دانشکده علوم ریاضی و آمار

مدرس: دکتر مجتبی رفیعی

نیمسال اول ۱۴۰۰-۱۴۰۱

ساختمان داده‌ها و الگوریتم‌ها

جلسه ۳۰

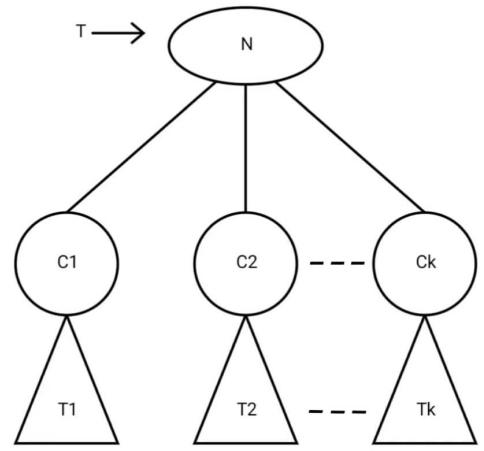
نگارنده: ریحانه دهقانی

۱۲ آذر ۱۴۰۰

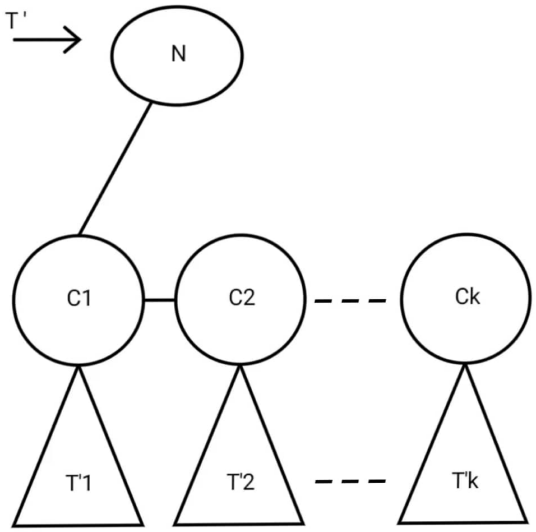
فهرست مطالب

۱ بررسی رابطه بین پیمایش‌های درختk-تایی ودرخت دودویی‌معادل

۱.۱ شهود اثبات-پیش‌ترتیب



$\mathrm{Preorder(T)}=\mathrm{N\ C_1\ T_1\ C_2\ T_2 ... C_k\ T_k}$

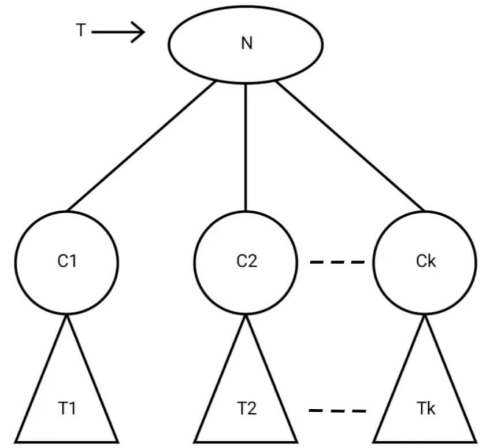


$\mathrm{Preorder}(T')= \mathrm{N\ C_1\ T'_1\ C_2\ T'_2 ...\ C_k\ T'_k}$

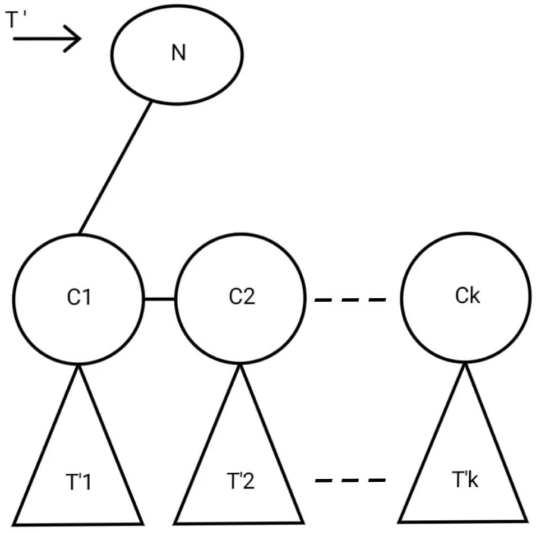
واضح است که جایگاه نودهای ملاقات‌شده در پیمایش پیش‌ترتیب T و ’T یکسان است ($\mathrm{Preorder(T)}= \mathrm{Preorder}(T')$).

می‌توان از ایده‌ی بالا برای رد یکسان بودن پیمایش میان‌ترتیب و پس‌ترتیب درخت T و T' استفاده کرد. در ادامه شهود کافی برای رابطه‌های مذکور آورده شده است.

۲.۱ شهود اثبات-پس‌ترتیب



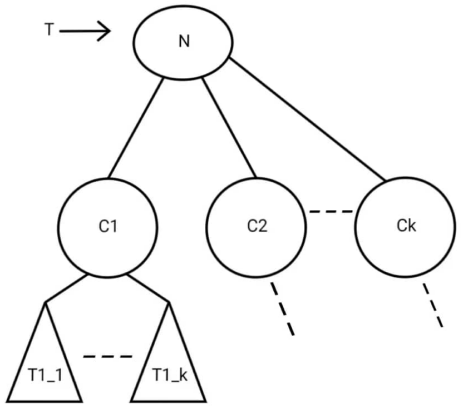
