بسم الله الرحمن الرحيم

دانشگاه تهران

دانشکده علوم

گروه ریاضی وعلوم کامپیوتر

تولید کد های متناظر با درختان K تایی

نگارش :

محمد فرجی مهماندار

استاد راهنما :

دکتر هایده اهرابیان

پایان نامه برای دریافت درجه کارشناسی ارشد در علوم کامپیوتر

بهمن ۱۳۸۱

جمهوری اسلامی ایران

دانشگاه تهران

دانشکده علوم

بسمه تعالی

اداره کل تحصیلات تکمیلی دانشگاه

احتراماً به اطلاع میرساند که جلسه دفاع از پایان نامه دوره **کارشناسی ارشد علوم کامپیوتر آقای محمد فرجی مهماندار** تحت عنوان:

تولید کدهای متناظر با درختان k- تایی

در تاریخ ۸۱/۱۲/۱۱ در **گروه ریاضی و علوم کامپیوتر دانشکده علوم دانشگاه تهران** برگزار گردید.

هیأت داوران بر اساس کیفیت پایان نامه، مقالات انتشار یافته، استماع دفاعیه و نحوه پاسخ به سؤالات، **پایاننامه** ایشان را برای دریافت **درجه کارشناسی ارشد** در **رشته علوم کامپیوتر** معادل با ٦ واحد با نمره **۱۹/- تمام** با درجه **عالی** مورد ارزشیابی قرار داد.

هیأت داوران

سمت	نام و نام خانوادگی	مرتبه دانشگاهی	دانشگاه	امضاء
۱. استاد راهنما	**دکتر هایده اهرابیان**	استادیار	تهران	
۲. استاد داور	**دکتر حسن صالحی فتح آبادی**	دانشیار	تهران	
۳. استاد داور	**دکتر سیدمهدی تشکری**	استادیار	امیرکبیر	

معاون تحصیلات تکمیلی گروه	مدیر گروه	معاون تحصیلات تکمیلی دانشکده
دکتر سیامک یاسمی	دکترعمید رسولیان	دکتر مجید مدرس

چکیده :

دراین پایان نامه برخی از الگوریتم های ترتیبی وموازی ارائه شـده بـرای تولیـد دنبالـه هـای متنـاظر بادرختان k تایی بررسی می گردد.الگوریتم های ارائـه شـده بـه واسـطه ترتیـب ونـوع تولیـد متفـاوت می باشند. دسته اول الگوریتم های ترتیبی می باشند کـه درختـان k تـایی را بـاترتیب $B-Order$ تولید می کنند ودرآن، الگوریتم های زکس [14] و کـورش [6] و راسـکی [11] مـورد بررسـی قـرار می گیرد و دسته دوم الگوریتم هایی هستند که درختان را با ترتیب های متفـاوتی تولیـد مـی کننـد وباجزئیات بیشـتری مـورد بررسـی قـرارگرفتـه انـد کـه درآن الگوریتـم هـای اهرابیـان ونـوذری [2] درترتیب $B-Order$ و باروناجیان و راسکی [4] درترتیب $A-Order$ *می باشد.*

گروه سوم شامل، الگوریتم های موازی تولید دنباله های متناظربادرختـان k تـایی مـی باشـد ودر آن الگوریتم هایی پائولو وواجنوسکی [12] و واجنوسکی وفیلیپس [13] مورد بحث قرارگرفته است .

تمام الگوریتم های بحث شده در این پایـان نامـه ، پیـاده سـازی شـده و یـک جـدول مقایسـه ای از پیچیدگی زمانی الگوریتم های تولید نسبت به یکدیگر نشان داده شده است.

تشکر وقدر دانی

مراتب سپاس گزاری عمیق خودم را از استاد ارجمند، سر کار خانم دکتر اهرابیان، به خاطر راهنمایی‌های مفید و ارزنده‌شان در طول دوران تحصیل و نگـــارش ایــن پایان نامه ابراز می‌دارم. امیدوارم در آینده نیز از راهنمایی‌های ارزنده‌شـــان بــهره بیشتر ببرم .

