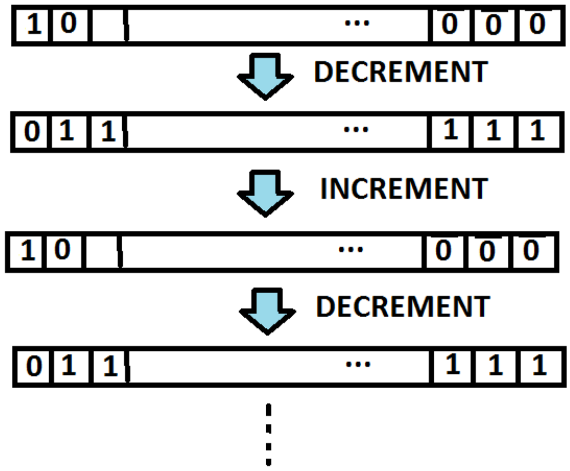
ادامه تحلیل سرشکنی

ساختمان داده
دانشکده ریاضی. دانشگاه صنعتی خواجه نصیرالدین طوسی. پاییز ۱۴۰۲

۱ شمارنده دودویی، کاهش و افزایش

در مثال شمارنده دودویی دیدیم که دنباله‌ای از n عمل افزایش Increment زمان اجرایش $O(n)$ است و این را با روشهای مختلف تحلیل سرشکنی نشان دادیم. حال اگر بخواهیم علاوه بر عمل افزایش، عمل کاهش Decrement را هم داشته باشیم (یعنی یک واحد از مقدار شمارنده کم کنیم)، دنباله‌ای از n عمل کاهش و افزایش زمان اجرایش $O(n)$ نخواهد بود. با فرض اینکه شمارنده k بیت دارد، فرض کنید مقدار کنونی شمارنده $2^{k-1}-1$ باشد. اگر اعمال Increment و Decrement را بصورت تناوبی روی این شمارنده انجام دهیم، مقدار شماره بین 2^{k-1} و $2^{k-1}-1$ نوسان می‌کند و هر بار کل بیتهای باید تغییر کند. لذا دنباله ای از n عمل افزایش و کاهش می‌تواند زمان اجرایش $O(nk)$ باشد.



۱.۱ یک ترفند برای زمان سرشکنی بهتر

در یک ایده متفاوت برای تسریع افزایش و کاهش، استفاده از دو شمارنده دودویی به جای یک شمارنده است. فرض کنید که متغیر X را داریم که در شروع کار صفر است و با عمل Increment یک واحد به آن افزوده می‌شود و با عمل Decrement یک واحد از آن کاسته می‌شود. برای شبیه‌سازی این متغیر از دو شمارنده دودویی P و N به طول k بیت استفاده می‌کنیم طوری که همیشه $X = P - N$ برقرار باشد. اینجا فرض ما بر این است که مقدار X منفی نمی‌شود و از $2^k - 1$ نیز بیشتر نمی‌شود. در شروع کار قرار می‌دهیم $P = 0$ و $N = 0$ که معادل $X = 0$ است. هر بار یک که عمل Increment داریم مقدار P را یک واحد افزایش می‌دهیم و هر بار یک عمل Decrement داریم مقدار N را یک واحد افزایش می‌دهیم. با این ترفند عمل Decrement را با Increment شبیه‌سازی می‌کنیم.

یک مشاهده به ما می‌گوید چون مقدار واقعی شمارنده $X = P - N$ است، اگر موقعیتی پیش آمد که داشته باشیم

$$P[i] = N[i] = 1$$

با خیال راحت می‌توانیم قرار دهیم

$$P[i] \leftarrow 0, \qquad N[i] \leftarrow 0$$

جون صفر کردن این بیتها تغییری در مقدار $P - N$ ایجاد نمی‌کند. لذا می‌توانیم فرض کنید در هر دو بیت متناظر P و N حداکثر یکی از بیتها 1 است. به عبارت دیگر، برای هر i داریم

$$P[i] \wedge N[i] = 0$$

مزیت این خاصیت این است که P و N می‌توانند بطور نامحدود افزایش یابند و $P-N$ معادل با مقدار شمارنده مورد نظر یعنی X باشد.
با این اوصاف عمل Increment و Decrement را می‌توانیم بصورت زیر پیاده سازی کنیم.

```
Increment(P,N):
  i ← 0
  while P[i] = 1
      P[i] ← 0
      i ← i+1
  if N[i] = 1
      N[i] ← 0
  else
      P[i] ← 1
```

```
Decrement(P,N):
  i ← 0
  while N[i] = 1
      N[i] ← 0
      i ← i+1
  if P[i] = 1
      P[i] ← 0
  else
      N[i] ← 1
```

$$\begin{array}{ccccccccccccc}
P=10001 & & P=10010 & & P=10011 & & P=10000 & & P=10000 & & P=10000 & & P=10001 \\
N=01100 & \xrightarrow{++} & N=01100 & \xrightarrow{++} & N=01100 & \xrightarrow{++} & N=01000 & \xrightarrow{--} & N=01001 & \xrightarrow{--} & N=01010 & \xrightarrow{++} & N=01010 \\
P-N=5 & & P-N=6 & & P-N=7 & & P-N=8 & & P-N=7 & & P-N=6 & & P-N=7
\end{array}$$

تحلیل سرشکنی. از روش تابع پتانسیل برای تحلیل سرشکنی استفاده می‌کنیم. مشابه حالت قبلی، از تابع پتانسیل زیر استفاده می‌کنیم.

$$\Phi(D) = \text{تعداد ۱ ها در شمارنده های } N \text{ و } P$$

فرض کنید در عمل i ام، b_{01} تعداد بیتهایی باشد که از 0 به 1 تغییر کرده و b_{10} تعداد بیتهایی باشد که از 1 به 0 تغییر کرده است. داریم

$$c_i = b_{01} + b_{10}$$

همچنین

$$\Phi(D_i) - \Phi(D_{i-1}) = b_{01} - b_{10}$$

لذا داریم

$$a_i = c_i + \Phi(D_i) - \Phi(D_{i-1}) = 2b_{01} \leq 2$$

چون در هر عمل Increment یا Decrement حداکثر یک 0 به 1 تغییر می‌یابد. نیتجه اینکه زمان n عمل متوالی Increment و Decrement حداکثر $2n$ است.