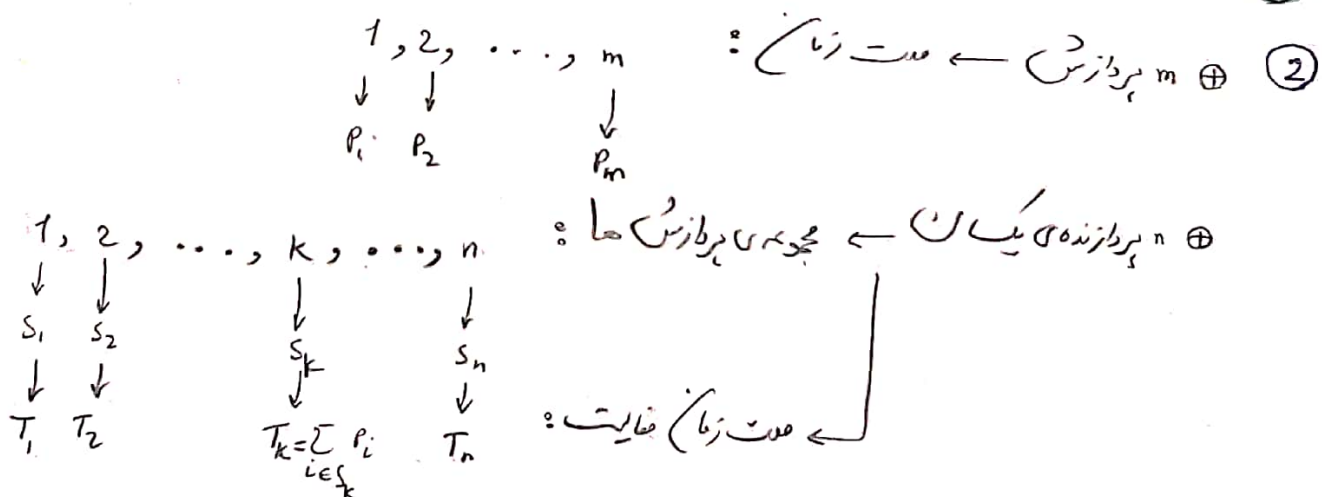


تکلیف ششم الگوریتم مسرسته

رهنمب علیداران - 810151236

- ① \oplus رسیدن زمانه این الگوریتم $O(n)$ است. زیرا در این الگوریتم با یک بار پیمایش لیست وزن های مربوطه n وسیله، می توان نیچه را بدست آورد.
- \oplus نیچه ای که این الگوریتم به عنوان حداقل مقدار جعبه های لازم به ما می دهد، حداقل 2 برابر جواب جعبه است. پس 2- تقریب است. دو جعبه ای را که به صورت متوالی در آن ها وسیله تکرار داریم را در نظر بگیرد. مجموع وزن وسیله هایی که در این دو جعبه قرار گرفته است، حتماً از C بزرگتر است! زیرا اگر بزرگتر از C شود، همی این وسیله ها را در یک جعبه قرار می دایم. پس حداقل نصف هر کدام از این دو جعبه خاص مانده و حذف رفته است. این منتهی برای C جعبه های دیگر نیز صادق است. \rightarrow پس اگر جواب جعبه R باشد، جواب این الگوریتم حداکثر $2R$ است. ϕ



$$\text{Min } M = \text{Max } T_k \quad 1 \leq k \leq n$$

\oplus هدف :

الف) هرگز است. حال از 1 تا m می‌نویسیم. بررسی می‌کنیم که T مربوط به کدام یک از هرزنده‌ها، minimum است. (اگر چند هرزنده، minimum بودند، به صورت رندم یکی را انتخاب می‌کنیم). سپس هرگز است. به مجموعه S مربوط به آن هرزنده اضافه می‌کنیم و P آن هرگز است را هم به T مربوط به این هرزنده اضافه می‌کنیم.

