# فصل ۱

# روش پیشنهادی

## ۱.۱ مقدمه

پس از آشنایی با روش‌های پیشین که برای حل مسئله مشابه مورد استفاده قرار گرفته‌اند، حال می‌توانیم به معرفی و تشریح روش‌های پیشنهادی خود برای حل مسئله پیش رو بپردازیم. در این فصل ابتدا داده‌های ورودی مسئله را همراه با فرضیات در نظر گرفته شده بیان می‌کنیم و پس از آن دو روش پیشنهادی متفاوت را بیان خواهیم نمود. در روش اول که به رویکردهای پیشین نزدیک‌تر است با تغییری از جنس روش‌های نوین در مراحل میانی به یک روش جدید می‌رسیم که به علت افزایش سرعت همگرایی می‌توان فرض و داده‌های جدیدی را از طریق حذف و تغییر تعداد کپی[۱] به آن افزود و پاسخ گرفت. اما روش دوم کاملا متفاوت بوده و با رویکردی جدید در حوزه یادگیری ماشین همراه است که به کمک یادگیری تقویتی به حل مسئله مورد نظر می‌پردازد.

## ۲.۱ معرفی دادگان ورودی

قبل از وارد شدن به بخش روش‌های پیشنهادی نیاز است تا دادگان ورودی را مشخص و معرفی نماییم. دادگان ورودی در این پایان‌نامه همگی به صورت فایل‌های خام اسکی[۲] هستند که حاوی اطلاعات جهش‌های ماتریس

[1] Copy number variation (CNV)
[2] Ascii

ژن-سلول (SNV) و اطلاعات مربوط به حذف و تغییر تعداد کپی هستند.

در ادامه جدول ۱.۱ را برای معرفی اندیس‌های بکار گرفته شده در روابط مربوط به روش پیشنهادی اول معرفی می‌نماییم.

جدول ۱.۱: اندیس‌های به کار رفته در روابط روش پیشنهادی اول

| | |
|---|---|
| $D$ | ماتریس داده نویزی در دسترس که مقادیر ۰ و ۱ در آن قرار دارد |
| $E$ | ماتریس داده حقیقی بدون نویز که به دنبال آن هستیم |
| $T$ | درخت فیلوژنی جهش‌ها |
| $\sigma$ | بردار انتصابات |
| $X_T$ | ماتریس متناظر درخت $T$ |
| $N$ | تعداد سلول‌های نمونه |
| $M$ | تعداد جهش‌ها |
| $\mathcal{N}$ | مجموعه سلول‌های متمایز از هم |
| $\alpha$ | نرخ خطای مثبت کاذب |
| $\beta$ | نرخ خطای منفی کاذب |

## ۳.۱ روش پیشنهادی برای مدیریت داده‌های از دست رفته

در ادامه این بخش به معرفی روش‌های پیشنهادی پرداخته خواهد شد اما در ابتدا به دلیل وجود داده‌های از دست رفته در پایگاه‌داده‌های مورد استفاده لازم است تا به بررسی و رویکردای برای حل این مشکل پرداخته شود و در ادامه پس از معرفی روش پیشنهادی برای مددریت این داده‌های از دست رفته، هر کدام از روش‌های پیشنهادی به تفضیل شرح داده شود.

همان‌گونه که در داده‌های حقیقی مشاهده شد در پایگاه داده‌های حقیقی ما با اطلاعات از دست رفته مواجه هستیم و به همین دلیل نیز سعی کردیم تا در پایگاه داده مجازی تولید شده نیز به مشابه داده‌های حقیقی، شامل اطلاعات از دست رفته باشد. در این بخش به رویکرد روش محاسبه استاتیک برای مدیریت این داده‌های از دست رفته می‌پردازیم و در بخش بعد به معرفی روشی برای بدست آوردن درخت فیلوژنی پرداخته خواهد شد. همان‌گونه که در ادامه بررسی خواهد شد، این اطلاعات از دست رفته در پایگاه داده‌های مختلف نرخ‌های متفاوتی دارد که تاثیر این تغییرات نیز در روشی پیشنهادی بررسی خواهد شد.

### ۱.۳.۱ روش محاسبه استاتیک

در این روش قصد داریم تا به یک‌باره بتوانیم مقادیر مناسب برای داده‌هایی که از دست رفته‌اند را تخمین بزنیم. در این روش باید توجه شود که ما لزوما به دنبال جایگذاری مقدار از دست رفته با مقدار درست واقعی نیستیم. اگرچه چنین بیانی در نگاه اول ممکن است تعجب‌آور باشد اما با دقت بیشتر متوجه خواهیم شد که ما در آینده برای خطاهای موجود در پایگاه داده مدل‌سازی‌های محدودی داریم. مدل‌هایی که بهترین آن‌ها نیز ممکن است با واقعیت نویز افزوده شده به دادگان متفاوت باشد. در نتیجه اگر مطمئن بودیم که تمام داده‌هایی که موجود می‌باشند بدون خطا هستند در آن صورت ما نیز به دنبال یافتن جایگذاری با مقدار واقعی بودیم اما در حال حاضر که درصدی از داده‌های در دسترس خود همراه با خطا می‌باشند، ما به دنبال جایگذاری‌ای هستیم که بتواند در مجموع با مدل‌سازی خطایی که در نظر می‌گیریم بیشترین سازگاری را داشته باشد کما اینکه ممکن است در حقیقت جایگزاری اشتباهی انجام داده باشیم. حال با توجه به توضیحی که بیان شد به تشریح این روش می‌پردازیم.

با توجه به فرض مدل مکان‌های بی‌نهایت می‌دانیم که جهش‌های اتفاق افتاده در والد در تمامی نسل‌های آینده باقی خواهد ماند. بنابرین اگر تمامی جهش‌های نمونه (سلول) $a$ در نمونه‌ای دیگر مانند $b$ قرار داشته باشد، بنابرین می‌توان نتیجه گرفت که $a$ یکی از اجداد $b$ خواهد بود. همین فرضیه هسته اصلی روش پیشنهادی درنظر گرفته شده را تشکیل می‌دهد. بنابرین اگر جهش $i$ در سلول $a$ از دست رفته است، با توجه به اینکه آن جهش در سلول $b$ چه وضعیتی دارد می‌توان تصمیم‌گیری کرد. اگر $b(i) = ۰$ باشد، در این صورت $a(i)$ حتما باید ۰ باشد وگرنه فرض اولیه مدل مکان‌های بی‌نهایت نقض خواهد شد. اما اگر $b(i) = ۱$ باشد، آنگاه نتیجه خاصی نمی‌توان گرفت و باید به دنبال نمونه والد $a$ یعنی نمونه $d$ باشیم. حال اگر $d(i) = ۱$ باشد، آنگاه $a(i)$ حتما باید ۱ باشد. اما اگر $d(i) = ۰$ بود آنگاه انتخاب هر مقداری برای $a(i)$ تقریبا آزاد خواهد بود زیرا با فرض اولیه تناقضی ندارد و اینکه ساختار فیلوژنی را تغییر نمی‌دهد. اما از آنجایی که خود داده‌های در دسترس شامل خطا می‌باشند و هر نمونه‌ای که حاوی اطلاعات از دست رفته است لزوما یک نواده یا یک والد ندارد، مجموعه‌ای از سلول‌های فرزند یا والد خواهند بود که متناسب با پارمترهای خطایی که در نظر می‌گیریم و فاصله ژنی‌ای که دارند می‌توانند در تصمیم‌گیری تاثیرگزار باشند. صورت دقیق‌تر توضیحات داده شده را می‌توان به صورت فرمولی که در ادامه آمده است به نمایش درآورد.

در ابتدا تابعی به نام $F_s(D_{ij})$ تعریف می‌کنیم که به نوعی با توجه به ارزشی که به سلول‌های نواده شده از سلول

$j$ می‌دهد سعی دارد تا اطمینان ۰ بودن داده از دست رفته $D_{ij}$ را بیان کند.

برای محاسبه این تابع می‌دانیم که ابتدا سلول‌های مختلف با توجه به احتمال نواده بودنشان باید رتبه‌بندی شوند و وزن بگیرند. پس از آن هر سلول متناسب با ارزش تاثیرگزاری خود می‌تواند در مورد جایگاه جهش $i$ برای سلول $j$ نظر دهد.

$$F_s(D_{ij}) = \sum_{n \in \mathcal{N}} (۱ - D_{mj}) \prod_{m=۱}^{M} W(D_{mn}, D_{mj}) \tag{۱.۱}$$

در فرمول ۱.۱ مجموعه $\mathcal{N}$ برابر با مجموعه سلول‌های متمایز از هم است. زیرا که در بسیاری از پایگاه‌داده‌ها از یک نمونه سلول ممکن است چندین نمونه وجود داشته باشد که وجود آن‌ها باعث بایس در محاسبات ما خواهد شد. همچنین تابع $W_s(c, p)$ به ارزش‌دهی جهش $c$ در برابر $p$ به عنوان نواده بودن می‌پردازد که در فرمول ۲.۱ تعریف شده است.

$$W(c,p) = \begin{cases} ۱ & \text{if} \quad c = ۱, p = ۱ \\ ۱ - \xi & \text{if} \quad c = ۱, p = ۰ \\ ۰ & \text{if} \quad c = ۰, p = ۱ \\ ۱ & \text{if} \quad c = ۰, p = ۰ \end{cases} \tag{۲.۱}$$

مقدار $\xi$ عددی بین $(۰, ۱)$ است که پارامتری در جهت میزان ارزش‌دهی به نوادگان با فواصل مختلف می‌باشد. هرچه این عدد بزرگتر باشد به معنی کم‌ارزش‌تر شدن نوادگان با فواصل بیشتر است و بلاعکس.

به همین صورت برای اولاد سلول $j$ نیز می‌توان مشابه حالت قبل عمل کرد که روابط آن به صورت فرمول ۳.۱ خواهد شد.

$$F_a(D_{ij}) = \sum_{n \in \mathcal{N}} D_{mj} \prod_{m=۱}^{M} W(D_{mj}, D_{mn}) \tag{۳.۱}$$

حال دو نکته در استفاده از روابط بالا باقی خواهد ماند.

