دانشکده‌ی علوم ریاضی

آنالیز الگوریتم‌ها — ۱۵ بهمن ۹۱

جلسه‌ی ۱: مقدمه

مدرّس: دکتر شهرام خزائی — *نگارنده: مریم غرقانی*

۱ مقدمه

الگوریتم‌ها در همه حوزه‌های علوم کامپیوتر، از جستجو در اینترنت گرفته تا وصل شدن موبایل شما به شبکه، کاربرد دارند. در این درس با روش‌های مختلف حل مسئله آشنا می‌شویم و با بررسی پیچیدگی زمانی الگوریتم‌ها سر و کار داریم. برای یك مسئله ممکن است چندین راه حل یا الگوریتم متفاوت وجود داشته باشد. به طور کلی وقتی با یك مسئله مواجه می‌شویم ابتدا سعی می‌کنیم یك الگوریتم برای حل آن ارائه دهیم و در صورت پیدا کردن یك الگوریتم، به دنبال یك راه حل بهتر می‌گردیم. در واقع هدف، پیدا کردن الگوریتم کارا است. در ادامه روش تقسیم و حل[1] برای حل مسئله را تعریف می‌کنیم و به بررسی چند مثال می‌پردازیم:

۲ الگوریتم‌های تقسیم و حل

در این روش یك مسئله را به زیر مسئله‌های کوچکتر تقسیم می‌کنیم، هر یك از زیر مسئله‌ها را حل می‌کنیم و سپس جواب آنها را با هم ادغام می‌کنیم تا به جواب مسئله اصلی برسیم.

۱.۲ حاصل‌ضرب دو عدد n رقمی

پیچیدگی زمانی الگوریتم معمولی حاصل‌ضرب $O(n^2)$ است. برای مثال حاصل‌ضرب دو عدد ۵۴۶۵ و ۱۴۲۳ را به صورت زیر محاسبه می‌کنیم:

```
      ۵۴۶۵
      ۱۴۲۳
   ---------
     ۱۶۳۹۵
+   ۱۰۹۳۰۰
+  ۲۱۸۶۰۰۰
+  ۵۴۶۵۰۰۰
   ---------
   ۷۷۷۶۶۹۵
```

حال دو عدد n رقمی x و y را به روش تقسیم و حل در هم ضرب می‌کنیم:

[1] divide and conquer

$$x = \begin{array}{|c|c|}\hline a & b \\ \hline\end{array} \Rightarrow x = 10^{n/2}a + b$$
$$y = \begin{array}{|c|c|}\hline c & d \\ \hline\end{array} \Rightarrow y = 10^{n/2}c + d$$

$$\Rightarrow xy = 10^{n}ac + 10^{n/2}(ad + bc) + bd$$

پیچیدگی زمانی روش بالا هم $O(n^2)$ است. چون ۴ حاصل‌ضرب را باید محاسبه کنیم (اثبات این موضوع را در جلسات بعد با استفاده از قضیه اصلی خواهیم دید.). برای اینکه پیچیدگی زمانی الگوریتم تقسیم و حل را کم کنیم باید تعداد ضرب‌ها را کاهش دهیم. برای این کار می‌توانیم به جای محاسبه ad و bc در عبارت بالا، حاصل‌ضرب $P = (a+b)(c+d)$ را بدست آوریم و حاصل‌ضرب‌های ac و bd را از P کم کنیم تا به عبارت $ad+bc$ برسیم. این ایده اولین بار توسط گوس برای محاسبه حاصل‌ضرب اعداد مختلط مطرح شد و تعداد حاصل‌ضرب‌ها را از ۴ به ۳ کاهش می‌دهد. در جلسه آینده نشان خواهیم داد که پیچیدگی زمانی الگوریتم بازگشتی‌ای که از سه حاصل‌ضرب استفاده می‌کند کمتر از $O(n^2)$ است.