(ب)

$$\oplus m=6 \rightarrow \begin{matrix} P_1=3 & P_2=1 & P_3=6 \\ P_4=4 & P_5=6 & P_6=4 \end{matrix}$$

$$\oplus n=2 \rightarrow \begin{matrix} S_1=\{1\} & T_1=0 \\ S_2=\{1\} & T_2=0 \end{matrix}$$

$$\oplus \begin{matrix} S_1=\{1\} & T_1=3 \\ S_2=\{1\} & T_2=0 \end{matrix}$$

$$\oplus \begin{matrix} S_1=\{1\} & T_1=3 \\ S_2=\{2\} & T_2=1 \end{matrix}$$

$$\oplus \begin{matrix} S_1=\{1\} & T_1=3 \\ S_2=\{2,3\} & T_2=7 \end{matrix}$$

$$\oplus \begin{matrix} S_1=\{1,4\} & T_1=7 \\ S_2=\{2,3\} & T_2=7 \end{matrix}$$

$$\oplus \begin{matrix} S_1=\{1,4,5\} & T_1=13 \\ S_2=\{2,3\} & T_2=7 \end{matrix}$$

$$\oplus \begin{matrix} S_1=\{1,4,5\} & T_1=13 \\ S_2=\{2,3,6\} & T_2=11 \end{matrix}$$

⊕ مرحله ①: فرض می‌کنیم یک از هرزنده‌های $1 \leq i \leq 2$ انتخاب شود.

⊕ مرحله ②: $T_1 > T_2$ ← هرزنده 2 انتخاب می‌شود.

⊕ مرحله ③: $T_1 > T_2$ ← هرزنده 3 انتخاب می‌شود.

⊕ مرحله ④: $T_2 > T_1$ ← هرزنده 1 انتخاب می‌شود.

⊕ مرحله ⑤: زخمی نمی‌شوند.

⊕ مرحله ⑥: $T_1 > T_2$ ← هرزنده 5 انتخاب می‌شود.

$$\boxed{M=13} \leftarrow 11$$

ج) \oplus اگر مقدار مجبوری مثل M^* داشته باشیم، خواهیم داشت:

\oplus اگر پردازش با بیشترین زمان اجرا را داشته باشیم، M^* حداقل به اندازه‌ی این تعیین است. چرا که این پردازش باید توسط پردازنده‌ی اجرا شود پس داریم:

$$M^* \geq \max_{1 \leq k \leq m} t_k$$

\oplus در بهترین حالت (min)، میانگین زمان اجرای تمام پردازش‌ها را به عنوان زمان پردازش پردازنده‌ها خواهیم داشت. پس M^* حواره حداقل به اندازه‌ی این میانگین خواهد بود.

$$M^* \geq \frac{1}{n} \sum_{k=1}^m t_k$$

\oplus رفته در مرحله‌ی از استوریتم تقریبی ما هستیم که می‌خواهیم پردازش‌ها را به آن‌ها پردازش است، به پردازنده‌ی شماره j ، assign کنیم. پس سینه T_j این پردازنده، از بین تمام T ها، بهترین T بوده است. و همچنین از میانگین زمان اجرای پردازنده‌ها هم کمتر بوده است. پس داریم:

مقدار T_j پردازنده‌ی از میانگین تمام پردازش‌ها

$$\left. \begin{array}{l} T_j - t_i \leq \frac{1}{n} \sum_{k=1}^m t_k \\ t_i \leq \max_{1 \leq k \leq m} t_k \end{array} \right\} \Rightarrow T_j = T_j - t_i + t_i \leq \frac{1}{n} \sum_{k=1}^m t_k + \max_{1 \leq k \leq m} t_k$$


پس این یک upper bound برای جواب استوریتم تقریبی ما است.

