Algorithms and Data Structures
Conspectus

Didubeli Soso

September 21, 2023 - October 1, 2023

1 Reviewing

Basically, I knew everything from "*Discrete Structures*", solving difference equations was simply an exercise, nothing to write down. Only new thing was *master theorem*.

Master Theorem: Let a, b, c be nonnegative constants. The solution to the recurrence

$$T(n) = \begin{cases} b, & \text{for } n = 1, \\ aT(n/c) + bn, & \text{for } n > 1 \end{cases}$$

for n a power of c is

$$T(n) = \begin{cases} O(n), & \text{if } a < c, \\ O(n \log n), & \text{if } a = c, \\ O(n^{log_c a}), & \text{if } a > c. \end{cases}$$

$\sum_{i=1}^{k} \left(\frac{n}{2^i} \cdot \log_2 \frac{n}{2^i} \right) + 2 \cdot \log_2 2$

2 Binary Search

Require: Sorted array $a[1..n]$, $a[1] < a[2] < \cdots < a[n]$ and element x.

Ensure: $\begin{cases} m, & \text{if there is an } 1 \leq m \leq n \text{ with } a[m] = x \\ -1, & \text{otherwise} \end{cases}$

$l \leftarrow 0$
$r \leftarrow n + 1$
while $l + 1 < r$ **do**
  $m \leftarrow \lfloor \frac{l+r}{2} \rfloor$
  if $a[m] = x$ **then**
    return m
  end if
  if $a[m] < x$ **then**
    $l \leftarrow m$
  else
    $r \leftarrow n$
  end if
end while
return -1

Algorithm 1: Binary Search

Binary Search finds desired element in $O(\log n)$ time. Input array must be sorted!

3 Sorting

Given input sequence $(a(1), \ldots, a(n))$ or set $\{a(1), \ldots, a(n)\}$ sorting algorithms output sequence $(a(\pi(1)), \ldots, a(\pi(n))$ or set $\{a(\pi(1)), \ldots, a(\pi(n)\}$ which are sorted by certain order(ascending, descending, or custom). Sorting is done by comparisons.

3.1 Merge Sort

Given $(a(1), \ldots, a(n))$, $(b(1), \ldots, b(m))$ with $(a(1) \leq \cdots \leq a(n))$, $(b(1) \leq \cdots \leq b(nm))$, **Merge Sort** outputs merged(sorted) sequence $(c(1), \ldots, c(n+m))$ with $(c(1) \leq \cdots \leq c(n+m))$.

$$\text{merge}((a(1),\ldots,a(n)),(b(1),\ldots,b(m))) = \begin{cases} a(1)\circ\text{merge}((a(2),\ldots,a(n)),(b(1),\ldots,b(m))), & a(1)\le b(1)\\ b(1)\circ\text{merge}((a(1),\ldots,a(n)),(b(2),\ldots,b(m))), & a(1)>b(1)\end{cases}$$

This is done in $n+m-1$ comparisons.

$$\text{sort}((a(1),\ldots,a(n)) = \text{merge}(\text{sort}((a(1),\ldots,a(n/2)),\text{sort}((a(n/2+1),\ldots,a(n)))$$

$S(1)=0$, $S(n) < n/2+n/2+2\cdot S(n/2) = 2\cdot S(n/2)+n$ Let $n=2^k$, $k\in\mathbb{N}$.

$f(1)=a$, $f(n)=2\cdot f(n/2)+b\cdot n$
After guessing the general formula: $f(n)=2^x\cdot f(n/2^x)+x\cdot b\cdot n$
Stop recursion at $n/2^x=1 \implies x=\log n$.
Conjecture: $f(n)=n\cdot f(1)+b\cdot n\cdot\log(n)=a\cdot n+b\cdot n\cdot\log(n)$

$$S(n) < n\cdot\log(n)$$

NO deterministic sorting algorithm can run faster than $O(n\cdot\log(n))$

3.2 Quicksort

Input: $(a(1),\ldots,a(n))$ or set $\{a(1),\ldots,a(n)\}$(It is assumed that $a(i)$ are mutually distinct).

Random experiment: choose "splitter" $s\in\{a(1),\cdots,a(n)\}$. We are dealing with uniform distribution, so all n splitters are equally likely.

$$A_< = \{a\in A \mid a<s\} \text{ and } A_> = \{a\in A\mid a>s\}$$
$$\text{sort}(A)=\text{sort}(A_<)\circ s\circ\text{sort}(A_>)$$

Expected runtime is $T(n)\le n+(1/n)\cdot\sum_{i=1}^{n}(T(i-1)+T(n-i))$

4 Elementary Probability Theory

- $W = (S, p) \leftarrow$ probability space, describing a random experiment.
- $S \leftarrow$ set, finite or countable, sample space.
- $s \in S \leftarrow$ sample, possible outcome of the experiment.
- $p : S \to [0, 1] \leftarrow$ probability function.
- $\sum_{s \in S} p(s) = 1 \leftarrow p(s)$: probability that the outcome is s.
- $A \subseteq S \leftarrow$ event.
- $p(A) = \sum_{a \in A} p(s) \leftarrow$ probability of A.
- $a \in S \leftarrow$ elementary event.

$A, B \subseteq S, p(A) \neq 0 \implies p(B \mid A) = \dfrac{p(B \cap A)}{p(A)}$ [Probability of B given A]

$A, B \subseteq S$ independent iff $p(A \cap B) = p(A) \cdot p(B)$

$A \subseteq S_1, B \subseteq S_2 \leftarrow$ events of the single experiments.
$e_1(A) = A \times S_2, e_2(B) = S_1 \times B \leftarrow$ embedding into $S_{\text{(Other events do not matter)}}$.

Lemma 2:
Embedded events have the probability of the original events in the original space.

Lemma 3:
Embedded events from different probability spaces are independent.

Lemma 4:
$S_1 \times \cdots \times S_n$ is a probability space.

Lemma 5:
Embedded events $e_i(A_i) = S_1 \times \cdots \times S_{i-1} \times A_i \times S_{i+1} \times \cdots \times S_n \subseteq S$ have the probability of the original events A_i in the original space.

Lemma 6:
Embedded events $e_1(A_1). \ldots, e_n(A_n)$ are mutually independent.

$X : S \to \mathbb{R} \leftarrow$ random variable.

Expected value of the random variable X is $E(X) = \sum_{a \in S} X(a) \cdot p(a)$

Lemma 7:
If $X, Y : S \to \mathbb{R}$ are random variables and $c \in \mathbb{R}$ is a constant, then:

- $E(c \cdot X) = c \cdot E(X)$
- $E(X + Y) = E(X) + E(Y)$

Lemma 8:
Let $X_i : S \to \mathbb{R}$, $i \in \{1, \ldots, n\}$ be random variables. Then:

$$E(\sum_{i=1}^{n} X_i) = \sum_{i=1}^{n} E(X_i)$$

Lemma 9:

$$E(X) = E(X_1) + E(X_2)$$