماعی اکید کافی است یکی از درخت‌های اولیه را انتخاب کرده و تقسیم‌هایی از آن را که در $W(T)$ وجود ندارد، حذف کنیم. در نهایت و با توجه به اینکه $W(T) \subseteq W(T_x)$ به درختی می‌رسیم که مجموعه تقسیم‌های آن برابر $W(T)$ است.

## ۲.۴ درخت اجماعی اکثریت

در یک درخت اجماعی اکثریت، $W(T)$ شامل تقسیم‌هایی است که در بیشتر از نصف (بیش از $\frac{m}{2}$) از $W(T_i)$ ها آمده است. برای ساخت درخت اجماعی اکثریت به صورت زیر عمل می‌کنیم.

- مجموعه تقسیم‌هایی درخت نهایی ($W(T)$) را طبق تعریف درست می‌کنیم.
- برای هر درخت $T_i$، ریشه را روی یک گونه‌ی دلخواه، مثلا گونه‌ی ۱ قرار می‌دهیم (گونه انتخاب شده برای تمام درخت‌ها باید یکسان باشد). همینطور درخت را طبق لم۱ طوری منقبض می‌کنیم که تقسیم‌هایی که در $W(T)$ نیستند حذف شوند.
- به ازای هر تقسیم c، $B_c$ را آن طرف تقسیم تعریف می‌کنیم که شامل برگ ۱ نباشد.

**لم ۲.** به ازای هر دو تقسیم c و d، عضو $W(T)$ یا $B_c \subseteq B_d$ یا $B_d \subseteq B_c$ و یا $B_c \cap B_d = \emptyset$

اثبات: از آن‌جایی که تقسیم‌های موجود در $W(T)$ در بیش از نصف درخت‌ها وجود دارند، پس حداقل یک درخت وجود دارد که هردو تقسیم c و d در آن وجود داشته باشد. حال در این درخت یا d در مسیر رسیدن از یال c به ریشه‌ی ۱ است، یا c در مسیر d است و یا هردو یال از مسیرهای جداگانه به ریشه‌ی ۱ می رسند (در اینجال منظور از یال x، یعنی یالی که تقسیم روی آن تقسیم x را نتیجه می‌دهد).

**لم ۳.** اگر همه‌ی تقسیم‌های $c_i$ که در آن $B_{c_i}$ شامل گونه‌ی x باشند را در نظر بگیریم و آن‌ها را $B_{x_i}$ بنامیم، یک زنجیره از B ها می توان ساخت که $B_{x_1} \subseteq B_{x_2} \subseteq ... \subseteq B_{x_t}$ باشد

اثبات: از آنجایی که همه‌ی $B_{x_i}$ها گونه‌ی x را دارند و اشتراکشان تهی نیست، پس طبق لم ۲ باید زیرمجموعه‌ی یکدیگر باشند. اگر آن‌ها را به ترتیب اندازه (تعداد عناصر) مرتب کنیم، رابطه‌ی لم ۳ برقرار خواهد بود. در واقع این زنجیره، زنجیره‌ی یال‌ها از برگ x تا ریشه‌ی ۱ را در درخت $T$ نشان می‌دهد.

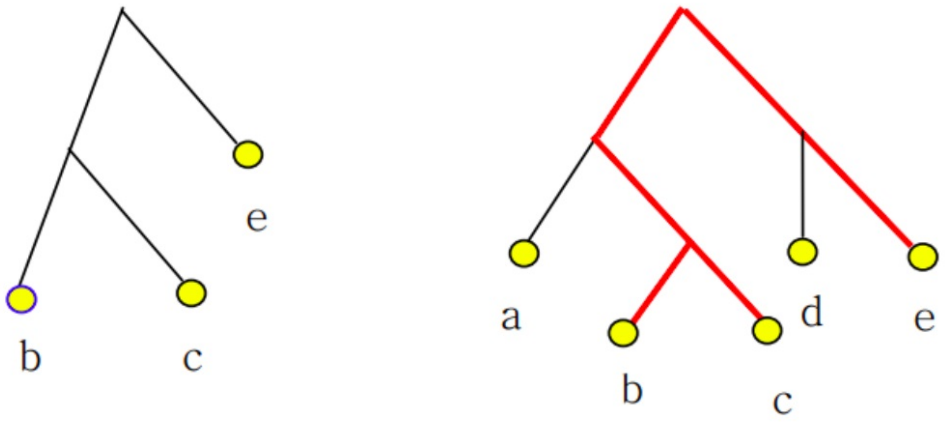
**لم ۴.** حتما در یک درخت یال مربوط به $B_{x_i}$ فرزند مستقیم $B_{x_{i+1}}$ است.

