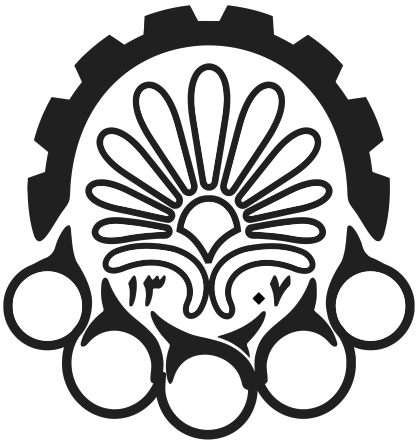
معماری افزاره‌های شبکه

دکتر صبائی



دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

دانشکده مهندسی کامپیوتر

رضا آدینه پور ۴۰۲۱۳۱۰۵۵

تمرین سری دوم

۱۱ آبان ۱۴۰۳

معماری افزاره‌های شبکه

تمرین سری دوم

رضا آدینه پور ۴۰۲۱۳۱۰۵۵

دانشکده مهندسی کامپیوتر

سوال اول

فرض کنید یک مسیریاب با استفاده از الگوریتم تطابق بیشترین طول پیشوند (Longest Prefix Matching)، جلورانی (Forwarding) بسته‌ها را انجام می‌دهد. در صورتی که جدول جلورانی مسیریاب به صورت زیر باشد:

Prefix	Next Hop
1010*	A
101*	B
101011*	C
100*	D

گام بعدی بسته‌های دریافتی با آدرس‌های مقصد زیر را بدست آورید:

- 10101101

پاسخ

طولانی ترین تطابق با *101011 است بنابراین مقصد C است.

- 10111101

پاسخ

طولانی ترین تطابق با *101 است بنابراین مقصد B است.

- 10001101

پاسخ

طولانی ترین تطابق با *100 است بنابراین مقصد D است.

سوال دوم

روش‌های مختلفی برای جستجوی آدرس IP در جدول جلورانی وجود دارد. هر کدام از این روش‌ها ویژگی‌هایی دارند.

- اهداف اصلی یک روش جستجوی آدرس IP چیست؟

پاسخ

اهداف اصلی یک روش جستجوی آدرس IP شامل موارد زیر است:

- **تطابق سریع و دقیق**: یافتن سریع و دقیق‌ترین تطابق (Longest Prefix Match) برای آدرس IP، به طوری که بسته به درستی و بدون تأخیر به مقصد هدایت شود.
- **بهره‌وری بالا**: استفاده بهینه از منابع سخت‌افزاری مانند حافظه و پردازشگر، تا روش جستجو بتواند با کمترین هزینه پردازشی و مصرف حافظه کار کند.
- **قابلیت مقیاس‌پذیری**: روش باید بتواند با افزایش اندازه جدول جلورانی و تعداد آدرس‌ها، همچنان به‌خوبی کار کند و عملکرد مطلوبی داشته باشد.
- **پایداری و مقاومت در برابر تغییرات**: امکان تطبیق و بروزرسانی آسان جدول جلورانی با ورود و خروج آدرس‌ها و یا تغییر مسیرها.

- معیارهای ارزیابی یک روش جستجوی آدرس IP کدام‌اند و چگونه می‌توان کارایی یک روش جست‌وجوی آدرس IP را در شبکه‌های بزرگ ارزیابی کرد؟ (پاسخ خود را با توجه به مواردی مانند کارایی حافظه، زمان جستجو، مقیاس‌پذیری و موارد مشابه دیگر توضیح دهید.)

پاسخ

- **کارایی حافظه**:
  - میزان استفاده از حافظه یکی از معیارهای مهم برای ارزیابی روش‌های جستجو است. روش‌های کارا از حافظه بهینه استفاده می‌کنند، به‌ویژه در جدول‌های جلورانی بزرگ.
  - **روش ارزیابی**: اندازه جدول و ساختار داده‌های ذخیره شده برای هر روش را مقایسه می‌کنیم. روش‌هایی که از ساختارهای داده فشرده‌تری مانند درخت‌ها و یا درخت‌های باینری استفاده می‌کنند، معمولاً کارایی حافظه بالاتری دارند.
- **زمان جستجو**:
  - زمان لازم برای یافتن تطابق در جدول، معیار مهم دیگری است. در مسیریاب‌ها، این معیار باید تا حد ممکن کمینه باشد.
  - **روش ارزیابی**: می‌توان میانگین زمان جستجو یا پیچیدگی زمانی روش‌ها (مثلاً $\mathcal{O}(\log n)$ یا $\mathcal{O}(1)$ برای جستجوهای خاص) را بررسی کرد. روش‌های سریع‌تر زمان جستجو را کاهش می‌دهند و عملکرد بهتری دارند.

