دانشکده‌ی علوم ریاضی

ساختمان داده — ۲۱ مهر ۹۲

جلسه‌ی ۸: تصادفی‌سازی

مدرّس: دکتر شهرام خزائی — *نگارنده: آرمیتا اردشیری و فاطمه کرمانی*

۱ مسأله تصادفی‌سازی

در این مسأله قصد داریم از آرایه ورودی داده شده، آرایه‌ای کاملا تصادفی به عنوان خروجی تولید کنیم.

- **ورودی:** آرایه $\langle a_۱, a_۲, \cdots, a_n\rangle$
- **خروجی:** یک جایگشت کاملا تصادفی از ورودی

فرض کنید آرایه ورودی A به طول n داده شده است. الگوریتم ارائه شده برای تصادفی‌سازی به صورت زیر می‌باشد: آرایه X از رکوردهای با مؤلفه‌های val و priority را در نظر می‌گیریم؛ به این صورت که رکورد iام آرایه صورت

$$X[i] = \langle X[i].\text{val}, X[i].\text{priority}\rangle$$

است که مؤلفه $X[i].\text{val}$ برابر با مقدار $A[i]$ و مؤلفه $X[i].\text{priority}$ عددی تصادفی در بازه $[۱, t]$ می‌باشد که مقدار t را در ادامه تعیین خواهیم کرد. حال با مرتب کردن آرایه X برحسب مؤلفه priority، آرایه خروجی بدست می‌آید. شبه کد این الگوریتم به صورت زیر می‌باشد:

Algorithm 1 Algorithm: RANDOMIZE

```
function RANDOMIZE(array A[1 : n])
    for i = 1 to n do
        X[i].val ← A[i]
        X[i].priority ← RANDOM(1, t)
    Sort X with respect to the key "priority"
    Output X.val = (X[1].val, ··· , X[n].val)
```

برای درک بهتر الگوریتم مثال زیر را در نظر بگیرید:

مثال ۱ *آرایه A به صورت $A = (۱, ۲, ۳, ۴)$ داده شده است. فرض کنید تابع مقادیر تصادفی زیر را تولید کرده باشد:*

$$X.\text{priority} = (۳۶, ۳, ۶۲, ۱۹)$$

با توجه به الگوریتم ذکر شده، آرایه X به صورت زیر می‌باشد:

$$X = (\langle ۱, ۳۶\rangle, \langle ۲, ۳\rangle, \langle ۳, ۶۲\rangle, \langle ۴, ۱۹\rangle)$$

این آرایه را بر حسب مؤلفه priority *مرتب می کنیم، و آرایه* X *به صورت زیر حاصل می‌شود:*

$$X = (\langle ۲,۳\rangle, \langle ۴,۱۹\rangle, \langle ۱,۳۶\rangle, \langle ۳,۶۲\rangle)$$

و خروجی نهایی به صورت زیر است:

$$(۲,۴,۱,۳)$$

نکته حائز اهمیت در این مسأله این است که در تولید آرایه X.priority، اعداد نباید تکراری باشند. با توجه به اینکه برای تولید اعداد تصادفی از $\textsc{Random}(۱,t)$ استفاده کرده‌ایم، سوالی که مطرح می‌شود این است که عدد t را چقدر بزرگ انتخاب کنیم تا در انتخاب n عدد تصادفی با احتمال خوبی (مثلاً بیشتر از $۱ - \frac{۱}{۱۰۰n}$) اعداد تکراری نباشند؟
برای حل این مسأله ابتدا به سراغ حل مسأله‌ی روز تولد می‌رویم و سپس به طرز مشابهی مسأله بالا برای پیدا کردن عدد t حل می‌کنیم.