هم چنین از آقای دکتر صالحی و دکتر تشکری که داوری ایــن پایــان نامــه را بــر عهده داشتند و کلیه دوستان و همکاران که در نگارش این پایان نامه مـــرا یــاری نموده اند، تقدیر وتشکر می کنم .

فهرست

عناوین	صفحه

فهرست

مقدمه :

تاکنون الگوریتم های زیادی برای تولید درختان k تایی باتعداد گره های ثابت ارائه شده است. هرکدام از این الگوریتم ها ، درختان k تایی را باروش خاصی بصورت یک دنباله کدگذاری کرده وسپس این دنباله ها را تولید می کنند . الگوریتم های تولید، درخت متناظر با این دنباله ها را با ترتیب های متفاوتی تولید می کنند .دراین خصوص، زکس وپائولو[7,14] دوترتیب برای درختان k تایی به نامهای $A - Order$, $B - Order$ تعریف نموده اند . کنوت [5] نیز مبحث الگوریتم های رتبه گذاری و بازیابی دنباله ازرتبه درخت رابرای درختان مطرح نمود .ازجمله الگوریتم هایی که دراین مبحث ارائه شده ،الگوریتم تولید درختان k تایی ارائه شده توسط زکس [14] می باشدکه دنباله ای بنام $BitString$ را تولید می کند. الگوریتم تولید pew-دنباله ارائه شده توسط باروناجیان – راسکی [4] و $Ballot$-دنباله ارائه شده توسط اهرابیان ونوذری [2] نیز الگوریتم های دیگری هستند که دراین راستا ارائه شده است. دراین پایان نامه بشرح تعدادی ازاین الگوریتم ها می پردازیم .

این پایان نامه شامل چهارفصل می باشد . درفصل اول تعاریف اولیه گراف، درخت والگوریتم [5] به همراه جزئیات بیان می شوند .از این تعاریف درفصول بعدی استفاده خواهد شد. درفصل دوم الگوریتم های ترتیبی تولید درختان k تایی درترتیب $B - Order$ با استفاده از دنباله های ارائه شده توسط زکس [14] و کورش [6]و راسکی [11] بررسی می شود .درفصل سوم الگوریتم های ارائه شده توسط اهرابیان ونوذری باترتیب $B - Order$ و باروناجیان وراسکی باترتیب $A - Order$ باجزئیات بیشترمورد بررسی قرارگرفته است .

فصل چهارم شامل الگوریتم های موازی تولید دنباله های مربوط به درختان k تایی است .دراین فصل الگوریتم موازی تولید $BitString$ که توسط پائولو وواجنوسکی [12] ارائه شده ، والگوریتم موازی تولید p-دنباله که توسط واجنوسکی وفیلیپس [13] ارائه شده، بررسی می گردد .

تمام الگوریتم های بحث شده در این پایان نامه ، پیاده سازی شده و یک جدول مقایسه ای از پیچیدگی زمانی الگوریتم های تولید نسبت به یکدیگر نشان داده شده است.

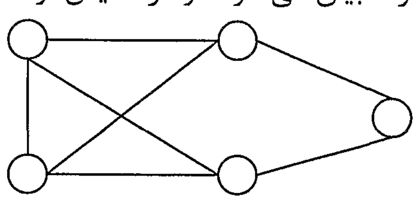
فصل اول

تعاریف ا ولیه

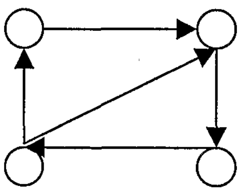
در این فصل، مفاهیم اولیه گراف، درخت و پردازش موازی بیان می‌شود و در ادامـه معیارهـایی بـرای مقایسه الگوریتم های ترتیبی وموازی مطرح می شوند.