ادامه پاسخ

- **مقیاس‌پذیری:**
  - روش باید بتواند با افزایش تعداد و طول آدرس‌ها (به خصوص در شبکه‌های بزرگ و پیچیده)، همچنان با سرعت و دقت بالا کار کند.
  - **روش ارزیابی:** عملکرد روش‌ها در حجم داده‌های بزرگ و همچنین میزان تأثیر افزایش تعداد آدرس‌ها بر زمان جستجو و حافظه مصرفی بررسی می‌شود.
- **انعطاف‌پذیری و بروزرسانی:**
  - روش باید در برابر تغییرات آدرس‌ها، افزودن یا حذف آن‌ها انعطاف‌پذیر باشد و امکان بروزرسانی سریع و آسان را فراهم کند.
  - **روش ارزیابی:** سرعت و سهولت عملیات درج، حذف و به‌روزرسانی آدرس‌ها در روش بررسی می‌شود. روش‌های با ساختار پویا (مانند ساختارهای درختی) معمولاً در این زمینه عملکرد بهتری دارند.

- دسته‌بندی روش‌های جستجوی آدرس IP در جدول جلورانی را با ذکر ویژگی‌های هر روش بیان کنید.

پاسخ

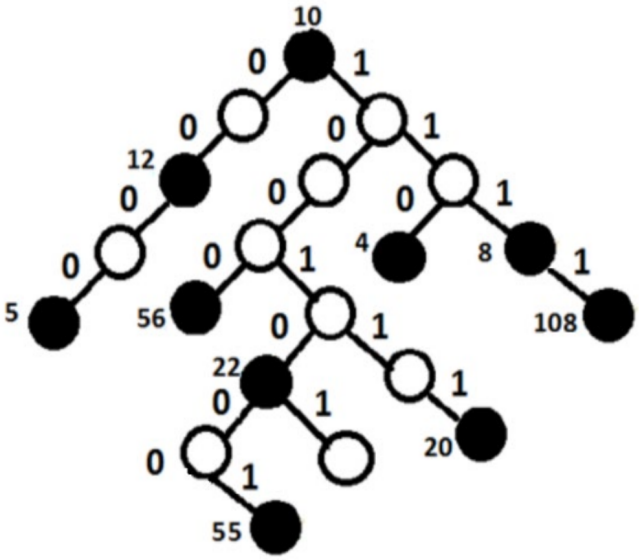
- **روش‌های مبتنی بر درخت (Trie-Based Methods):**
  - **ویژگی‌ها:** این روش‌ها از ساختار درختی مانند Binary Trie استفاده می‌کنند و آدرس‌های IP را به‌صورت بیتی ذخیره می‌کنند. به این صورت که هر سطح درخت نشان‌دهنده یک بیت از آدرس است و مسیر به پایین‌ترین سطح نشان‌دهنده طولانی‌ترین تطابق است.
  - **مزایا:** زمان جستجوی نسبتاً سریع و مقیاس‌پذیری خوب.
  - **معایب:** در شرایط خاص، ممکن است نیاز به حافظه زیادی داشته باشند و نگهداری و به‌روزرسانی آنها پیچیده باشد.
- **روش‌های مبتنی بر هشینگ (Hashing-Based Methods):**
  - **ویژگی‌ها:** از جدول‌های هش برای ذخیره پیشوندهای IP استفاده می‌کنند. در این روش، جستجو از طریق هش کردن انجام می‌شود.
  - **مزایا:** سرعت جستجوی بسیار سریع، به‌ویژه در شرایطی که جستجو با پیچیدگی $\mathcal{O}(1)$ انجام شود.
  - **معایب:** مقیاس‌پذیری کمتر نسبت به روش‌های دیگر و محدودیت در پردازش طولانی‌ترین پیشوندها به دلیل نبود ساختار سلسله‌مراتبی.
- **روش‌های مبتنی بر جستجوی باینری (Binary Search-Based Methods):**
  - **ویژگی‌ها:** از جستجوی باینری برای یافتن طولانی‌ترین پیشوند استفاده می‌کنند. این روش به‌خصوص برای شبکه‌هایی با پیشوندهای کوتاه‌تر مناسب است.
  - **مزایا:** زمان جستجوی ثابت و مناسب در اندازه‌های کوچک و متوسط.
  - **معایب:** کارایی حافظه کمتر نسبت به سایر روش‌ها در شبکه‌های بزرگ.