نکته اول وجود داده‌های دیگر از دست رفته در محاسبه توابع است که به دو صورت می‌توان با آن‌ها برخورد نمود.

رویکرد اول این است که در آن‌جایگاه ژنی از محاسبه آن خود داری شود و رویکرد دوم استفاده از از مقدار ۰/۵ یا فراوانی نسبی آن جهش در محسبات است که ما رویکرد اول را در این گزارش استفاده خواهیم کرد.
نکته دوم وجود خطا در داده‌هاست. برای مدیریت این مشکل می‌توان با مدل‌سازی خطا که به صورت فرمول ۴.۱ بیان می‌شود، برخورد کرد.

$$\begin{aligned} P(D_{ij}=1|E_{ij}=0)=\alpha, \qquad & P(D_{ij}=0|E_{ij}=0)=1-\alpha \\ P(D_{ij}=0|E_{ij}=1)=\beta, \qquad & P(D_{ij}=1|E_{ij}=1)=1-\beta \end{aligned} \tag{۴.۱}$$

پس از تعریف مدل‌سازی خطا می‌توان روابط قبلی را مجددا به صورتی که در ادامه آمده است بازنویسی کرد.

$$W_e(c,p)=\sum_{i,j\in\{0,1\}} P(c|E_c=i)P(p|E_p=j)W(i,j) \tag{۵.۱}$$

که در این صورت توابع $F_p$ و $F_a$ نیز به صورت زیر همراه با مدل‌سازی خطا بازتعریف خواهند شد.

$$\begin{aligned} \hat{F}_s(D_{ij}) &= \sum_{n\in\mathcal{N}}[1-D_{mj}(1-\alpha)]\prod_{m=1}^{M} W_e(D_{mn},D_{mj}) \\ \hat{F}_a(D_{ij}) &= \sum_{n\in\mathcal{N}} D_{mj}(1-\beta)\prod_{m=1}^{M} W_e(D_{mj},D_{mn}) \end{aligned} \tag{۶.۱}$$

حال پس از محاسبه مقادیر $\hat{F}_s$ و $\hat{F}_a$ می‌توان در مورد داده نامعلوم $D_{ij}$ به صورت فرمول ۷.۱ تصمیم گرفت.

$$D_{ij}=\begin{cases} 0 & \text{if} \quad \hat{F}_s \geq \hat{F}_a \\ 1 & \text{if} \quad \hat{F}_s < \hat{F}_a \end{cases} \tag{۷.۱}$$

همچنین با کمی دقت در فرمول‌بندی انجام شده اگر برای تمام $i,j$های ماتریس $D$ این مقادیر توابع $\hat{F}$ محاسبه شوند، خود می‌توانند معیاری برای ارزیابی پایگاه‌داده در دسترس و احتمال درستی فرض مدل مکان‌های بی‌نهایت باشند.

#### ۱.۱.۳.۱ تصادفی

پر کردن کاملا تصادفی میس‌ها. در این روش به صورت تصادفی مقادیر از دست رفته را مقدار دهی می‌کنیم. تنها نکته‌ای که در این روش وجود دارد این است که نباید این پرکردن تصادفی داده‌های از دست رفته باعث شود تا پارامترهای مدل‌سازی‌ای که از قبل در نظر گرفته بودیم با این روش نادقیق شوند.

## ۴.۱ روش پیشنهادی اول (درخت‌بازی)

### ۱.۴.۱ پیش‌پردازش

قبل از شروع باید بر روی داده‌ها یک پیش‌پردازش اعمال کنیم که وابسته به سیاست درنظر گفته شده می‌تواند باعث تغییر در پاسخ نهایی نیز شود. به این منظور داده‌هایی که miss شده‌اند با روش‌های زیر می‌تواند برای ورود به مرحله بعد تخمین زده شود.

جدول ۲.۱: پارامترهای مدل ریاضی

| | |
|---|---|
| $t_{ik}$ | زمان خدمت‌دهی به بیمار در مرحله $k$ام |
| $\tilde{t}_{ik}$ | زمان فاری خدمت‌دهی به بیمار در محله $k$ام |
| $t^p_{ik}$ | مقدار بدبینانه (حداکثر) برای زمان خدمت‌دهی به بیمار در مرحله $k$ام |
| $t^m_{ik}$ | محتمل‌ترین مقدار برای زمان خدمت‌دهی به بیمار در مرحله $k$ام |
| $t^o_{ik}$ | مقدار خوشبینانه (حداقل) برای زمان خدمت‌دهی به بیمار در مرحله $k$ام |

جدول ۳.۱: متغیرهای مدل ریاضی

| | |
|---|---|
| $X_{ild_k}$ | متغیر صفر-یک تخصیص بیمار به تخت/اتاق عمل |
| $S_{ild_k}$ | زمان شروع خدمت‌دهی به بیمار |
| $Y_{ijkl_k}$ | متغیر صفر-یک توالی بیماران |
| $V_{ni}$ | متغیر صفر-یک تخصیص جراح به بیمار |

# فصل ۲

# نتایج تجربی

## ۱.۲ پایگاه داده‌های ورودی

قبل از اینکه وارد روش پیشنهادی شویم به تشریح وردی‌های مسئله و داده‌هایی که مورد استفاده قرار خواهیم داد می‌پردازیم. داده‌های ورودی برابر ماتریس $D_{m\times n}$ می‌باشد که بعد اول $M$ برابر با ژن‌ها و بعد دوم $N$ برابر سلول‌های نمونه‌برداری شده می‌باشد. در هر خانه $d_{i,j}$ یک بردار داده قرار دارد که حاوی اطلاعات ژن $j$ در سلول $i$ می‌باشد.

### ۱.۱.۲ پایگاه داده مصنوعی [۱]

با توجه به این نکته که از درخت فیلوژنی حقیقی[۲] داده‌های حقیقی موجود اطلاعی نداریم، به سراغ ساخت پایگاه داده مصنوعی می‌رویم. با استفاده از این پایگاه داده مصنوعی می‌توانیم در مورد روش‌هایی که در ادامه بیان خواهیم کرد یک معیار ارزیابی نسبتا مناسبی داشته باشیم و تا حدودی از مشکلات روش‌های پیشنهادی آگاه شویم و به تصحیح آن بپردازیم. برای ساخت پایگاه داده مصنوعی که همان ماتریس ورودی $D_{m\times n}$ می‌باشد، از دو روش مختلف با دو فرض مختلف استفاده خواهیم کرد که در ادامه به تشریح هر کدام خواهیم پرداخت.
برای ایجاد پایگاه داده در این حالت ابتدا درختی تصادفی با پارامترهای $\zeta, n$ ایجاد می‌کنیم که $n$ تعداد ژن‌ها

[1] Synthetic Dataset
[2] Ground-truth Phylogeny Tree

(جهش‌ها) بوده و $\zeta$ عددی در بازه $(۰, \infty)$ است که یک پارامتر کنترلی است که وظیفه‌اش کنترل کلی تعداد نسل‌های مختلف را از یک جمعیت در درخت فیلوژنی می‌باشد. حال برای تولید پایگاه داده مصنوعی به ترتیب سه گام زیر باید انجام شود.

- ایجاد یک درخت فیلوژنی تصادفی
- تبدیل درخت فیلوژنی به ماتریس اطلاعات سلول-ژن $(E)$
- اضافه کردن نویز به ماتریس $E$ و تبدیل آن به ماتریس نویزی $D$

در ادامه هر بخش به صورت جداگانه به تفضیل شرح داده خواهد شد.

#### ۱.۱.۱.۲ ساخت درخت تصادفی

برای ساخت درخت تصادفی از دو روش مختلف استفاده شده است که هرکدام جداگانه توضیح داده شده است.

**روش اول: با استفاده از درخت تصادفی دودویی ژنولوژی**[۳]

در این روش همان‌گونه که از نام آن مشخص است با استفاده از درخت تصادفی دودویی ژنولوژی به ساخت ماتریس داده ورودی مسله می‌پردازیم که برای ساخت این دادگان از فرض‌های که در ادامه آمده است استفاده خواهیم کرد.

در مرحله اول که ساخت درخت است به این صورت عمل میکنیم که به تعداد $n$ گونه (سلول) در نظر می‌گیریم. سپس به ترتیب مراحل زیر را انجام می‌دهیم تا به درخت تصادفی مورد نظر برسیم.

- به هر کدام از $n$ گونه متمایز در ابتدا وزن $w_i = ۱$ را اختصاص می‌دهیم که متناسب با احتمال انتخاب هر گونه در مراحل بعدی خواهد بود.
- برای هر گونه $i$ تابع جرم احتمال را در ادامه به صورت $F_i = \frac{w_i}{\sum_{i=۱}^{n} w_i}$ در نظر میگیریم
- با استفاده از $F$ دو گونه متمایز $u, v$ را انتخاب می‌کنیم و به هم متصل می‌کنیم

[3] Random Binary Genealogical Tree

- به جای دو گونه $u, v$ یک گونه جدید $uv$ با وزن $w_{uv} = \frac{w_u + w_v}{\sqrt[۴]{\zeta}}$ را قرار می‌دهیم.

