Data Structures of Genome and Protein Sequences Indexing

Adeleh Asadi
PhD Candidate in Knowledge and Information Science;
University of Shiraz adelehasadi@gmail.com

Iranian Journal of Information Processing and Management

Iranian Research Institute for Information Science and Technology (IranDoc)
ISSN 2251-8223
eISSN 2251-8231
Indexed by SCOPUS, ISC, & LISTA
Vol. 31 | No. 2 | pp. 581-600
Winter 2016
https://doi.org/10.35050/JIPM010.2016.047

Received: 6, Apr. 2015 **Accepted: 10, Aug. 2015**

Abstract: Data structure is a tool for storage and retrieval of information which is named logic and mathematic way of specific data organization. Various sequences of genes and proteins in various creatures increases the amount of data in genome databases, and finding appropriate data structure and indexing are subject for many studies. String data structures are general data structure for genome indexing, and this article would review the many used three types of string data structure, suffix tree, suffix array, and Directed Acyclic Word Graphs.
This paper is a review of the literature related to three types of data, including genome databases indexing field, tree, postfix, postfix and graphs spiral array directly introduces the word. Findings of this research show that suffix tree and Directed Acyclic Word Graph (DAWG) structures need much space while suffix array needs less space. Against the Directed Acyclic Word Graph, suffix array can be stored on Memory Stick. Suffix tree and Directed Acyclic Word Graph are dynamic structures but as suffix array is a sorted out structure, it could hardly be changed.

Keywords: Data Structure; Genome; Indexing; Suffix Tree; Suffix Array; Directed Acyclic Word Graphs; Inverted File

ساختارهای داده نمایه‌سازی ژنوم و توالی‌های پروتئینی

عادله اسعدی شالی
دانشجوی دکتری علم اطلاعات و دانش‌شناسی؛ دانشگاه شیراز adelehasadi@gmail.com

پژوهش‌نامهٔ پردازش و مدیریت اطلاعات

دریافت: ۱۳۹٤/۰۱/۱۷ | **پذیرش:** ۱۳۹٤/۰۵/۱۹ | مقاله برای اصلاح به مدت ۲۴ روز نزد پدیدآور بوده است.

فصلنامه | علمی پژوهشی
پژوهشگاه علوم و فناوری اطلاعات ایران (ایرانداک)
شاپا (چاپی) ۸۲۲۳-۲۲۵۱
شاپا (الکترونیکی) ۸۲۳۱-۲۲۵۱
نمایه در LISTA، ISC، و SCOPUS
jipm.irandoc.ac.ir
دورهٔ ۳۱ | شمارهٔ ۲ | ص‌ص ۵۸۱-۶۰۰
زمستان ۱۳۹٤
https://doi.org/10.35050/JIPM010.2016.047

چکیده: ساختار داده ابزاری برای ذخیره و بازیابی بوده و به‌طور کلی، روش منطقی و ریاضی یک سازماندهی خاص از داده‌ها نامیده می‌شود. کشف توالی‌های مختلف ژنوم و پروتئین در جانداران مختلف نیاز به نمایه‌سازی آن و نوع ساختار دادهٔ متناسب، جهت بازیابی سریع‌تر را افزایش داده است. ساختارهای دادهٔ رشته‌ای در طول سال‌های اخیر پرکاربردترین ساختارهای داده نمایه‌سازی ژنوم بوده است.

به لحاظ روش پژوهش، این مقاله مروری بوده و با بررسی مقالات مختلف مرتبط، سه نوع ساختار داده نمایه‌سازی پایگاه‌های ژنوم از جمله رشته‌ای، درخت پسوندی، آرایهٔ پسوندی و نمودار مارپیچ مستقیم کلمه معرفی می‌شود.

نتایج پژوهش نشان می‌دهد که درخت پسوندی و نمودار مارپیچ مستقیم کلمه ساختارهایی با حجم بالا بوده و آرایهٔ پسوندی حجم کمتری را در حافظه اشغال می‌کند. درخت پسوندی و نمودار مارپیچ مستقیم کلمه، نسبتاً پویا بوده اما آرایهٔ پسوندی ساختاری مرتب‌شده است و تغییر داده‌ها در این ساختار به‌سختی صورت می‌گیرد. آرایهٔ پسوندی می‌تواند بر روی حافظه‌های جانبی ذخیره پیاده‌سازی شود؛ هرچند بازیابی داده‌ها در آن به‌کندی صورت می‌گیرد. اما در مورد نمودار مارپیچ مستقیم کلمه ذخیره‌سازی در حافظهٔ جانبی امکان نداشته و درخت‌های پسوندی نیز ناکارآمد هستند.

کلیدواژه‌ها: ساختار داده، ژنوم، نمایه‌سازی، درخت پسوندی، آرایهٔ پسوندی، نمودار مارپیچ مستقیم کلمه، فایل مقلوب

۱. مقدمه و بیان مسئله

سازماندهی مناسب داده‌ها برای بازیابی آسان و سریع آنها که به ساختار و یا ساختمان داده[1] معروف است، از موضوعات مهم در علوم اطلاع‌رسانی و کامپیوتر است. ساختار داده ابزاری برای ذخیره و بازیابی بوده و به‌طور کلی، روش منطقی و ریاضی یک سازماندهی خاص از داده‌ها نامیده می‌شود (جباریه ۱۳۷۹، ۴۸). ساختار داده روشی برای ساماندهی اطلاعات برای انجام هرچه بهتر عملیات پردازشی می‌باشد. هدف ساختمان داده ساماندهی داده‌ها در جهت کاهش هزینه‌های استفاده از منابع موجود و نیز کاهش استفاده از فضای حافظه، زمان پردازنده، میزان پهنای باند شبکه، میزان مراجعه به دیسک و یا مواردی مشابه است. الگوریتم طراحی‌شده بایستی با مصرف بهینهٔ منابع بتواند جواب مناسب را با توجه به میزان منابع در اختیار تولید نماید.