ادامه پاسخ

- **روش‌های مبتنی بر جداول TCAM**:
  - **ویژگی‌ها**: در این روش‌ها از حافظه TCAM استفاده می‌شود که امکان جستجوی همزمان چندین ورودی را فراهم می‌آورد.
  - **مزایا**: سرعت بسیار بالای جستجو و پشتیبانی از طولانی‌ترین پیشوند.
  - **معایب**: قیمت بالا و مصرف انرژی زیاد، همچنین نیاز به تنظیم دقیق برای بهینه‌سازی عملکرد.

سوال سوم

درخت باینری نشان‌داده شده در زیر، از روی جدول جلورانی در یک مسیریاب ایجاد شده است و جست و جو در جدول جلورانی برای پیدا کردن شماره پورت خروجی با پیمایش این درخت انجام می‌گردد. (در این درخت باینری، گره‌های مشکی، پیشوندهای آدرس جدول جلورانی هستند و مقادیر آن‌ها بیانگر شماره پورت خروجی است.)



شکل ۱: درخت باینری

۱. بسته‌ای با آدرس مقصد 90B28FF1 از کدام پورت خارج می‌گردد؟ (آدرس‌های مقصد در مبنای ۱۶ نمایش داده شده‌اند.)

پاسخ

بر اساس درخت داده شده، جدول مسیریابی به‌صورت زیر به‌دست می‌آید:

Prefix	Output Port
*	10
111*	8
110*	4
1000*	56
10010*	22
100100*	60
100111*	20
00*	12
0000*	5

اگر معادل باینری آدرس داده شده را بنویسیم:
(0X90B28FF1) = 1001 0000 1011 0010 1000 1111 1111 0001
می‌بینیم که براساس الگوریتم Longest Prefix Matching بیشترین Matching را با سطر ۶ جدول دارد.
پس درنتیجه، این بسته از پورت شماره ۶۰ خارج می‌شود.

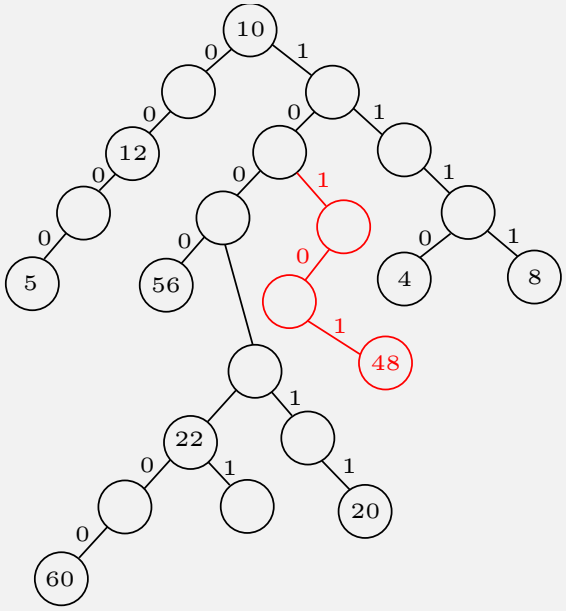
۲. بسته‌ای با آدرس مقصد A2AB11C3 از کدام پورت خارج می‌گردد؟

پاسخ

معادل باینری آدرس داده شده را می‌نویسیم:
(0XA2AB11C3) = 1010 0010 1010 1011 0001 0001 1100 0011
بیشترین Matching را با سطر ۱ جدول است. پس، این بسته از پورت شماره ۱۰ خارج می‌شود.

۳. آدرس پیشوندی *10101 با پورت خروجی ۵۰ را به درخت اضافه نمایید.

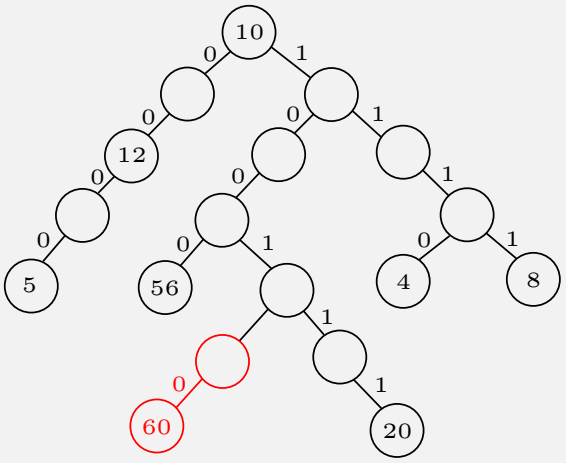
پاسخ



شکل ۲: درخت جدید

۴. آدرس پیشوندی *10010 با پورت خروجی ۲۲ را از درخت حذف نمایید.