\oplus همچنین با استفاده از دو عبارت \oplus و \oplus upper bound ای که پیدا کردیم، خواهیم داشت:

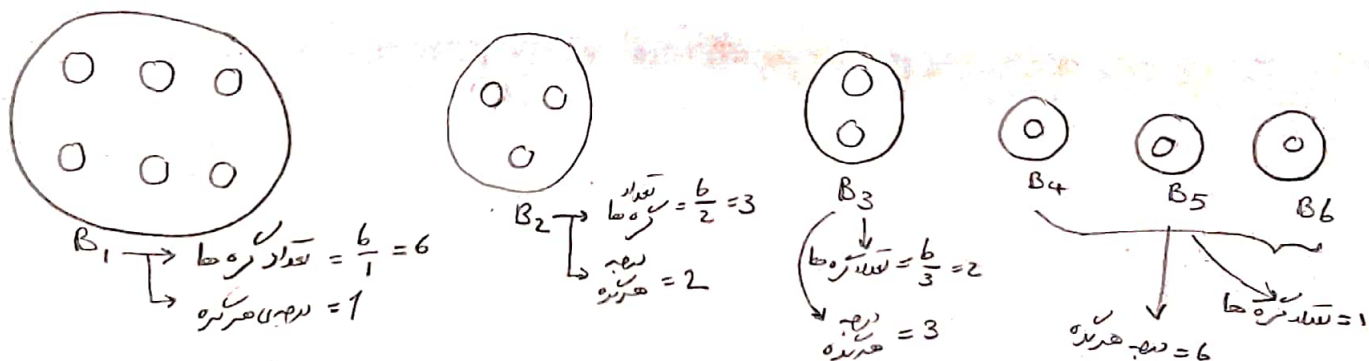
$$2M^* \geq \max_{1 \leq k \leq m} t_k + \frac{1}{n} \sum_{k=1}^m t_k \geq T_j \xrightarrow{\text{So}} T_j \leq 2M^*$$

پس سینه جوابی که این استوریتم تقریبی به ما می‌دهد، حداکثر دو برابر بدترین جواب عینه است. استوریتم 2- تقریب است.

A



6 = تعداد ژنرها
6 = درجه عددی



Scanned with CamScanner

④

⊕ با توجه به اینکه روشی که برای ساخت cycle داریم، در این الگوریتم یک حرفه شده است، بسیار شبیه به الگوریتم Prim برای ساخت MST است. cycle ای که در حقیقت به عنوان جواب به ما می دهد، شامل MST است. با حذف یکی از یال ها به MST این گراف خواهیم رسید. پس اگر جواب این الگوریتم تقریبی را برابر C در نظر بگیریم، داریم:

$$\textcircled{1} \quad \boxed{MST \leq C}$$

⊕ همچنین اگر روی MST یک full walk بزنیم، می دانیم که با توجه به اینکه از هر یال درج MST دوبار عبور کرده است، خواهیم داشت:

$$\textcircled{2} \quad \boxed{2MST = \text{full walk}}$$

⊕ یک full walk روی MST این گراف در نظر می گیریم. در حقیقت full walk، هر روی گراف دوبار ظاهر شده است. برای تبدیل این cycle به یک Ham، باید هر بار حلقه های را حذف کرد و یال ها جایگزین کرد، در حقیقت هر node یکبار دیده شده باشد. که با توجه به برقرار بودن نامساوی مثلث در گراف، وزن دو Ham در حقیقت داریم، حداکثر به اندازه ی وزن full walk

است. جواب الگوریتم ما (C)، یکی از این Ham ها است. پس داریم:

$$\textcircled{3} \quad \boxed{C \leq \text{full walk}}$$

⊕ اگر جواب بهینه ی مسئله را برابر C^* در نظر بگیریم، با حذف یکی از یال ها می توان به MST رسید.