الگوریتم به‌عنوان مراحل حل یک مسئله یا انجام یک کار، مجموعه‌ای متناهی از دستورالعمل‌هایی است که برای رسیدن به خروجی‌های مطلوب با شروع از یک حالت اولیه به‌کار می‌رود (شریف‌زاده ۱۳۸۶). در طراحی پایگاه‌های اطلاعاتی، الگوریتم‌های طراحی‌شده بستگی مستقیم به نوع ساختار داده و نوع داده دارد. استفاده از ساختارهای داده متناسب با نوع داده، نحوهٔ ارتباط داده‌ها و نحوهٔ دسترسی و استفاده از داده‌ها امکان طراحی الگوریتم‌های مناسب در برنامه‌نویسی و طراحی پایگاه‌های اطلاعاتی را خواهد داد. لذا، آشنایی با ساختارهای داده از اهمیت به‌سزایی برخوردار است.

۲. نوع نیازها در بازیابی اطلاعات مربوط به ژن

داده‌های مربوط به پایگاه‌های اطلاعاتی ژنوم و توالی‌های پروتئینی نوع خاصی از داده‌های حرفی است که با داده‌های حرفی مورد استفاده در متون متفاوت هستند. ژنوم انسانی حاوی بیش از سه میلیارد ذرهٔ رمزی DNA است. DNA از چهار بلوک شیمیایی به نام نوکلئوتیدها تشکیل شده که شامل آدنین (A)، گوانین (G)، سیتوسین (C)، و تیمین (T) هستند. این کد چهار حرفی میلیون‌ها و حتی میلیاردها بار در سراسر ژنوم تکرار شده است. ساختار دادهٔ پایگاه‌های ژنوم باید به‌گونه‌ای باشد که امکان بازیابی هر نقطه از این توالی وجود داشته باشد. در ادامه به شرح ویژگی‌های این نوع داده خواهیم پرداخت.

با گسترش تحقیقات در حوزهٔ علوم زیستی و کشف توالی‌های مختلف ژنوم و پروتئین جانداران مختلف، حجم پایگاه‌های اطلاعات ژنوم افزایش یافته است. به‌عنوان مثال، ژنوم انسانی

1. data structure

دارای در حدود ۳ بیلیون جفت نوکلئوتید بوده و پایگاه‌های اطلاعات ژنوم انسانی بیش از ۱۵ بیلیون جفت نوکلئوتید را در بر گرفته است و برای ذخیره‌سازی آن ۴۵ گیگابایت حافظه صرف خواهد شد (Sadakane & Shibuya 2001, 1).

بایواینفورمتیک[1] علمی است که با بهره‌گیری از علوم ریاضیات کاربردی، اطلاع‌رسانی، آمار و علوم کامپیوتر به بررسی مسائل زیست‌شناسی می‌پردازد. بیشتر پژوهش‌های این حوزه عبارت‌اند از:

- ◇ تطبیق[2] و مقایسهٔ توالی‌های ژن و پروتئین در داخل یک گونه و یا بین گونه‌های مختلف جانداران که ارتباط بین گونه‌های مختلف جانداران را مشخص کرده و روند تکاملی آنها را نشان می‌دهد؛
- ◇ یافتن ژن خاص و بازیابی نظیر به نظیر رشتهٔ مورد نظر از طریق رونویسی آرِان‌ای[3] از ژن (Sung 2005)؛
- ◇ تطبیق توالی‌های کوچک ژن و اتصال آنها و تشکیل ژنوم کامل؛
- ◇ پیش‌بینی ساختارهای پروتئینی و تجلی‌های ژن[4] ناشناخته با بررسی شباهت توالی‌های آن با توالی‌های ژن‌هایی با عملکرد مشخص (Bioinformatic 2007).

توالی‌های تکرار[5] ژن نیز در پژوهش‌ها نقش مهمی داشته و بررسی آنها در روند یک پژوهش به کرات ضرورت می‌یابد. بررسی ویژگی‌ها و موقعیت توالی‌های تکراری در ژنوم نقشی کلیدی در بررسی، ارزیابی و مقایسهٔ ژنوم دارد (Abouelhoda, Kurtz, & Ohlebusch 2002). حجم توالی‌های تکراری در ژنوم بسیار بالاست به‌گونه‌ای که ۵۰ درصد از ۳ بیلیون جفت ژنوم انسان را توالی‌های تکراری تشکیل داده است. هرچه جاندار تکامل یافته‌تر باشد، توالی‌های تکرار در ژنوم او بیشتر است؛ مثلاً در مقایسه با انسان، ۷ درصد از ژنوم کرم و ۳ درصد از ژنوم مگس دارای توالی تکرار هستند. وجود توالی‌های تکرار با بروز بیماری نیز رابطهٔ مستقیم داشته و منجر به بسیاری از بیماری‌های ژنتیکی می‌شود. در نتیجه، نیاز به بازیابی آنها از موارد دیگری بود که ضرورت ایجاد پایگاه‌های بازیابی ژن را بیشتر نمایان ساخت (Miyamoto 2004).

