امیر محمّد آقاپور ۹۶۳۱۰۰۸ تمرین ۳

۱.

الف) $N-1$ حالت

ب) اگر مرتّب سازی به صورت صعودی باشد، تابع هدف(در واقع تابع هزینه) تعداد عناصر در هر حالت که مقدارشان از سمت چپی کمتر است را برمیگرداند. اگر این مقدار صفر شود، به حالت سورت شده رسیده ایم. برای بیان به صورت ریاضی، تابع کمکی $compare$ را تعریف و تابع هدف f است که برای رسیدن به پاسخ باید کمینه(صفر) شود.

$$compare(x,y)=\begin{cases}1\ if\ x>y\\0\ if\ x<y\end{cases}$$

$$f(state)=\sum_{i=0}^{N-2} compare(state[i],state[i+1])$$

ج) بلی. در حالت $(2,0,1)$ ، مقدار تابع هدف برابر ۱ است و در دو حالت همسایه آن یعنی $(0,2,1)$ و $(2,1,0)$ مقدار تابع هدف به ترتیب برابر ۱ و ۲ است که بزرگتر مساوی مقدار تابع هدف است، پس در مینیمم محلی(شانه) قرار داریم.

د)

تپه نوردی:

$Iteration\ 1$

$current=(2,7,3,9,5)$

$successors\ of\ current=\{(7,2,3,9,5),(2,3,7,9,5),(2,7,9,3,5),(2,7,3,5,9)\}$

$f\ value\ of\ successors=\{2,1,1,1\}$

$current=a\ successor\ with\ minimum\ f\ value=(2,7,3,5,9)$

$Iteration\ 2$

$successors\ of\ current=\{(7,2,3,5,9),(2,3,7,5,9),(2,7,5,3,9),(2,7,3,9,5)\}$

$f\ value\ of\ successors=\{1,1,2,2\}$

$current=a\ successor\ with\ minimum\ f\ value=(2,3,7,5,9)$

Simulated Annealing با مقدار T در مرحله اوّل برابر ۴ و در مرحله دوم ۱. همچنین از نسخه ای از برنامه که به دنبال مینیمم کردن است استفاده میکنیم.

$Iteration\ 1$

$current=(2,7,3,9,5)$

$successors\ of\ current=\{(7,2,3,9,5),(2,3,7,9,5),(2,7,9,3,5),(2,7,3,5,9)\}$

$randomly\ selected\ successor=(7,2,3,9,5)$

$\Delta E=2-1=1>0$

$current=successor\ with\ probability\ e^{-\frac{1}{4}}\cong 0.77\rightarrow chosen\ as\ current$

$Iteration\ 2$

$successors\ of\ current=\{(2,7,3,9,5),(7,3,2,9,5),(7,2,9,3,5),(7,2,3,5,9)\}$

$randomly\ selected\ successor=(7,2,3,5,9)$

$\Delta E=1-2=-1<0$

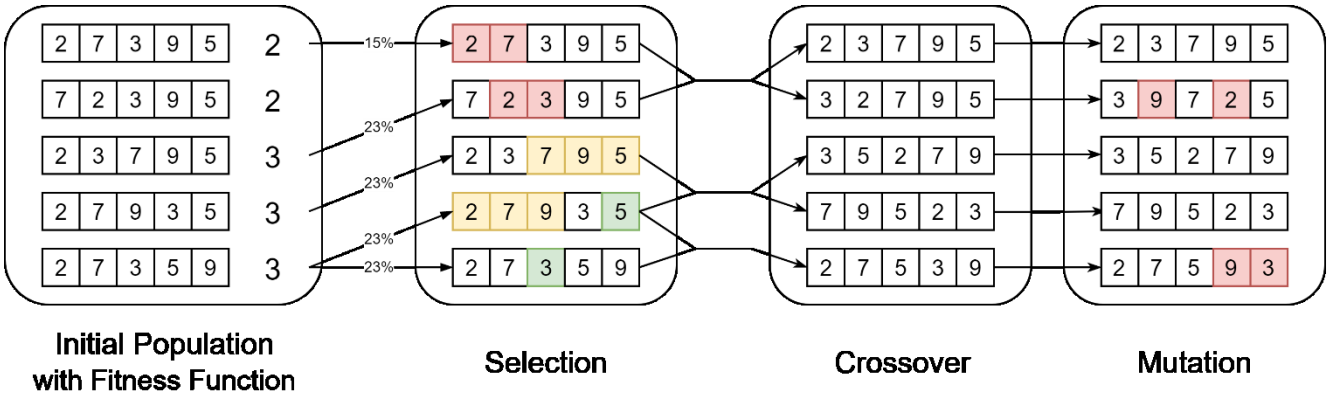
$current=successor\ because\ it\ is\ better(lower)\ =(7,2,3,5,9)$

ه)

تابع fitness را برابر $N-1-f$ تعریف میکنیم(تعداد دوتایی های مجاور سورت شده) و به دنبال زیاد کردن آن هستیم زیرا در حالت سورت شده f برابر صفر است.