اثبات: این دو یال طبق استدلال لم ۱، در یک درخت حتما باهم وجود دارند. اگر در آن درخت یال دیگری بین این دو یال وجود داشت، اندازه‌ی $B_x$ آن حتما بین اندازه‌ی دو یال دیگر بود، و چون تمام یال‌های درخت نیز پس از منقبض‌سازی در مراحل اولیه، حتما در $W(T)$ حضور دارند، پس این یال نیز باید در زنجیره‌ی $B_x$ها می‌بود و بنابراین باید بین $B_{x_i}$ و $B_{x_{i+1}}$ قرار می‌گرفت که با فرض مسئله در تضاد است.

- با توجه به لم های ۲ و ۳ و ۴، مرحله به مرحله درخت $T$ را می‌سازیم. به این صورت که به ازای هر یال c ، P[c] را یال بالاسر یال c در نظر می‌گیریم و در هر مرحله سعی می‌کنیم نزدیکترین یال بالاسر را انتخاب کنیم تا جایی که به یال والد c برسیم. مراحل کار در الگوریتم ۱ شرح داده شده است.

**الگوریتم ۱** ساخت درخت اجماعی اکثریت

۱: به ازای $i = ۱$ تا $m$:
۲: به ازای هر یال c در درخت $T_i$:
۳: یال q را یال والد c در $T_i$ در نظر می‌گیریم
۴: اگر $P[c] = \emptyset$ یا $|B_q| < |B_{P[c]}|$ آنگاه:
۵: $P[c] = q$
۶: پایان اگر
۷: پایان به ازای
۸: پایان به ازای




شکل ۵: ساخت ۳تایی از روی یک درخت ریشه‌دار

## ۳.۴ درخت $R^*$

درخت $R^*$ نیز یک مدل درخت اجماعی است که بر اساس سه‌تایی‌های درخت‌های موجود ساخته می‌شود.

### ۱.۳.۴ سه تایی

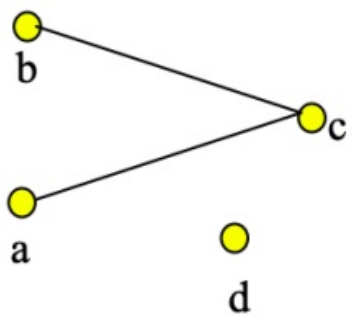
در یک درخت ریشه‌دار، با نگه‌داری ۳ برگ و حذف بقیه برگ‌ها و منقبض سازی درخت، به یک درخت ۳تایی می‌رسیم. در این ۳تایی والد دو برگ، فرزند والد برگ دیگر خواهد بود، یا به عبارت دیگر دو برگ به هم نزدیک‌تر هستند تا برگ سوم. برای نمایش این موضوع از نماد ab | c استفاده می‌کنیم که به این معنی است که برگ c در سه تایی، از برگ‌های a و b دورتر است. با توجه به این موضوع، برای هر ۳تایی ۳حالت مختلف داریم، براساس اینکه کدام برگ دورتر باشد. برای مثال سه‌تایی شکل ۵ به صورت bc | e قابل نمایش است.

### ۲.۳.۴ گراف باقی‌مانده

الگوریتم ساخت گراف باقی‌مانده، سه‌تایی‌های درخت را دریافت کرده و به ازای هر ab | c یک یال بین a و b می‌گذارد. در مرحله‌ی نهایی، یک گراف غیرهمبند خواهیم داشت که هربخش یکی از زیرشاخه‌های ریشه‌ی اصلی درخت هستند. به بیان دیگر این الگوریتم با حذف ریشه، مجموعه‌های باقی‌مانده را برمی‌گرداند. مثالی از روش ساخت گراف باقی‌مانده را در شکل ۶، و چند نمونه از مجموعه‌های باقی‌مانده چند درخت را در شکل ۷ می‌توان مشاهده کرد.