پس وزن MST، یک lower bound برای C^* است:

$$\textcircled{4} \quad \boxed{MST \leq C^*}$$

الگوریتم در حقیقت خواهیم داشت:

$$\frac{C}{C^*} \leq \frac{\text{full walk}}{MST} \stackrel{\textcircled{2}}{=} \frac{2MST}{MST} = 2 \Rightarrow \frac{C}{C^*} \leq 2$$

approximation-ratio = 2

یعنی الگوریتم 2-تقریبی است

⊕ k عدد مثلی ← ظرفیت : c_1, c_2, \dots, c_k
 ← وزن : w_1, w_2, \dots, w_k

⊕ هدف : ① حداکثر مجموع وزن مثلی ها M
 ② مجموع ظرفیت مثلی ها ، maximum شود

⊕ DP : $O(kM)$ (در این سیستم تقریبی باز) \rightarrow polynomial = ؟

! ← این مسئله نسبتاً پیچیده به سبب $0/1$ -knapsack است. این سیستم FPTAS به این دلیل ارائه می دهیم :

* مرحله 1 : ظرفیت های مثلی ها را با اعداد integer که بزرگ باشند، جایگزین می کنیم :
 برای این کار ، دو شرط را رعایت می کنیم : ① اعداد نباید بزرگ باشند (چون زمان running time بزرگ می شود و polynomial نمی شود).
 ② مقدار error در جواب داریم که باید کنترل شود.

⊕ این برای این تبدیل به integer روشی باید استفاده شود که این شرایط را رعایت شوند. اگر فقط به سادگی ، اعداد را رند به بالا round کنیم ، شرط دوم برقرار نخواهد بود. دقت پس روی error خواهیم داشت. پس این روش را به کار

می گیریم : ① یک عدد Δ در نظر می گیریم. بعد اعداد را به بازه های با طول Δ تقسیم می کنیم :
 و بعد ظرفیت های c_i را به کوچکترین Δ که از آن بزرگتر است ، تبدیل می کنیم :
 ② سپس حال برای کوچک شدن مقادیر ، چه را به Δ تقسیم می کنیم. که جفت integer خواهد بود.

⊕ Δ را به این شکل انتخاب می کنیم :

$$c_i^* = \left\lceil \frac{c_i}{\Delta} \right\rceil$$

که داریم برای سیستم های PTAS :
 approximation-result $\geq (1-\epsilon) \cdot OPT$
 $= OPT - \epsilon \cdot OPT$
 (maximization)

پس میزان error می گوییم داشته باشیم.

له همین حد داریم که error ایجاد شده روی ظرفیت هدف مشکلی حداکثر Δ است. زیرا در ابتدا حداکثر تا اندازه Δ ظرفیت ها را زیاد کردیم. در نهایت هم حداکثر Δ به Δ تقسیم کردیم. که scale کوچکی به دست می آید. پس فقط همان error در نهایت کار را داریم که حداکثر Δ است.

له یې:

⊕ هدف: $\text{error} \leq \epsilon \cdot \text{OPT}$ کچه اوسیم

⊕ $\text{error} \leq \Delta$ دوی نریت هرک از منځ ها داریم

Δ $\xrightarrow{\text{تعداد منځ های انتخاب شده در جواب احسنه}}$ $K \cdot \Delta$ \Rightarrow $\text{error} \leq K \cdot \Delta$

تعداد منځ های انتخاب شده در جواب احسنه $\leq K$ خواهد بود

له یې:

⊖ له: $K \cdot \Delta \leq \epsilon \cdot \text{OPT} \Rightarrow \Delta = \left(\frac{\epsilon}{K}\right) \cdot \text{OPT}$

⊖ حاله ی instance از منځ با نریت های integer و کوهله داریم.

* مرحله 2: ما از اوسیم هییه ی DP نه داریم، برای حل این instance از منځ که نریت های integer هستند، استفاده میکنیم و جواب هییه ی ما در $O(K \cdot C_{\max}^*)$ بدست می آید. یک زیر مجموعه از منځ ها را به عنوان جواب optimal به ما خواهد داد، نه مجموع نریت های آن برابر C_{\max}^* است.