برای عمل ترکیب از ordered crossover با بازه های مختلف استفاده میکنیم. به این صورت که از والد اوّل و دوم دو بازه با طول یکسان و با بازه مختلف انتخاب، سپس مقادیر آن ها را جابجا می کنیم با این شرط که اگرعددی برابر مقدار جایگزین شده خارج از بازه بود حذف و مقدار جایگزین شده با حفظ ترتیب به جای آن می رود.

برای عمل جهش که با احتمال کم انجام می شود، دو عدد با هم جابجا می شوند.



۲.

الف) اگر فضای حالت دارای ماکسیمم محلّی غیر گلوبال نباشد، تپه نورد به سرعت پاسخ بهینه را می یابد حال آنکه simulated annealing از نظر زمانی بدتر و حتی ممکن است جواب بهینه را نیابد(بخاطر زمانبندی کاهش دمای نامناسب، قبل از رسیدن به قلّه متوقف شود یا بدلیل استفاده از انتخاب های تصادفی، قلّه را نیابد)

ب) ایجاد تغییرات غیر وابسته به نسل قبل زیرا تغییر ایجاد شده ممکن است باعث بهبود شود. همچنین در طبیعت نیز جهش اتفاق می افتد و اگر مفید(دارای الگوی با برازندگی بیش از میانگین) باشد می ماند و در غیر این صورت در نسل های بعد حذف می شود.

ج) در تپه نوردی با شروع مجدد تصادفی فقط یه حالت را برای جست جو نگه میداریم(یک تپه نورد در هر لحظه) اما در پرتو محلی k جست و جو کننده در k قسمت رندوم وجود دارند. مشکل پرتو محلی هم کمبود تنوع در ناحیه های جست و جو و تمرکز سریع پرتو ها روی یک ناحیه با ارزش تر است. برای رفع این مشکل، بجای انتخاب بهترین ناحیه های بعدی برای بسط، احتمال متناسب با ارزش به آن ها نسبت و طبق آن احتمالات آن ها را بسط می دهیم که تبدیل به الگوریتم stochastic local beam search می شود.

۳.

الف) هر مجموعه از ضرایب را یک حالت در نظر گرفته(حالت n بعدی پیوسته است) و حالات همسایه را با افزایش یکی از ضرایب به مقدار ε بدست می آوریم. مقدار اپسیلون میتواند ثابت یا متغیّر و بستگی به حالت فعلی داشته باشد. به عنوان تابع هدف(سنجش بهتر بودنِ حالت های همسایه) می توان مجموع مربّعات اختلاف مقدار تابع و مقدار y نقاط داده شده را محاسبه کرد و آن را مینیمم کرد.(least square) یعنی اگر مقدار f یک همسایه کمتر بود، آن همسایه بهتر است.

$$f\big(state=(a_0,a_1,\dots,a_n)\big)=\sum_{\forall(x_i,y_i)}\big(y_i-(a_0+a_1x_i+\cdots+a_nx_i^n)\big)^2$$

انتخاب دیگر برای تابع هدف می تواند فاصله نقطه تا منحنی(طول پاره خط عمود بر منحنی و گذرا از نقطه) باشد.

ب)

Current = Initial State

While true

Neighbor = successor of Current with minimum value of f function

If f(Neighbor) < f(Current)

Current = Neighbor

else

return Current

ج) اگر طول گام ثابت کم باشد، سرعت رسیدن به هدف کم می شود و اگر زیاد باشد ممکن است مینیمم یا ماکسیمم محلی دقیق را نتوانیم پیدا کنیم.(در اطراف قله متوقّف شویم زیرا با برداشتن گام به آن طرف قله می رویم.) بنابراین رفتار الگوریتم خیلی وابسته به حالت شروع و رفتار تابع هدف دارد که مناسب نیست. همچنین در فضای پیوسته به دلیل وجود همسایه های زیاد، با برداشتن گام ثابت، فضا را گسسته کردیم(discretized) که باعث افزایش فاکتور انشعاب می شود.