۲ مسأله‌ی روز تولد

مسأله‌ی روز تولد به این صورت مطرح می‌شود: چه تعداد افراد باید در یک اتاق حاضر باشند تا احتمال این‌که دو نفر روز تولد یکسانی داشته باشند بیش از $\frac{۱}{۲}$ باشد؟
برای حل این مسأله، برای سهولت در محاسبه به سراغ حل مسأله کلی‌تر زیر می‌رویم: می‌خواهیم از مجموعه‌ای شامل N عضو، k عضو را با جایگزینی انتخاب کنیم. احتمال اینکه عضو تکراری داشته باشیم چقدر می‌باشد؟
احتمال متمم این پیشامد، یعنی اینکه همه k عضو متمایز باشند، به‌صورت زیر محاسبه می‌شود:

$$\begin{aligned}
\Pr\{\texttt{all } k \texttt{ samples are distinct}\} &= \tfrac{N}{N} \times \tfrac{N-۱}{N} \times \tfrac{N-۲}{N} \times \cdots \times \tfrac{N-k+۱}{N} \\
&= (۱)(۱-\tfrac{۱}{N})(۱-\tfrac{۲}{N})\cdots(۱-\tfrac{k-۱}{N}) \\
&= \textstyle\prod_{i=۱}^{k-۱}(۱-\tfrac{i}{N}) \\
&\leq \textstyle\prod_{i=۱}^{k-۱} \mathrm{e}^{-\frac{i}{N}} \\
&= \mathrm{e}^{-\frac{۱}{N}\sum_{i=۱}^{k-۱} i} \\
&= \mathrm{e}^{-\frac{\binom{k}{۲}}{N}}
\end{aligned}$$

رابطه نامساوی با توجه به رابطه $۱ + x \leq \mathrm{e}^x$ نوشته شده است. برای اینکه احتمال فوق کمتر از $\frac{۱}{۲}$ باشد، کافی است داشته باشیم:

$$\begin{aligned}
\mathrm{e}^{-\frac{\binom{k}{۲}}{N}} &\leq \tfrac{۱}{۲} && \Rightarrow \\
-\tfrac{\binom{k}{۲}}{N} &\leq \ln\tfrac{۱}{۲} && \Rightarrow \\
۲N\ln ۲ &\leq k(k-۱) && \Rightarrow \\
\tfrac{(۱+\sqrt{۱+(۸\ln ۲)N})}{۲} &\leq k
\end{aligned}$$

با توجه به عبارت بدست آمده، مشخص است k از مرتبه $\Theta(\sqrt{N})$ می‌باشد.
روش دیگری برای حل مسأله روز تولد استفاده از متغیر تصادفی نشانگر می‌باشد. در اینجا با این روش هم مسأله را حل می‌کنیم و نشان می‌دهیم همان مقدار $\Theta(\sqrt{N})$ برای k بدست می‌آید.
متغیر تصادفی X_{ij} را متغیر تصادفی نشانگر پیشامد اینکه نمونه‌های iام و jام یکسان باشند، تعریف می‌کنیم؛ یعنی:

$$X_{ij} = I\{\ i\,\text{th and } \&\ j\,\text{th sample are equal}\}$$

متغیر X تعداد زوج مقادیری را که دارای مقدار یکسان هستند را نشان می دهد و به صورت زیر تعریف می‌شود:

$$X = \sum_{i=1}^{k} \sum_{j=i+1}^{k} X_{ij}$$

برای وجود افراد با روز تولد یکسان با احتمال خوبی (مثلاً $\frac{1}{2}$) کافی است مقدار متوسط این متغیر، یعنی $\mathrm{E}[X]$، برابر با $\Theta(1)$ باشد. با محاسبه $\mathrm{E}[X]$ به عبارت زیر می‌رسیم:

$$\mathrm{E}[X] = \sum_{i=1}^{k} \sum_{j=i+1}^{k} E[X_{ij}] = \sum_{i=1}^{k} \sum_{j=i+1}^{k} \frac{1}{N} = \frac{\binom{k}{2}}{N}$$

عبارت بدست آمده را برابر با $\Theta(1)$ قرار می‌دهیم:

$$\frac{\binom{k}{2}}{N} = \Theta(1) \to k = \Theta(\sqrt{N})$$

توجه کنید که طبق مطالب بیان شده در جلسه گذشته مقدار $\mathrm{E}[X_{ij}]$ برابر با احتمال وقوع X_{ij} می‌باشد. پس در محاسبات بالا به جای $\mathrm{E}[X_{ij}]$ مقدار $\frac{1}{N}$ را قرار دادیم، که احتمال برابری روز تولد دو نفر می‌باشد.
مشاهده می‌شود که جواب بدست آمده مشابه حالت قبل می‌باشد، اما محاسبات ساده‌تر می‌شود.