⊕ در مرحله ی قبل گفته بودیم که از OPT استفاده میکنیم برای بدست آوردن Δ ؛ درحالی که جواب optimal را نداریم. پس آن کار ممکن نبود. \Leftarrow به جای از منځ Lower Bound جواب هییه ی استفاده میکنیم:

$$\Delta = \left(\frac{\epsilon}{K}\right) \cdot LB$$

له با توجه به اینکه $LB \leq OPT$ است، پس Δ کوچکتر می شود. پس error رولا به اوسیم کمتر می شود. پس خوب است به اوسیم و ratio مربوط به آن آیسیم وارد می شود.

\Leftarrow LB را به این شکل تعریف میکنیم: اگر n منځ ها (حداکثر به اندازه M باشد) منځ با بیشترین W را انتخاب میکنیم و LB خواهد بود.

* مرحله 3: جوابی که اوسیم در مرحله ی قبل به ما داده است را report میکنیم به عنوان جواب اوسیم تقریبی.

* برای اینکه اثبات کنیم FPTAS است، باید این 3 مورد را اثبات کنیم:

① اینکه جواب ما valid است و در منځ های جواب، کوچکتر صمدی M است.

② جواب ما حداکثر برابر $(1-\epsilon) \cdot \text{OPT}$ است. (چون maximization است.)

③ running time نسبت به ورودی ها، polynomial است. $(n^{\frac{1}{\epsilon}})$

له آيات اين في مورد:

1 ما ظرفيت هارا تغير ديم، پس M هارا نه پس وزن جوابي كه اين سيم به ما مي دهد، حدس M خواهر بود.

2) ما داریم:

① جواب بچينه براي مسئله با ظرفيت هاي int نو $R = (C^*)$ (زير مجموعه منفي ها)

② جواب بچينه براي مسئله با ظرفيت هاي اديري $R_{OPT} = (C)$ (زير مجموعه منفي ها)

3) ما داریم:

$$\textcircled{1} \sum_{i \in R} C_i^* \not\geq \sum_{i \in R_{OPT}} C_i^*$$

در مسئله با ظرفيت هاي C^* ، از هر مجموعه R ديگري، بزرگتر است.

$$\textcircled{3} \Delta = \left(\frac{\epsilon}{K} \right) \cdot LB$$

$$\textcircled{4} LB = \max_{1 \leq i \leq K} C_i \leq OPT$$

$$\textcircled{2} C_i^* = \left\lceil \frac{C_i}{\Delta} \right\rceil$$

4) ما داریم:

$$\textcircled{2} \rightarrow \frac{C_i}{\Delta} \leq C_i^* \leq \left(\frac{C_i}{\Delta} \right) + 1$$

$$\sum_{i \in R} C_i \not\geq \sum_{i \in R} \Delta (C_i^* - 1) = \Delta \cdot \left(\sum_{i \in R} C_i^* \right) - |R| \cdot \Delta$$

$$\begin{aligned} & \textcircled{1} \text{ و } \textcircled{2} \rightarrow \Delta \cdot \left(\sum_{i \in R_{OPT}} C_i^* \right) - K \cdot \Delta \not\geq \sum_{i \in R_{OPT}} C_i - K \cdot \Delta \xrightarrow{\textcircled{3}} \sum_{i \in R_{OPT}} C_i - K \cdot \left(\frac{\epsilon}{K} \right) \cdot LB \\ & \xrightarrow{\textcircled{4}} \sum_{i \in R_{OPT}} C_i - \epsilon \cdot LB \not\geq \sum_{i \in R_{OPT}} C_i - \epsilon \cdot OPT = OPT - \epsilon \cdot OPT = (1 - \epsilon) \cdot OPT \end{aligned}$$

پیدا کنیم:

$$\sum_{i \in R} c_i \leq (1-\epsilon) \cdot OPT$$

approximation-ratio

پیدا کنیم $(1-\epsilon)$ - تقریب است.

3] این running time نسبت به k polynomial است:

1) در مرحله 1 که c^* را بدست آوریم، در $O(k)$ برای $k-L$ مربوط به مقادیر c^* را می‌توانیم.

2) در مرحله 3، report کردن زیر مجموعه‌ی بدست آمده، در $O(k)$ انجام می‌شود.