د) با استفاده از گرادیان تابع هدف(∇f که همان مشتق جزئی نسبت به متغیّر هاست). یعنی طبق قسمت «الف» قرار می دهیم $\varepsilon_i = \frac{\partial f}{\partial a_i}$ و حالت بعدی با جمع هر درایه ی حالت فعلی با ε_i بدست می آید. اندازه گرادیان، میزان تغییرات مقدار تابع را مشخص میکند؛ بنابراین وقتی تغییرات زیاد است(دور از قله) اندازه گام ها زیاد و سرعت نزدیک شدن به هدف بیشتر و وقتی به قله نزدیک می شویم، گرادیان به صفر نزدیک می شود و اندازه گام ها کاهش و دقت یافتن مقدار مینیمم افزایش میابد. محاسبات و کد:

$$\nabla f = \left(\frac{\partial f}{\partial a_0}, \frac{\partial f}{\partial a_1}, \dots, \frac{\partial f}{\partial a_n}\right)$$

$$\varepsilon_i = \frac{\partial f}{\partial a_i} = \sum_{\forall (x_i, y_i)} 2x_i(y_i - a_0 - a_1 x_i - \cdots - a_n x_i^n)$$

$$state' = (a_0 + \varepsilon_0(state), a_1 + \varepsilon_1(state), \dots, a_n + \varepsilon_n(state))$$

Current = Initial State

While true

If $\nabla f \neq 0$

Current = Current + $\nabla f(Current)$

else

return Current

برای بهتر کردن روش فوق میتوان به به جای استفاده مستقیم از گرادیان، آن را در قرینه وارون ماتریس هسین ضرب کرد و سپس با حالت فعلی جمع کرد. یعنی:

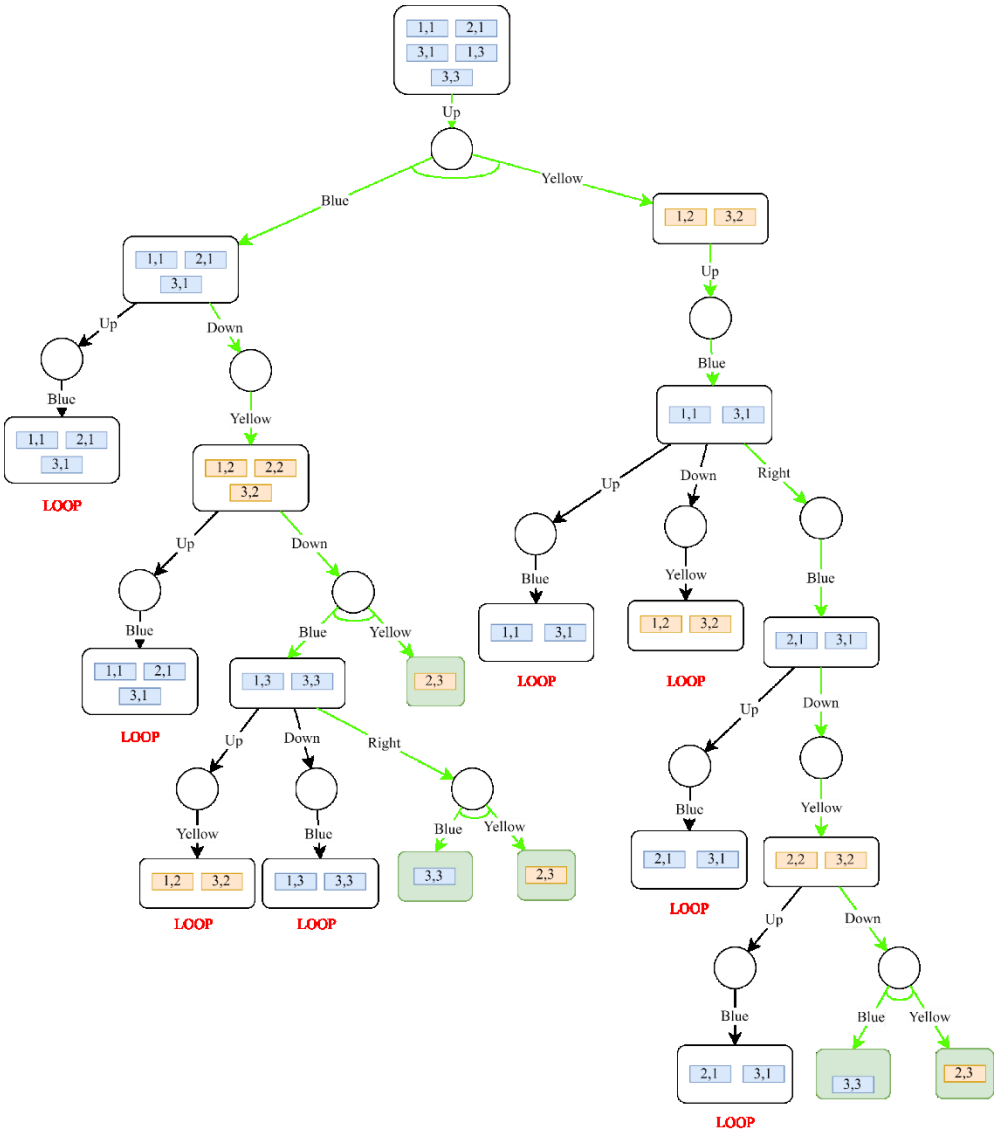
$$Current = Current - H_f^{-1}(Current)\nabla f(Current)$$

۴.

