گزارش پروژه دوم درس طراحی الگوریتم‌ها (پاراگراف)

امیرعلی صادقی فرشی (۹۹۱۲۸۳۴)

الگوریتم: الگوریتم به کار رفته در این پروژه به صورت زیر است:

۱- برای تمام i و jها که $1 \leq i,j \leq n$ که در آن n تعداد کلمات است و $j \geq i$، هزینه‌ی قرار دادن کلمات i تا j در یک سطر را حساب می‌کنیم. بدین‌ترتیب یک ماتریس بالامثلثی تشکیل می‌شود که آن را PartialCosts می‌نامیم.

۲- برای هر $1 \leq w \leq n-1$، Cost(w) را هزینه‌ی ساختن پاراگراف فقط با استفاده از کلمات ۱ تا w-ام تعریف می‌کنیم. برای $w = 1$ واضح است که $\text{Cost}(1) = \text{PartialCosts}(1,1)$. برای wهای بعدی داریم:

$$Cost(w) = min\big(Cost(w-i) + PartialCosts(w-i+1, w)\big)$$
$$for: i = 1,2,\dots,w$$

که در عبارت بالا Cost(0) را صفر در نظر می‌گیریم. در واقع در اینجا برای محاسبه‌ی هزینه‌ی w-ام به ازای نقاط شکست مختلف و قرار دادن باقی کلمات در یک سطر جدید، بهترین هزینه را انتخاب می‌کنیم.

برای $w = n$ کمی رابطه‌ی بالا تغییر خواهد یافت چون سطر آخر با هر تعداد جای خالی، هزینه‌ای ندارد و این موضوع لحاظ شده است. در واقع در فرایند مینیمم‌گیری، فقط هزینه‌ی سطرهای ماقبل آخر با هم جمع می‌شود و از میان آن‌ها بهترین انتخاب می‌شود.

۳- پس از اتمام مرحله‌ی پیش، $Cost(n)$ برابر هزینه‌ی ساختن این پاراگراف خواهد بود.

۴- برای بازسازی فرم نهایی پاراگراف، از نقاط شکست ذخیره‌شده استفاده می‌کنیم. بدین‌صورت که از آخرین درایه‌ی آن شروع می‌کنیم و به صورت بازگشتی تا جایی پیش می‌رویم که شماره‌ی اندیس با مقدار موجود در آن یکی باشد. چرا که این نقطه خودش نقطه‌ی شکست خودش است پس آخرین کلمه‌ی سطر اول خواهد بود. آن سطر را چاپ کرده و سپس در هر فراخوانی، کلمات مابین مقدار آن درایه و شماره‌ی اندیس آن را چاپ می‌کنیم. بدین‌صورت تمام سطرها چاپ می‌شوند.

تحلیل زمانی:

۱- مرحله‌ی ۱ در الگوریتم بالا شامل $1 + 2 + \cdots + n = \frac{n(n+1)}{2}$ عملیات است.

۲- در مرحله‌ی ۲، برای هر w، حداکثر w عمل جمع و $w-1$ عمل مقایسه باید انجام شود. واژه‌ی حداکثر بدین‌جهت به کار برده شده که اگر به ازای یک i مقدار PartialCosts برابر بی‌نهایت باشد، مابقی iها محاسبه نخواهند شد چون اگر به ازای آن i، آن مجموعه از کلمات در یک سطر جا نشوند، با افزوده شدن یک کلمه دیگر باز هم در یک سطر جا نخواهند شد. در نتیجه هزینه‌ی این مرحله برابر است با:

$$\sum_{w=1}^{n} (2w-1) = n^2$$

جمع هزینه‌ی این دو مرحله که هزینه‌ی به دست آوردن هزینه‌ی مینیمم است $\theta(n^2)$ می‌باشد. برای به دست آوردن فرم پاراگراف که همان مرحله‌ی ۴ می‌باشد کافی است حداکثر n بار به صورت بازگشتی تا ابتدای آرایه‌ی مربوط به نقاط شکست حرکت کنیم. لذا این مرحله $\theta(n)$ خواهد بود. جمع این دو که هزینه‌ی نهایی است، برابر $\theta(n^2)$ است.

شبیه‌سازی: برای شبیه‌سازی این کد، مثال زیر به آن ورودی داده شد و تصویر خروجی ترمینال را در زیر مشاهده می‌کنیم. همان‌طور که می‌بینیم در سطر اول یک فاصله (هزینه 1^3)، در سطر دوم دو فاصله (هزینه 2^3) و سطر سوم نیز به دلیل سطر آخر بودن هزینه‌ی صفر خواهد داشت. در نتیجه هزینه‌ی کل برابر $1 + 8 = 9$ می‌باشد. آرایه‌ی Costs که در بالا ذکر شد در زیر با نام All Costs آمده است. همچنین Breakpoints همان نقاط شکست است که به ازای هر کلمه ذخیره شده است.

```
Word Lengths: [2, 3, 2, 8, 6, 3, 3]
Breakpoints: [0, 1, 2, 2, 3, 3, 5]
All Costs: [512, 64, 1, 9, 73, 9, 9]
Final Cost: 9

Paragraph formation:
||-|||-||-
|||||||--
||||||-|||
|||-------

Process finished with exit code 0
```