$\text{Postorder(T)}= \text{T}_1\ \text{C}_1\ \text{T}_2\ \text{C}_2 \ldots\ \text{T}_\text{k}\ \text{C}_\text{k}\ \text{N}$



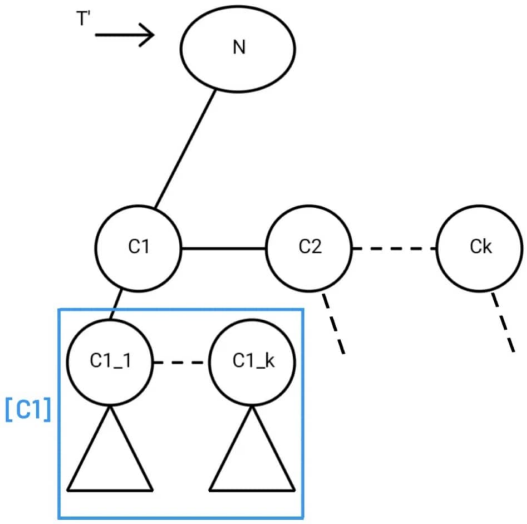
$\text{Postorder}(T')= T'_1\ T'_2 \ldots\ T'_\text{k}\ \text{C}_\text{k} \ldots \text{C}_2\ \text{C}_1\ \text{N}$

نتیجه: مشاهده پیمایش‌های بالا به وضوح بیانگر آن است که $(Postorder(T) \neq Postorder(T'))$.

۳.۱ شهود اثبات-میان‌ترتیب



$$\text{Inorder(T)}= \text{T}_{1\text{-}1}\ \text{C}_1\ \text{T}_{1\text{-}2}...\text{T}_{1\text{-}k}\ \text{N}\ \text{T}_{2\text{-}1}\ \text{C}_2\ \text{T}_{2\text{-}2}...\ \text{T}_{2\text{-}k}...\ \text{T}_{k\text{-}1}\ \text{C}_k\ \text{T}_{k\text{-}2}...\ \text{T}_{k\text{-}k}$$



$$\text{Inorder}(T')= [\text{C}_1]\ \text{C}_1\ [\text{C}_2]\ \text{C}_2...[\text{C}_k]\ \text{C}_k\ \text{N}$$

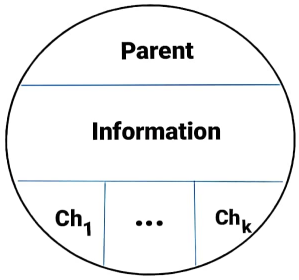
نتیجه: مشاهده پیمایش‌های بالا به وضوح بیانگر آن است که $(Inorder(T) \neq Inorder(T'))$.