۱-۱- گراف[1]

بنابر تعریف یک گراف ساده زوج مرتب (V,E) است که در آن V یک مجموعهٔ حتماً ناتهی از عناصری به نام رئوس و E یـک مجموعـهٔ متشـکل از زوج هـایی بـه نـام یـال هاسـت. V را مجموعهٔ رئوس[2] و Eرا مجموعهٔ یالها[3] می گوییم. یک گراف با مجموعه رئـوس V و مجموعـهٔ یال‌های E را معمولاً با $G(V,E)$نشان می دهیم. اگر e یک عضو مجموعه E باشد که دو رأس u,v را بهم متصل می کنـد، مـی گوئیـم رئـوس u,v بـر یـال e واقـع شـده اسـت. در ایـن صورت u,v را مجاور گویند. اگر دو رأس واقع بر یک یال بر هم منطبق باشـند، آنگـاه یـال را حلقه[4] می گویند. یال ها در یک گراف می توانند یک طرفه و یا دو طرفه باشند. اگر یـال هـا یک طرفه باشند، گراف، جهت دار نامیده می شود. یک گراف فاقد جهت بصورت یال هـای دو طرفه بیان می گردد و در نمایش نوک پیکان ها نشان داده نمی شود.(شکل ۱-۱)



(الف) -گراف بدون جهت



(ب)- گراف جهت دار

شکل۱-۱-گراف های جهتدار وبدون جهت

1. Graph
2. Vertices
3. Edges
4. Loop

مسیر[1] و دور[2] یک گراف:

یک مسیرساده از رأس u به رأس v در گراف $G(V,E)$ یک دنباله غیرتهی و متناهی از رئــوس و یــال های متمایز مانند $u=u_0,e_1,u_1,e_2,...,u_{n-1},e_n,u_n=v$ می باشد که در آن e_i یال $u_{i-1}u_i$ است. اگر در یک مسیر رأس ابتدا و رأس انتها، یکسان باشند این مسیر، دور نامیده می‌شود.

درخت[3]: گراف ساده و همبند و بدون دور، درخت نامیده می شود. اگر u,v دو رأس غیرمجـاور در درخت T باشند آنگاه $T+uv$ دقیقاً شامل یک دور می شود.

اگر T یک گراف با nرأس و mیال باشد آنگاه گزاره های زیر معادلند:

T یک درخت است .

T بدون دور است و $m=n-1$.

T همبند است و $m=n-1$.

T همبند است و با حذف هر یال، ناهمبند می شود.

هر دو رأس در T فقط با یک مسیر، بهم متصل اند.

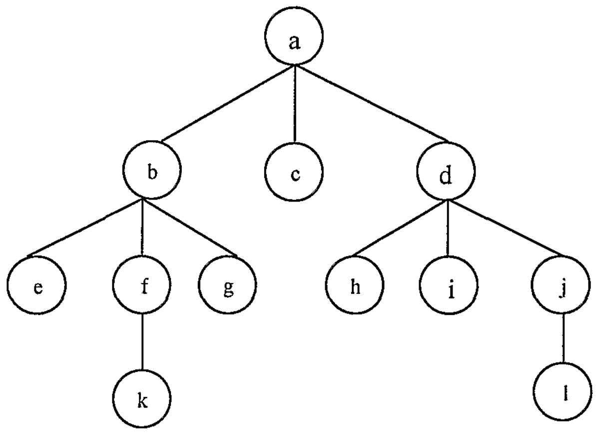
T بدون دور است ولی از بهم وصل کردن دو رأس آن، فقط یک دور بدست می آید.

در درخت رئوس، گره[4] و یال ها، شاخه[5] نامیده می‌شوند.