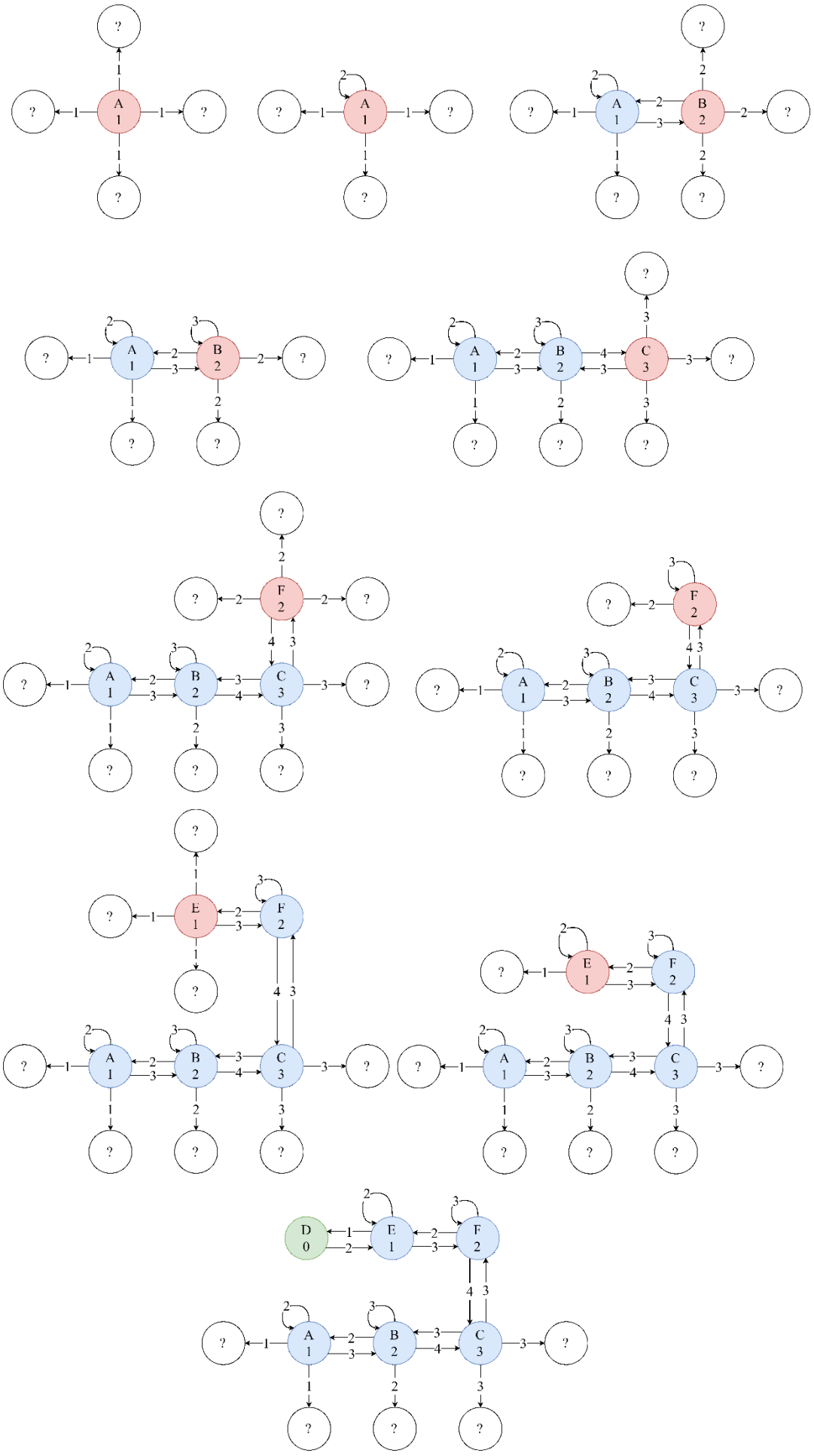
ترتیب انجام اعمال در هر مرحله بالا پایین راست چپ است. زیرا درختی که تمام برگ های آن به هدف ختم شده است به عنوان جواب برگردانده می شود. عامل با انجام اعمال مشخص شده در نود های AND و با چک کردن سنسور خود در نود های OR می تواند به جواب برسد.