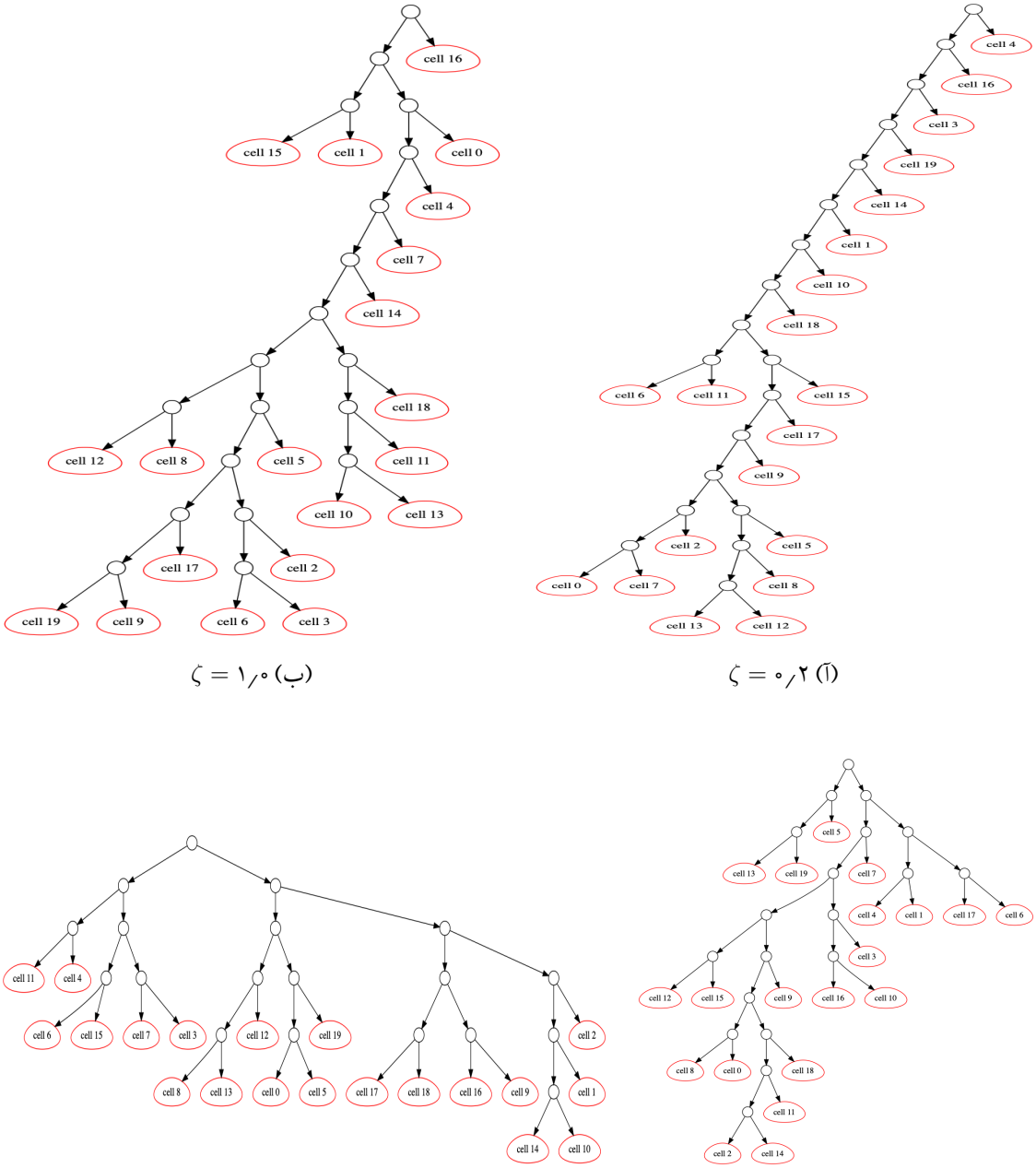
- تعداد گونه‌ها یک واحد کم شده است. بررسی می‌کنیم اگر تعداد گونه‌های باقی‌مانده از ۲ کمتر باشد درخت تصادفی ساخته شده است و پایان کار است. در غیر این صورت به مرحله اول بازمی‌گردیم.

پارامتر $\zeta$ به گونه‌ای کنترل‌کننده میزان ناپایداری در طی نسل‌ها می‌باشد. بطوریکه نمونه‌ای از نتایج مقادیر مختلف آن برای $n = ۲۰$ در شکل ۱.۲ آورده شده است. پس از ساخت درخت تصادفی به سراغ مرحله بعد یعنی تبدیل درخت به ماتریس ژن–سلول $E$ می‌رویم.
در ادامه با توجه به اینکه تعداد دلخواه جهش‌ها چه عددی بوده است یکی از گام‌های زیر را برمی‌داریم.

- اگر تعداد جهش‌ها $M > N$ بوده باشد در آن صورت به صورت تصادفی به تعداد دفعات اختلاف یکی از انشعاب‌ها در درخت را به صورت تصادفی انتخاب کرده و آن جهش اضافه شده را تا تمامی نوادگان پیش خواهیم برد.

- اگر تعداد جهش‌ها $M < N$ بوده باشد آنگاه مجددا به اندازه تعداد اختلاف انشعاب‌هایی را انتخاب کرده و این بار جهش در آن انشعاب را تا تمامی نوادگان حذف می‌کنیم.

به این ترتیب تمامی سلول‌ها را با تعداد جهش‌های انتخابی خواهیم داشت. در نهایت برای اخرین تغییر در جهش‌ها می‌توان یک گام دیگر برداشت که آن تولید یه عدد تصادفی کوچکتر از $\frac{M}{۲}$ است که به آن تعداد می‌توان جهش‌های موجود را از انشعابی برداشت و بر روی انشعابی دیگر قرار داد. با این کار ممکن است تعداد جهش‌ها در انشعاب‌های مختلف تغییر کند و چه بسا به مدل‌های واقعی نزدیکتر شود که البته در این پایان‌نامه از گام آخر صرف‌نظر کرده‌ایم.
حال کار ما با پخش تصادفی جهش‌ها در پایگاه‌داده مجازی پایان یافته است. تا به اینجا ما در فرض خود از هر نمونه جمعیت مختلف یک سلول داشته‌ایم. اما در بعضی مواقع در پایگاه داده‌های واقعی ممکن است از یک جمعیت بیش از یک نمونه وجود داشته باشد که البته این امر لزوما درست نیست به این دلیل که بعد از افزوده شدن نویز به داده‌ها ممکن است برخی سلول‌ها جهش‌هایشان مشابه هم شود. اما به هرحال اگر چنین چیزی را بخواهیم که داشته باشیم با انتخاب تصادفی برخی سلول‌ها(برگ‌ها) در درخت و کپی کردن آن‌ها می‌توان به چنین مقصودی رسید.




(آ) $\zeta = ۰/۲$ (ب) $\zeta = ۱/۰$

(ج) $\zeta = ۸$ (د) $\zeta = ۱۰۰$

شکل ۱.۲: درخت فیلوژنی تصادفی تولید شده برای $n = ۲۰$ و $\zeta$های مختلف

### روش دوم: با استفاده از درخت تصادفی جهش‌های ژنی [۴]

این روش نیز تا حدود زیادی مشابه روش قبل است با این تفاوت که در اینجا به جای اینکه درخت تصادفی را

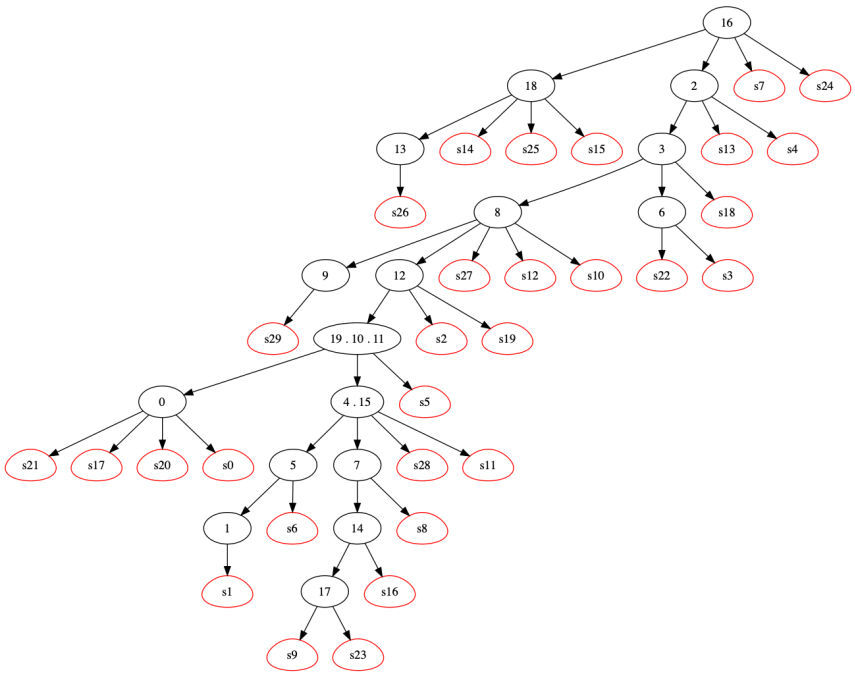
[4] Random Mutation History Tree

با توجه سلول‌ها از پایین به بالا بسازیم، ابتدا یک درخت تصادفی بدون در نظر گرفتن سلول‌ها ایجاد می‌کنیم و سپس به تخصیص جهش‌ها به آن می‌پردازیم و در نهایت برای آخرین مرحله به تعداد دلخواه سلول را به درخت اضافه کرده و درخت را تکمیل می‌کنیم. در گام اول به تعداد $M+1$ نود در نظر میگیریم. مشابه حالت قبل با طی مراحلی بکه در ادامه آمده است به ساختار یک درخت تصادفی می‌رسیم.

- به هر کدام از $m$ نود متمایز در ابتدا وزن $w_i = 1$ را اختصاص می‌دهیم که متناسب با روند حرکتی تومور به سمت آن جهش‌ها در مراحل بعدی خواهد بود.
- برای هر نود $i$ تابع جرم احتمال را در ادامه به صورت $F_i = \frac{w_i}{\sum_{i=1}^{n} w_i}$ بیان می‌شود در نظر می‌گیریم.
- با استفاده از $F$ دو نود متمایز $u, v$ را انتخاب می‌کنیم و به هم متصل می‌کنیم
- به جای دو گونه $u, v$ یک نود جدید $uv$ با وزن $w_{uv} = \frac{w_u + w_v}{\sqrt[4]{\zeta}}$ را قرار می‌دهیم.
- تعداد نودها یک واحد کم شده است. بررسی می‌کنیم اگر تعداد نودهای باقی‌مانده از ۲ کمتر باشد به مرحله بعد می‌رویم و در غیر این صورت به مرحله اول بازمی‌گردیم.
- در این مرحله تمامی برگ‌های درخت ساخته شده را حذف می‌کنیم و تنه باقی‌مانده را به عنوان درخت تصادفی جهش‌ها در نظر می‌گیریم.