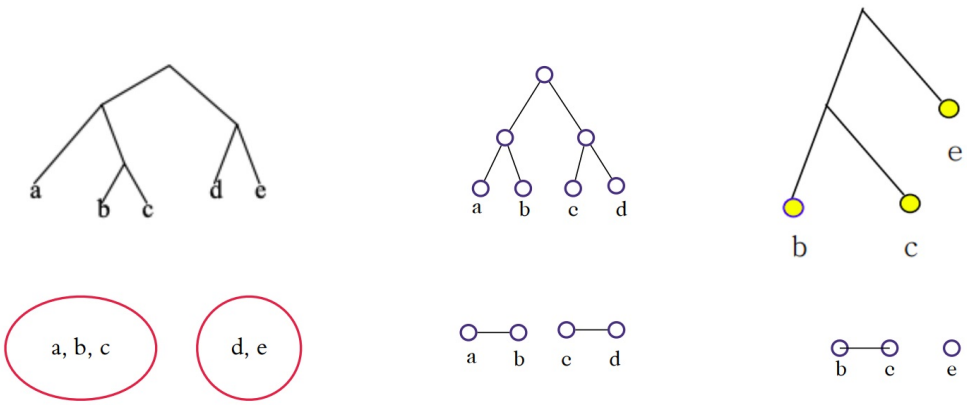
### ۳.۳.۴ ساخت درخت از روی سه‌تایی‌ها

در این بخش قصد داریم الگوریتمی ارائه دهیم که با داشتن سه‌تایی‌ها، درختی که آن سه‌تایی‌ها را داشته باشد بسازد. الگوریتم ۲ بدین منظور ارائه شده است. یک مثال از اجرای این الگوریتم برای ورودی bc | a و ac | d و de | b را می‌توان در شکل ۸ مشاهده کرد.