۲ درخت ترای (Trie)

همانطور که در بخش‌های قبلی گفتیم درختk-تایی به طور بالقوه‌ای پتانسیل هدررفت حافظه را دارا می‌باشد. با این‌حال این داده‌ساختار برای برخی کاربردها مناسب بوده و طور گسترده ای از آنها استفاده می‌شود.در ادامه سعی داریم در این راستا درخت ترای را به عنوان یک درختk-تایی معرفی کنیم.

"درخت ترای" که با نامهای دیگری نظیر "درخت دیجیتال" و "درخت پیشوندی" نیز شناخته می‌شود نوعی درختk-تایی جست وجو است که برای پیدا کردن کلیدهای مشخص در داخل یک مجموعه مورد استفاده قرار می‌گیرد.

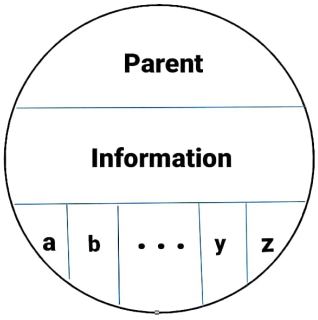
کلیدها در این ساختار بیشتر از نوع رشته ای هستند و به‌طور کامل در یک نود قرار نگرفته‌اند، بلکه با پیمایش درخت و جابجایی بین نودها از طریق یال بین آنها، کاراکترهای تشکیل‌دهنده کلید به‌مرور مشخص می‌شوند.
نمایش گرافیکی نودهای درخت ترای در حالت کلی به‌صورت زیر است:



اشاره‌گرهای Parent و ch۱ تا Chk شبیه به نودهای درخت در جلسات قبلی می‌باشد. با این‌حال بخش Information می‌تواند شامل هرگونه اطلاعات مفیدی متناسب با کاربرد داده‌ساختار باشد، مثل: تعداد تکرار زیررشته تا نود جاری در کل رشته‌های موجود در مجموعه، معنای کلید یا توضیحات در رابطه با آن و... .

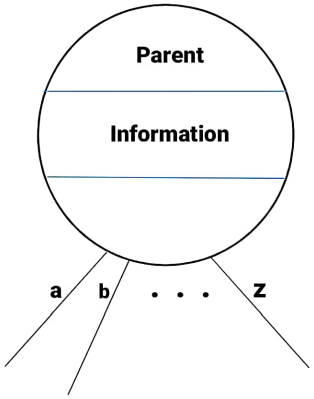
مثال۱:

درخت ترای برای کلیدهای book و ball و thin و the و tester و good را ترسیم کنید.
برای این ترسیم نودهای درخت را به‌صورت زیر درنظر می‌گیریم:

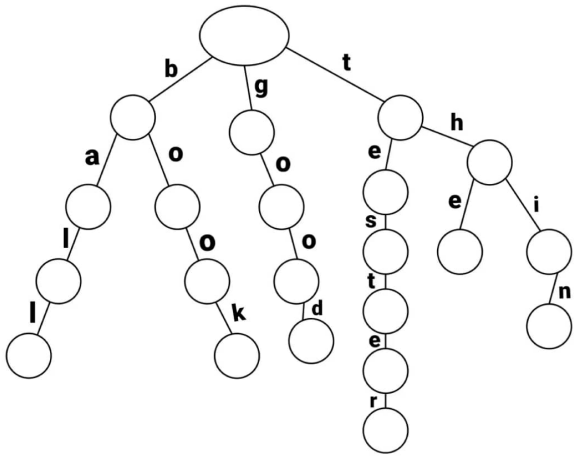


به‌ترتیب بچه ها از چپ‌به‌راست با حروف لاتین برچسب می‌خورند.درواقع اولین اشاره‌گر قراردادمی‌شود که برای حرف a است و همین‌طور الی آخر.