پس از به پایان رسیدن مراحلی که بیان شد درخت تصادفی آماده است و حال نوبت به تخصیص دادن خود ژن‌ها به هرکدام از این نودهای درخت است. برای این منظور به هرکدام از $M$ نود یک ژن را به صورت تصادفی تخصیص می‌دهیم. پس از آن برای نهایی سازی درخت جهش‌ها از پارامتر دلخواه $A = \lfloor \gamma * (M-1) \rfloor$ استفاده می‌کنیم که $\gamma$ عددی بین $(0, 1)$ است و $A$ تعداد یال‌هایی است که در درخت باید برداشته شود و دو نود آن با یکدیگر ادغام شود. این کار باعث می‌شود تا در درخت جهش‌ها در برخی نودها به جای یک جهش چند جهش داشته باشیم که بتواند به مدل داده‌های واقعی نزدیکتر باشد.

پس از تکمیل درخت جهش‌ها نوبت قرار دادن نمونه‌هایی بر روی آن است. به همین منظور با فرض اینکه $N \geq M$ است. به تعداد $M$ تا از سلول‌ها را به هر کدام از نودهای درخت جهش به عنوان برگ‌های جدید اضافه می‌کنیم و برای $N-m$ سلول باقی مانده همین کار را این‌بار به صورت تصادفی انجام می‌دهیم. در نهایت درخت تصادفی جهش‌ها ساخته شده است که نمونه‌ای از آن را در شکل ۲.۲ قابل مشاهده است.




شکل ۲.۲: درخت جهش تصادفی با پارامترهای $N = ۳۰, M = ۲۰, \zeta = ۱, \gamma = ۰/۱۵$

### ۲.۱.۱.۲ تبدیل درخت به ماتریس ژن-سلول

با داشتن درخت (تولید شده با هرکدام از روش‌ها تفاوتی ندارد) در ادامه از فرض‌های مختلف در تولید ماتریس $E$ می‌توان استفاده کرد.

#### فرض مدل مکان‌های بی‌نهایت[۵]

در این حالت فرض می‌کنیم که هر جهش اتفاق افتاده در درخت فیلوژنی در تمامی نسل‌های پس از آن باقی می‌ماند و هیچ‌گاه از بین نمی‌رود. در چنین حالتی درخت حاصل از این روش درختی یکتا بوده که به نام درخت فیلوژنی کامل[۶] شناخته می‌شود.

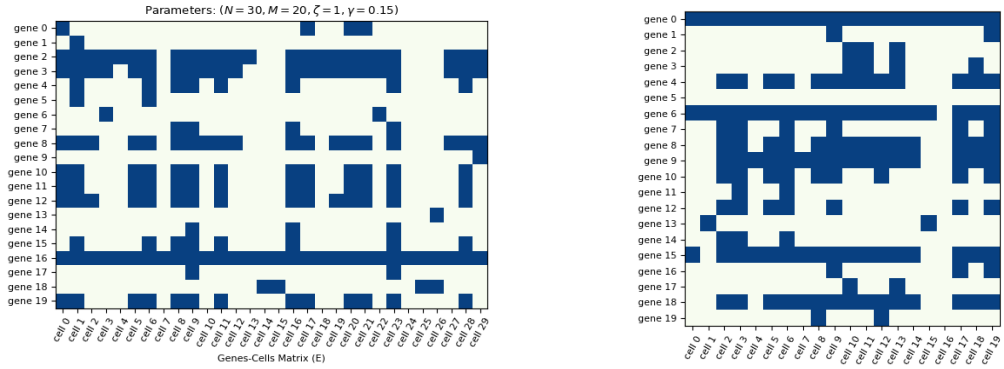
در این قسمت باید با استفاده از درخت تصادفی تولید بتوانیم ماتریس جهش‌ها را برای سلول‌های مختلف با فرض مکان‌های بی‌نهایت بدست آوریم. در ابتدا ماتریس $E$ را به ابعاد $M \times N$ ایجاد می‌کنیم و برای هر درایه $i, j$ در

[5] Infinite Site Models
[6] Perfect Phylogeny Tree

آن که $i$ شماره جهش و $j$ شماره سلول است به صورت فرمولی که در ادامه آمده است مقداردهی می‌کنیم.

$$E_{i,j} = \begin{cases} ۱ & \text{if} \quad \text{mutation } i \text{ is an ancestor of cell } j \\ ۰ & \text{o.w} \end{cases} \tag{۱.۲}$$

به این ترتیب با فرض مدل مکان‌های بی‌نهایت ماتریس بدون خطا $E$ را داریم که برای تصاویر دو روش درخت مرحله قبل در شکل ۳.۲ بدست آمده‌اند.




(آ) ماتریس درخت شکل ۱.۲.ب (ب) ماتریس درخت شکل ۲.۲

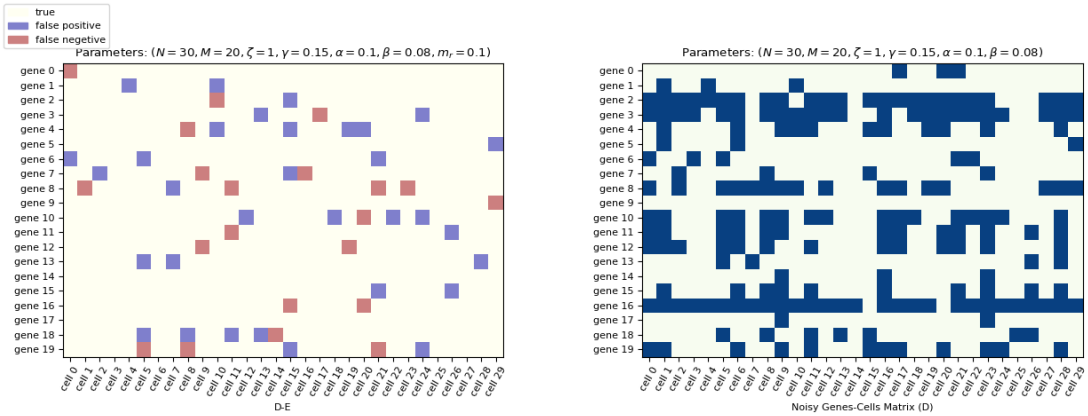
شکل ۳.۲: ماتریس‌های ژن-سلول ($E$) بدست آمده از درخت‌های تصادفی ساخته شده

### ۳.۱.۱.۲ اضافه کردن نویز به ماتریس ژن-جهش

برای قسمت نهایی آماده سازی پایگاه داده مجازی نیاز است تا به ماتریس $E$ با پارامتر $\Theta = (\alpha, \beta, m_r)$ نویز اضافه کنیم و آن را به ماتریس $D$ تبدیل کنیم که $\alpha = P(D_{ij} = ۱ | E_{ij} = ۰)$ و $\beta = P(D_{ij}|E_{ij})$ است و همچنین $m_r \in (۰, ۱)$ که نرخ داده‌های از دست رفته را مشخص می‌کند.

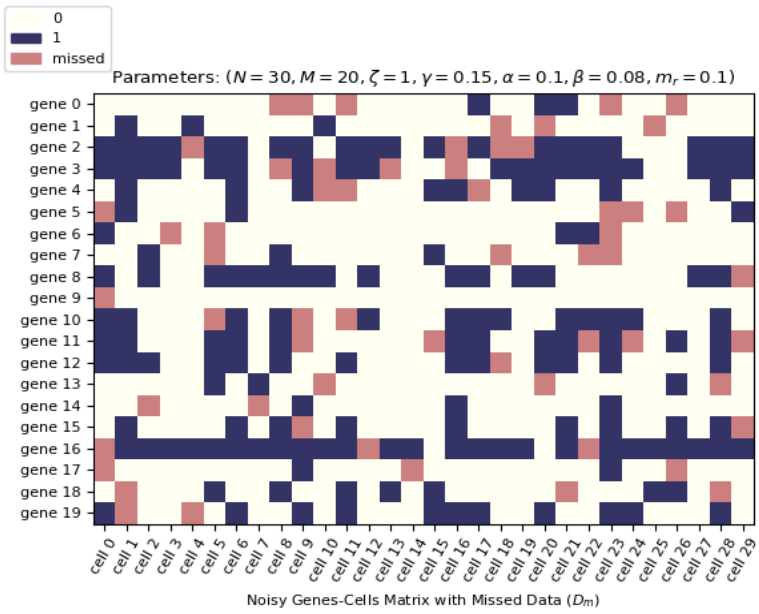
برای این منظور به ازای تمامی درایه‌های ۰ ماتریس $E$ هربار یک عدد تصادفی با توزیع یکنواخت بین $[۰, ۱)$ بوجود می‌آوریم و اگر عدد تولید شده کوچکتر از $\alpha$ بود آنگاه ان درایه در ماتریس $D$ را برابر با ۱ قرار می‌دهیم. به همین ترتیب مجددا این بار برای درایه‌های ۱ ماتریس $E$ این‌کار را تکرار می‌کنیم و اگر عدد تصادفی تولید شده کوچکتر از $\beta$ شد، درایه متناظر را در ماتریس $D$ برابر با ۰ قرار می‌دهیم.

پس از اتمام کار نوبت به اضافه کردن داده‌های از دست رفته است. برای این منظور با نرخ $m_r$ بعضی از درایه‌های ماتریس $D$ را برابر با ۲ قرار می‌دهیم که به منزله در دسترس نبودن اطلاعات است. نام ماتریس نهایی را که شامل داده‌های از دست رفته است $D_m$ می‌گزاریم. در ادامه تصاویر اضافه شدن نویز به ماتریس شکل ۲.۳.ب در شکل ۴.۲ آمده است.




(آ) ماتریس نویزی با $\alpha = ۰/۱, \beta = ۰/۰۸$

(ب) نویزی اضافه شده با پارمترهای $\alpha = ۰/۱, \beta = ۰/۰۸$

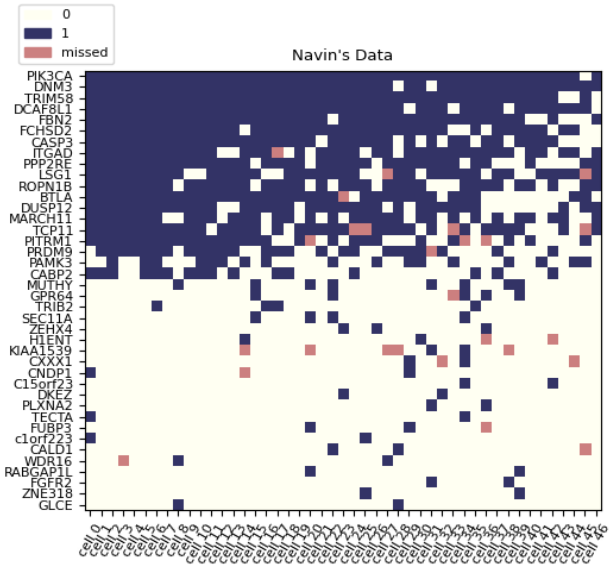



(ج) ماتریس نویزی به همراه داده‌های از دست رفته با پارامترهای $\alpha = ۰/۱, \beta = ۰/۰۸, m_r = ۰/۱$

شکل ۴.۲: ماتریس‌های ژن-سلول همراه با نویز و داده‌های از دست رفته شکل ۲.۳.ب که برای ورودی مسله آماده شده است.

### ۲.۱.۲ پایگاه داده حقیقی [۷]

به عنوان پایگاه داده حقیقی از پایگاه داده استفاده شده در مقاله SCITE به عنوان پایگاه داده حقیقی اصلی استفاده خواهیم کرد که ماتریس داده ورودی آن به صورت شکل ۵.۲ می‌باشد. همچنین پایگاه‌داده حقیقی Xu




شکل ۵.۲: داده‌های حقیقی Navin در مقاله SCITE

نیز که در مقاله SCITE مورد استفاده قرار گرفته است در شکل ۶.۲ آمده است.

[7] Real Dataset


شکل ۶.۲: داده‌های حقیقی Xu در مقاله SCITE

## ۲.۲ روش پیشنهادی بدست آوردن درخت فیلوژنی

پس از تخمین داده‌های از دست رفته، در این بخش به معرفی روش پیشنهادی برای یافتن درخت فیلوژنی می‌پردازیم. در روش‌های گذشته که رویکرد آن در ادامه بیان شده است به موفقیت نرسیدیم و این‌بار در نظر داریم تا با استفاده از یک درخت در ساختار شبکه ژن‌ها بتوانیم به یک درخت فیلوژنی مناسب دست یابیم.

- استفاده از شبکه ژن‌ها

۷ استفاده از یک گراف ابتدایی و سپس تغییر و هرس کردنش تا رسیدن به درخت جهش‌ها

۳ استفاده از یک درخت نمونه نابهینه و تغییر اتصالات تا رسیدن به درخت بهینه جهش‌ها

- استفاده از شبکه سلول‌ها

۷ استفاده از یک گراف سلول‌ها و بهینه‌کردن ارتباطات بین آن‌ها و سپس تبدیل آن به درخت فیلوژنی

در رویکرد اول ما از شبکه‌های ژنی استفاده خواهیم نمود. این شبکه‌ها نودهایی معادل با یک ژن متمایز را در نظر می‌گیرند. در گذشته با استفاده از شبکه‌ای کامل با وزن‌های متفاوت که بر حسب اطلاعات ورودی به الگوریتم تعیین می‌شد، متاسفانه به موفقیت خاصی نرسیدیم. همچنین مشابه همین رویکرد را در ساختار

شبکه‌های سلولی دنبال کردیم که مجددا پیشرفت قابل ملاحظه‌ای حاصل نشد. به همین جهت این‌بار در این گزارش با تغییری اساسی به دنبال یافتن روشی مناسب برای استنتاج درخت فیلوژنی می‌باشیم.

## ۱.۲.۲ استفاده از شبکه ژن‌ها برای یافتن درخت فیلوژنی

در این رویکرد با استفاده از شبکه‌ای که نودهایی معادل ژن‌ها داشته باشد سعی داریم تا به درخت فیلوژنی بهینه برسیم.

### ۱.۱.۲.۲ استفاده از یک درخت نمونه نابهینه و تغییر اتصالات تا رسیدن به درخت فیلوژنی بهینه

در این روش قصد داریم تا با شروع از یک درخت نمونه که در ابتدا به صورت تصادفی از اتصال ژن‌ها بوجود آمده است، به بهینه‌ترین درخت ممکن برسیم. این روش به صورت تکراروار با تغییر اتصالات درخت سعی در بدست آوردن درختی مطلوب‌تر دارد که شرایط و روابط تاثیرگزار در آن به تفضیل شرح داده خواهد شد. در واقع این روش پیشنهادی یک جست‌وجوی حریصانه می‌باشد که طی شرایطی می‌توان انتظار داشت که به پاسخ بهینه دست یافته شود. این روش به نام روش زنجیره مارکو مونت-کارلو[۸] شناخته می‌شود که در بسیاری از مقالات مرتبط نیز مورد استفاده قرار گرفته شده است.

برای شروع یک درخت تصادفی $T$ را با نودهایی معادل ژن‌های پایگاه داده ورودی در نظر می‌گیریم که در گام اول به صورت تصادفی ساخته شده است. در گام‌های بعدی یک نود $n_1$ را از درخت $T$ به صورت تصادفی انتخاب می‌کنیم. سپس زیردرخت با ریشه این نود را از درخت کم می‌کنیم. حال در درخت باقی‌مانده یک نود دیگر $n_2$ را به صورت تصادفی انتخاب می‌کنیم و آن زیر درخت قبلی با ریشه $n_1$ را به $n_2$ متصل می‌کنیم و درخت جدید را $T_n$ نام‌گزاری می‌کنیم. پس از آن با احتمال،

$$P = \min\left(1, \frac{Eng(T)}{Eng(T_n)}\right) \tag{۲.۲}$$

درخت جدید بدست آمده $T_n$ را به عنوان نتیجه این گام می‌پذیریم و در غیر این صورت درخت این گام نیز همان درخت سابق $T$ باقی خواهد ماند. در رابطه ۲.۲، تابع $Eng$ برای یک درخت در واقع انرژی آن درخت را محاسبه

[8] Markov Chain Monte Carlo

می‌کند و ما به دنبال پایدارترین درخت هستیم که کمترین انرژی را داشته باشد. تعریف این تابع برای یک درخت به این صورت است که با توجه به نمونه‌هایی که در دادگان ورودی $D$ قرار دارد و اینکه کدام ژن بالاتر یا پایین‌تر از دیگر ژن‌ها قرار دارد به درخت یک نمره انرژی منصوب می‌کند که به صورت فرمول ۳.۲ بیان می‌شود.

$$Eng(T) = ||E - \hat{E}|| \tag{۳.۲}$$

که در اینجا $\hat{E}$ ماتریس تخمین زده شده روش پیشنهادی با توجه به درخت نهایی بدست آمده خواهد بود. در واقع ماتریس $E$ همان ماتریس صحیح بدون خطا است که جهش‌های مختلف را به ازای سلول‌های مختلف مشخص می‌کند. هنگامی که ما درخت ساخته شده فرضی $T$ را داشته باشیم می‌توانیم در دو گام به $\hat{E}$ برسیم. توجه به این نکته ضروری است که اگر در واقعیت فرض ما که همان مکان بی‌نهایت بود کامل برقرار باشد و $E$ را داشته باشیم، حتما باید بتوانیم به درختی با $Eng(T) = ۰$ دست یابیم. اما از آنجایی که ما $D$ را به عنوانی از تخمین $E$ داریم بنابرین محاسبه خطای ما نیز تخمینی از خطای واقعی خواهد بود نه خود آن که در اصل به صورت فرمول ۴.۲ می‌شود.

$$Eng(T) \approx Err(T) = ||D - \hat{E}|| \tag{۴.۲}$$

می‌دانیم که هر نود از این درخت $T$ یک مکان برای اتصال سلولی می‌تواند باشد که در این صورت معنی آن اینگونه خواهد بود که سلول ضمیمه شده به آن نود تمام جهش‌های والد خود را داشته است. بنابرین در گام اول نیاز است تا هر مکان از درخت مشخص شود که چه نمونه‌هایی می‌تواند تولید نماید. این اطلاع توسط ماتریس $A$ مشخص می‌شود که به صورت زیر از روی درخت ساخته خواهد شد.

$$A_{i,j} = \begin{cases} ۱ & \text{اگر } i = j \text{ یا جهش } i \text{ والد جهش } j \text{ باشد} \\ ۰ & \text{در غیر این صورت} \end{cases} \tag{۵.۲}$$

حال در گام دوم کافی می‌توانیم با توجه به یک معیار بهترین انتخاب را برای ضمیمه کردن سلول‌ها (نمونه‌های) موجود به درخت داشته باشیم. به همین جهت در ماتریس $D$ که هر ستون آن برابر با نمایش یک سلول است، می‌تواند با هر ستون از ماتریس $A$ مقایسه شود و بهترین ستونی که از $A$ انتخاب شود برابر با جایگاه مناسب

ضمیمه شدن نمونه با مقداری خطا به درخت $T$ است. حال با توجه به اینکه فرض مکان‌های بی‌نهایت را داشتیم ماتریس $E$ را به صورت زیر می‌سازیم.

$$\hat{E}_{i,j} = A_{i,\sigma_j} \tag{۶.۲}$$

که $\sigma_i$ برابر با بهترین نود (ژن) برای اتصال نمونه بردار $d_j$ است که بهترین جایگاه به صورت فرمول زیر انتخاب می‌شود.

$$\begin{aligned}\sigma_j = \arg\max_{x\in[۱\to M]} \sum_{i=۱}^{M} \Big[ & \\ & A_{i,x}D_{i,j}(۱-\beta) + (۱-A_{i,x})(۱-D_{i,j})(۱-\alpha) + \\ & A_{i,x}(۱-D_{i,j})\beta + (۱-A_{i,x})D_{i,j}\alpha \\ \Big] &\end{aligned} \tag{۷.۲}$$

حال با داشتن ماتریس $\hat{E}$ می‌توان خطای درخت را محاسبه نمود و با هدایت mcmc طبق فرمول ۸.۲ به درخت بهینه $T_{op}$ رسید.

$$T_{op} = \min_{T\in \text{All possible } T} \Big( ||D - \hat{E}_T|| \Big) \tag{۸.۲}$$

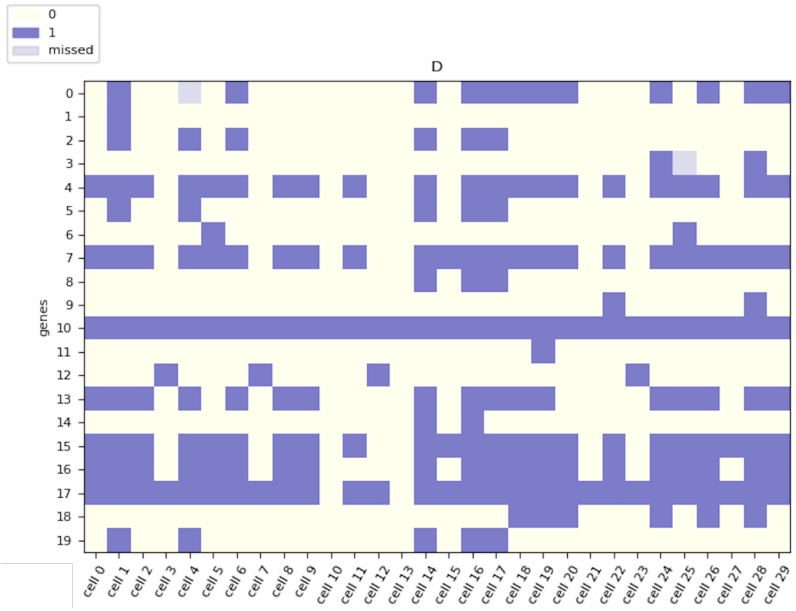
## ۳.۲ نتایج تجربی

در این بخش به نتایج بدست آمده برای روش پیشنهادی می‌پردازیم و برای هر دو داده مصنوعی و حقیقی نتایج بدست آمده را تحلیل خواهیم نمود.

### ۱.۳.۲ نتایج بر روی پایگاه داده مصنوعی

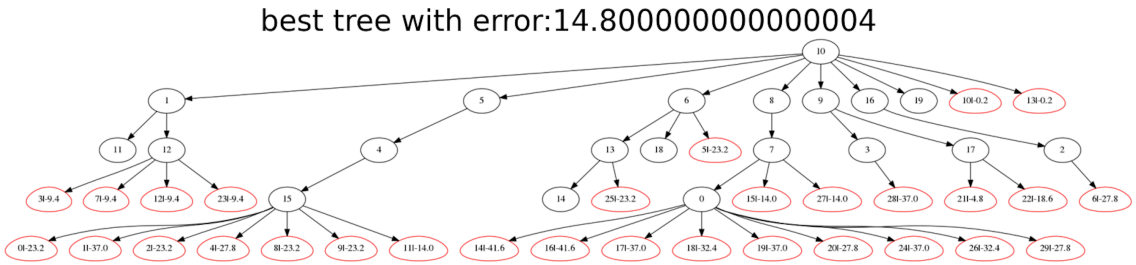
همان‌گونه که در بخش دوم توضیح داده شد با توجه به سختی دسترسی به پایگاه داده‌های حقیقی و اینکه در آن‌ها نیز حقیقت داده‌ها ($E$) وجود ندارد تصمیم به ایجاد پایگاه داده‌ای مصنوعی گرفته شد که با کمک آن بتوان ارزیابی مناسبی از روش پیشنهادی و میزان کارایی و مقاومت روش را نسبت به تغییر پارامترها سنجید.

فرض کنید ماتریس ورودی شکل ۷.۲ را در اختیار داریم و میخواهیم بهترین درخت فیلوژنی را برای آن بیابیم.




شکل ۷.۲: نمونه‌ای تصادفی از ماتریس ورودی $D$

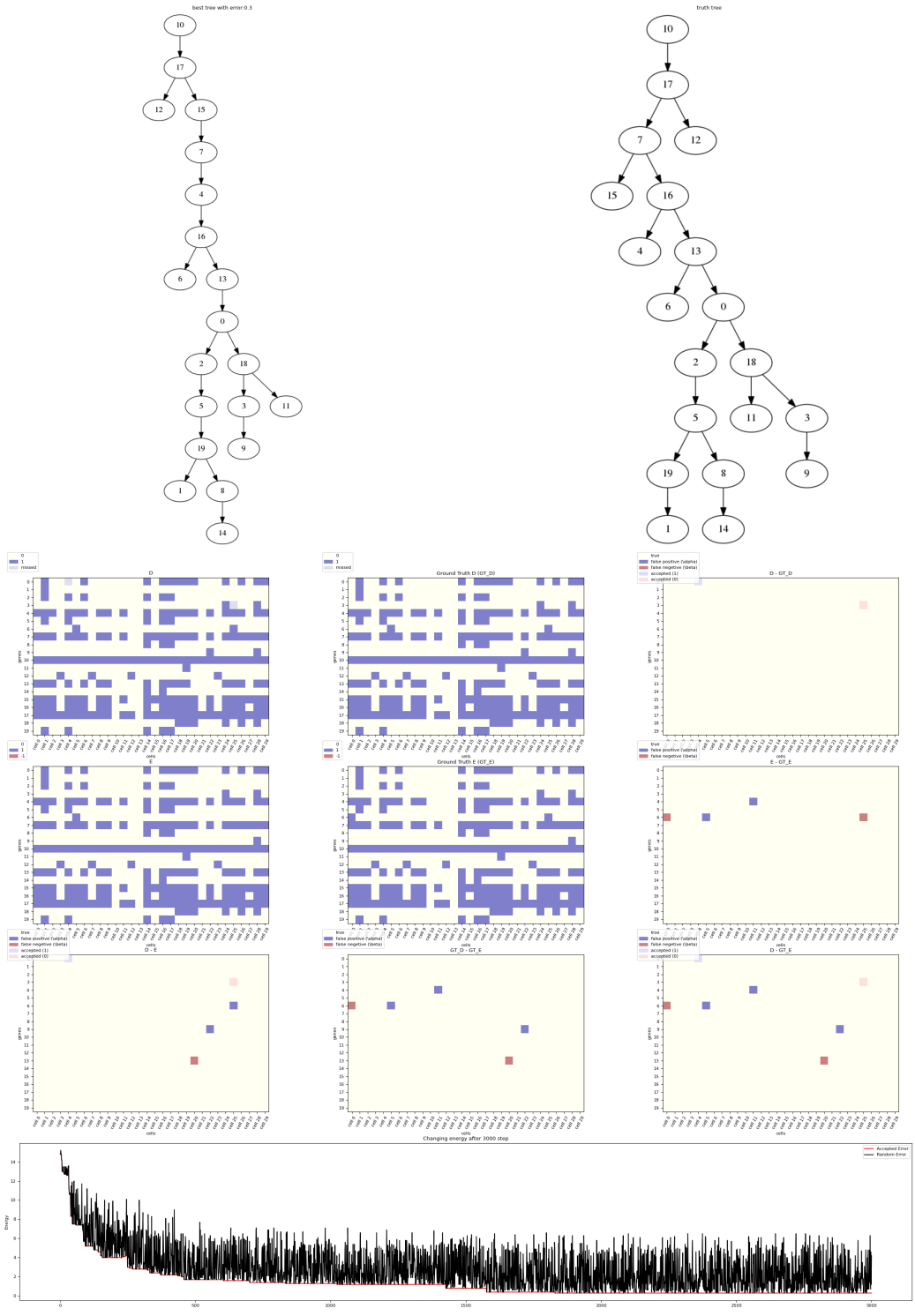
حال یک درخت تصادفی به صورت شکل ۸.۲ می‌سازیم. در درخت شکل ۸.۲ نمونه‌ها (سلول‌ها) با رنگ قرمز به درخت متصل شده‌اند که البته این ضمیمه بهترین ضمیمه ممکن است و میزان انرژی (خطای) هر ضمیمه نیز در کادر قرمز رنگ سلول‌ها به صورتی عددی منفی نوشته شده است.




شکل ۸.۲: درخت تصادفی ایجاد شده به عنوان درخت اولیه شکل ۷.۲

پس از این مرحله اگر ۳۰۰۰ گام MCMC را اجرا نماییم می‌توانیم نتیجه حاصله را در شکل ۹.۲ مشاهده کنیم. در این شکل دو درخت وجود دارد که درخت سمت راستی درخت حقیقی است که به دنبال آن بودیم و درخت سمت چپ بهترین درخت یافته شده است. همچنین در پایین شکل، ۹ ماتریس مشاهده می‌شود که ماتریس‌ها سمت راست و پایین به نوعی بیان‌کننده میزان خطا بین ۴ ماتریس سمت چپ بالا می‌باشند. در بالای هر ماتریس نام آن نوشته شده است و در نهایت در انتهای تصویر نیز روند کاهش خطا و تلاش‌های MCMC در گام‌های مختلف قابل مشاهده است. فقط نکته‌ای که وجود دارد این است که خطای نوشته شده در تصاویر برابر ۰/۱ مقیاس نوشته شده است.

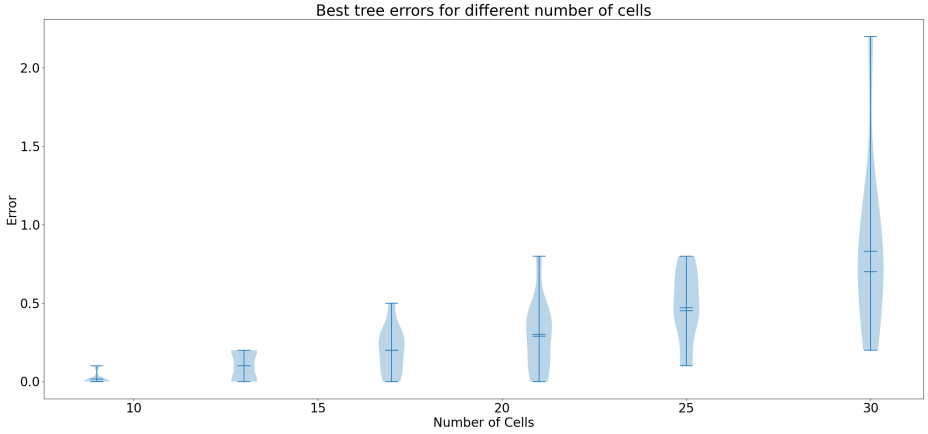
همان‌گونه که مشخص است در ماتریس $D$ دو داده از دست رفته وجود دارد که یکی از آن‌ها در حقیقت جهش یافته و دیگر خیر. اگر ما در محاسبات خود این دو داده را در محاسبه خطا در نظر نگیریم و با تغییر ۳ داده دیگر می‌توانیم به ماتریس $\hat{E}$ (در شکل به نام $E$ نوشته شده است) برسیم که معادل بهترین درخت بدست آمده است. که این یعنی ماتریس $D$ ما با ۵ تغییر بدست ما رسیده است. حال اگر حقیقت داده‌ها و درخت اصلی را مشاهده کنیم می‌بینیم که در آنجا نیز ۵ خطا وارد شده است که ۲ تای آن‌ها را درست کشف شده است. بنابرین الگوریتم بدون اطلاع از حقیقت توانسته با حداکثر ۵ خطا به یک درخت فیلوژنی مناسب دست بیابد که در ساختار نیز شباهات بسیار زیادی به حقیقت دارد. بنابرین روش پیشنهادی توانسته درخت فیلوژنی را با صحت $\frac{۲۰*۳۰-۵}{۲۰*۳۰} = ۰/۹۹۱\overline{۶}$ بازسازی کند که عددی قابل قبول می‌باشد.




شکل ۹.۲: نتیجه اجرای روش پیشنهادی برای ماتریس شکل ۷.۲

اما برای بررسی مناسب‌تر تعدادی تست را به ازای $M$ و $N$های مختلف اجرا نمودیم که به صورت خلاصه نتایج حاصل از آن در ادامه قابل مشاهده است.

در شکل ۱۰.۲ مقدار خطای درخت بهینه یافته شده قابل مشاهده است که نشان می‌دهد هر چه تعداد نمونه‌ها افزایش پیدا میکند و اندازه ماتریس ورودی بزرگ‌تر می‌شود، مقدار خطا نیز افزایش می‌یابد. در این اجرا تعداد جهش‌ها نیز عددی بین تعداد نمونه‌ها و نصف تعداد نمونه‌ها بوده است. حال برای اینکه متوجه شویم آیا این

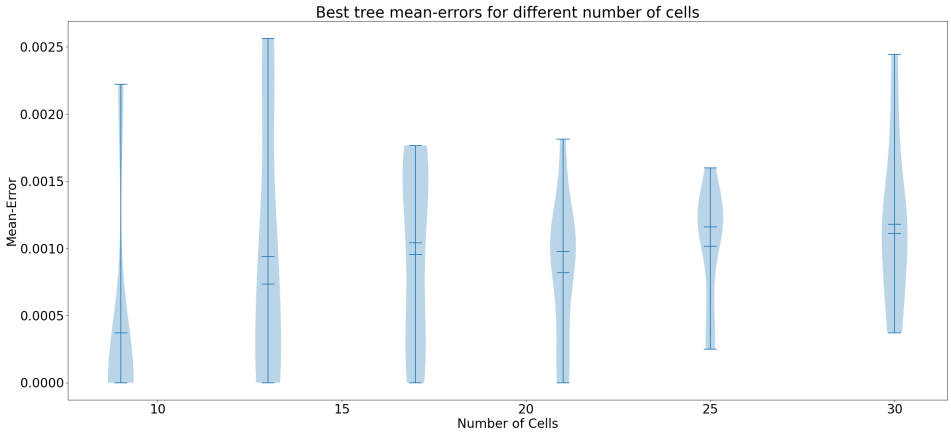



شکل ۱۰.۲: نتیجه اجرای روش پیشنهادی برای تعداد نمونه‌های مختلف

افزایش خطا بخاطر ضعف روش پیشنهادی است یا ماهیت داده‌های ورودی میزان خطا را در هر اجرا بر تعداد خانه‌های ماتریس $D$ تقسیم می‌کنیم که در آن صورت به نمودار شکل ۱۱.۲ می‌رسیم. در این نمودار جدید مشخص می‌شود که با افزایش اندازه ماتریس ورودی روش پیشنهادی سعی می‌کند تا خطا را به ازای هر داده کنترل کند که نشان از کارآمدی روش پیشنهادی می‌باشد.

### ۲.۳.۲ نتایج بر روی داده‌های حقیقی

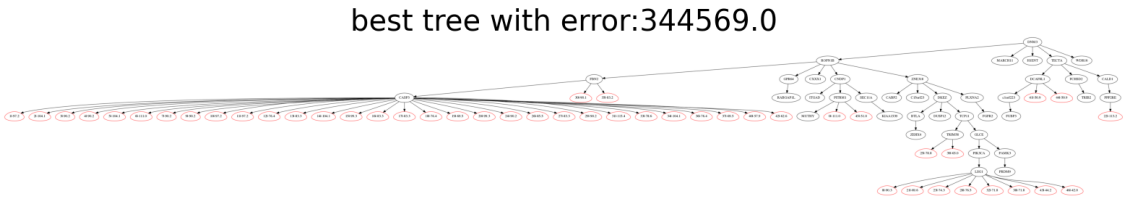
در این قسمت به ارائه گزارش و نتایج حاصل از روش‌های پیشنهادی با استفاده از داده‌های حقیقی برای بدست آوردن درخت فیلوژنی خواهیم پرداخت.




شکل ۱۱.۲: نتیجه اجرای روش پیشنهادی برای تعداد نمونه‌های مختلف

### ۱.۲.۳.۲ نتایج بهینه‌سازی درخت ژنی

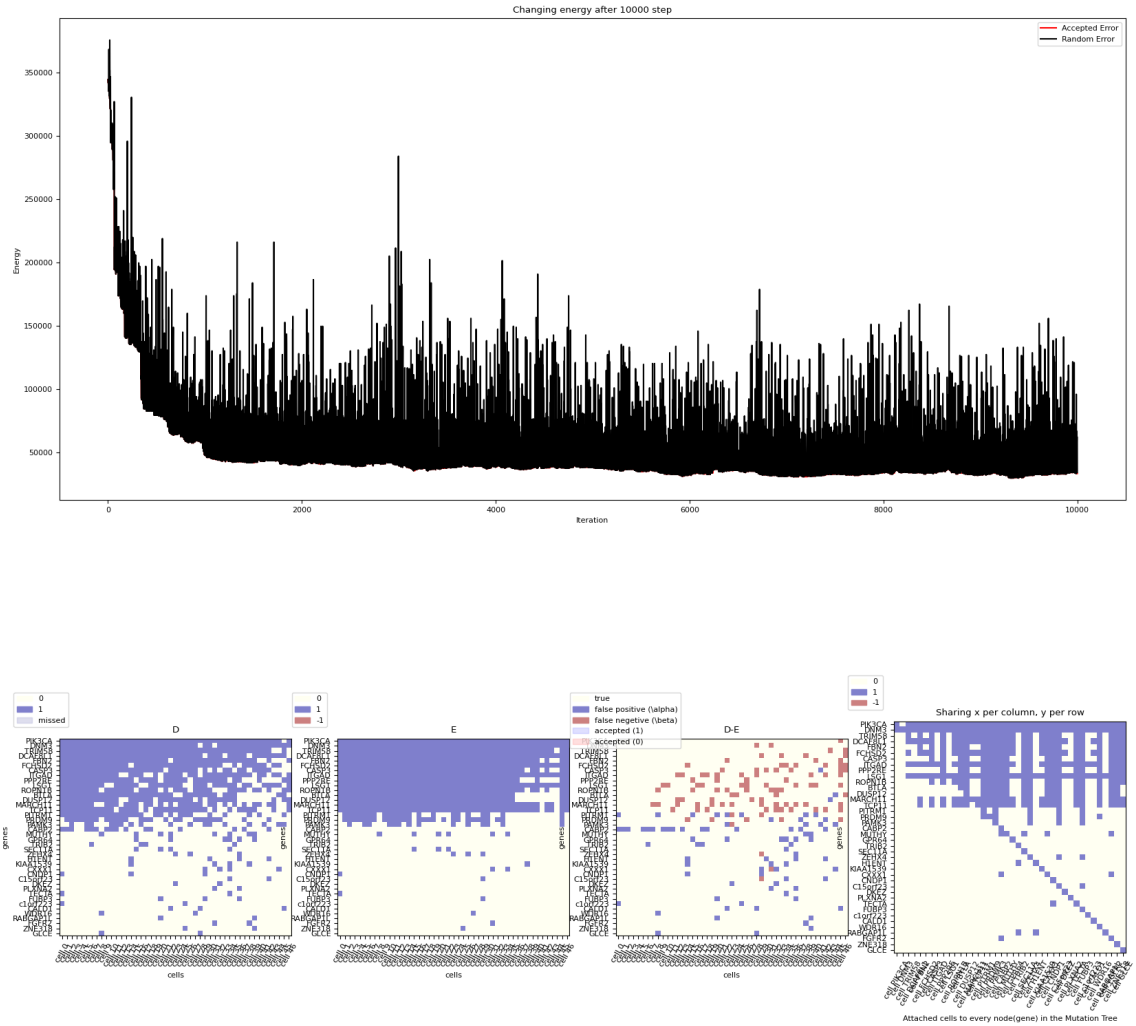
همان‌طور که در فصل قبل بیان شد، یکی از روش‌های بدست آوردن درخت فیلوژنی استفاده از یک درخت تصادفی نابهینه ژنی بود که طی تکرار گام‌هایی سعی در تغییر اتصالات و یافتن درخت بهینه داشت که بتواند روند صحیح تغییرات ژنی را در تومور مورد نظر نمایش دهد. نتیجه بدست آمده بر روی پایگاه داده حقیقی Navin به شرح زیر می‌باشد که عکس ۱۲.۲ درخت تصادفی اولیه الگوریتم را نشان می‌دهد که انرژی آن نیز بالای تصویر نوشته شده است.