3) در مرحله 2، که می‌توانیم با c^* حل کنیم، به دست می‌آید:

⊕ استوریتم (DP) روی k با فرمت های integer، دارای $O(k \cdot c_{tot}^*)$ است.

$$c_{tot}^* = \sum_{i \in R} c_i^*$$

⊕ داریم:

$$1) c_i^* = \left\lceil \frac{c_i}{\Delta} \right\rceil$$

$$2) \Delta = \left(\frac{\epsilon}{k} \right) \cdot LB$$

$$3) LB = \max_{1 \leq i \leq k} c_i$$

1) در k c_i^* محدود دارد
2) هر کدام از c_i^* ها از maximum این ها کوچکتر است.

$$c_{tot}^* = \sum_{1 \leq i \leq k} c_i^* \leq k \cdot \max_i c_i^* = k \cdot \max_i \left\lceil \frac{c_i}{\Delta} \right\rceil$$

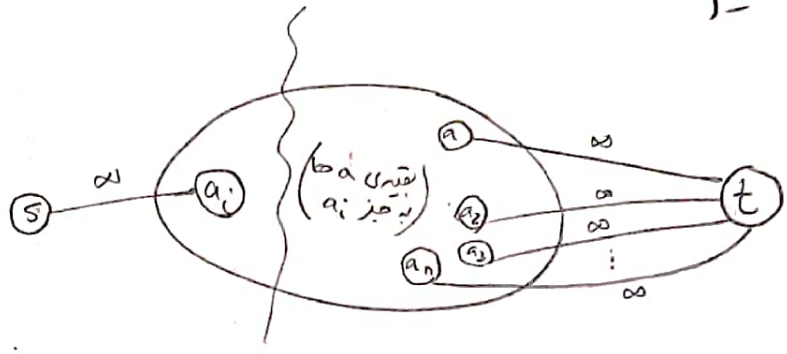
$$= k \cdot \left\lceil \frac{\max_i c_i}{\Delta} \right\rceil \stackrel{2)}{=} k \cdot \left\lceil \frac{\max_i c_i}{\left(\frac{\epsilon}{k} \right) \cdot LB} \right\rceil \stackrel{3)}{=} k \cdot \left\lceil \frac{1}{\frac{\epsilon}{k}} \right\rceil = k \cdot \left\lceil \frac{k}{\epsilon} \right\rceil = O\left(\frac{k^2}{\epsilon}\right)$$

FPTAS ϵ k poly
است $\frac{1}{\epsilon}$ k poly

پیدا کنیم $(1-\epsilon)$ - تقریب است
است $O\left(\frac{k^3}{\epsilon}\right)$

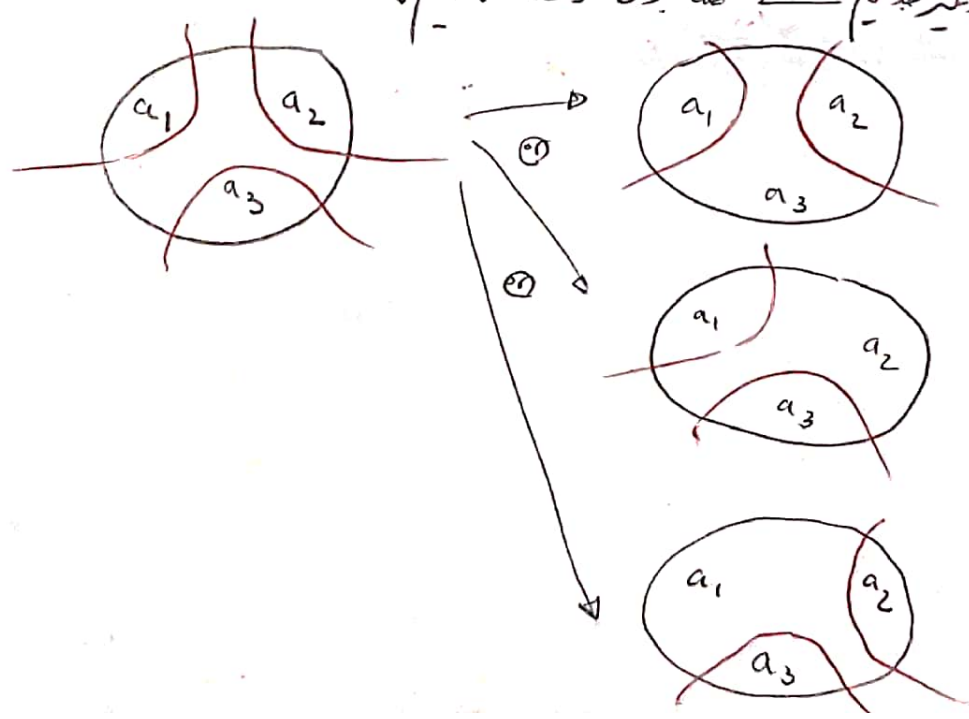
1) $O(k)$ ← مرحله 1
2) $O\left(\frac{k^2}{\epsilon}\right)$ ← مرحله 2
3) $O(k)$ ← مرحله 3

