دانشکده مهندسی کامپیوتر
دانشگاه صنعتی شریف

ژنومیک محاسباتی

مطهری و فروغمند
پاییز ۱۴۰۰

یک الگوریتم برای بازسازی درخت تبارزایی ویژگی‌مبنا

جلسه نهم

نگارنده: ایوب وزیری مقدم

۱ مروری بر مباحث گذشته

در هم‌ردیفی چندگانه که در جلسه قبل به آن اشاره شد، چند رشته S_i و ماتریس مشابهت حروف به عنوان ورودی مسئله داده می شود و در خروجی، مشابه هم‌ترازی دوگانه، یک جدول با **i** سطر که هر سطر همان رشته S_i است که به هرکدام تعدادی فضای خالی (**Gap**) اضافه شده است. برای حل مسائل هم‌ترازی چندگانه، می‌توان همانند مسئله هم‌ترازی دوگانه، با استفاده از روش برنامه‌ریزی پویا راه حل ارائه داد.

روش هایی که برای حل مسائل هم‌ترازی چندگانه مطرح شدند عبارت‌اند از روش ستاره‌مرکز،روش‌های پیش‌برنده مانند **ClustalW** و روش‌های تکراری مانند **MUSCLE** .

۲ درخت تبارزایی ویژگی‌مبنا

در مسئله درخت تبارزایی ویژگی‌مبنا، ماتریس ویژگی‌ها به عنوان ورودی در نظر گرفته می‌شود به طوری که سطرهای ماتریس نشان‌دهنده گونه‌ها و ستون‌های آن نشان‌دهنده ویژگی‌ها است و در نهایت یک درخت روی گونه‌ها به عنوان خروجی برگردانده می‌شود.

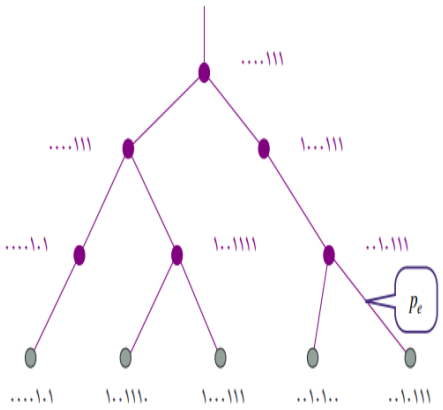
۱.۲ مدل احتمالاتی

برای حل یک مسئله درخت تبارزایی ابتدا باید یک تابع هدف داشته باشیم که معمولاً یک مدل احتمالاتی است و ما باید تلاش کنیم که محتمل‌ترین حالت را پیدا کنیم. مدل احتمالاتی که در مسئله درخت تبارزایی ویژگی‌مبنا داریم یک مدل دوحالته است زیرا هر یک از اسیدهای نوکلئیک از نظر

ساختاری یا پورین[۱] هستند یا پریمیدین[۲]، به همین دلیل ورودی‌ها رشته‌هایی شامل صفر و یک هستند.

۲.۲ مسئله درستنمایی بیشینه اجدادی

این مسئله یک سری رشته‌های صفر و یک که برگ‌های درخت هستند را به عنوان ورودی می‌گیرد و یک درخت روی این برگ‌ها،یک سری رشته‌های صفر و یک روی رأس‌های میانی و به ازای هر یال یک احتمال را به عنوان خروجی برمی‌گرداند. اگر یک درخت با یک سری رشته‌های صفر و یک روی رأس‌های میانی و یک سری احتمال روی هر یک از یال‌ها داشته باشیم آنگاه می‌توانیم یک احتمال به آن درخت نسبت دهیم. اگر فرض کنیم هر یک از مکان‌ها به صورت مستقل از هم تغییر می‌کنند در این صورت هر مکان با احتمال P_e تغییر می کند و با احتمال $(۱ - P_e)$ بدون تغییر باقی می‌ماند، بنابراین هدف ما در این مسئله پیدا کردن محتمل‌ترین درخت است.



شکل ۱: یک نمونه از درخت خروجی

ANCESTRAL MAXIMUM LIKELIHOOD VERSION I

Input: A set S of m binary sequences, each of length n.

Goal: Find a tree T with m leaves, an assignment $e \longmapsto p_e \in [0,1]$ of edge probabilities, and a labelling $\lambda : V(T) \to \{0,1\}^n$ of the vertices such that

1) The m labels of the leaves are exactly the sequences from S, and
2) $\prod_{e \in E(T)} p_e^{d_e}(1-p_e)^{n-d_e}$ (where d_e is the Hamming distance of the two labels across the edge e) is maximized.

شکل ۲: شبه کد الگوریتم درستنمایی بیشینه اجدادی نسخه ۱

۳.۲ بیشینه کردن تابع هدف

حال به دنبال این هستیم که P_e را طوری محاسبه کنیم که تابع هدف زیر بیشینه شود:

$$\text{بیشینه کن} \quad P_e^{d_e}(۱ - P_e)^{n-d_e}$$

بنابراین گرادیان تابع هدف را نسبت به P_e به دست آورده و مساوی صفر قرار می‌دهیم:

[۱] Purine
[۲] Pyrimidine

$$\Rightarrow \quad d_e P_e^{d_e-1}(1-P_e)^{n-de} - (n-d_e)P_e^{d_e}(1-P_e)^{n-d_e-1} = 0$$
$$\Rightarrow \quad d_e(1-P_e) - (n-d_e)P_e = 0$$

بنابراین به جای P_e ها مقدار d_e/n را در تابع هدف قرار می‌دهیم:

$$\Pi_{e\in E(T)}(\frac{d_e}{n})^{\frac{d_e}{n}}(1-\frac{d_e}{n})^{1-\frac{d_e}{n}}$$
$$\sum_{e\in E(T)}(\frac{d_e}{n}log(\frac{d_e}{n}) + (1-\frac{d_e}{n})log(1-\frac{d_e}{n}))$$

که به این عبارت در صورت داشتن یک توزیع احتمال آنتروپی[۳] می‌گویند:

$$\sum_{e\in E(T)} -H_2(\frac{d_e}{n})$$

بنابراین کافی است که تابع هزینه بدست آمده را کمینه کنیم، در این صورت شبه‌کد الگوریتم درستنمایی بیشینه به صورت زیر تغییر می کند:

ANCESTRAL MAXIMUM LIKELIHOOD VERSION II

ANCESTRAL MAXIMUM LIKELIHOOD (VERSION II)
Input: A set S of m binary sequences, each of length n.
Goal: Find a tree T with m leaves and a labelling $\lambda : V(T) \to \{0,1\}^n$ of the vertices such that
1) The m labels of the leaves are exactly the sequences from S, and
2) $\sum_{e\in E(T)} H(d_e/n)$ is minimized.

$-p\log_2(p) - (1-p)\log_2(1-p)$

شکل ۳: شبه کد الگوریتم درستنمایی بیشینه اجدادی نسخه ۲

۴.۲ مسئله حداکثر صرفه‌جویی

اگر از تابع هزینه به‌دست آمده در قسمت قبل H را حذف کنیم یعنی به جای آنکه $H(\frac{d_e}{n})$ را کمینه کنیم، $\frac{d_e}{n}$ را کمینه کنیم که معنی آن این است که می‌خواهیم صرفه‌جویانه‌ترین درخت را پیدا کنیم یعنی درختی را پیدا کنیم که جمع $\frac{d_e}{n}$ ها کمینه شود که این همان مسئله حداکثر صرفه‌جویی است. بنابراین دو مسئله حداکثر صرفه‌جویی و درستنمایی شبیه به هم هستند، با این تفاوت که وزن یال‌ها را با دو تابع مختلف حساب می‌کنیم.

[۳] Entropy

MAXIMUM PARSIMONY

Input: A set S of m binary sequences, each of length n.

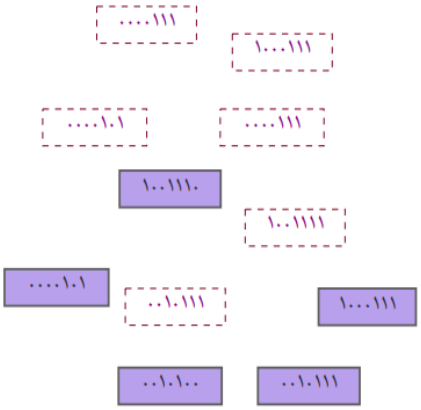
Goal: Find a tree T with m leaves and a labelling $\lambda : V(T) \to \{0,1\}^n$ of the vertices such that

1) The m labels of the leaves are exactly the sequences from S, and
2) $\sum_{e \in E(T)} (d_e/n)$ is minimized.

شکل ۴: شبه کد الگوریتم حداکثر صرفه‌جویی

فرض کنیم یک گراف عظیم داریم که رأس‌های آن شامل همه رشته‌های صفر و یک باشد و بعضی از این رشته‌های صفر و یک را به عنوان ورودی S به ما داده باشند. ساختن یک درخت فیلوژنی روی این رأس‌ها مانند این است که یک زیردرخت همبند پیدا کنیم که مجموعه S برگ‌های این زیردرخت باشند، به طوری که وزن این زیردرخت کمینه باشد.

تذکر: اگر در درخت ساخته شده بعضی از رأس‌های مجموعه S به عنوان گره میانی باشند می توانیم از همان راس یک فرزند با طول صفر تولید کنیم و فرزند تولید شده را به عنوان ورودی در نظر بگیریم.