bc | a, ac | d

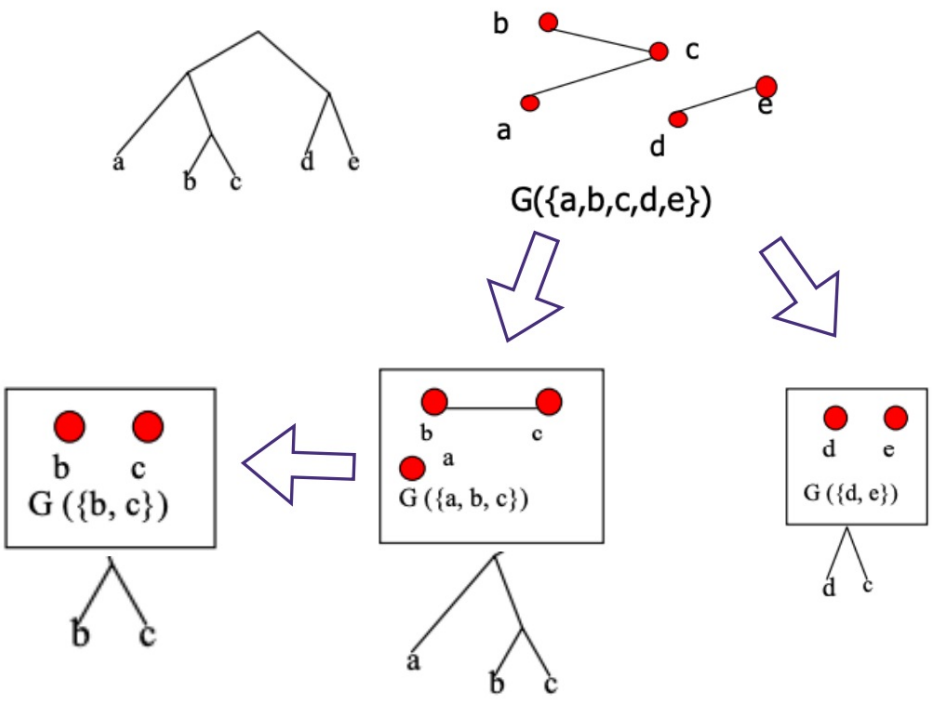
شکل ۶: یک نمونه گراف باقی‌مانده




شکل ۷: چند مجموعه‌ی باقی‌مانده

**الگوریتم ۲** ساخت درخت از روی سه‌تایی‌ها

۱: گراف باقی‌مانده را بساز
۲: اگر ناهمبند بود:
۳: برای هرزیرمجموعه درخت را بساز (بازگشتی)
۴: یک راس ریشه‌ی جدید اضافه کن و همه‌ی درخت‌ها را زیردرخت آن بگذار
۵: این درخت را برگردان
۶: اگر نه:
۷: باید حتما تک راس باقی‌مانده باشد، وگرنه نمی‌توان درخت ساخت
۸: آن تک راس را به عنوان درخت بازسازی شده برگردان




شکل ۸: مثال از اجرای الگوریتم ۲

#### ۴.۳.۴ ساخت درخت $R^*$

برای ساخت درخت $T$ از روی درخت‌های $\{T_1, T_2, ..., T_m\}$ به روش $R^*$، از هر درخت مجموعه‌ی ۳تایی‌ها را در یک ماتریس علامت می‌زنیم. در آن ماتریس به ازای هر ۳برگ مختلف، شیوه‌های ساخت ۳تایی که ۳حالت می‌تواند داشته باشد، در نظر گرفته می‌شود و اینکه از هر شیوه چه تعداد درخت، ۳تایی را به آن صورت دارند هم ثبت می‌شود. در نهایت شیوه‌ای که بیشترین تعداد را داشته باشد، به عنوان ۳تایی نهایی آن ۳برگ برای درخت $T$ انتخاب می‌شود. به این صورت از $TL(T_1), ..., TL(T_m)$ ها به $TL(T)$ می‌رسیم ($TL$ مجموعه‌ی سه‌تایی‌هاست). حال با داشتن مجموعه‌ی سه‌تایی‌های درخت $T$، با استفاده از الگوریتم ۲ درخت را می‌سازیم.

## ۵ روش‌های نوشتن درخت

در این بخش روش‌هایی را برای تبدیل ساختار یک درخت به متن جهت انتقال راحت ارائه داده می‌دهیم.

### ۱.۵ قالب Newick

این قالب حالت‌های مختلف را پوشش می‌دهد اما قواعد کلی آن به این صورت است که هر پرانتز بیانگر یک راس میانی است و در آن فرزندان با ، از همدیگر جدا شده‌اند. اگر راسی اسم داشته باشد در سمت راست پرانتز مربوط به آن راس نوشته می‌شود، یا اگر برگ باشد به صورت مستقیم نوشته می‌شود. اگر یک راس فاصله تا ریشه داشته باشد، بعد از رعایت قواعد قبلی نام گذاری، در آخر در سمت راست اطلاعات هر راس : گذاشته شده و سمت راست آن میزان فاصله نوشته می‌شود. چند مثال این قالب در شکل ۹ نمایش داده شده است.

### ۲.۵ قالب NEXUS

این قالب علاوه بر ساختار درخت، اطلاعات دیگری را نگه می‌دارد، برای مثال ویژگی‌های گونه‌های موجود و قواعد تبدیل نام آن‌ها به اعداد برای استفاده در ساختار درخت. یا مثلا ماتریس هم‌ردیفی بین گونه‌های موجود. در نهایت ساختار درخت طبق ساختار Newick داده می‌شود. مثالی از قالب NEXUS در شکل ۱۰ قابل مشاهده است.

```
(,,(,));                                no nodes are named
(A,B,(C,D));                            leaf nodes are named
(A,B,(C,D)E)F;                          all nodes are named
(:0.1,:0.2,(:0.3,:0.4):0.5);            all but root node have a distance to parent
(:0.1,:0.2,(:0.3,:0.4):0.5):0.0;        all have a distance to parent
(A:0.1,B:0.2,(C:0.3,D:0.4):0.5);        distances and leaf names (popular)
(A:0.1,B:0.2,(C:0.3,D:0.4)E:0.5)F;      distances and all names
((B:0.2,(C:0.3,D:0.4)E:0.5)F:0.1)A;     a tree rooted on a leaf node (rare)
```