⊕ گزرات را بر این شکل تغییر می دهیم:



یک source با نام s و یک sink با نام t تعریف می کنیم. بین s و a_i یک یال با وزن ∞ ایجاد می کنیم. به ازای بقیه a_i ها و t هم همین کار را می کنیم. به این تغییر، اگر بخواهیم $\min \text{cut}$ در این گزرات داشته باشیم، حتماً به شکل رسم شده در بالا خواهد بود. a_i را در سطح s قرار دهیم، و بقیه را در سطح t قرار دهیم. به این ترتیب، به ازای a_i یال های با وزن ∞ را انتخاب نکنیم، بلکه حتماً حتماً a_i را انتخاب می کنیم. cut را نتیجه خواهد بود که به یک کار در polynomial انجام می شود.

⊕ برای جدا کردن هر یک از a_i در گزرات از دیگر a_i ها، یک cut به شکل بالا باید ایجاد کنیم. اگر بخواهیم a_1 را از بقیه جدا کنیم، باید a_1 را از a_2, a_3, \dots, a_n جدا کنیم، به $n-1$ cut نیاز داریم. a_1 را از a_2, a_3, \dots, a_n جدا کنیم. مثلاً برای $n=3$ داریم:



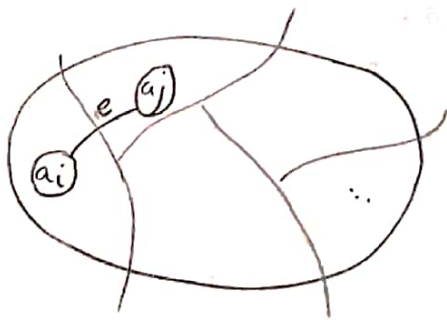
برای جدا کردن 3 نره،
دوتا از 3 cut کافیست.
به این 3 حالت
برای انتخاب cut ها
خواهیم داشت

⊕ کفشیتم حداکثر n گره به صورت 2 به 2 از یلوگیر، انتخاب $cut \in [n-1]$ کافی است. پس یک استوریج approximation بسازیم که از این $cut \in [n]$ در براریم که هر کدام، یکی از a_i طرا از یلوگیر حداکثر $n-1$ راب به صورت رندوم انتخاب کند و تمام گره طرا دوبه دو از هم جدا کند. به یلوگیر به این روند، جوابی که این استوریج به ما می دهد (C)، عواره. وزن میگیری مؤاهداست نسبت به حالتی که $cut \in [n-1]$ راب بخوی انتخاب کنیم که minimum شود مجموع وزن آن ها (یعنی $cut \in [n-1]$ انتخاب کنیم). و $cut \in [n]$ که از یلوگیر وزن میگیری را انتخاب کنیم. پس داریم:

$$\textcircled{1} C \leq \frac{n-1}{n} \sum_i \text{Mincut}(a_i, A - \{a_i\})$$

\swarrow $\left(1 - \frac{1}{n}\right)$ \searrow
 (رندوم انتخاب می کند) $cut \in [n-1]$ به صورت