جواب برگردانده شده درخت خواهد بود:

```
Go Up
if blue:
    Go Down
    Go Down
    if yellow:
        return 2,3
    else:
        Go Right
        if blue:
            return 3,3
        else:
            return 2,3
else:
    Go Up
    Go Right
    Go Down
    Go Down
    if Blue:
        return 3,3
    else:
        return 2,3
```

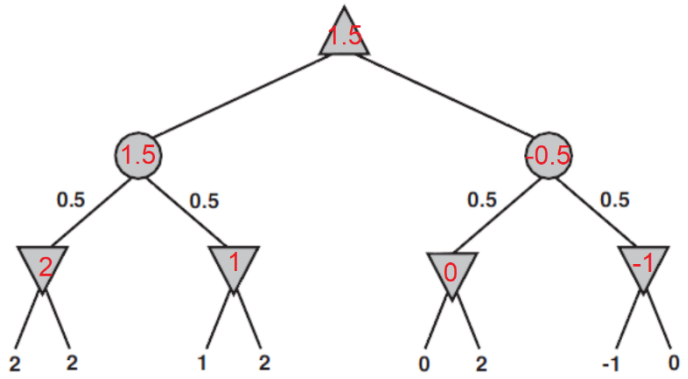


۵. اعمال در جهت خود رسم شده اند. مقدار تخمینی رسیدن از نود به هدف($H(s)$) داخل هر نود و مقدار هزینه هر مسیر با یادگیری های انجام شده روی هر یال نوشته شده است. همچنین فرض شده اعمال برگشت پذیر اند و در صورت انجام یه حرکت، هزینه و حالت بعدی برعکس آن حرکت نیز یاد گیری می شود.(ترتیب از چپ به راست)

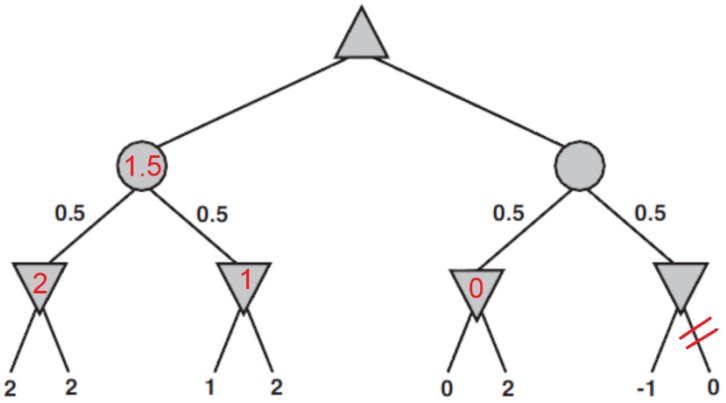


۶.

الف) بهترین حرکت از ریشه، نود چپ با امید ۱,۵ است.



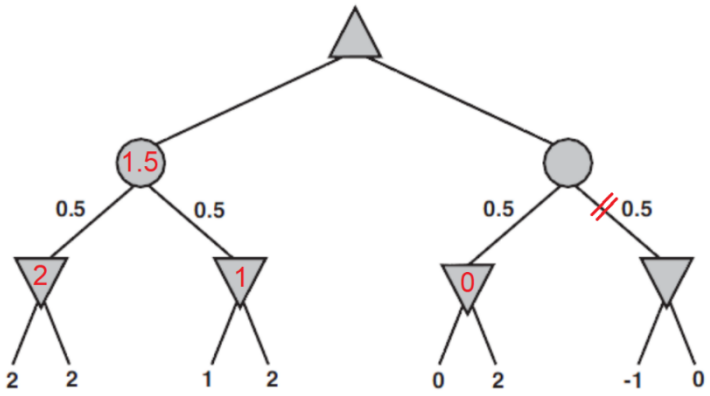
ب)



شاخه مشخص شده هرس α می شود زیرا :

$$v = 0 \times 0.5 - 1 \times 0.5 = -0.5, \alpha = 1.5 \rightarrow v \leq \alpha$$

ج)



نود مشخص شده حرس به علت شاخه شانس می شود زیرا اگر مقدار ۲ هم برگردانده شود(که بیشترین مقدار است)، مقدار گره شانس برابر ۱ خواهد شد که از مقدار آلفا که ۱,۵ است کمتر است.