متداولترین نوع درخت، درختی است که یک گره به نام ریشه دارد و هیچ شاخه ای به آن وارد نمـی شود سایر گره های درخت، دارای یک شاخهٔ ورودی یکطرفه می باشند و می توان با شــروع پیمـایش درخت از ریشه به تمامی گره ها رسید. گره های دیگر در مجموعه های $T_1,T_2,...,T_n$ که هر کدام خود یـک درخــت می‌باشــند، افــراز مــی شــوند. در ایــن صــورت $T_1,T_2,...,T_n$ را زیردرخــت هــای ریشه می‌نامند.تعداد زیردرخت های یک گره درجه آن گره نامیده می شود. درجهٔ یک درخت معادل

1. Path
2. Cycle
3.Tree
4.Node
5.Branch

با بالاترین درجهٔ گره های آن درخت می باشد. گره ای که دارای درجه صفر است، برگ نامیده می شود.در شکل ۱-۲-مجموعه $\{e,k,g,h,i,l\}$ برگ های درخت T می باشند.ریشه های زیردرختهای یک گره ،فرزندان[1] آن گره نامیده می شوندونیز آن گره والد[2] فرزندان خود می باشند.



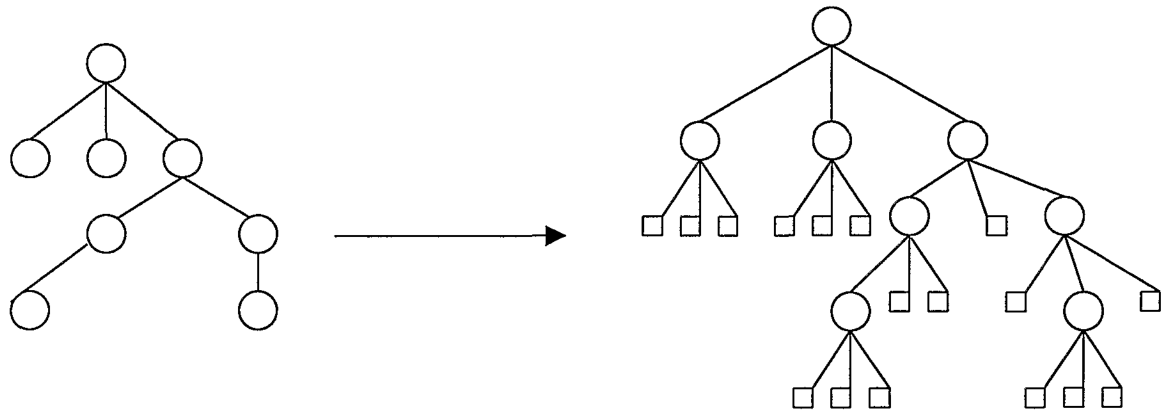
شکل ۱-۲- درخت T

درختهای k تایی:

درختهای k تایی ساختمان داده هایی می باشند که شامل یک مجموعه متناهی از n گره که یا تهی است $(n=0)$ یا شامل یک گره ریشه وحداکثر k فرزند متصل به آن می‌باشد به طوری که هرفرزند یک زیردرخت k تایی می باشد (تعریف بازگشتی). برای $k=2$ ما به حالت خاصی بنام درخت دودویی می‌رسیم که هر گره یک فرزند راست و یک فرزند چپ دارد.

یک درخت k تایی را منظم[3] گوییم اگر هر گره درخت یا صفر یا دقیقاً k فرزند داشته باشد برای این منظور کافی است که هر زیردرخت خالی در درخت را با یک گره خاصی نمایش دهیم این گره ها را گره خارجی می نامندودر این حالت گره های درخت راگره های داخلی می نامند.(شکل ۱-۳) باتوجه به این نکته می توان هر درخت k تایی با n گره را به یک درخت منظم که متناظر با آن درخت می باشد تبدیل نمود.