۳ محاسبه مقدار t

حال به مسأله اصلی برای پیدا کردن مقدار t بازمی‌گردیم . هدف انتخاب t به گونه ای است که وقتی از بین t عدد، n عدد را به تصادف و با جایگذاری انتخاب می‌کنیم، احتمال وجود عضو تکراری خیلی کم (مثلاً کمتر از $\frac{1}{100n}$) باشد.
با توجه به رابطه احتمالی که در بالا به‌دست آوردیم کافی است داشته باشیم:

$$e^{-\frac{\binom{n}{2}}{t}} = 1 - \frac{1}{100n}\ .$$

اگر از تقریب $e^{-x} \approx 1 - x$ برای x های کوچک استفاده کنیم، داریم:

$$\frac{\binom{n}{2}}{t} \approx \frac{1}{100n} \Rightarrow \frac{n^2}{2t} \approx \frac{1}{100n} \Rightarrow\ t \approx 50n^3.$$

با بدست آمدن مقدار مناسب برای t، الگوریتم Randomize نوشته شده در بخش قبل را به صورت زیر اصلاح می‌کنیم:

Algorithm 2 Algorithm: Randomize

```
function Randomize(array A[1 : n])
    for i = 1 to n do
        X[i].val ← A[i]
        X[i].priority ← Random(1, 50n^3)
    Sort X with respect to the key "priority"
    Output ⟨X[1].val, ··· , X[n].val⟩
```

سؤال ۱ *الگوریتم فوق چند بیت تصادفی استفاده می‌کند؟*

۴ تصادفی‌سازی درجا

۱.۴ الگوریتم

در این بخش می‌خواهیم الگوریتمی برای تصادفی‌سازی به صورت درجا ارائه دهیم. منظور از الگوریتم درجا این است که در حین اجرای الگوریتم از حافظه اضافی برای ذخیره اطلاعات استفاده نمی‌شود، برخلاف الگوریتم بخش پیشین که در آن از آرایه X.priority برای ذخیره اولویت‌ها استفاده کردیم.

Algorithm 3 Algorithm: RANDOMIZE-IN-PLACE

```
function RANDOMIZE-IN-PLACE(array A[1 : n])
    for i = 1 to n do
        j ← RANDOM(i, n)
        Swap A[i] ↔ A[j]
```

نکته ۱ *دقت کنید که در حلقه* **for**، *در آخرین تکرار (یعنی به ازای $i = n$) عملاً محتویات آرایه تغییر نمی‌کند. بنابراین، می‌توان حلقه را تا $i = n - 1$ اجرا نمود. با این وجود، برای ساده کردن اثبات صحت الگوریتم، اجرای حلقه تا $i = n$ در نظر گرفته می‌شود.*

۲.۴ اثبات درستی الگوریتم

قضیه ۱ *الگوریتم ارائه شده، یک جایگشت کاملاً تصادفی از آرایه ورودی تولید می‌کند.*

برهان. ابتدا تعریف زیر را ارائه می‌کنیم.