پاسخ



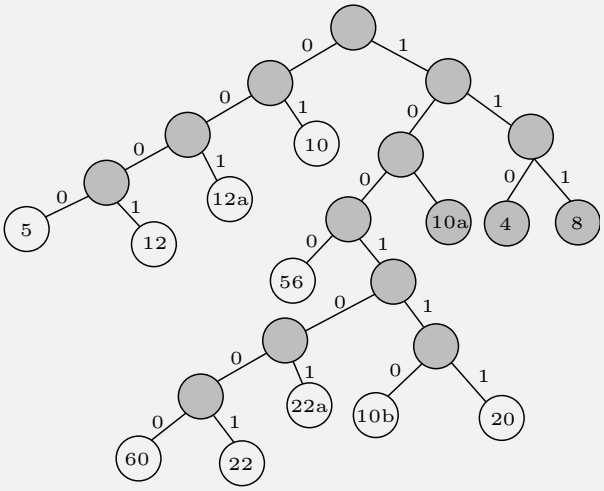
شکل ۳: درخت جدید

سوال چهارم

درخت سوال ۳ را به فرم disjoint-prefix binary trie تبدیل نمایید. حافظه مورد نیاز جهت نگهداری درخت سوال ۳ و سوال ۴ را مقایسه کنید.

پاسخ

درخت Disjoint حاصل به‌صورت زیر است:



شکل ۴: درخت Disjoint

به این درخت باید ۴ گره جدید دیگر اضاف شود تا درخت کامل شود. در اینجا مجموعا ۱۳ برگ داریم و در این روش فقط برگ‌ها را در حافظه ذخیره می‌کنیم و درنتیجه کلا اطلاعات ۱۳ برگ را درحافظه داریم. اما در سوال ۴ کل درخت که شامل ۱۹ گره است را باید در حافظه نگهداری کنیم. **درنتیجه، در این روش، حافظه مصرفی کاهش می‌یابد.**

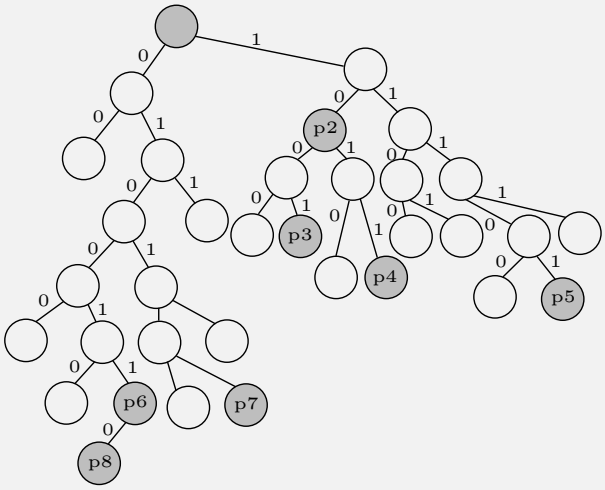
سوال پنجم

Path-compressed trie جدول جلورانی زیر را رسم کنید.

P1	*
P2	10*
P3	1001*
P4	1011*
P5	11101*
P6	010011*
P7	010101*
P8	0100110*

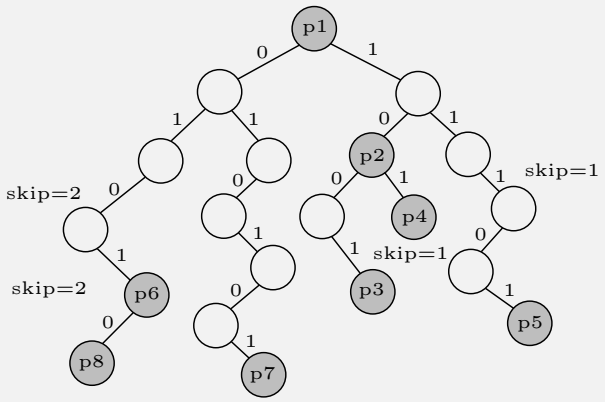
پاسخ

ابتدا Binary Trie جدول را به‌صورت زیر رسم می‌کنیم:



شکل ۵: درخت Binary

سپس درخت Path Compress را به‌صورت زیر رسم می‌کنیم:



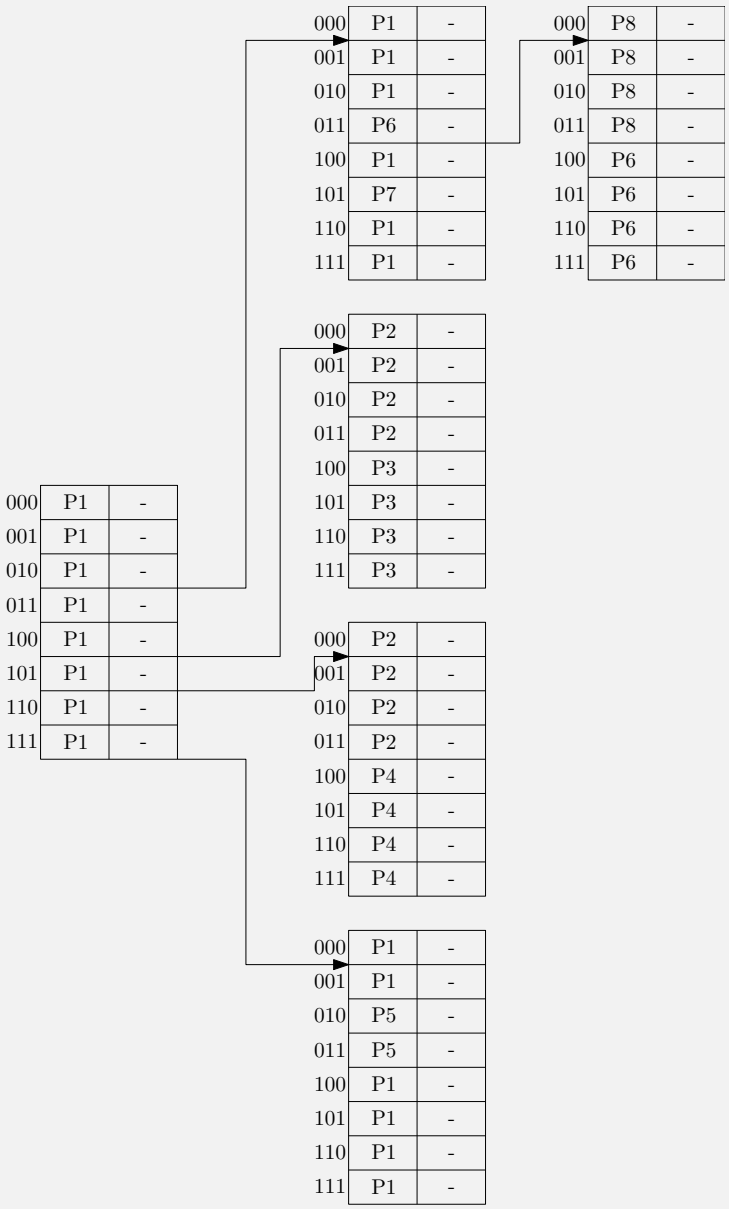
شکل ۶: درخت Path Compress

سوال ششم

Multi Bit Trie سه بیتی جدول مسیریابی سوال ۵ را رسم نموده و سپس درخت به‌دست آمده را به disjoint-prefix تبدیل نمایید.

پاسخ

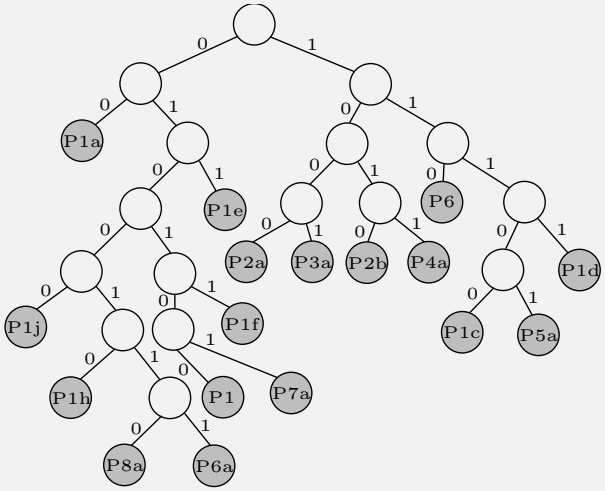
درخت Multi Bit Trie به‌صورت زیر رسم می‌شود:



شکل ۷: درخت Multi Bit Trie

و درخت Disjoint-Prefix را به‌صورت زیر رسم می‌کنیم:

پاسخ



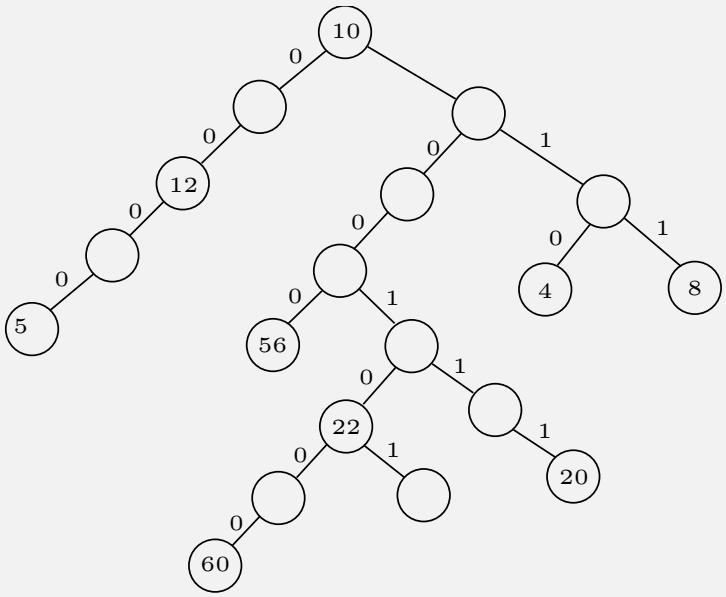
شکل ۸: درخت Disjoint-Prefix

سوال هفتم

LC-trie نظیر درخت سوال ۳ را رسم نمایید. رابطه تعداد branchها و strideها چیست؟

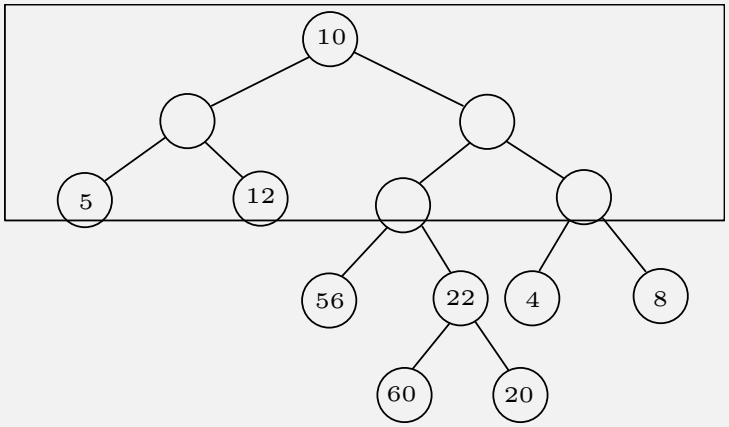
پاسخ

ابتدا درخت One Bit Trie را به‌صورت زیر رسم می‌کنیم:



شکل ۹: درخت One Bit Trie

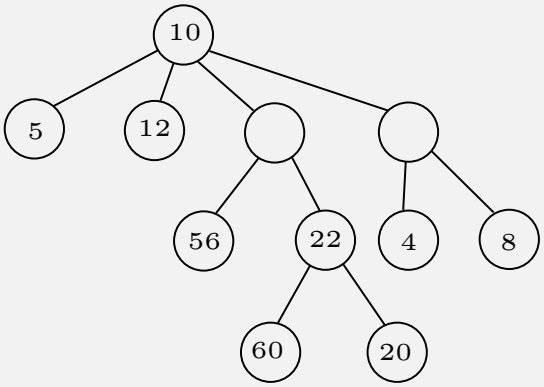
درخت Path Compressed Trie به‌صورت زیر می‌شود:



شکل ۱۰: درخت Path Compressed Trie

پاسخ

و درنهایت LC-Trie به‌صورت زیر رسم می‌شود:



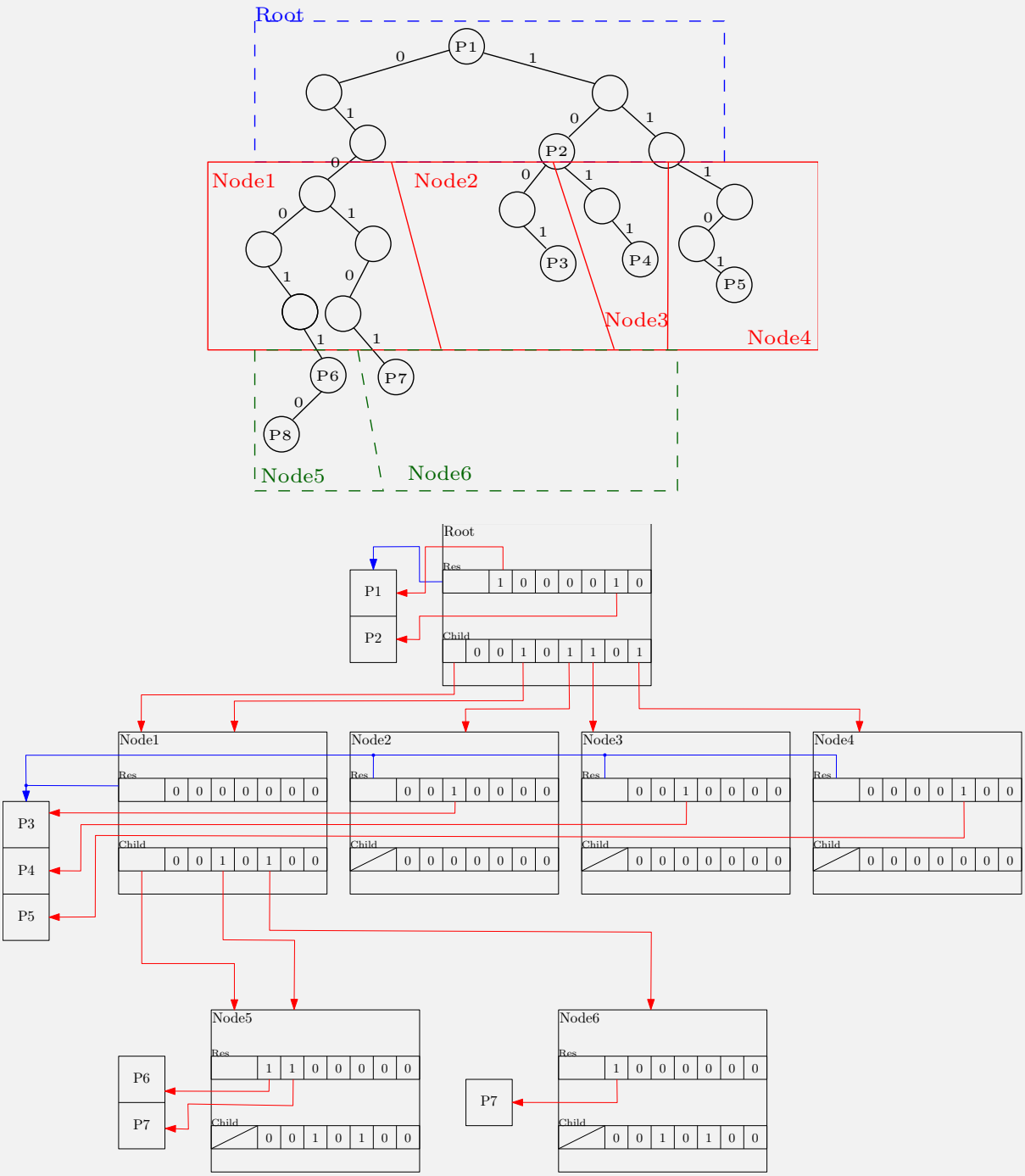
شکل ۱۱: درخت LC-Trie

مقدار Stride با تعداد Branch رابطه مستقیم دارد.
هرچقدر که مقدار Stride بیشتر باشد، تعداد Branch ها بیشتر می‌شود.

سوال هشتم

برای جدول مسیریابی سوال ۵، بر اساس روش Tree Bitmap ساختار داده را به طور کامل رسم نمایید. (Stride=3)

پاسخ



سوال نهم

در الگوریتم Binary search on prefix range در صورتی که n داده، m بیتی داشته باشیم و از k-way search استفاده کنیم.

- در این الگوریتم چه ارتباطی بین n، m و k وجود دارد؟

پاسخ

در هر جست‌و جو، $\frac{1}{k}$ از جدول مرحله بعدی جست‌وجو باقی می‌ماند زیرا باید، $log(k)$ بیت را در جدول جست‌وجو کنیم. جدولی $2n$ ردیفی داریم که دارای m بیت در هر ردیف است.

- در بدترین حالت مرتبه زمانی جستجو و حافظه مورد نیاز را محاسبه نمایید.

پاسخ

در بدترین حالت، پیچیدگی حافظه و زمان به‌ترتیب برابر هستند با $O(N)$ و $O(log(n))$

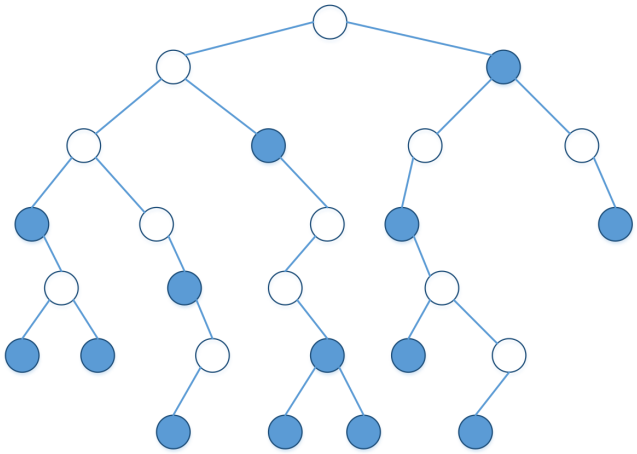
- در صورتی که عمل جستجو توسط پردازنده‌ای با فرکانس ساعت 3.2 MHz و بر روی یک جدول جلورانی با 90000 سطر (entries) انجام شود، بیشترین زمان مورد نیاز و متوسط زمان مورد نیاز و میزان حافظه مورد نیاز را محاسبه نمایید.

پاسخ

با فرض اینکه هربار جدول به k قسمت تقسیم شود و سپس مقایسه انجام شود، هر دفعه k دسترسی به حافظه خواهیم داشت.