برای راحتی کار و سادگی در ترسیم روی هر یال درخت، حرف مربوطه نوشته می‌شود، بنابراین هر نود درخت به‌صورت زیر در می‌آید:



بنابراین درخت ترای به‌صورت زیر ترسیم می‌شود:



با توجه به مثال بالا، واضح است که داده‌ساختار درختی ترای، گزینه مناسبی برای ساخت یک "دیکشنری" است. همچنین از آنجایی‌که در پیمایش درخت در هر نود، پیشوند کلیدها را داریم، می‌تواند گزینه مناسبی برای "تکمیل خودکار" در زبان‌های برنامه‌نویسی یا "جست‌وجو در گوگل" (موتورهای جست‌وجو) باشد.

در ادامه، مثال قبل را برای پشتیبانی از دیکشنری با استفاده از درخت ترای کامل می‌کنیم. برای قسمت اطلاعات (Information) هر نود، می‌توان داده‌های زیر را درنظر گرفت:

۱): متغیر flag : نشان می‌دهد که مسیر پیموده‌شده تا نود کنونی یک واژه معتبر است یا خیر،

۲): معنی کلمه لاتین،

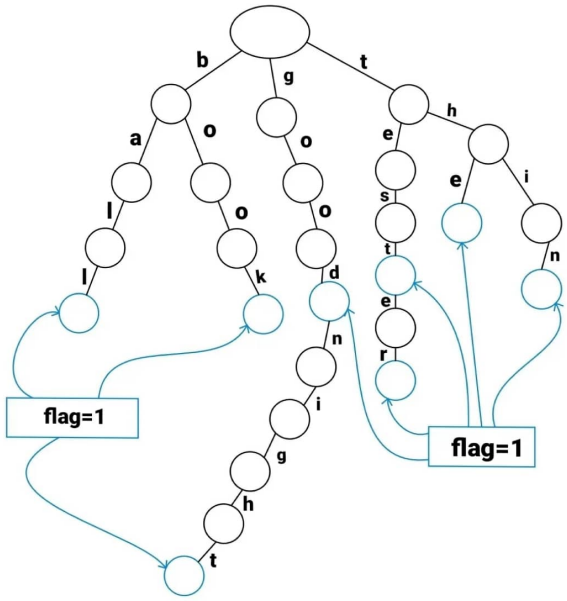
۳): تعداد کلماتی که حاوی پیشوند پیمایش‌شده است،

۴): کلمات مترادف،

۵): پویا بودن محتوای نودها-تعداد جست‌وجوهای کلید موردنظر.

۶): و... .

مثال۲: برای سادگی تنها از اطلاع flag استفاده کرده و با اضافه کردن کلیدهای test و goodnight به مثال قبل، درخت ترای آن را بازترسیم می‌کنیم.



توجه: flag مابقی نودها صفر است (flag=۰)

۱.۲ عملیات روی درخت ترای

عملیات روی درخت ترای شامل موارد زیر است:

۱): **عمل جستجو:** برای جستجوی یک کلید در درخت ترای نیاز است که حداکثر به تعداد کاراکتر های آن رشته، درخت را پیمایش کنیم بنابراین پیچیدگی جست‌وجو از مرتبه O(m) است.(جاییکه m تعداد کاراکترهای رشته موردنظر است).

۲): **عمل درج:** برای عمل درج یک کلید می‌بایست گام‌های زیر را دنبال کنیم:

-از ریشه شروع به پیمایش کرده و اگر کلید موردنظر در پیمایش دیده‌شد، دو حالت پیش می‌آید: