دانشکده مهندسی کامپیوتر
دانشگاه صنعتی شریف

ژنومیک محاسباتی

مطهری و فروغمند
پاییز ۱۴۰۰

بازسازی درخت تبارزایی

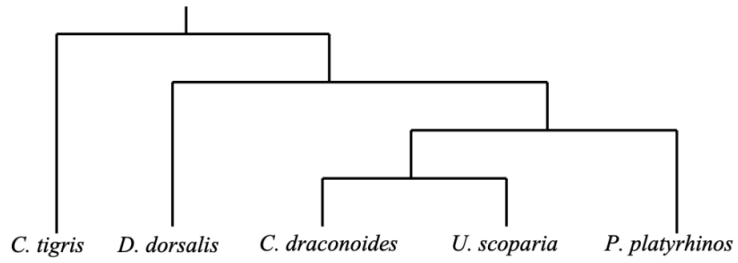
جلسه چهارم

نگارنده: سید پوریا لقایی

۱ مروری بر مباحث گذشته

در جلسه‌های گذشته مباحث هم‌ترازی رشته‌ها مطرح شد. در این جلسه مباحث بازسازی درخت‌های تبارزای مطرح می‌شود.

۲ نمونه‌ای از درخت‌های تبارزایی

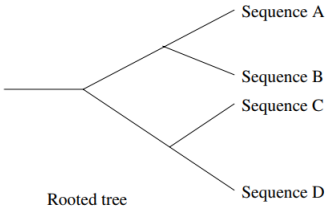


شکل ۱: درخت تبارزایی

در مدل‌های تکاملی سعی می‌کنیم با داشتن اطلاعاتی از گونه‌ها درخت تبارزایی آن را بسازیم. به عنوان مثال ممکن است برای یک ویروس درخت تبارزایی بسازیم تا در نهایت منشاء آن را پیدا کنیم.

۱.۲ ساخت درخت تبارزایی با ریشه

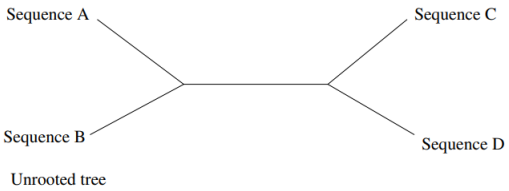
درخت تبارزایی در اصل یک درخت با ریشه است اما در بسیاری از موراد توانایی ساختن درخت ریشه دار را نداریم. معمولا درخت های بدون ریشه ساخته می شوند و با استفاده از تکنیک هایی، از روی آن درخت ریشه‌دار را استخراج می کنیم.



شکل ۲: درخت تبارزایی ریشه‌دار

۲.۲ درخت تبارزایی بدون ریشه

اطلاعاتی که درخت بی‌ریشه به ما می‌دهد در مورد شباهت بین گونه‌ها است. اگر بتوانیم جای ریشه را در درخت بی‌ریشه پیدا کنیم به همان درخت ریشه‌دار (که در اصل در مسائل هم به دنبال آن هستیم) تبدیل می‌شود. نکته دیگر آنکه همه اطلاعات موجود در درخت ریشه‌دار در درخت بی‌ریشه نیز موجود است ولی فقط جای ریشه را نداریم. با توجه به اینکه ساختن درخت ریشه‌دار مسئله سخت است، فرض کنید یک الگوریتمی در زمان چند جمله‌ای برای ایجاد درخت بی ریشه داشته باشیم. در این صورت یک مسئله داریم با این عنوان که چطور از روی درخت بی‌ریشه، درخت ریشه‌دار را ایجاد کنیم.



شکل ۳: درخت تبارزایی بدون ریشه

۳ روش‌های تبدیل درخت بدون ریشه به ریشه دار

از راه حل‌های موجود برای حل این مسئله می توان از موارد زیر نام برد:

- اضافه کردن یک گونه متفاوت و دور نسبت به گونه‌های موجود در درخت بی‌ریشه (حالتی از تبدیل درخت بی‌ریشه به ریشه‌دار که در آن اطلاعاتی به مجموعه اضافه کرده‌ایم)
- وسط بزرگترین یال را به عنوان ریشه در نظر بگیریم

برای ساختن درخت تبارزایی باید ارتباط گونه‌ها با هم را داشته باشیم. به دو صورت ارتباط بین گونه‌ها مدل می شود:

- ماتریس ویژگی گونه‌ها
- ماتریس فاصله بین گونه‌ها

در بعضی از شرایط ها که تعداد ویژگی ها زیاد است ساخت درخت تبارزایی از روی ماتریس ویژگی کار مناسبی نیست. در حال حاضر درخت ساختن برای ویروسی مثل کرونا که داده های آن بسیار زیاد شده، کار بسیار سخت است. معمولا با تغیر درخت های قدیمی ویروس کرونا، درخت جدید ان را می سازند.

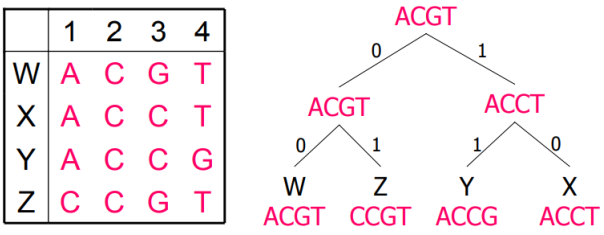
۱.۳ معیارهای سنجش درخت

حال که ساختن درخت ها را بررسی کردیم باید معیاری داشته باشیم که بتوانیم بهترین درخت را تشخیص دهیم. معیار های موجود شامل دو دسته می شوند :

- معیار بیشینه صرفه جویی
- معیار های احتمالاتی (بررسی تطابق درخت با داده های زیست شناسی)

۴ بازسازی ویژگی مبانی درخت تبارزایی

در این روش ورودی، ماتریس ویژگی گونه‌ها بوده. به عنوان نمونه در ژنوم انسان، ابتدا یک هم ترازی بین رشته‌ها ایجاد می شود و سپس از روی آن ماتریس ویژگی را استخراج می کنند. همانطور که بالاتر هم گفته شد، یک معیار برای تشخیص بهترین درخت، بیشینه صرفه جویی است. اگر درختی داشته باشیم که توپولوژی، برگ‌ها و نودهای میانی آن مشخص باشد و همینطور جمع میزان تغیرات بین همه نودها و فرزندان انها، به عنوان هزینه درخت محسوب شود، مسئله ای تعریف می‌کنیم تحت این عنوان که بین درخت‌های ممکن، درختی را بیابیم که عدد جمع میزان تغیرات بین نودها و فرزندانشان، کمینه باشد.



شکل ۴: مثال بیشینه صرفه‌جویی

۱.۴ حل مسئله بیشینه صرفه جویی (sankoff)

همانطور که بالاتر هم گفته شد اگر یک ماتریس ویژگی و یک توپولوژی برای درخت به عنوان ورودی داده شود (حتی جای گره های برگ را نیز داشته باشیم)، و از ما خواسته شود با این شرایط صرفه‌جویانه ترین درخت را پیدا کینم راه حلی چند جمله‌های برای این مسئله وجود دارد. الگوریتمی به نام sankoff که برای هر ویژگی یک بار جداگانه اجرا می‌شود. یک امتیاز تحت عنوان $A[v\ ,\ c]$ تعریف می‌کنیم که به معنی بهترین امتیاز زیر درخت v به ازای انتصاب حرف c به آن است. بر همین اساس یک رابطه بازگشتی می‌نویسیم :

$$A[v,c] = \sum_{u:v\to u} minscore(c,c') + A[u,c']$$

فرمول بالا میزان امتیاز برای هر نود میانی است . برای برگ‌ها هم رابطه زیر را داریم :

