دانشگاه صنعتی امیرکبیر
دانشکده مهندسی کامپیوتر و فناوری اطلاعات

معماری سوئیچ و روترهای با کارآیی بالا
تمرین اول

پرهام الوانی

۲۴ آبان ۱۳۹۶

۱ سوال اول

هر عنصر می‌تواند در هر یک از ۱۰۰۰۰ بلوک موجود قرار بگیرد. فرض می‌کنیم یکی از ۱۰۰۰۰ بلوک خالی باشد، به این ترتیب هر عنصر ۹۹۹۹ انتخاب خواهد داشت. بنابراین احتمال خالی بودن یکی از بلوک‌ها به شرح زیر است:

$$\frac{\binom{10^4}{1} * (10^4 - 1)^{5000}}{10^{4^{5000}}} \tag{۱.۱}$$

اگر فرض کنیم متغیر تصادفی x نشان‌دهنده‌ی تعداد بلوک‌های خالی باشد، امید ریاضی x متوسط تعداد بلوک‌های خالی خواهد بود به این ترتیب داریم:

$$E[x] = \sum_{n=0}^{10^4} n \frac{\binom{10^4}{n} * (10^4 - n)^{5000}}{10^{4^{5000}}} \tag{۲.۱}$$

ابتدا دو عنصر را انتخاب کرده و در یک بلوک مشخص قرار می‌دهیم و سایر عناصر ۹۹۹۹ انتخاب خواهند داشت.

$$\frac{\binom{10^4}{1} * \binom{5000}{2} * (10^4 - 1)^{5000-2}}{10^{4^{5000}}} \tag{۳.۱}$$

اگر فرض کنیم متغیر تصادفی y نشان‌دهنده‌ی تعداد بلوک‌های دو عنصری باشد متوسط تعداد بلوک‌های دو عنصری امید ریاضی y خواهد بود به این ترتیب داریم:

$$E[x] = \sum_{n=0}^{2500} n \frac{\binom{10^4}{n} * \prod_{i=0}^{n-1} \binom{5000-2i}{2} * (10^4 - n)^{5000-2n}}{10^{4^{5000}}} \tag{۴.۱}$$

۲ سوال دوم

برای جستجو در اولین گام بلوک مورد نظر مشخص می‌شود و سپس جستجو در همان بلوک صورت می‌پذیرد. از آنجایی که جستجو در بلوک صورت می‌پذیرد پس زمان آن به اندازه‌ی بلوک وابسته است، اندازه‌ی هر بلوک در صورتی که تعداد عناصر آن‌ها با یکدیگر برابر باشد n/m می‌باشد.