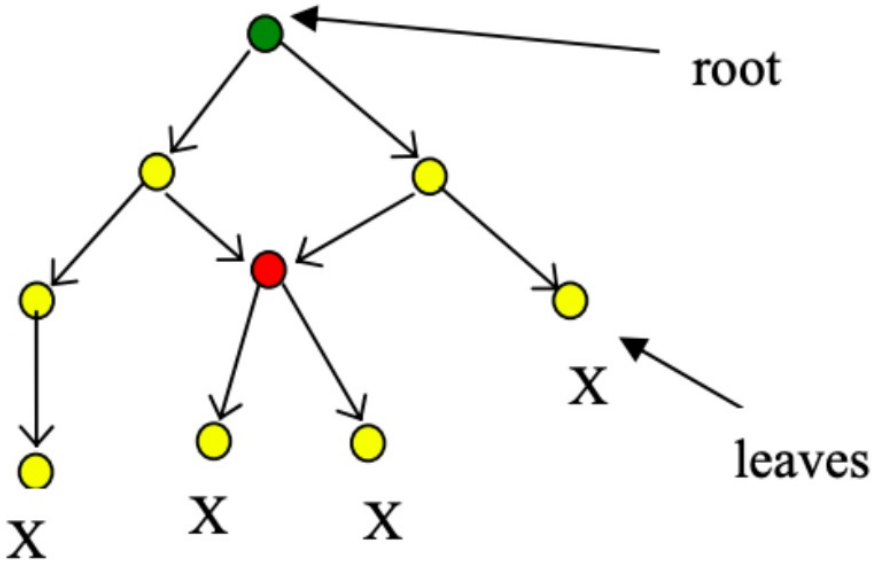
شکل ۹: قالب Newick

```
#NEXUS
Begin TAXA;
  Dimensions ntax=4;
  TaxLabels SpaceDog SpaceCat SpaceOrc SpaceElf
End;

Begin data;
  Dimensions nchar=15;
  Format datatype=dna missing=? gap=- matchchar=.;
  Matrix
    [ When a position is a "matchchar", it means that it is the same as the first entry
at the same position. ]
    SpaceDog   atgctagctagctcg
    SpaceCat   ......??...-.a.
    SpaceOrc   ...t.......-.g. [ same as atgttagctag-tgg ]
    SpaceElf   ...t.......-.a.
  ;
End;

BEGIN TREES;
  Tree tree1 = (((SpaceDog,SpaceCat),SpaceOrc,SpaceElf));
END;
```

شکل ۱۰: قالب NEXUS




شکل ۱۱: یک شبکه‌ی تبارزایی

## ۶ ابردرخت (Supertree)

ابردرخت یک درخت تبارزایی است که به جای اینکه مستقیم از روی گونه‌های مختلف ساخته شود، از روی درخت‌های تبارزایی روی زیرمجموعه‌های گونه‌ها (مثلا گونه‌های شبیه بهم از یک رده) ساخته می‌شود. این کار باعث می‌شود که بتوان روش‌های درخت اجماعی را روی زیرمجموعه‌های شبیه و کوچک از گونه‌ها اجرا کرد و از پیچیدگی زیاد مسئله جلوگیری نمود.

## ۷ شبکه‌های تبارزایی (Phylogenetic Networks)

شبکه‌های تبارزایی مانند درخت‌های تبارزایی هستند، با این تفاوت که هر موجود می‌تواند بیش از یک والد داشته باشد. کاربرد اصلی آن در باکتری‌ها و فرایندهای انتقال ژن افقی آنان است که در آن باکتری‌های یک نسل می‌توانند با یک‌دیگر تبادل ژنتیکی داشته باشند و فرزندانی تولید کنند که از ژنتیک چندین گونه به ارث می‌برد. شکل ۱۱ مثالی از این شبکه‌های تبارزایی است.

## مراجع

*[For21] Hadi Foroughmand.* Computational genomics course slides. *Sharif University, 2021.*

*[Sun09] Wing-Kin Sung.* Algorithms in bioinformatics: A practical introduction. *CRC Press, 2009.*