1.Children
2.Parent
3.Regular



شکل ۱-۳-یک درخت سه تایی وتوسعه یافته آن

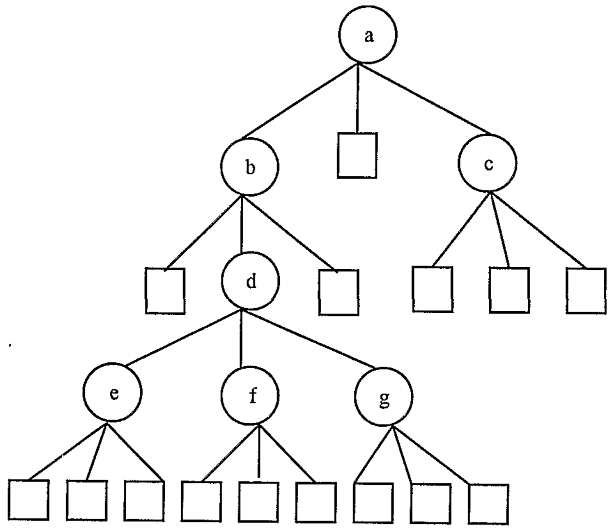
پیمایش[1] درخت ها

پیمایش یک درخت، یک ترتیب خطی از اطلاعات موجود در گـره هـای آن را ایجـاد مـی نمـاید. در هنگام پیمایش یک درخت با یک گره و زیردرختانش به طرز مشابهی رفتار می کنیـم بـرای درختـان دودویی سه پیمایش به صورت زیر تعریف می شود LDR, LRD, DLR که این سه حالت را بـه ترتیـب پیش ترتیب[2]، پس ترتیب[3] و میان ترتیب[4] می نامیم. و در آن L حرکت به چپ ، R حرکت به راسـت و D چاپ داده یک گره می باشد. در واقع هیچ روش معمولی، برای اینکه بتوان یـک پیمـایش میـان ترتیب را به درختان k تایی تعمیم داد وجود ندارد ولی پیمایش پیش ترتیب و پس ترتیب درختان k تایی کاملاً معلوم می باشد.در شکل ۱-۴ پیمایش های پس ترتیب وپیش ترتیب درخـت۳ تـایی بـا ۷ گره نشان داده شده است.

۱-۲- تعریف الگوریتم و پیچیدگی آن

الگوریتم مجموعهٔ محدودی از دستورالعمل هایی است که با دنبال کردن آنها، هـدف خـاصی بـرآورده می شود. موارد زیر در هر الگوریتم قابل بررسی است:

1.Traversal
2.Preorder
3.Postorder
4.Inorder



$abdefgc$:پیمایش پیش ترتیب $efgdbca$:پیمایش پس ترتیب

شکل ۱-۴- پیمایش های پس ترتیب وپیش ترتیب درخت kتایی

ورودی: یک الگوریتم می تواند یک یا چند ورودی داشته باشد که از محیط خارج تأمین می شوند.

خروجی: الگوریتم باید حداقل، یک کمیت به عنوان خروجی داشته باشد

قطعیت: الگوریتم باید، پس از طی مراحل محدودی خاتمه یابد.

برای مقایسهٔ الگوریتم ها با یکدیگر معمولاً از دو معیار تعداد عملیات انجام شده (سرعت) و میزان حافظه مصرفی، استفاده می شود سرعت یک الگوریتم مهمتر از میزان حافظه مصرفی بوده و رکن اصلی مقایسه می باشد.

پیچیدگی یک الگوریتم بر پایه تعداد ورودی n مشخص می گردد این کار با حذف عامل های ثابت در الگوریتم صورت می گیرد، چرا که مقادیر ثابت بدلیل تفاوت در پیاده سازی مـی باشد. جهت نشان دادن سرعت یک الگوریتم معمولاً از نشانه گذاری O ، $(Order)$استفاده می شود و براساس تعریف داریم:

$$f(n)=O(g(n)) \ : \ \exists \ \ c>0 \ \ ,n_0>0 \ ,\forall n>n_0 : 0<f(n)\le cg(n).$$

اغلب الگوریتم های مهم به یکی از زمان های زیر نیاز دارند:

زمان ثابت $f(n)=c=O(1)$

زمان خطی $f(n) = an + b = O(n)$

زمان چند جمله ای $f(n) = an^k + bn^{k-1} + ... + c = O(n^k)$

زمان الگوریتمی لگاریتم $f(n) = a\log(n) + b = O(\log(n))$

زمان نمایی $f(n) = ab^n + c = O(b^n)$

۱-۳- نمایش درخت k تایی

معمولترین روش ها برای نمایش یک درخت (گراف) نمایش آرایه ای و[1] پیونـدی[2] مـی باشـد و روش دیگری که ما در این رساله مورد بررسی قرار می دهیم نمایش دنباله ای[3] می باشد.

