مدرس: دکتر مجتبی رفیعی

نیمسال اول ۱۴۰۰-۱۴۰۱

ساختمان داده‌ها و الگوریتم‌ها

جلسه ۵: انواع الگوریتم‌ها از حیث رویکرد حل مساله

نگارنده: آیدا عمومی

۵ آبان ۱۴۰۰

فهرست مطالب

۱ جمع‌بندی - گام‌های حل مساله

در حالت کلی، برای حل یک مسأله می‌بایست گام‌های زیر را به ترتیب دنبال کنیم:

۱): بررسی دقیق مساله: در نظر گرفتن حالت های خاص، فرضیات و ...،

۲): نوآوری: پیدا کردن یک راه حل برای مساله،

۳): بیان رسمی (Formal) راه حل مساله در قالب یک الگوریتم،

۴): نشان دادن صحت و درستی الگوریتم ارایه شده،

۵): تحلیل کارایی الگوریتم،

۶): و نهایتا پیاده سازی الگوریتم.

توجه

با گام‌های ۱ تا ۳ در درس طراحی الگوریتم آشنا می‌شویم. محوریت درس ساختمان داده بیشتر روی گام‌های ۴ تا ۶ است.

۲ انواع الگوریم‌ها از حیث رویکرد حل مساله

انواع الگوریتم‌ها از حیث رویکرد حل مساله به قرار زیر است:

۱. **الگوریتم‌های قطعی / الگوریتم‌های غیر قطعی**[1]،

۲. **الگوریتم‌های تقریبی**[2]،

۳. **الگوریتم‌های تصادفی** [3].

۱.۲ الگوریتم قطعی

به الگوریتم‌های اطلاق می‌شود که اگر در یک وضعیت مشخصی باشیم آنگاه با توجه به شرایط فعلی، وضعیت بعدی کاملا مشخص است. در ادامه مثالی برای این نوع از الگوریتم‌ها آورده شده است.

عنوان مساله: جستجو

صورت مساله: یک آرایه از اعدادی که لزوما مرتب نیستند داده شده است، قرار است یک مقدار خاص مانند x را در آن جستجو کنیم.

Algorithm 1 DSearch $(A[1 \dots n, x])$

```
▷ Array A is not necessarily sorted.
for i = 1 to n do
    if (A[i] == x) then
        return True
    else
        return False
```

الگوریتم فوق، یک الگوریتم قطعی با پیچیدگی $\mathcal{O}(n)$ است؛ یعنی در بدترین حالت n تا مقایسه انجام می‌شود.

۲.۲ الگوریتم غیر قطعی

در مقابل الگوریتم‌های قطعی، *الگوریتم‌های غیر قطعی* را داریم؛ به الگوریتم‌های اطلاق می‌شود که اگر در یک وضعیت مشخصی باشیم آنگاه با توجه به شرایط فعلی، چندین وضعیت بعدی خواهیم داشت. لازم به ذکر است که در حالت کلی، تاکنون چنین امکانی تنها در تئوری مطرح است.

برای بیان الگوریتم‌های غیر قطعی نیازمند در نظر گرفتن توابعی هستیم که در ادامه شرح هر یک از آنها آورده شده است:

۱. **تابع choose(S)**: این تابع به طور هم زمان $|S|$ کپی از سیستم ایجاد می‌کند و هر یک را به یکی از عناصر مجموعه S اختصاص می‌دهد.

[1] Deterministic Algorithms/Non-deterministic Algorithms
[2] Approximation Algorithms
[3] Randomized Algorithms

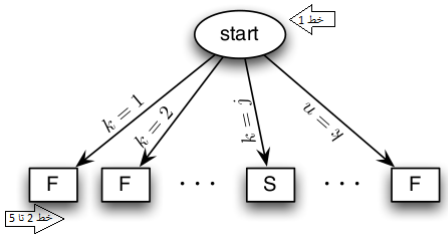
۲. **تابع ()success**: این تابع به معنی توقف الگوریتم به همراه پیدا کردن جواب است.

۳. **تابع ()failure**: این تابع به معنی توقف الگوریتم به همراه عدم پیدا کردن جواب است.

Algorithm 2 NDSearch $(A[1 \dots n, x])$

```
1: ▷ Array A is not necessarily sorted.
2: i ← choose(S = {1, ..., n})
3: for i = 1 to n do
4:     if (A[i] == x) then
5:         success()
6:     else
7:         failure()
```

با فرض اینکه $A[j] = x$ است، شکل زیر نحوه اجرای الگوریتم غیرقطعی جستجو را نشان می‌دهد.