شکل ۱۲.۲: درخت تصادفی اولیه

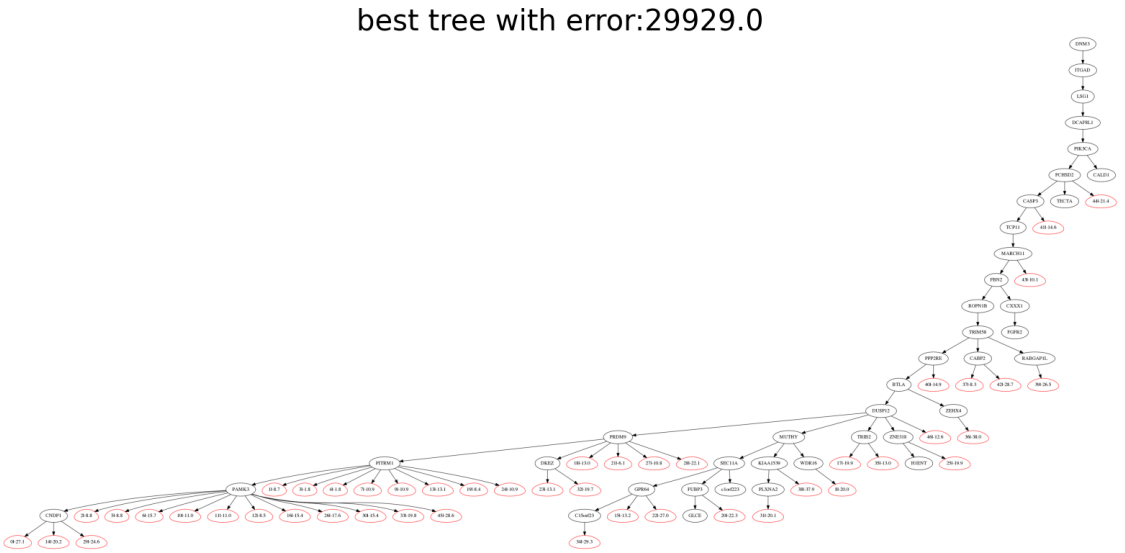
تصویر ۱۳.۲ نیز نمودار تغییر انرژی را طی گام‌های مختلف نمایش پیشنهادی مشخص می‌کند. در نهایت تصویر بهترین درخت یافته شده به همراه انرژی آن.

در نهایت برای مقایسه نیز تصویر درخت حاصله در مقاله اصلی SCITE را در شکل ۱۵.۲ نمایش داده شده




شکل ۱۳.۲: نمودار تغییر انرژی در طی گام‌های مختلف

است. همان‌طور که مشخص است مقدار انرژی بدست آمده برای خروجی الگوریتم پیشنهادی بهتر (کمتر) از انرژی درخت SCITE می‌باشد که دلیل بر بهینه‌تر بودن درخت روش پیشنهادی ارائه شده در این گزارش است.




شکل ۱۴.۲: بهترین درخت یافته‌شده و خروجی الگوریتم برای مقاله SCITE

## ۴.۲ گام‌های آتی

در ادامه برای تکمیل روش پیشنهادی در دو قسمت نیاز به بهبود وجود دارد.
قسمت اول مربوط به درخت اولیه است و قسمت دیگر مربوط به سرعت MCMC می‌باشد.

### ۱.۴.۲ بهبود در ساخت درخت اولیه

در حال حاضر ما درخت اولیه را به صورت تصادفی انتخاب می‌کنیم که می‌توان در این مرحله درخت اولیه را با استفاده از مفروضات مدل مکان‌های بی‌نهایت و با توجه به ماتریس ورودی بهبود بخشید. این کار باعث می‌شود تا شروع الگوریتم از نقطه بهتری باشد که در این صورت هم گام‌های لازم برای رسیدن به درخت بهینه می‌تواند کمتر شود و هم اینکه احتمال قرار گرفتن در نقاط اکسترمم نسبی را کاهش می‌دهیم.

### ۲.۴.۲ افزایش سرعت همگرایی MCMC

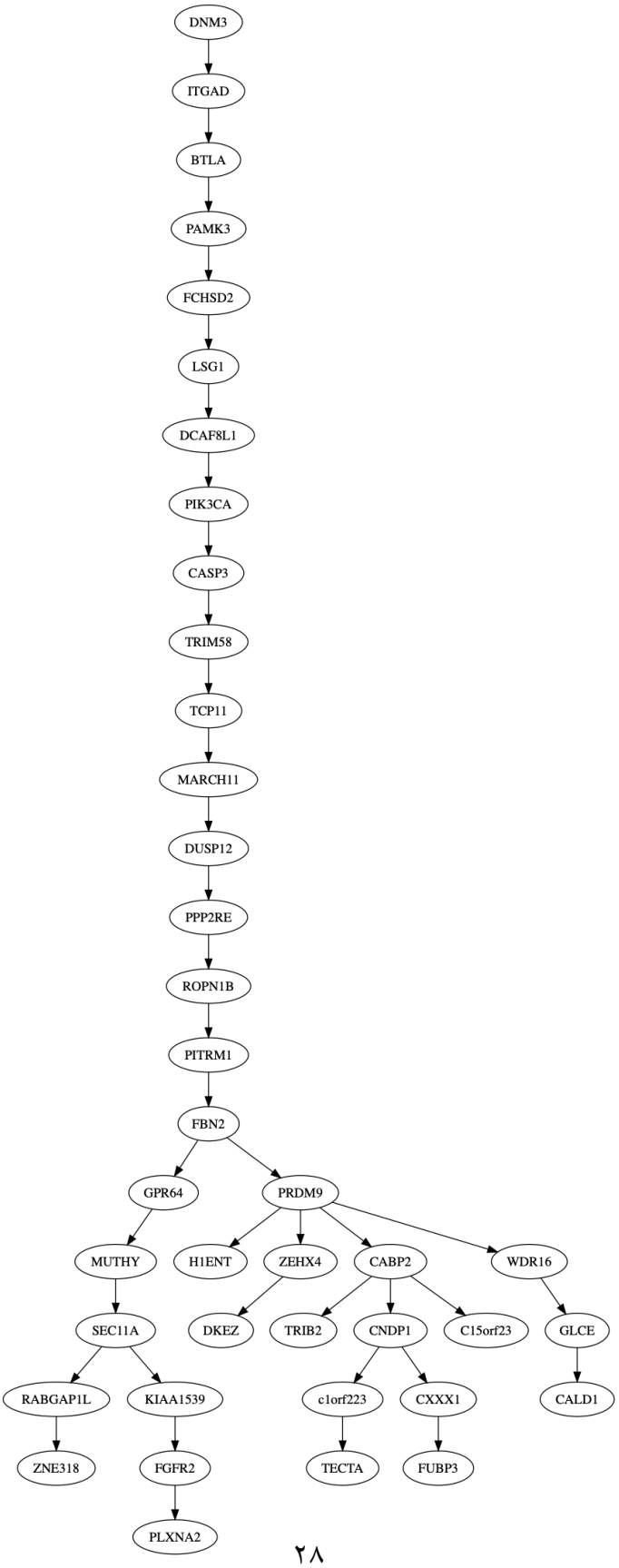
در این بخش نیاز است تا در دو قسمت روش پیشنهادی بهبود یابد.

### ۱.۲.۴.۲ تنوع در گام‌ها با استراتژی معقول

برای این بخش کاری که باید انجام شود این است که بتوان برای افزایش سرعت همگرایی از روش‌های مختلف در گام‌ها استفاده کرد. برای مثال در حال حاضر می‌توان از سه روش مختلف در هر گام استفاده نمود. روش اول تعویض دو نود در درخت می‌باشد. روش دوم جدایی یک زیر درخت و اتصال آن به محلی دیگر می‌باشد و در نهایت روش سوم تعویض دو زیر درخت با یکدیگر می‌باشد. با انتخاب یک استراتژی مناسب بین هرکدام از این روش‌ها در گام‌های مختلف احتمالا بتوان سرعت همگرایی را افزایش داد.

### ۲.۲.۴.۲ قرار دادن احتمال وزن‌دار به ازای هر انتخاب

در حال حاضر ما در هرکدام از روش‌های مختلف که در بخش قبل برای گام‌های MCMC بیان کردیم، انتخاب نودها را به صورت کاملا یکنواخت انجام می‌دهیم. در صورتی که احتمالا بتوان با تعریف فرمولی مناسب این احتمال انتخاب بین نودهای مختلف در درخت را از حالت یکنواخت خارج کرد و در نتیجه مجددا سرعت همگرایی الگوریتم را افزایش داد.






شکل ۱۵.۲: درخت بدست آمده در مقاله SCITE