دانشگاه صنعتی امیرکبیر
(پلیتکنیک تهران)
دانشکده مهندسی کامپیوتر

درس بیوانفورماتیک
تمرین پنج۵

امیرمهدی زرین‌نژاد
۹۷۳۱۰۸۷

نیم‌سال دوم ۱۴۰۰

سوال ۱)

الف) برای مجموعه رشته‌های پروتئینی زیر، یک موتیف به صورت عبارت منظم بنویسید.

```
AYGTTSKK
AYPTTSIK
AVHTTSIK
AYMTTSIK
AVZTTSIK
```

A-[YV]-X-T(2)-S-[IK]-K

ب) عبارت منظم مقابل با کدام یک از رشته‌های زیر exact match می‌شود. برای هر رشته دلیل بیاورید.

M-[TG]-X-{M}-A(2)-P-[YPC]

- MMTGAAPP:
  باتوجه به regex ارائه شده، رشته در ایندکس(لوکیشن) دوم باید T یا G باشد اما این رشته داده شده کاراکتر M دارد که در [TG] وجود ندارد و exact match نیست.

- MTTTAAPC:
  عبارت منظم داده‌شده با این رشته exact match می‌شود زیرا رشته با فرمت regex تطابق دارد:

  M T T T AA P C
  M-[TG]-X-{M}-A(2)-P-[YPC]

  لوکیشن اول باید M باشد که درست است. لوکیشن دوم می‌تواند T یا G باشد که T است و مطابقت دارد. لوکیشن سوم هرچیزی می‌تواند باشد. لوکیشن چهارم هرچیزی به جز M باید باشد که T است. لوکیشن پنجم و ششم باید A باشند که هستند. لوکیشن هفتم باید P باشد که هست و لوکیشن آخر هم می‌تواند Y یا P یا C باشد که C است و درست است.

- MGTMAAPP:
  در این رشته،کاراکتر چهارم M است درحالی که برای کاراکتر چهارم در regex فرمت {M} آمده که هرچیزی به جز M را معنی می‌دهد. پس این کاراکتر با عبارت منظم متناقض است و رد می‌شود.

- MTGAAPPY:
  این رشته نیز با فرمت داده شده مطابقت ندارد. برای اثباتش می‌توانیم از چپ شروع کنیم و مطابقت را بررسی کنیم:

  M T G A APPY
  M-[TG]-X-{M}-A(2)-P-[YPC]

  تا A اول مطابقت دارد اما در regex کاراکترهای ۵و۶ باید A باشند ولی در رشته داده شده فقط کاراکتر پنجم A است و کاراکتر ششم P است.

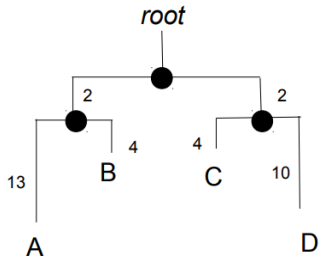
  ویا مثلا از راست بررسی کنیم:

  M T G A AP P Y
  M-[TG]-X-{M}-A(2)-P-[YPC]

  بازهم می‌بینیم که کاراکتر لوکیشن ۶ باید A باشد درحالی که P است.

سوال ۲)

به درخت حقیقی زیر که فاصله ۴ گونه از یکدیگر را نشان میدهد، دقت کنید. در این درخت فواصل هر دو گونه، از مجموعه فواصل شاخه‌های بین به دست می‌آید. به طور مثال فاصله گونه A تا C برابر با ۴+۲+۲+۱۳ = ۲۱ میباشد.



الف) ابتدا فواصل بین تمامی گونه‌ها را استخراج کرده و ماتریس فاصله را رسم کنید.

d(A, B) = 13 + 4 = 17
d(A, C) = 13 + 2 + 2 + 4 = 21
d(A, D) = 13 + 2 + 2 + 10 = 27
d(B, C) = 4 + 2 + 2 + 4 = 12
d(B, D) = 4 + 2 + 2 + 10 = 18
d(C, D) = 4 + 10 = 14

	A	**B**	**C**	**D**
A	-	17	21	27
B	17	-	12	18
C	21	12	-	14
D	27	18	14	-

ب) با استفاده از ماتریس فاصله بدست آمده در قسمت (الف)، به روش UPGMA درخت را رسم کنید.

B و C کوتاه‌ترین فاصله را دارند پس برای ادغام انتخاب می‌شوند:

$d(BC, A) = \frac{d(B,A)+d(C,A)}{2} = \frac{17+21}{2} = 19$

$d(BC, D) = \frac{d(B,D)+d(C,D)}{2} = \frac{18+14}{2} = 16$

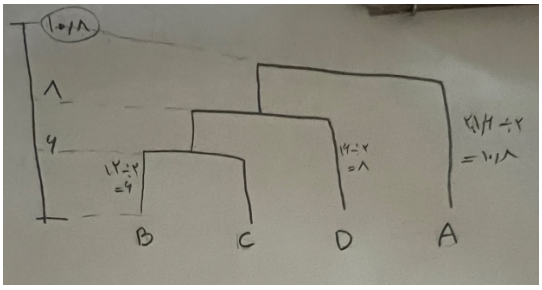
	BC	**A**	**D**
BC	-	19	16
A	19	-	27
D	16	27	-

حال BC و D کوتاه‌ترین فاصله را دارند و باهم ترکیب می‌شوند:

$$d(BCD, A) = \frac{d(B,A)+d(C,A)+d(D,A)}{3} = \frac{17+21+27}{3} = 21.67$$

	BCD	**A**
BCD	-	21.67
A	21.57	-

و درخت هم به شکل زیر می‌شود:



ج) با استفاده از ماتریس فاصله بدست آمده در قسمت (الف)، به روش NJ درخت را رسم کنید.

rA = 17+21+27 = 65 → r'A = 65/2 = 32.5
rB = 17+12+18 = 47 → r'B = 47/2 = 23.5
rC = 21+12+14 = 47 → r'c = 47/2 = 23.5
rD = 27+18+14 = 59 → r'D = 59/2 = 29.5

$$d'_{ij} = d_{ij} - \frac{1}{2}(r_i + r_j)$$

d'AB = dAB – (rA + rB) /2 = 17-(65+47) /2 = -39
d'AC = dAC – (rA + rC) /2 = 21-(65+47) /2 = -35
d'AD = dAD – (rA + rD) /2 = 27-(65+59) /2 = -35
d'BC = dBC – (rB + rC) /2 = 12-(47+47) /2 = -35
d'BD = dBD – (rB + rD) /2 = 18-(47+59) /2 = -35
d'CD = dCD – (rC + rD) /2 = 14-(47+59) /2 = -39

➔ d':

	A	**B**	**C**	**D**
A	-	-39	-35	-35
B	-39	-	-35	-35
C	-35	-35	-	-39
D	-35	-35	-39	-

A و B یا C و D هردو d’ با مقدار ۳۹- دارند که کم‌ترین است و یکی از این دورا انتخاب می‌کنیم برای ادغام.
مثلا A و B را انتخاب می‌کنیم:

dAU = (dAB + (r’A – r’B)) /2 = (17+(32.5-23.5)) /2 = 13
dBU = 4
dCU = ((dAC – dUA) + (dBC-dUB)) /2 = ((21-13)+(12-4)) /2 =8
dDU = ((dAD – dUA) + (dBD-dUB)) /2 = ((27-13)+(18-4)) /2 =14

حال با توجه به مقادیر به‌دست آمده، ماتریس فاصله را به‌روز می‌کنیم:

	U(AB)	**C**	**D**
U(AB)	-	8	14
C	8	-	14
D	14	14	-

rU(AB) = 22 r’U(AB) = 22
rC = 22 r’C = 22
rD = 28 r’D = 28

d’U(AB)C = dU(AB)C – (rU(AB) + rC) /2 = 8- (22+22) /2= -14
d’U(AB)D = dU(AB)D – (rU(AB) + rD) /2= 14- (22+28) /2= -11
d’CD = dCD – (rC + rD) /2 = 14- (22+28) /2= -11