سوال دهم

یکی از روش‌های کاهش هزینه توان مصرفی در TCAM روش Trie-Based Table Partitioning که خود شامل Subtree Splitting و Post-Order Splitting است. با اعمال این دو روش بر روی درخت زیر، جدولی مشابه جدول صفحه ۶۴ کتاب بدست آورید. گره‌های آبی رنگ شامل آدرس‌های Prefix هستند. (مقدار اندازه بلوک‌های حافظه را ۴ در نظر بگیرید؛ $b = 4$)

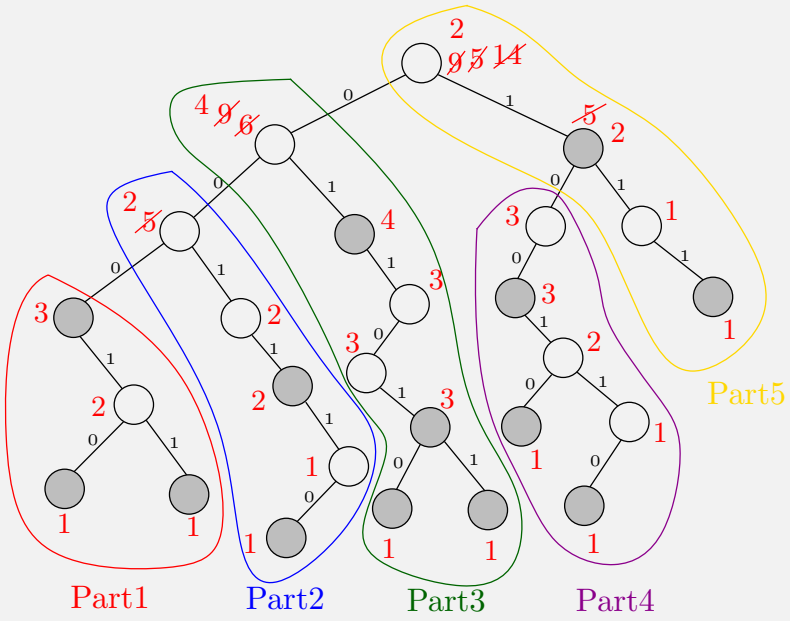


شکل ۱۲: درخت باینری

پاسخ

مقدار Value را برای هر گره می‌نویسیم و در هر مرحله که تکه‌ای از درخت را جدا می‌کنیم، این مقدار را آپدیت می‌کنیم و مقدار و مقدار آن را با رنگ قرمز درکنار هر Node مشخص می‌کنیم.
باید $2 \leq Value \leq 4$ را جدا کنیم.

۱. **Preorder:**

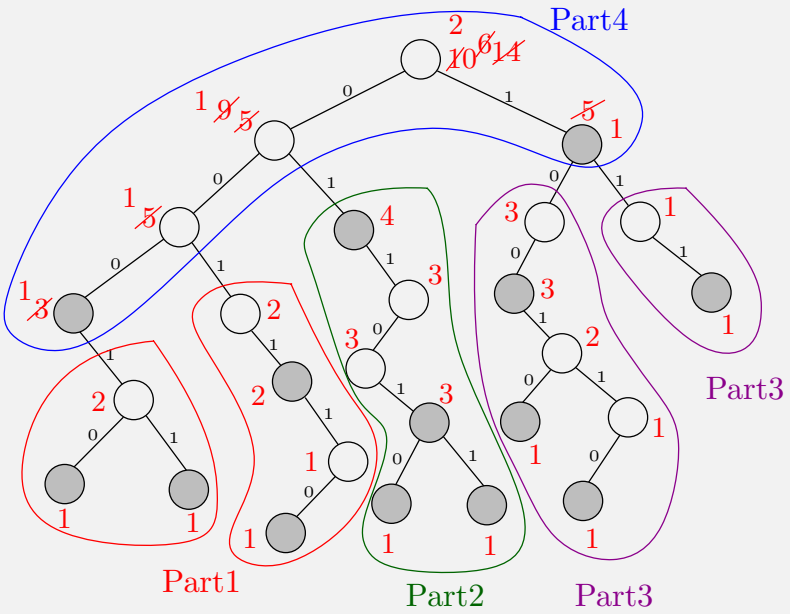


شکل ۱۳: Preorder

پاسخ

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

۲. **Postorder:**



شکل ۱۴: Postorder

Prefix	Bucket Prefixes	Bucket Size	Covering Prefix
0001*,001*	00010*,00011*,0011*,001110*	4	0001*,0011*
01*	01*,01101*,011010*,011011*	4	01*,01101*
10*,11*	100*,10010*,100110*,111*	4	100*,111*
*	000*,1*	2	1*,000*