تعریف ۱ *یک kـ جایگشت از یک دنباله n عضوی، یک دنباله k تایی دلخواه از بین n عضو دنباله، بدون تکرار، است.*

تعداد kـ جایگشت‌های یک دنباله n عضوی برابر $\frac{n!}{(n-k)!}$ است. باید ثابت کنیم که خروجی الگوریتم هر یک از nـ جایگشت‌های ممکن را با احتمال $\frac{1}{n!}$ تولید می‌کند. برای اثبات درستی الگوریتم، از ناوردایی حلقه به فرم زیر استفاده می‌کنیم:

ناوردایی حلقه: درست بعد از مقدار دهی متغیر حلقه‌ی **for** با مقدار i، هرکدام از $(i-1)$ ـ جایگشت‌های ممکن با احتمال $\frac{(n-i+1)!}{n!}$ در $i-1$ خانه‌ی اول آرایه A قرار می‌گیرند.

نشان می‌دهیم این ویژگی ناوردایی قبل از اجرای اولین حلقه برقرار است، در طول اجرای هر حلقه برقرار می‌ماند و از این ویژگی در پایان حلقه برای اثبات درستی الگوریتم استفاده می‌کنیم.

- **مرحله آغاز:** قبل از شروع حلقه اول مقدار i برابر با ۱ است. بنابراین طبق ویژگی ناوردایی، زیر آرایه $A[1 \cdots 0]$ باید با احتمال $1 = \frac{(n-i+1)!}{n!}$ هر ۰ـ جایگشت ممکن را داشته باشد. با توجه به اینکه زیرآرایه $A[1 \cdots 0]$ تهی می‌باشد و یک ۰ـ جایگشت هم عنصری ندارد، زیرآرایه $A[1 \cdots 0]$ هر ۰ـ جایگشتی را با احتمال ۱ شامل می‌شود.
- **مرحله نگه‌داری:** فرض می‌کنیم قبل از اجرای حلقه‌ی i ام، هر کدام از $(i-1)$ ـ جایگشت‌ها به احتمال $\frac{(n-i+1)!}{n!}$ در زیرآرایه $A[1 \cdots i-1]$ ایجاد می‌شوند. می‌خواهیم نشان دهیم پس از اجرای حلقه‌ی i ام، هر i ـ جایگشت با احتمال $\frac{(n-i)!}{n!}$ در زیرآرایه $A[1 \cdots i]$ ایجاد می‌شود.

احتمال وقوع یک جایگشت خاص مثل $(x_۱, x_۲, \cdots, x_{i-۱}, x_i)$ را E می‌نامیم. برای تولید این جایگشت لازم است زیرآرایه‌ی $(x_۱, x_۲, \cdots, x_{i-۱})$ در $A[۱ \cdots i-۱]$ ایجاد شود، که احتمال آن را $E_۱$ می‌نامیم. همچنین لازم است در اجرای حلقه i ام الگوریتم، عنصر x_i در جایگاه i ام قرار گیرد که احتمال آن را با $E_۲$ نشان می‌دهیم. در نتیجه داریم:

$$\Pr\{E\} = \Pr\{E_۱ \cap E_۲\} = \Pr\{E_۲ \mid E_۱\} \Pr\{E_۱\}$$

با توجه به اینکه در حلقه i ام الگوریتم، مقدار x_i به صورت تصادفی از میان $n - i + ۱$ مقدار موجود در زیرآرایه $A[i \cdots n]$ انتخاب می‌شود، پس احتمال $\Pr\{E_۲ \mid E_۱\}$ برابر است با $\frac{۱}{n-i+۱}$.

$$\begin{aligned}
\Pr\{E_۱ \cap E_۲\} &= \Pr\{E_۲ \mid E_۱\} \Pr\{E_۱\} \\
&= \frac{۱}{n-i+۱} \times \frac{(n-i+۱)!}{n!} \\
&= \frac{(n-i)!}{n}
\end{aligned}$$

- **مرحله پایان:** در حلقه پایانی مقدار i برابر است با $n + ۱$ و آرایه $A[۱ \cdots n]$ یک n ـ جایگشت تصادفی می‌باشد که احتمال آن برابر است با:

$$\frac{(n - n)!}{n!} = \frac{۱}{n!}$$

در نتیجه هر جایگشت به احتمال برابر ایجاد می‌شود و الگوریتم عملیات تصادفی‌سازی را به درستی انجام می‌دهد. ■