➔ d’:

	U(AB)	**C**	**D**
U(AB)		-14	-11
C	-14		-11
D	-11	-11	

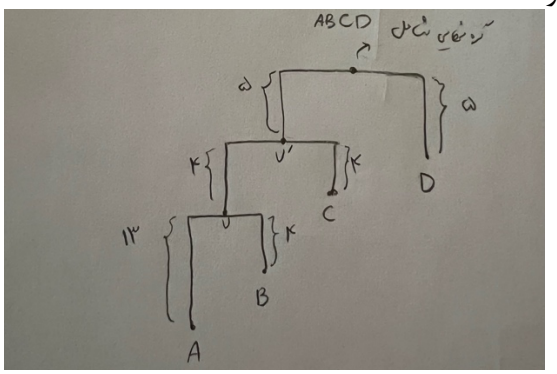
از بین این مقادیر می‌بینیم که U(AB) و C کوچک‌ترین مقدار را دارد و برای ترکیب انتخاب می‌شوند و گره جدید را U’ می‌نامیم:

dCU’= (dCU + (r’C – r’U)) /2 = (8+(22-22) /2 =4
dUU’ = 4
dDU’ = ((dDU – dUU’) + (dDC – dCU’)) /2 = ((14-4)+(14-4)) /2 =10

➔ ماتریس فاصله را باتوجه به مقادیر بدست آمده بروز می‌کنیم

	U'(ABC)	D
U'(ABC)	-	10
D	10	-

n برابر ۲ شده‌است و ۰ = ۲-n می‌شود پس نمی‌توان ادامه داد و الگوریتم همینجا تمام می‌شود.
و درخت هم به شکل زیر می‌شود:



د) درخت بدست آمده در قسمت (ب) و (ج) را با درخت اصلی مقایسه کنید و در صورت وجود تفاوت، دلیل را بیان کنید.

یک تفاوتی که وجود دارد در ترتیب ترکیب شدن گره‌ها با یکدیگر است که علتش تفاوت در معیار انتخاب و اولویت‌دادن به گره‌ها برای انتخاب است.

در UPGMA از همان فاصله‌های اولیه استفاده می‌کنیم و درصورت نیاز هم از متوسط این فاصله‌ها استفاده می‌کنیم(در حالتی که بیش از دو تاکسون در حال ادغام باشند). اما در NJ پارامترهای جدیدی تعریف می‌کنیم و معییار فاصله را بروز می‌کنیم و بر اساس آن‌ها انتخاب را انجام می‌دهیم. بر این اساس در قسمت ب ترتیب ترکیب گره‌ها به صورت B-C و BC- D و BCD-A است اما در قسمت ج به صورتAB و ABC و ABCD است. این درحالیست که درخت اولیه داده شده ترتیبی متفاوت از هردوی این روش‌ها دارد.

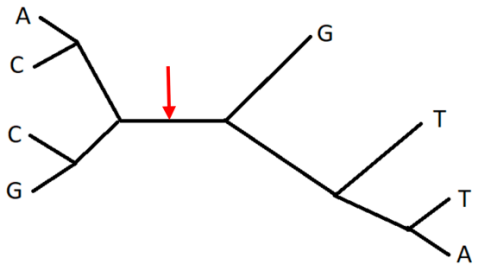
هم‌چنین باتوجه به پارامترهای جدیدی که در NJ تعریف می‌شود، فاصله یک تاکسون نسبت به همه گره‌ها دخیل می‌شوند و جامع‌تر از UPGMA است که فقط فاصله تاکسون‌های مشارکت کننده را دخیل می‌کند.

تفاوت دیگری که وجود دارد و یک پیش‌رفت برای NJ به حساب می‌آید این است که در UPGMA تفاوت فاصله‌ها روی درخت مشخص نمی‌شود و تاکسون‌ها همگی در یک سطح قرار می‌گیرد. اما در NJ این فواصل در درخت هم مشهود و قابل دریافت هستند و واقعیت را بهتر نشان می‌دهد(چراکه در واقعیت سرعت تغییر همه تاکسون‌ها یکسان نیست و طول برنچ‌ها باید متفاوت باشد). در درخت اولیه داده‌شده هم به نحوی این تفاوت فاصله‌ها همراه با مقدارشان نشان داده‌شده‌اند.

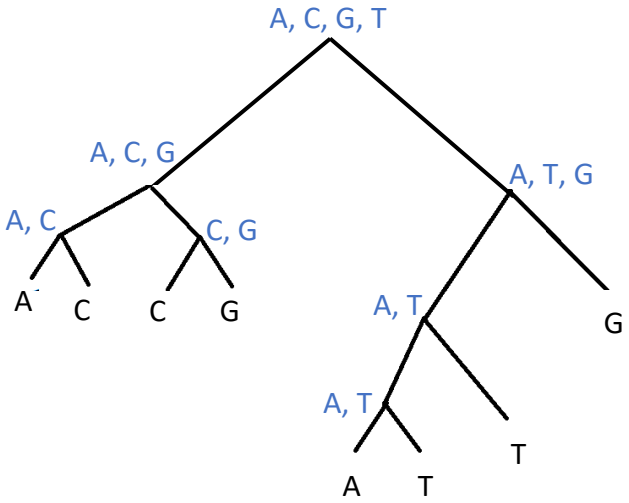
پس همان‌طور که در توضیحات آمد، فاصله تاکسون‌ها در درخت اولیه همانند درخت NJ مشهود است و این برخلاف درختی است که UPGMA تولید کرده(چراکه تاکسون‌هایش همگی در یک سطح هستند و تفاوت فاصله‌ها و تفاوت سرعت تغییرات در درخت دریافت نمی‌شود) و از این نظر درخت NJ بیش‌تر به درخت اولیه شباهت دارد. درخت NJ از نظر ترتیب ترکیب شدن کمی با درخت اولیه تفاوت دارد اما شباهتش از UPGMA بیش‌تر است. چراکه در NJ و درخت اولیه هردو ترکیب AB وجود دارد. فقط بعد از این مرحله در NJ ابتدا AB با C ترکیب شده است اما در درخت اولیه CD ترکیب شده‌اند و بعد با AB ترکیب شده‌اند. در حالی که در UPGMA ابتدا BC ترکیب شده‌اند و در ادامه هم شباهتی با درخت اولیه وجود ندارد.

سوال ۳)

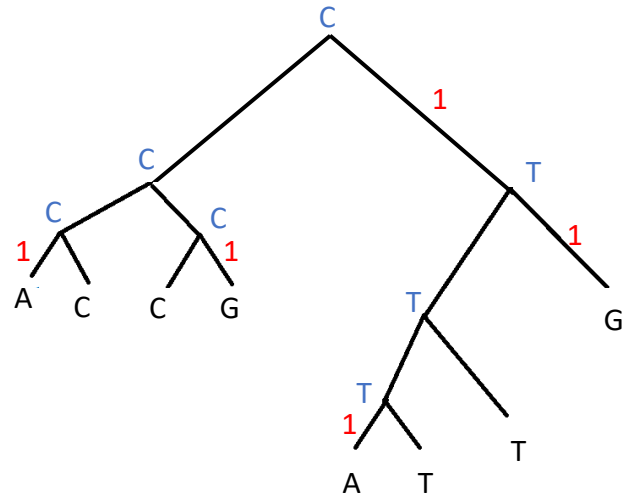
فرض کنید درخت زیر ریشه دار است. با در نظر گرفتن ریشه در نقطه قرمز، حداقل تعداد جهش‌ها در درخت زیر را پیدا کنید و بررسی کنید که در صورت تغییر محل ریشه درخت آیا تغییری در تعداد جهش‌ها اعمال می‌شود و یا خیر.



ابتدا از برگ‌ها به سمت ریشه‌ها می‌آییم و در هر مرحله اجتماع می‌گیریم:



حال گره‌ها را انتخاب می‌کنیم به طوری که کم‌ترین جانشینی را نتیجه دهد تا برسیم به برگ‌ها:



همانطور که می‌بینیم با حداقل ۵ جهش می‌توان درخت را تشکیل داد که در شکل مشخص شده‌اند.
تغییر در محل ریشه نیز تغییری درنتیجه حاصل نمی‌کند زیرا اگر درخت را از جای دیگری بشکنیم تنها نقطه شروع این مسیرها را تغییر می‌دهد بازهم همین تعداد جهش‌ها را نیاز خواهد داشت. (در واقع موقعیت نسبی برگ‌ها نسبت به یکدیگر و گره‌ها و ارتباطات ثابت می‌ماند)

سوال ۴)

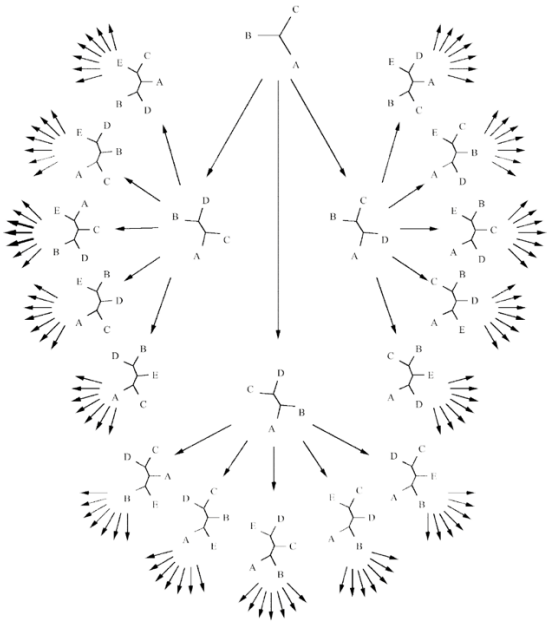
برای سوالات پارسیمونی بزرگ، دو روش exhaustive و branch-and-bound را به طور خلاصه شرح دهید و مقایسه کنید. همچنین اطمینان برای پیدا کردن اپتیمم سراسری را در این دو الگوریتم بررسی کنید.

۳روش کلی جست‌وجو وجود دارد:

Exhaustive search, Branch-and-Bound و Heuristic search method

که از میان این سه، روش‌های Exhaustive search و Branch-and-Bound به پاسخ exact می‌رسند و روش‌های هیوریستیک exact نیستند. حل مسئله بزرگ پارسیمونی نیاز دارد تمام فضای حالات (یعنی همه درخت‌های ممکن) را بررسی کند تا جواب بهینه را پیدا کند.

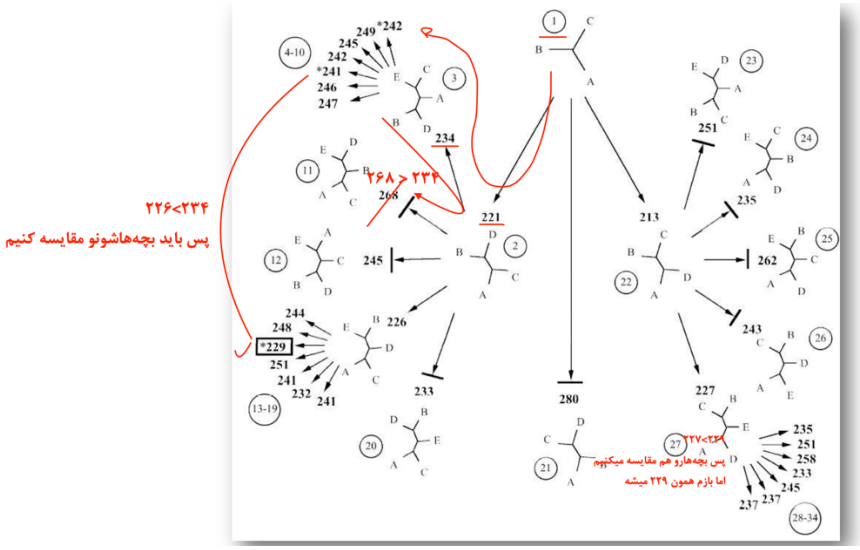
روش Exhaustive search به این صورت عمل می‌کند تمام درخت‌های ممکن را بررسی می‌کند.
روش پیاده‌سازی‌اش هم به این صورت است که ابتدا یک درخت بدون ریشه شامل ۳ تاکسون را ایجاد می‌کند. (که این ۳ گره اولیه می‌توانند رندم انتخاب شوند)
سپس تاکسون بعدی(گره چهارم) را باید اضافه کند. برای این کار تمام محل‌های ممکن برای اضافه کردن این گره را تست می‌کند و در هر کدام از این حالت‌ها امتیاز درخت بدست آمده را محاسبه می‌کند.
همین کار را برای تاکسون پنجم هم می‌کند و تمامی محل‌های ممکن برای اضافه کردن این تاکسون را تست می‌کند و به درخت‌های حاصل امتیاز می‌دهد.
این روند ادامه پیدا می‌کند تاجایی که تمامی تاکسون‌ها به درخت اضافه شوند و نهایتا درختی که بهترین امتیاز را کسب کرده انتخاب می‌شود.



روش Branch-and-Bound به طور کلی شبیه به روش قبلی عمل می‌کند(شروع از ۳ تاکسون، یک به یک اضافه کردن بقیه تاکسون‌ها و محاسبه امتیاز درخت‌ها) با این تفاوت که از هرس کردن بهره می‌برد و کمی از عملیات و محاسبات اضافه جلوگیری می‌کند.

درواقع ابتدا یک درخت فاصله برای همه‌ی تاکسون‌های موردنظر ایجاد می‌کند(مثلا با استفاده از UPGMA یا NJ). سپس حداقل تعداد جانشینی (min number of substitution) را برای این درخت محاسبه

می‌کند و به این صورت یک upper bound (یا lower bound درصورتی که امتیازات وارونه باشد) برای مقایسه و هرس کردن تعریف می‌کند. (که یک درخت پاریسمونی بیشینه باید مساوی یا کوتاه‌تر از این درخت بر مبنای فاصله باشد و اگر در جایی از رشد درخت این طور نبود، درخت را از آن‌جا رشد نمی‌دهیم و اصطلاحا هرس می‌کنیم) پس در این روش همانند روش قبل یک درخت ۳تایی اولیه ایجاد می‌کنیم و آن را توسعه می‌دهیم هم‌چنین درهرمرحله امتیاز بهترین درخت تولید شده تا به آن جای کار را نگهداری می‌کنیم و باتوجه به این امتیاز درخت را رشد می‌دهیم. (جاهایی را که می‌دانیم امتیاز بدتر می‌شود و از بهترین امتیازی که داشتیم بالاتر نمی‌رود هرس می‌کنیم و اصلا ادامه نمی‌دهیم).
زمانی که به انتهای درخت جست‌وجو برسیم، یا درخت بهینه را داریم که همان را نگه می‌داریم یا یک درخت نیمه بهینه داریم و آن را رد می‌کنیم. نهایتا زمانی که تمامی مسیرهای ممکن از ۳تاکسون اولیه را جست‌وجو کنیم، الگوریتم به پایان می‌رسد و درخت با بیش‌ترین پارسیمونی یافت می‌شود.
مثالی از این روش و هرس کردن:



با توجه به توضیحات داده شده هردوی این روش‌ها بهترین درختی که می‌توان پیدا کرد را پیدا می‌کنند. در روش اول این کار با جست‌جوی تمامی حالات ممکن انجام می‌شود و روش دوم هم همین‌کار را می‌کند اما آن‌هایی که اطمینان داریم به جواب بهینه منجر نمی‌شوند را هرس می‌کند و کمی کار را سبک‌تر می‌کند.
پس هردوی این روش‌ها بار محاسباتی زیادی دارند اما روش دوم کمی سبک‌تر از اولی است. (روش اول تا ۱۰ تاکسون و روش دوم تا ۲۰ تاکسون را می‌توانند در ارد معقولی پردازش کنند و نه بیش‌تر)

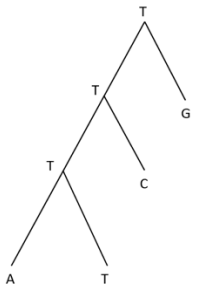
سوال ۵)

احتمال رخداد درخت‌های زیر را طبق ماتریس احتمال جهش داده شده پیدا کنید.

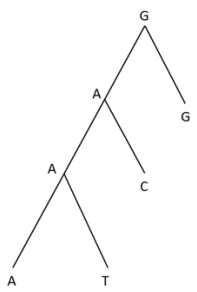
From/ To	A	C	G	T
A	0.55	0.2	0.15	0.1
C	0.05	0.7	0.15	0.1
G	0.15	0.05	0.6	0.2
T	0.25	0.05	0.1	0.6

در نظر داشته باشید که $P(A) = P(C) = P(G) = P(T) = 0.25$ و برای مثال احتمال $P(A \rightarrow C) = 0.2$

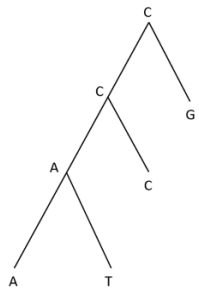
الف



ب



پ



اگر احتمال کامل را بخواهیم حساب کنیم بهتر است که احتمالات پیشین را هم دخیل کنیم و ۰.۲۵ ها را هم برای هر گره ریشه ضرب کنیم. (اما در مقایسه تاثیری ندارد چون احتمالات اولیه یکسان دارند و هرسه حالت در یک عدد 0.25 ضرب می‌شوند):

الف) P(درخت) =

$$P(T) \cdot P(T \rightarrow T) \cdot P(T \rightarrow T) \cdot P(T \rightarrow A) \cdot P(T \rightarrow T) \cdot (T \rightarrow C) \cdot P(T \rightarrow G)$$
$$= 0.25 \times 0.6 \times 0.6 \times 0.25 \times 0.6 \times 0.05 \times 0.1$$
$$= 0.0000675$$

ب) P(درخت) =

$$P(G) \cdot P(G \rightarrow A) \cdot P(A \rightarrow A) \cdot P(A \rightarrow A) \cdot P(A \rightarrow T) \cdot (A \rightarrow C) \cdot P(G \rightarrow G)$$
$$= 0.25 \times 0.15 \times 0.55 \times 0.55 \times 0.1 \times 0.2 \times 0.6$$
$$= 0.00013612$$

پ) P(درخت) =

$$P(C) \cdot P(C \rightarrow C) \cdot P(C \rightarrow A) \cdot P(A \rightarrow A) \cdot P(A \rightarrow T) \cdot P(C \rightarrow C) \cdot P(C \rightarrow G)$$
$$= 0.25 \times 0.7 \times 0.05 \times 0.55 \times 0.1 \times 0.7 \times 0.15$$
$$= 0.00005053$$

اگر درخت ML را هم بخواهیم انتخاب کنیم، درخت وسطی می‌شود زیرا احتمال رخدادش بیش‌تر از بقیه است.

بدون درنظر گرفتن احتمالات پیشین:

الف) P(درخت) = P(T → T) . P(T → T) . P(T → A) . P(T → T) . P(T → C) . P(T → G)

$= 0.6 \times 0.6 \times 0.25 \times 0.6 \times 0.05 \times 0.1$

= 0.00027

ب) P(درخت) = P(G → A) . P(A → A) . P(A → A) . P(A → T) . P(A → C) . P(G → G)

$= 0.15 \times 0.55 \times 0.55 \times 0.1 \times 0.2 \times 0.6$

= 0.0005445

پ) P(درخت) = P(C → C) . P(C → A) . P(A → A) . P(A → T) . P(C → C) . P(C → G)

$= 0.7 \times 0.05 \times 0.55 \times 0.1 \times 0.7 \times 0.15$

= 0.0002

درخت ML بازهم درخت وسطی می‌شود زیرا احتمال رخدادش بیش‌تر از بقیه است.