شکل ۵: نمونه‌ای از ورودی مسئله حداکثر صرفه‌جویی

۵.۲ مسئله درخت اشتاینر

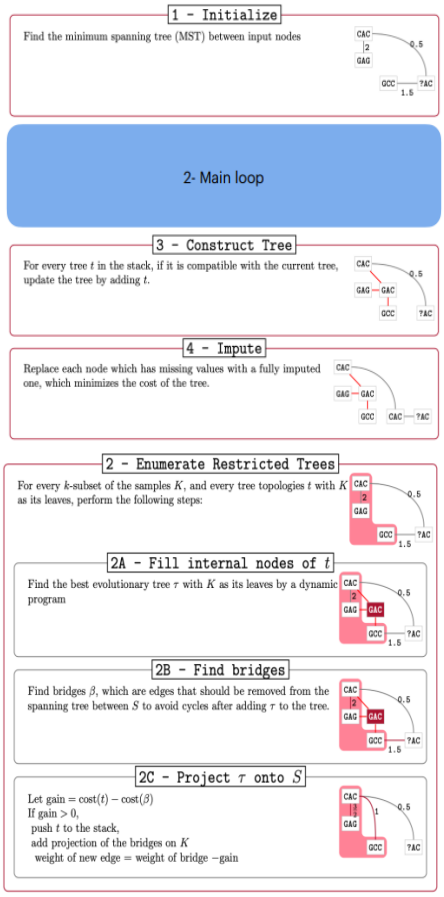
- ورودی: یک گراف و یک سری رأس‌های ترمینال
- خروجی: یک زیردرخت همبند شامل رأس‌های ترمینال با هزینه کمینه

برای حل این مسئله نه تنها الگوریتمی وجود ندارد بلکه اگر $P \neq NP$ آنگاه به ازای هر $\epsilon > 0$ نمی‌توان یک الگوریتم $(1+\epsilon)$ـ تقریب برای آن پیدا کرد. بنابراین نمی توان الگوریتمی با ضریب $\frac{96}{95}$ بهینه پیدا کرد. بهترین الگوریتم تقریبی برای مسئله درخت اشتاینر یک الگورتیم با ضریب تقریب 1.39-تقریب است. اگر شرط نامساوی مثلثی روی گراف برقرار باشد آنگاه مسئله درستنمایی بیشینه اجدادی همان مسئله درخت اشتاینر است بنابراین یک الگوریتم با ضریب تقریب 1.39-تقریب برای مسئله بیشینه درستنمایی اجدادی وجود دارد. آیا با الگوریتم 1.39-تقریب که برای مسئله درخت اشتاینر وجود دارد می‌توان مسئله حداکثر صرفه‌جویی را حل کرد؟ آیا مشکلی وجود دارد؟ مشکلی که وجود دارد این است که اندازه گراف ما خیلی بزرگ است زیرا 2^n راس دارد. از طرفی الگوریتم درخت اشتاینر گراف را به عنوان ورودی می‌گیرد و یک زیردرخت همبند با راس‌های ترمینال با هزینه کمینه را به عنوان خروجی برمی‌گرداند. بنابراین اگر گراف با 2^n راس به این الگوریتم داده شود در نتیجه زمان اجرای این الگوریتم چندجمله‌ای نخواهد بود. بنابراین ما نیاز داریم بدون اینکه گراف ورودی را بسازیم الگوریتم درخت اشتاینر را اجرا کنیم. برای این

کار الگوریتم‌های تقریبی مناسبی وجود دارد که بدون اینکه کل گراف ورودی را در نظر بگیرد درخت اشتاینر را به عنوان خروجی بدهد. یکی از این الگوریتم‌ها الگوریتم برمن[۴]، است.

۶.۲ الگوریتم درخت Berman

این الگوریتم راس‌های ترمینال را به عنوان ورودی می‌گیرد و یک درخت پوشای کمینه **T** بین راس‌های ترمینال می‌سازد. سپس به ازای هر زیرمجموعه k-عضوی از راس‌های ترمینال، به ازای هر توپولوژی برای این k-راس، بهترین درخت (کم‌هزینه‌ترین درخت) با هر یک از این توپولوژی‌ها را با استفاده از الگوریتم سنکوف[۵]، پیدا می‌کند. سپس بررسی می کند که آیا اضافه کردن این درخت می تواند کمکی کند؟ اگر این طور بود آن درخت را به پشته اضافه می‌کند و **T** را به‌روزرسانی می‌کند. سپس در انتهای کار درخت‌های اضافه شده از انتها را یکی یکی به **T** اضافه می‌کند، بنابراین درخت حاصل یک درخت اشتاینر است.



شکل ۶: مراحل الگوریتم درخت Berman

- زمان اجرا: اگر تعداد ورودی‌ها (ترمینال‌ها) **M** و تعداد اعضای زیرمجموعه **K** و طول راس‌ها **N** باشد در نتیجه زمان اجرای الگوریتم Belman برابر است با:

$$O(M^k N)$$

۳ ارجاع و منابع

[AV66] Alon N, Chor B, Pardi F, Rapoport A. Approximate maximum parsimony and ancestral maximum likelihood. IEEE/ACM Trans Comput Biol Bioinform. 2010 Jan-Mar;7(1):183-7.

[۴] Berman
[۵] Sankoff