شکل ۱: نحوه اجرای الگوریتم غیرقطعی برای جستجو

۳.۲ الگوریتم‌های تقریبی

به الگوریتم‌هایی اطلاق می‌شود که با آن اقدام به حل مساله به منظور دستیابی به یک جواب نزدیک جواب اصلی و نه دقیقا جواب اصلی می‌کنیم. علت استفاده از چنین رویکردی به عنوان مثال می‌تواند این باشد که حل مساله برای جواب دقیق **کارا** نیست، اما می‌توان به طور کارایی جواب تقریبی بدست آورد. در ادامه مثالی برای این نوع از الگوریتم‌ها آورده شده است.

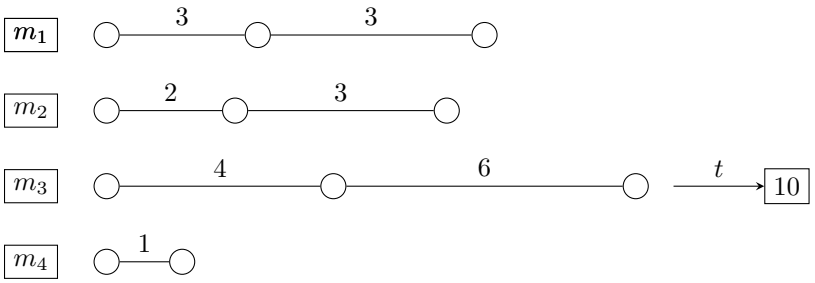
عنوان مساله: تخصیص پردازش

صورت مسأله: تقسیم n پردازش روی m ماشین به نحوی که زمان کلی پردازش همه آنها کمینه بشود.

Process: 3, 2, 4, 1, 3, 3, 6 → n=7

Machine: m_1, m_2, m_3, m_4 → m=4

راه حل اول: پردازش‌ها را به ترتیب بین ماشین‌ها تقسیم می‌کنیم.



زمان مورد نیاز برای پردازش $t = 10$ می‌باشد.

تجربیات لاتک‌نویسی

- **ایجاد نیم‌فاصله:** اگر دستوی های shift+space یا shift+B ایجاد نیم‌فاصله نکردند، می‌توانید با دستور shift+ctrl+2 نیم‌فاصله را درج کنید.

- **عدم نمایش EndIf و EndFor:** اگر در الگوریتم خود دستورات If یا For داشته باشید در پایان باید از EndIf و EndFor استفاده کنید. خروجی اجرای آن مشابه زیر است:

```
1. for i=1 to n do
2.     a=i+1
3. end for
```

اگر در ابتدای سند پکیج را اضافه بکنید، آن قطعه از الگوریتم فوق به صورت زیر نمایش داده می‌شود:

```
1. for i=1 to n do
2.     a=i+1
```

- **استفاده از پاورقی:** ما اکثراً از پاورقی برای نوشتن معادل انگلیسی کلمات استفاده می‌کنیم. دستور پاورقی {}\footnote است. هنگامی که از این دستور استفاده شود متن لاتین یا انگلیسی دچار جابجایی خواهد شد. در نتیجه ترجیحاً از دستور {}\LTRfootnote استفاده کنید.

- **ایجاد فاصله مشخص:** از دستور {}\hspace می‌توانید باری ایجاد فاصله با مثدار مشخص مثلاً 10mm استفاده نمایید.

- **چپ به راست کردن لیست:** برای چنین کاری می‌توانیم یکی از دو روش زیر را به کار گیریم:
  - کل لیست را در {}\lr قرار دهیم
  - بین دستورات {latin}\begin و {latin}\end قرار دهیم

- **درج برخی علائم:** برای درج علائم مختلف باید کد دستوری آن‌ها رااستفاده نماییم. برخی از این علائم به عنوان مثال به صورت زیر به قابل ملاحصه هستند:

- \: \textbackslash
- {} : \{\}
- → : $\rightarrow$
- ← : $\leftarrow$
- ↑ : $\uparrow$
- ↓ : $\downarrow$
- ↔ : $\leftrightarrow$
- ↕: $\updownarrow$
- ↙ : $\swarrow$
- ↘ : $\searrow$
- ↖ : $\nwarrow$
- ↗ : $\nearrow$
- △ : $\triangle$
- ▽ : $\triangledown$
- ◁ : $\triangleright$
- ▷ : $\triangleright$
- △ : $\bigtriangleup$
- ▽ : $\bigtriangledown$