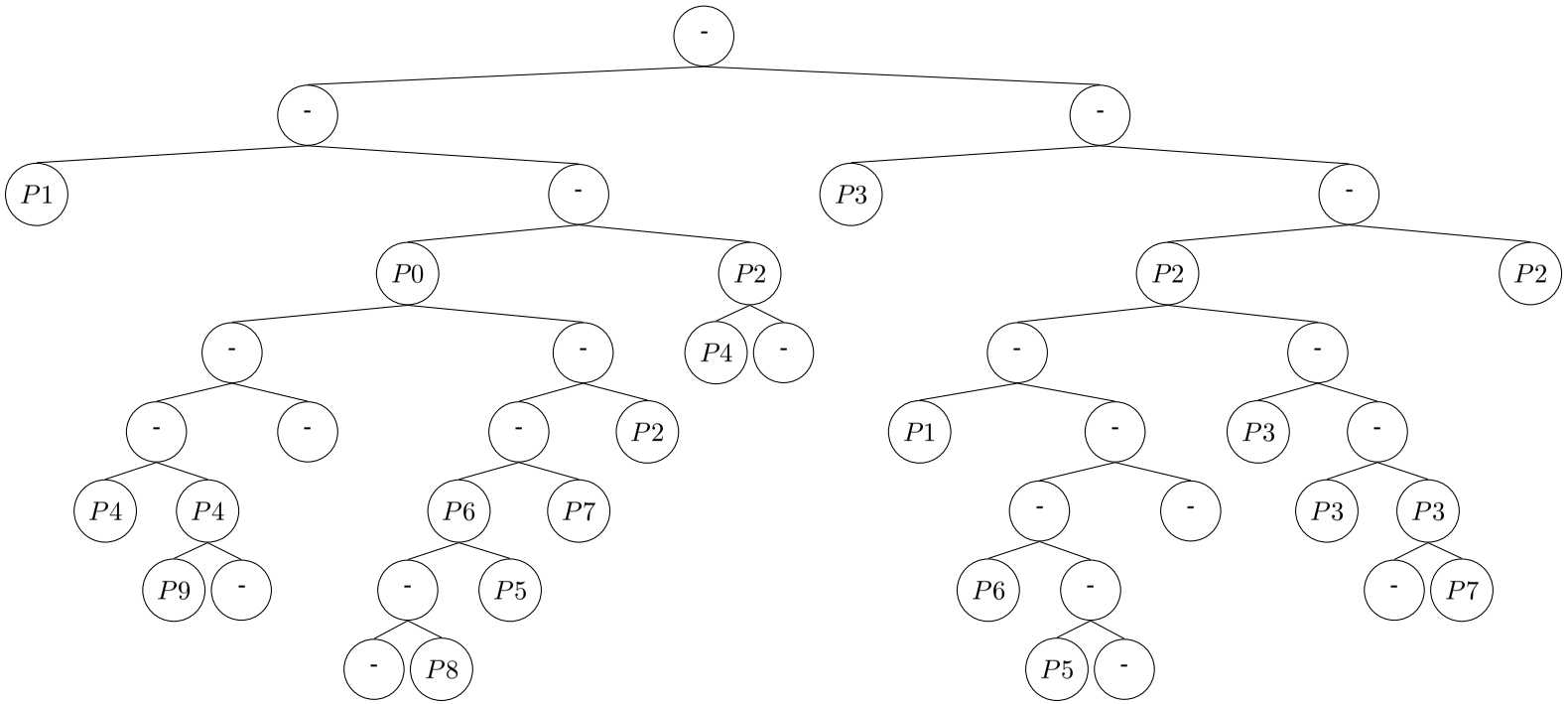
$$search = O(n/m)$$

حافظه‌ی مصرفی این روش اگر تعداد عناصر همه‌ی بلوک‌ها با یکدیگر برابر باشد به اندازه‌ی عناصر موجود خواهد بود ولی در صورتی که این امر اتفاق نیافتد حافظه‌ی مصرفی این روش از تعداد عناصر موجود بیشتر خواهد بود.

$$memroy = \Omega(n/m)$$

برای به روزرسانی تنها نیاز است که بلوک موردنظر پیدا شود و در ادامه محتوای آن به روزرسانی شود بنابراین زمان اجرای به روزرسانی با جستجو تفاوتی ندارد.

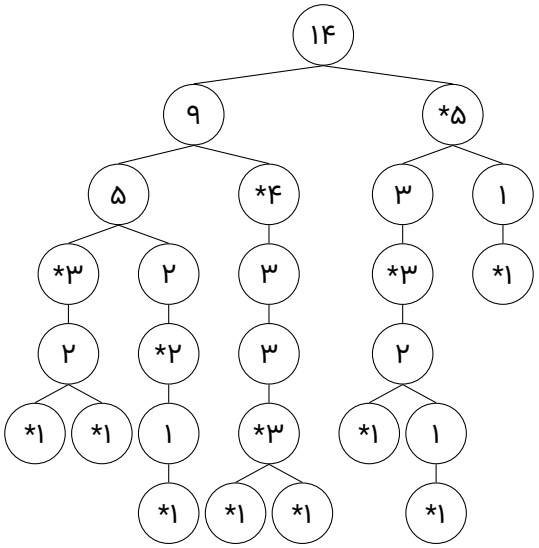
۳ سوال سوم



ד

۴ سوال چهارم

در ابتدا درخت دودویی که با تعداد prefixها شماره‌گذاری شده است، رسم می‌کنیم.



روش subtree-spliting

Index	Bucket Prefixes	Bucket Size	Covering Prefix
000^*	$000^*, 00010^*, 00011^*$	3	000^*
00^*	$0011^*, 001110^*$	2	—
$0*$	$01^*, 01101^*, 011010^0, 011011^*$	4	—
10^*	$100^*, 10010^*, 100110^*$	3	1^*
$*$	$1^*, 111^*$	2	—

ورش post-order spliting

Index	Bucket Prefixes	Bucket Size	Covering Prefix
$000^*, 00111^*$	$000^*, 00010^*, 00011^*, 001110^*$	4	$000^*, 0011^*$
$00^*, 011^*$	$0011^*, 01101^*, 011010^*, 011011^*$	4	01^*
$0^*, 10^*$	$01^*, 100^*, 10010^*, 100110^*$	4	1^*
$*$	$1^*, 111^*$	2	—