$$A[v,c] = \{\circ \quad if \quad c = a \quad else \quad +\infty\}$$

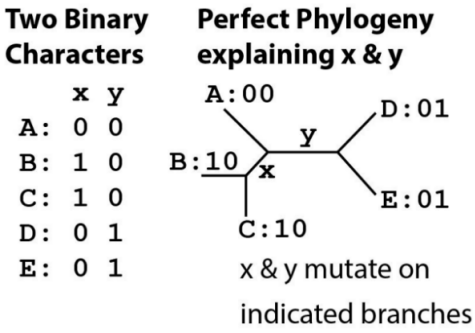
جواب نهایی برابر با حرفی است که کمترین مقدار امتیاز در ریشه را دارد. درخت موجود برای این مسئله تعداد $2m-1$ راس خواهد داشت(m برگ و $m-1$ نود میانی). برای هر ویژگی به تعداد گونه ها (m) ماتریس A پر می شود. تعداد کل ویژگی‌ها هم n تاست. پس پیچیدگی زمانی کلی الگوریتم (mn) است. پیچیدگی مکانی هم به دلیل اینکه به ماتریس ویژگی های پرشده قبلی نیاز نداریم و می توانیم انها را دور بریزیم، (m) است.

۵ یافتن بهترین توپولوژی درخت

مسئله بعدی این است که بخواهیم بهترین توپولوژی درخت را نیز به دست اوریم. در این شرایط باید همه درخت ها را در نظر بگیریم. مشخص است که این مسئله راه حل چند جمله ای ندارد. اگر بخواهیم یک نود را به درخت اضافه کنیم باید روی یک یال اضافه شود. تعداد یال های درخت بدون ریشه برای n گونه و راس میانی درجه سه برابر است با : $!!(2n-5)$. در کل مسائل یافتن درخت تبارزای صرفه جویانه، قابل حل در زمان چند جمله‌ای نیست. به همین دلیل روی این مسئله الگوریتم‌های هیوریستیک و تقریبی ارائه می شود

۶ درخت بی نقص

درخت بی نقص درختی است که مقدار هزینه آن دقیقا برابر با تعداد ستون‌ها(ویژگی‌ها) باشد. نکته این درخت این است که به ازای هر دو ویژگی که در نظر بگیریم همه حالات را در سطرهای آن پیدا نکنیم.



شکل ۵: درخت بی نقص

۷ روش برآورد درست نمایی بیشینه

فرض کنید یک سکه داریم که احتمال شیر و خط امدن آن را نمی‌دانیم. طبق مشاهدات می‌توانیم احتمال دیدن شیر و خط را تقریب بزنیم. هر چقد میزان مشاهدات ما بیشتر بود تقریبی که داریم بهتر است. در روش برآورد درست نمایی بیشینه ما مدل احتمالاتی را انتخاب می کنیم که احتمال مشاهده مد نظر ما را بیشینه کند. مدل احمالاتی از درخت و احتمال روی یال‌ها تشکیل شده است. اگر درخت واحتمالات را داشته باشیم و بدانیم روی هر راسی چه رشته‌هایی وجود دارد، می توانیم احتمال مشاهده را حساب کنیم. مسئله دیگر این است که فرض کنیم تنها برگ‌ها موجود باشد و توپولوژی درخت را نیز بدانیم. حال آیا می‌توانیم مسئله را با روش برآورد درست نمایی بیشینه حل کنیم؟ در این صورت باید همه حالات را در نظر بگیریم. این محاسبات بسیار طولانی بوده و ممکن است در زمان چند جمله‌ای قابل اجرا نباشد. اما با یک رابطه بازگشتی می‌توان این مسئله را در زمان بهتری حل کرد :

$$\prod_i (\frac{۱}{۲} L_i(r, ۰) + \frac{۱}{۲} L_i(r, ۱))$$

حالت پایه برای برگ V نیز برابر است با:

$$L_i(v, s) = ۱_{s=M[v,i]}$$

و همینطور یک رابطه بازگشتی برای راس میانی u با فرزندان v و w :

$$L_i(u, s) = \left[\sum_{y \in \{۰,۱\}} L_i(v, y) Pr(v_i = y | u_i = s)\right] \left[\sum_{x \in \{۰,۱\}} L_i(w, x) Pr(w_i = x | u_i = s)\right]$$

مراجع

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

.[Sun09]