نمایش آرایه ای درخت:

در این حالت درخت به ترتیب از ریشه شماره گذاری می شود و گره های هر سطح از چپ بـه راسـت شماره گذاری می شوند سپس اطلاعات گره در یک آرایه ذخیره می شوند به طوری که اطلاعات گـره i ام در محل i از این آرایه قرار می گیرد و برای هر گره با اندیس i و $1 \le i \le n$ داریم :

اگر $i \ne 1$ باشد در اینصورت پدر i در $\left\lfloor \frac{i}{k} \right\rfloor$ می باشد و اگر $i = 1$ باشد i ریشه است. بـه عبـارت دیگـر فرزندان گره i ام در محل های $ik-1$ تا $ik+k-2$ آرایه قرار می‌گیرند. این روش ارائه درخت را برای همه درخت ها به آسانی می توان پیاده‌سازی کرد و تنها عیب این روش این است که فضای زیـادی از حافظه را نیاز دارد و درج یا حذف گره های درخت مستلزم جابجایی گره هاست که باعث تغییر شماره سطح گره ها می شود.(شکل ۱-۵)

	i	$\cdots$	$ik-1$	ik	$\cdots$	$ik+k-2$	

شکل ۱-۵- نمایش آرایه ای درخت k تایی- گره i ام به همراه فرزندانش

1.Array representation
2.Linked list
3.Sequence representation

نمایش پیوندی درخت:

با بکار بردن این نحوۀ نمایش می توان مشکلات مربوط به نمایش آرایه ای را حل کرد در این روش هر گره شامل یک مقدار داده و k اشاره گر به فرزندانش خواهد بود که در آن اشاره گر i ام به فرزند i ام گره اشاره خواهد کرد.قابل ذکر است که حجم حافظه برای این نوع نمایش نیز زیاد می باشد.

$Data$	$link(1)$	....	$link(k)$

شکل ۱-۶- ساختار یک گره در نمایش پیوندی

نمایش دنباله ای درخت:

در این روش گره ها و یا یال های درخت بر اساس یک روش از پیش تعریف شده برچسب گذاری می شوند، سپس با استفاده از یکی از روش های پیمایش درخت (پیش ترتیب – پس ترتیب و میان ترتیب که مخصوص دودویی است)، درخت پیمایش می شود و این اطلاعات بصورت یک دنباله لیست می گردد. دنباله های تولید شده یک رابطه یک به یک با درختان دارند و هر دنباله بیانگر یک درخت منحصر به فرد می باشد و به صورت رشتۀ بیتی $(1-0)$ یا رشته عددی می توانند باشند. منحصر به فرد بودن نمایش و قابلیت تبدیل دنباله به درخت شرایط لازم برای قابل قبول بودن دنباله می باشد علاوه بر آن طول دنباله ها و کارایی الگوریتم تولید آنها از نکات اصلی در نمایش درخت ها بوسیله دنباله ها می باشد.

الگوریتم تولید درخت:

الگوریتمی که تمام دنباله های متناظر با درختان k تایی با n گره را که تعداد آنها برابر $B(n,k)$ بوده واز فرمول زیر بدست می آید ایجاد کند الگوریتم تولید درخت نامیده می شود:

$$B(n,k)=\frac{1}{n(k-1)+1}\times\frac{(kn)!}{n!(nk-n)!}=\frac{1}{n(k-1)}\binom{kn}{n}.$$