نکته ۱ *هیچ وقت راضی نباشید. همیشه از خود بپرسید که آیا راه حل بهتری وجود دارد؟*

سؤال ۱ *یك الگوریتم کارا برای ضرب ماتریس‌ها به صورت بازگشتی ارائه دهید.*

۲.۲ الگوریتم مرتب‌سازی ادغامی

یك آرایه با n درایه داریم که می‌خواهیم آن را مرتب کنیم. پیچیدگی زمانی بیشتر الگوریتم‌های مرتب‌سازی $O(n^2)$ است. مرتب‌سازی ادغامی[۲] یکی از الگوریتم‌های مرتب‌سازی است که با روش تقسیم و حل آرایه را مرتب می‌کند. این الگوریتم آرایه ورودی را به دونیمه با طول تقریبا مساوی تقسیم و آنها را به طور بازگشتی مرتب می‌کند. سپس نیمه‌های مرتب شده با هم ادغام می‌گردند.

مثال ۱ *شل زیر این روند را بر روی یک آرایه نشان می‌دهد.*

۵	۴	۱	۸	۷	۲	۶	۳

$\Downarrow$

۱	۴	۵	۸		۲	۳	۶	۷

$\Downarrow$

۱	۲	۳	۴	۵	۶	۷	۸

تابع MergeSort این که خود از تابع Merge استفاده می‌کند این الگوریتم را برای حالتی که طول آرایه توانی از دو باشد نشان می‌دهد که تبدیل آن به الگوریتمی که در حالت کلی کار کند سر راست است.

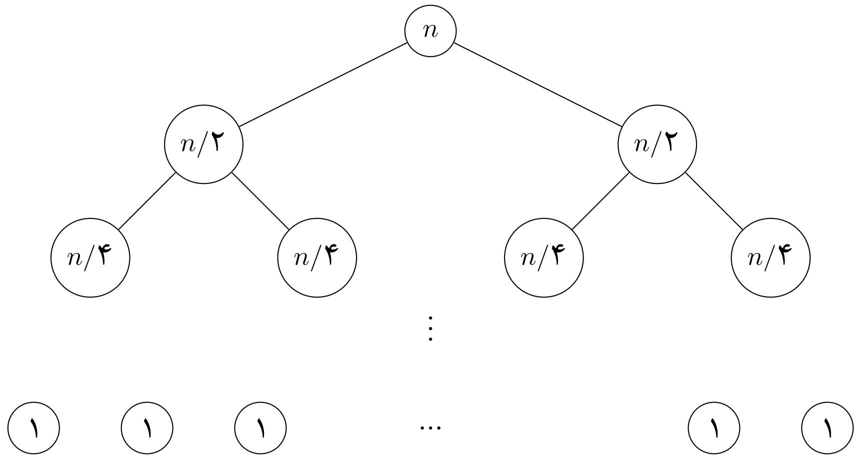
[۲] merge sort

Algorithm 1 Algorithm: MergeSort

```
function MergeSort(array A[1 : n])
    [assumes n is a power of two]
    L ← left half of A
    R ← right half of A
    A ← MergeSort(L)
    B ← MergeSort(R)
    C ← Merge(A, B)
    return C
```

```
function Merge(array A[1 : m], array B[1 : m])
    [assumes A[i] = B[i] = 0 for i > m]
    i ← 1, j ← 1
    for k = 1 to m do
        if A[i] < B[j] then
            C[k] ← A[i]
            i ← i + 1
        else
            C[k] ← B[j]
            j ← j + 1
    return C
```

با توجه به الگوریتم بالا، حداکثر زمان اجرای رویه ادغام برای آرایه به طول m برابر $۶m$ خط کد است. حال زمان اجرای این الگوریتم را محاسبه می‌کنیم. برای این کار درخت بازگشت این الگوریتم را رسم می‌کنیم:



همانطور که مشاهده می‌کنید، عمق درخت $\log n + ۱$ است. با توجه به شبه کدی که در بالا نوشتیم، برای ادغام دو آرایه مرتب شده mعضوی باید حداکثر $۶m$ خط کد اجرا شود. در سطح jام درخت بالا، آرایه‌های مرتب شده به طول $\frac{n}{۲^j}$ داریم و در کل ۲^j عمل ادغام باید انجام دهیم. پس در سطح jام از درخت، در مجموع حداکثر $۲^j ۶(\frac{n}{۲^j}) = ۶n$ خط کد اجرا می‌شود. زمان اجرای الگوریتم مرتب‌سازی ادغامی برابر است با مجموع پیچیدگی همه سطح‌ها که حداکثر برابر است با $۶n(\log n + ۱)$.