{ کل mincut طرا جمع میزنیم. در بدترین حالت، هم با هم برابر هستند. $cut \in [n-1]$ از این $\text{mincut} \in [n]$ که با هم برابر هستند را اند جوامیم در قله بگیریم، $\left(1 - \frac{1}{n}\right)$ را که همان $\frac{n-1}{n}$ است و یلوگیر $cut \in [n-1]$ از n مساوی را جمع کرده ایم، در \sum ضرب می کنیم.



⊕ گراف راب این صورت در نظر میگیریم:

یا e راب بتوان بخش از جواب OPT در نظر میگیریم؛ زیرا اگر عضو جواب OPT نباشد و آن را در گراف ما اضافه کنیم، دو e نه و a_j به وسیله ی آن به هم متصل خواهند بود. پس جواب OPT حتماً باید e را حذف کند.

که مجموعه ی $F(K)$ راب این طل تعریف می کنیم: به ازای هر e که انتقالی بین دو گره a_i و a_j ایجاد کرده است، آن e یا راب در مجموعه ی $F(i)$ و $F(j)$ اضافه می کنیم. به این ترتیب، هر e که جز جواب OPT است، دو گره ی a_i و a_j را از هم جدا می کند ($i \neq j$)، جز در مجموعه ی $F(i)$ و $F(j)$ است. پس اگر وزن $F(K)$ ها

رابطہ جمع ہر ایک، باقیہ ہر ایک، OPT ، در دو مجزہ از $F(K)$ حاصل ہر ہر، خواہیم داشت:

$$\textcircled{2} \sum_i W(F(i)) = 2 \cdot OPT$$

\oplus دسیم کہ طبق تقریب، $F(i)$ شامل تمام یں حاصل است کہ i از تقریبی n گزری A $(i \neq j \& n < j < 1 \& j)$ جدا می کند. پس میزان مجموع وزن این $F(i)$ حاصل به اندازه $MinCut$ ای است کہ این جدا سازی را می تواند داشت باشد:

$$\textcircled{3} W(F(i)) \geq MinCut(a_i, A - \{a_i\})$$

$$\oplus \quad \begin{array}{c} \textcircled{2} \text{ طبق} \\ \downarrow \\ 2OPT = \sum_i W(F(i)) \end{array} \geq \begin{array}{c} \textcircled{3} \text{ طبق} \\ \downarrow \\ \sum_i MinCut(a_i, A - \{a_i\}) \end{array}$$

\leftarrow پس:

$$\Rightarrow \oplus \quad 2OPT \geq \sum_i MinCut(a_i, A - \{a_i\})$$

$$\oplus \quad C \leq \frac{n-1}{n} \sum_i MinCut(a_i, A - \{a_i\})$$

$$\begin{array}{c} \textcircled{3} \text{ طبق} \\ \downarrow \\ \leq \left(\frac{n-1}{n}\right) 2OPT = 2\left(1 - \frac{1}{n}\right) OPT \end{array}$$

$$\Rightarrow C \leq 2\left(1 - \frac{1}{n}\right) OPT \Rightarrow \frac{C}{OPT} \leq \frac{2\left(1 - \frac{1}{n}\right)}{1}$$

approximation-ratio

$n=3$ ، $n=2$ برای $2(1 - \frac{1}{n})$ - تقریب است. \leftarrow مثلاً

$$\begin{array}{l} \downarrow \\ 2\left(1 - \frac{1}{2}\right) = 1 \Rightarrow \text{optimal} \Rightarrow \text{min cut} \\ 2\left(1 - \frac{1}{3}\right) = 2\left(\frac{2}{3}\right) = \frac{4}{3} \quad \checkmark \end{array}$$