بر اساس موارد مطرح‌شده در این بخش می‌توان این‌گونه نتیجه گرفت که در پژوهش‌های مربوط به حوزهٔ بایواینفورمتیک، پایگاه‌های اطلاعاتی با امکان بررسی دقیق توالی‌های مختلف به‌دست‌آمده از نمونه‌های مختلف، تعیین نوع توالی جدید بر اساس توالی‌های قبل و یا گونه‌های

1. bioinformatic
2. alignment
3. RNA
4. gene expression
5. repetitive sequences

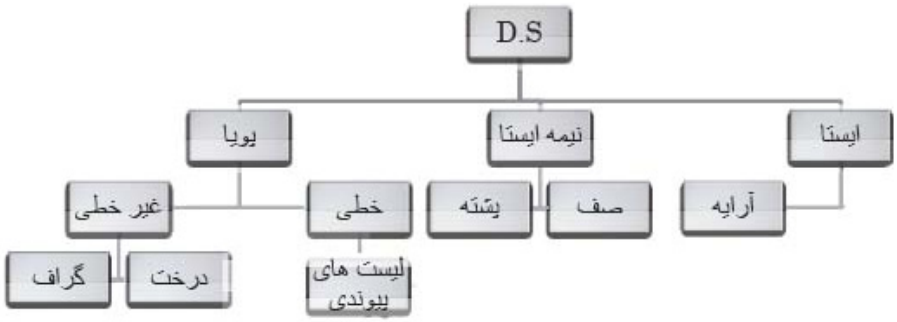
دیگر و بررسی تفاوت گونه‌های مختلف از طریق بررسی توالی ژنوم و پروتئین آنها و نیز امکان بازیابی هر نقطه از این توالی ژنوم مطلوب است.

سازماندهی اطلاعات و چگونگی دستیابی آسان و سریع آنها از ارکان علم اطلاع‌رسانی به‌شمار می‌رود. شیوه‌های سازماندهی اطلاعات بسته به نوع داده‌های آنها متغیر است. ژن‌ها و پروتئین‌ها رشته‌هایی حاوی اطلاعاتی هستند که از تکرار ۴ حرف به‌وجود آمده‌اند و همین ساختار تفاوت‌هایی را در نحوهٔ سازماندهی آنها ایجاد خواهد کرد. بررسی نیازهای اطلاعاتی محققان زیست‌شناسی و ساختارهای داده و شیوه‌های نمایه‌سازی ژنوم و پروتئین‌ها ما را با شیوه‌های سازماندهی این نوع داده‌ها آشنا خواهد کرد. این مقاله به معرفی انواع ساختار داده‌های عمومی و کاربرد آنها در سازماندهی ژنوم می‌پردازد.

۳. انواع ساختارهای داده

ساختارهای داده‌ها انواع گوناگونی دارند که هرکدام مناسب برنامه‌های مختلفی هستند. در کل، ساختارهای داده به سه نوع کلی ایستا، نیمه‌ایستا و پویا تقسیم می‌شوند. هر یک از انواع این ساختارها دارای ویژگی‌های خاصی هستند. به‌عنوان مثال، ایجاد تغییر در ساختارهای ایستا هزینه‌بر بوده و امکان تغییر وجود ندارد. در نوع پویا تغییرات مجاز بوده و در نوع نیمه‌ایستا تغییرات در شرایطی خاص امکان‌پذیر است.

آرایه‌ها[1] ساختارهای داده‌ای از نوع ایستا، پشته و صف از نوع نیمه‌ایستا و درخت‌ها[2]، گراف‌ها[3] و لیست‌های پیوندی[4] از نوع پویا هستند. شکل ۱ انواع ساختارهای داده ایستا، نیمه ایستا و پویا را نشان می‌دهد. در ادامه هر یک از این ساختارها به‌طور اجمالی معرفی می‌گردد.



شکل ۱. انواع ساختارهای داده (امیر جلیلی ایرانی ۱۳۹۱)

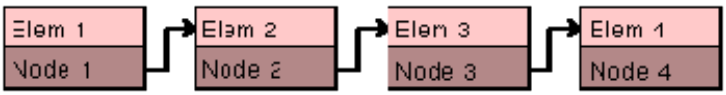
1. array 2. tree 3. graph 4. linked list

آرایه: آرایه‌ها پرکاربردترین ساختار دادهٔ مورد استفاده برای ذخیرهٔ مجموعه‌ای از عناصر اطلاعاتی هستند (Parlante 2001). یک آرایه، نمایه‌ای خطی[1] و شامل مجموعه‌ای از عناصر است که به‌صورت متوالی در حافظه ذخیره شده و اگر مرتب نشده باشد، به‌صورت خطی قابل بازیابی می‌باشد. اطلاعات اگر پراکنده بوده و مرتب نشده باشد، کامپیوتر مجبور است برای پیداکردن دادهٔ nام از یک مجموعهٔ داده، به‌ترتیب از ابتدای آرایه پردازش نموده، عنصر مورد جست‌وجو را با عنصر اول، دوم و ... تا دستیابی به عنصر مورد نظر مقایسه کرده، و به دادهٔ مورد نظر برسد که به این ترتیب در صورتی که حجم داده‌ها زیاد باشد، زمان زیادی صرف جست‌وجو خواهد شد. در مقابل، جست‌وجوی دودویی فقط بر روی آرایهٔ مرتب‌شده امکان‌پذیر است. بدین‌صورت که در آرایهٔ مرتب‌شده مانند فرهنگ لغت، جست‌وجو از وسط لیست شروع شده و با مقایسهٔ واژهٔ مورد جست‌وجو با واژهٔ قرارگرفته در وسط لیست برای حرکت بر روی لیست به‌منظور دستیابی به واژهٔ مورد نظر مشخص می‌شود و جست‌وجو تا دستیابی به واژهٔ مربوطه ادامه می‌یابد. سرعت بازیابی اطلاعات در روش دوم بیشتر است. تنها نقطهٔ ضعف این الگوریتم هزینه‌بربودن حذف و اضافهٔ داده‌ها از آرایه می‌باشد.

صف و پشته: پشته و صف جزء ساختار داده‌های نیمه‌ایستا هستند. گاهی می‌خواهیم در عملِ حذف و یا اضافه‌نمودن داده‌ها در هر مکانی از آرایه، محدودیت ایجاد کنیم؛ به‌طوری که فقط عملیات حذف و اضافه در ابتدا و انتهای لیست صورت گیرد. در این حالت از پشته استفاده می‌کنیم. پشته، لیست خطی است که عملیات حذف و اضافه تنها از یک طرف آن صورت می‌گیرد. در هر لحظه فقط عنصر بالائی پشته قابل دسترس است. صف، ساختمان داده‌ای است که کلیهٔ عملیات اضافه از یک سر آن و کلیهٔ عملیات حذف از انتهای دیگر آن انجام می‌پذیرد. صف در نرم‌افزارهایی که صف انتظار را برای دسترسی به منبعی برقرار می‌کنند، کاربرد دارد (تنها، آیت۱۳۸۷).

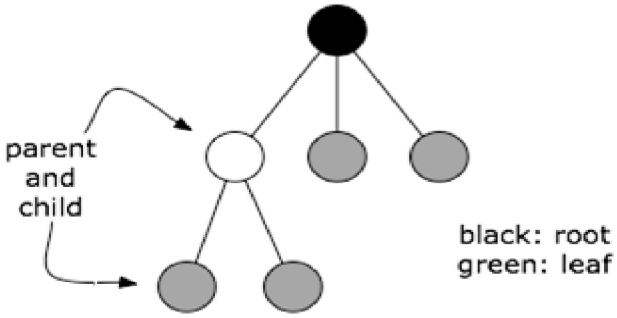
لیست‌های پیوندی: لیست پیوندی نوع دیگری از ساختارهای دادهٔ خطی پویا است و شامل مجموعه‌ای از عناصری است که با یک نظم خاص کنار هم قرار گرفته‌اند. به این ترتیب که هر عنصر داده‌ای اشاره‌گری[2] به عنصر بعد از خود دارد. لیست‌های پیوندی ساختارهای دادهٔ پویا بوده و حذف، اضافه و افزایش حجم اطلاعات در آن امکان‌پذیر است. شکل ۱ نشان‌دهندهٔ یک لیست پیوندی است.

1. linear index
2. pointer



شکل۲. ساختار یک لیست پیوندی (Kovacs 1998)

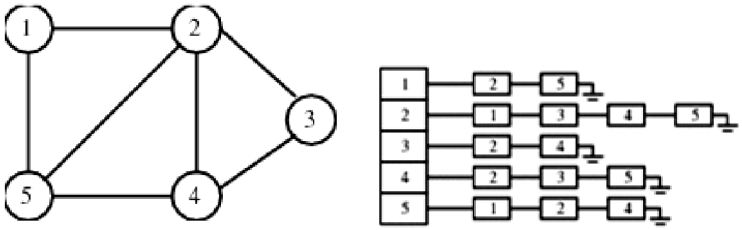
درخت‌ها و گراف‌ها: این دو نوع ساختار داده مشابه بوده و غیرخطی هستند. درخت‌ها ساختار داده‌ای هستند که با یک گره[1] ریشه آغاز شده و هر ریشه دارای یک و یا چند برگ می‌باشد. ریشه در بالای ساختار نمایش داده می‌شود و برگ‌ها در زیر ریشه قرار می‌گیرند. درخت‌ها ساختار سلسله‌مراتبی دارند. این بدان معنی است که هر گره می‌تواند تنها یک گره والدین داشته باشد. شکل۳ ساختار یک درخت را نشان می‌دهد.



شکل ۳. ساختار یک درخت (Black 2004)

گراف، ساختار دادهٔ پویا و غیرخطی است که از گره‌ها تشکیل شده و هر گره مانند درخت‌ها داده‌ای را شامل می‌شود، ولی بر خلاف درخت‌ها هر گره می‌تواند با گره‌های دیگر در ارتباط باشد (Bell college n.d.). همچنین، بین اجزای آن رابطهٔ سلسله‌مراتبی مثل درخت‌ها (رابطهٔ والدین و فرزندان در درخت پسوندی) برقرار نیست و رابطهٔ بین اجزای آن بر اساس احتمال وجود برقرار شده است (Mehta & Sahni 2005). شکل زیر ساختار یک گراف را نشان می‌دهد.

1. node



شکل٤. ساختار یک گراف (Skiena 2008)

٤. شاخص‌های ارزیابی ساختارهای داده

برای ارزیابی پایگاه‌ها و ساختارهای دادهٔ به‌کار رفته در پایگاه، چند خصیصه را مورد بررسی قرار می‌دهند. این خصیصه‌ها عبارت‌اند از:

- ◇ حجم کم مورد نیاز برای ساختن و بازیابی اطلاعات؛
- ◇ سرعت جست‌وجو؛
- ◇ انعطاف‌پذیری، پویایی و امکان به‌روزرسانی آسان؛
- ◇ در پایگاه‌های اطلاعاتی بزرگ، نمایه باید به سمت حافظه‌های جانبی پیش برود (Navarro 2003).

٥. ساختارهای داده مناسب پایگاه‌های اطلاعاتی ژنوم

معروف‌ترین ساختار دادهٔ متن مورد استفاده در موتورهای کاوش و پایگاه‌های اطلاعاتی متون که به‌دلیل پویایی و حجم پایین، کاربرد زیادی دارند، نمایه مقلوب[1] و یا فایل مقلوب[2] هستند (Puglisi, Smyth, & Turpin 2005)[3]. فایل‌های مقلوب نوعی لیست پیوندی هستند (مقسمی ۱۳۸۶) که به‌عنوان ساختار دادهٔ کمکی[4] و نمایه‌ای برای فایل اصلی به‌کار می‌رود (پائو ۱۳۷۸؛ مقسمی ۱۳۸۶؛ Black 2994). به این ترتیب، فایل مقلوب بر اساس کلمات موجود در متون یک پایگاه اطلاعاتی مرتب‌شده و در فهرست فایل مقلوب هر کلمه دارای شماره‌های مدارک مرتبط با آن است و کلمات به شماره‌های مدارک از طریق یک اشاره‌گر (شبیه به یک لیست پیوندی) پیوند دارند. زمانی که به‌دنبال یک ساختار داده در کلمهٔ محور باشیم، فایل‌های مقلوب بسیار مناسب

---

1. inverted index
2. inverted file

۳. در متون، گاه این نمایه با نام Inverted list و یا Postings file نیز مطرح می‌شود.

۴. این نوع ساختار داده‌ها به نمایه و یا شاخص خارجی (external index) معروف است (Black 2004).

است. این فایل حجم زیادی اشغال نکرده و می‌تواند موقعیت کلمات در متون مختلف را نیز نگهداری کند و جست‌وجوی دقیق‌تری را ارائه دهد. تطبیق‌پذیری این ساختار امکان چندین نوع پرس‌وجو[1] (عین کلمه، استفاده از عملگرهای بولی، رتبه‌بندی کلمات و غیره) را می‌دهد (Grossi & Vitter 2005). به‌دلیل اینکه جست‌وجو در این ساختار باید از نوع عبارت و واژه باشد، این ساختار به نمایهٔ کلمه‌محور معروف است و برای سازماندهی توالی‌های ژن و نیز زبان‌های شرق آسیا که ساختاری رشته‌ای دارند، مناسب نیست.

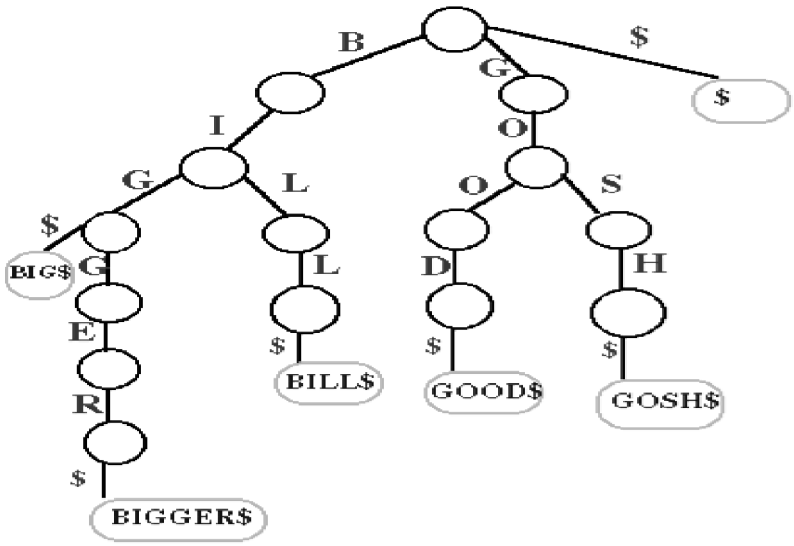
ساختارهای تمام‌متن مانند آرایهٔ پسوندی، درخت پسوندی و گراف مارپیچ مستقیم کلمه از جمله ساختار داده‌های ارائه‌شده برای حل این مشکل هستند. بسیاری از مسائل زیست‌شناختی مثل، جست‌وجوی توالی‌های ژنی، بازیابی توالی‌های تکرار، اتصال توالی‌ها، تطبیق توالی ژن و سازماندهی توالی ژن‌ها از طریق این ساختار امکان‌پذیر است. این نوع ساختارهای داده به تمام‌متن[2] معروف است که در آن، کل متن و حروف موجود در متن قابل بازیابی بوده و در تلاش است نمایه را جایگزینی برای کل متن نماید (Grossi 2004). این نوع ساختار داده در نمایه‌سازی خودکار پایگاه‌های ژنوم به‌کار برده می‌شوند. این مقاله به معرفی سه نوع ساختار دادهٔ رشته‌ای[3]، درخت پسوندی، آرایهٔ پسوندی و گراف مارپیچ مستقیم کلمه خواهد پرداخت. قبل از بررسی این سه نوع ساختار، *تری‌ها*[4] که مبنای ساختن ساختارهای دیگر است، معرفی می‌شود.

٦. ساختار داده تری

تری که برگرفته از کلمهٔ بازیابی[5] بوده و نوعی ساختار داده است که برای سازماندهی و جست‌وجوی فرهنگ‌ها استفاده می‌شود (Mehta & Sahni 2005, 1-28). به‌علاوه، با توجه به اینکه این ساختار داده توانایی جست‌وجوی حرف به حرف را دارد، از این ساختار در تحلیل‌های لغوی برای ترجمهٔ متون، جست‌وجوهای کتابشناختی، بررسی غلط‌های املایی، پردازش زبان طبیعی استفاده می‌شود (Aoe, Morimoto, & Sato 1992). این ساختار در دههٔ ۱۹۶۰ توسط «فردکین»[6] ارائه شده و یکی از قدیمی‌ترین ساختارهای داده به‌شمار می‌رود (Gent et al 2007). یک ساختار دادهٔ *تری*، درختی است برای جست‌وجوی رشته‌ای با تعداد حروف الفبای پایهٔ مشخص، که هر یک از لبه‌های آن با نویسه‌ای مشخص شده، و مسیر ریشه تا برگ نشان‌دهندهٔ یک رشته می‌باشد (Sung 2005). ساختار *تری* در اصل درختی است که برای ذخیره و بازیابی رشته‌ای از نشانه‌ها[7] به‌کار

1. query
2. full text
3. string data structure
4. trie
5. retrieval
6. Fredkin
7. symbols

می‌رود و ایدهٔ مهم و متمایزتری از ساختار درخت‌ها این است که در *تری*، توالی‌های دارای پیشوندهای مشابه، گره‌ها و لبه‌های مشترک خواهند داشت (Gent et al. 2007). به‌عنوان مثال، اگر کلمات BIG, BIGGER, BILL, GOOD GOSH کلمات یک متن در نظر گرفته شود، ساختار *تری* آن به شکل زیر خواهد بود.



شکل ۵. ساختار تری (McGill University 1997)

در ساختار *تری* هر یک از حروف کلمات به‌عنوان یک گره از ساختار درختی در نظر گرفته شده است. به این ترتیب که تمامی کلماتی که دارای حروف آغازین مشابه باشند، در زیر یک شاخه قرار خواهند گرفت و حروف به‌ترتیبِ حضور در متن از ریشه تا انتهای شاخه (آخرین حرف یک رشته) ذخیره خواهد شد. این ساختار برای بازیابی متن می‌تواند به‌کار رود، اما هنوز نمی‌توان آن را در نمایه‌سازی ژنوم به‌کار برد (امکان بازیابی بخشی از یک کلمه وجود ندارد). نوع دیگری از *تری‌ها* که پایهٔ ساختن درخت‌های پسوندی است، *تری* پسوندی است که در آن تمامی پسوندهای ممکن در یک کلمه در نظر گرفته شده و روابط درختی بین آنها برقرار می‌شود. پسوند رشتهٔ[1] S به طول n، زیررشته‌ای از رشتهٔ S است که ابتدای آن از هر قسمت از رشتهٔ S و انتهای این زیررشته n می‌باشد. برای مثال، اگر رشته $GOOGOL در نظر گرفته شود، (که $

۱. رشته، مجموعهٔ کاراکترهایی را گویند که در انتهای آنها علامت $ وجود داشته باشد.

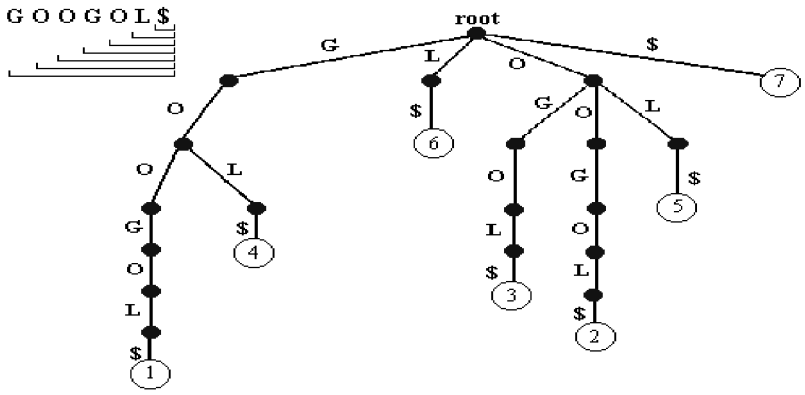
نشان‌دهندهٔ انتهای رشته باشد) مجموعهٔ پسوندهای $GOOGOL به شکل زیر خواهد بود:

GOOGOL$
OOGOL$
OGOL$
GOL$
OL$
L$
$

و *تری پسوندی* زیر را خواهد داشت:

رشته: $ L O O G O G

موقعیت: ۷ ۶ ۵ ۴ ۳ ۲ ۱



شکل ٦. تری پسوندی رشته $GOOGOL (McGill University 1997)

تری پسوندی امکان جست‌وجوی هر قسمت از یک کلمه را فراهم می‌آورد. در توالی DNA نیز از طریق این ساختار می‌توان هر بخشی از توالی ژنوم را بازیابی نمود (McGill University 1997). مشکل عمده در استفاده از این ساختار، حجم زیاد نمایه و نیز سرعت پایین ایجاد این ساختار است. پیچیدگی زمانی جست‌وجوی آن $O(md)$ (m اندازه کلمهٔ جست‌وجو و d تعداد حروف الفبا[1] است) و به‌طور میانگین فضای مورد نیاز برای ذخیرهٔ داده‌ها بر اساس این ساختار برابر $O(n^2)$ می‌باشد (Mäkinen, Navarro 2005)؛ به این معنی که برای ذخیره‌سازی n تعداد کاراکتر به فضایی به اندازهٔ n^2 نیاز است. در نتیجه، هزینهٔ بالایی را به سیستم تحمیل خواهد نمود. هر کاراکتر

۱. به‌عنوان مثال، توالی‌های ژن ۴ حرف الفبا دارند.

گره خاصی را اشغال می‌نماید و شاخه‌هایی با کاراکترهای مشابه جداگانه تکرار می‌شوند. این موارد منجر به افزایش حجم ساختار می‌شود. ساختار دادهٔ درخت پسوندی با فشرده‌سازی و ساختار نمودار مارپیچ مستقیم کلمه با خلاصه‌سازی ساختارهایی برای اصلاح *تری پسوندی* به‌شمار می‌روند (Crochemore & Lecroq n. d.).

۷. درخت پسوندی

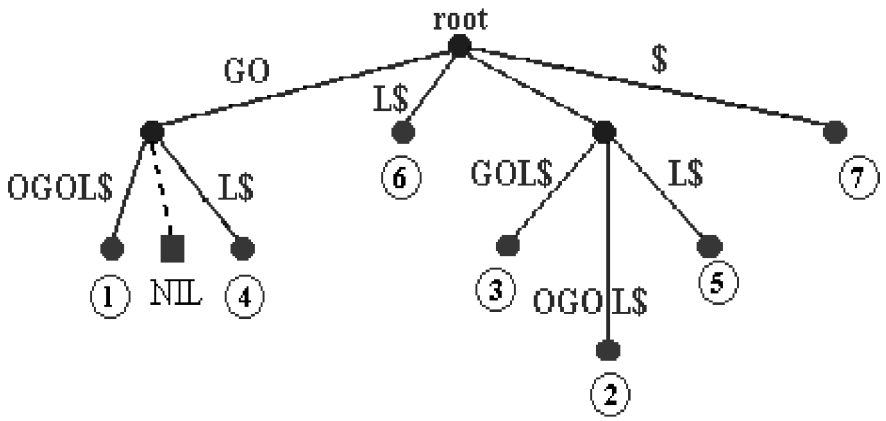
جست‌وجوی متون ساختاربندی‌نشده مثل توالی‌های ژن و برخی از زبان‌های شرق آسیا، مشکل جدیدی در نمایه‌سازی متون ایجاد نمود. آرایه و درخت پسوندی[1] از ساختارهای دادهٔ پرکاربرد متون ساختاربندی‌نشده هستند (Grossi & Vitter 2005). الگوریتم ایجاد درخت پسوندی برای نخستین بار در سال ۱۹۷۳ توسط واینر[2] ارائه شد و کاربردی‌های زیادی در نمایه‌سازی پیدا کرد (Crochemore & Lecroq n. d.).

درخت پسوندی رشتهٔ S به نشانهٔ T(S) شکل فشردهٔ *تری پسوندی* بوده و دارای ویژگی‌های زیر است:

- ◇ درخت پسوندی دارای n انتهای شاخه می‌باشد که از شماره ۱ تا n شماره‌گذاری شده است.
- ◇ هر گره (به استثنای گره انتهایی) حداقل دارای دو گره فرزند است. به این ترتیب که در *تری پسوندی* هر گره که دارای یک فرزند باشد با گره بعدی یکی می‌شود (Sung 2005).
- ◇ همهٔ لبه‌های خارج شده از گره‌ها با نویسه‌های متفاوت شروع می‌شوند.
- ◇ مجموعهٔ راه‌های ایجاد شده از ریشه تا انتهای شاخه‌ها، شامل کلیهٔ پسوندهای موجود برای رشتهٔ S هستند (Grossi & Italiano 1996).

درخت پسوندی رشتهٔ $GOOGOL به شکل زیر خواهد بود:

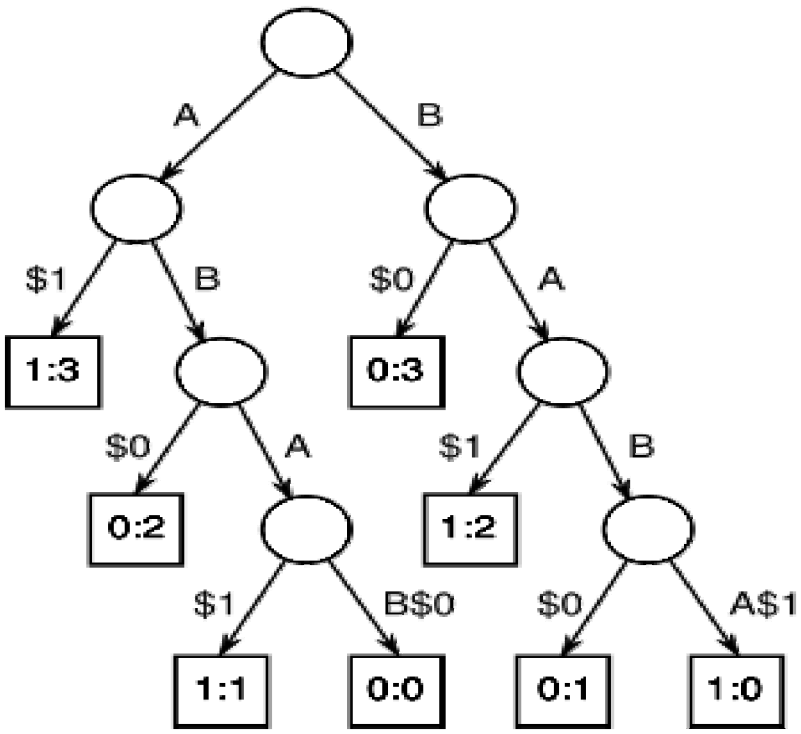
1. suffix tree
2. Weiner



شکل ۷. درخت پسوندی رشته $GOOGOL (McGill University 1997)

ساخت یک رشته با فضای حافظه‌ای $O(n^2)$ در *تری* پسوند به $O(n)$ در ساختار درخت بهبود یافته است (Grossi & Vitter 2005) و زمان بازیابی زیررشته m برابر با $O(n + m)$ است. همچنین، زمان ساخت این ساختار در بدترین حالت برابر با $O(n^2)$ می‌باشد (Stoye 2006).

درخت پسوندی تعمیم‌یافته امکان استفاده از درخت پسوندی در چندین رشته و متن‌های مختلف را می‌دهد و از این طریق می‌توان با استفاده از ساختار درخت پسوندی به جست‌وجو در یک پایگاه اطلاعاتی وسیع مانند توالی‌های DNA و پروتئین پرداخت. مثال زیر ساختار درخت تعمیم‌یافته را بهتر نمایان می‌کند. شکل زیر درخت پسوندی تعمیم‌یافته برای رشته‌های ABAB و BABA است.



شکل ۸. درخت پسوندی تعمیم‌یافته (Wikipedia 2002)

به این ترتیب، با اختصاص کاراکترهای انتهایی اختصاصی برای هر رشته می‌توان رشته‌ها را از هم جدا نمود. از سوی دیگر، برای جداسازی هر یک از لبه‌ها بخش اول جفت شاخص برای مشخص‌کردن رشتهٔ مربوطه به‌کار رفته و بخش دوم ابتدای زیررشته را نشان می‌دهد. در نتیجه، در کل برای نمایش برچسب لبه‌ها، سه قسمت (i, j, l) خواهد داشت که i نشان‌دهندهٔ رشتهٔ مربوطه، j ابتدای زیررشته و l انتهای آن خواهد بود (Mehta & Sahni 2005, 4-29).

۸. آرایهٔ پسوندی

«مانبر[1] و مایرز[2]»در سال ۱۹۹۳ آرایهٔ پسوندی[3] را برای نمایه‌سازی طراحی کردند که از نظر ساختار شباهت زیادی با درخت پسوندی داشت، اما ساختار آن به شکل درخت نبود (Schurmann, & Stoye 2005). آرایهٔ پسوندی آرایه‌ای است از پسوندهای یک رشته به شکل یک

1. Manber 2. Myers 3. suffix array

فرهنگ مرتب‌سازی شده. این ساختار برای تطبیق رشته‌ها، تحلیل ژنوم و فشرده‌سازی متون استفاده می‌شود (Dementiev, Karkkainen, & Sanders 2006).

اگر S[1..n] یک رشته به طول n و با حروف الفبایی Σ باشد و $ به‌عنوان کاراکتر پایانی منحصربه‌فرد رشته در نظر گرفته شود، به‌ازای هر i = 1, 2, ….., n، S[i…n] پسوندی از رشتهٔ S خواهد بود. آرایهٔ پسوندی (SA[1..n]) مجموعهٔ پسوندهای S با ترتیب الفبایی می‌باشد. برای مثال، شکل زیر پسوندهای رشته $GOOGOL است؛

موقعیت پسوند

۱. GOOGOL$
۲. OOGOL$
۳. OGOL$
۴. GOL$
۵. OL$
۶. L$
۷. $

به این ترتیب آرایهٔ پسوندی این رشته به مطابق جدول ۱ خواهد بود (Hon, Lam, & Sadakane 2007).

جدول ۱. آرایهٔ پسوندی رشته $GOOGOL

i	SA[i]	پسوند رشته S
۱	۷	$
۲	۴	GOL$
۳	۱	GOOGOL$
۴	۶	L$
۵	۲	OOGOL$
۶	۳	OGOL$
۷	۵	OL$

با بررسی این ساختار و درخت پسوندی می‌توان دریافت که این ساختار حجم کمتری خواهد داشت. آرایهٔ پسوندی برای نمایه‌سازی ژنوم در حدود ۴ برابر حجم کمتر نسبت به درخت پسوندی اشغال می‌کند (Manber & Myers 1993). جست‌وجوی دودویی با توجه به ترتیب فرهنگی آن سرعت بازیابی اطلاعات را بالا برده است. اما همین، ساختار به‌روزرسانی اطلاعات را مشکل نموده است.

مقایسهٔ این ساختار با فایل‌های مقلوب که یکی از پرکاربردترین ساختار جست‌وجو در پایگاه‌های اطلاعاتی به‌شمار می‌رود، نشان می‌دهد که:

- ◇ جست‌وجوهای عبارتی آرایه‌های پسوندی در مقایسه با فایل‌های مقلوب بهتر انجام می‌شود؛
- ◇ آرایهٔ پسوندی محدودیت جست‌وجوی صرف کلمات را ندارد؛
- ◇ فایل‌های مقلوب، فضا و زمان جست‌وجوی مناسب‌تری نسبت به آرایهٔ پسوندی دارند (Seki 2005).

با وجود کاربردهای گستردهٔ آرایهٔ پسوندی، هنوز حجم زیاد ساختار و زمان زیاد برای ایجاد ساختار، از محدودیت‌های آن به‌شمار می‌رود.

۹. نمودار مارپیچ مستقیم کلمه

نمودار مارپیچ مستقیم کلمه[1] نوعی گراف است که امکان بازیابی بخش‌هایی از یک کلمه را می‌دهد و یکی از ساختارهای مورد استفاده در پایگاه‌های دادهٔ ژنوم است. نمودار مارپیچ مستقیم کلمه شکل خلاصه‌شدهٔ *تری* است. در این ساختار تعداد گره‌ها کمتر شده است و تقریباً مشابه *تری* روابط بین کاراکترهای موجود در رشته را ارائه می‌نماید (Mehta & Sahni 2005, 2-30). نخستین بار بولمر[2] این ساختار را طراحی نمود. به‌طور کلی، یک نمودار مارپیچ مستقیم کلمه شامل ویژگی‌های زیر است:

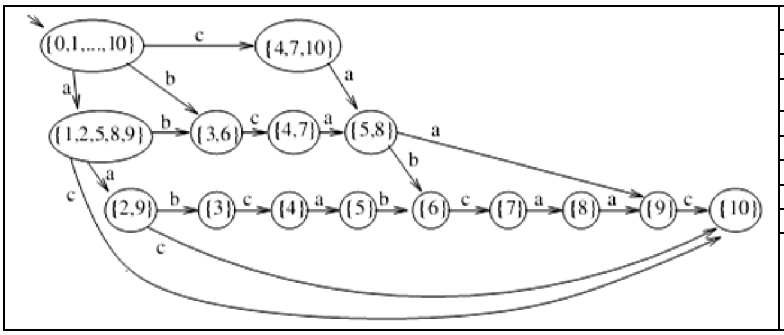
- ◇ دو گره کاملاً مشخص به‌عنوان گره ابتدایی و گره پایانی وجود دارد؛
- ◇ لبه‌ها با زیررشته‌ای از رشتهٔ اصلی S برچسب‌گذاری می‌شوند؛
- ◇ برچسب‌های لبه‌هایی که یک گره مشابه را ترک می‌کنند، نباید نویسهٔ مشابه داشته باشند؛
- ◇ هر پسوند از رشتهٔ S مربوط به مسیری از گراف است که از گره ابتدایی آغاز شده و به گره پایانی ختم می‌شود. به این ترتیب که مطابق ترتیب حروف پسوندها، برخی از برچسب‌های لبه‌ها به هم ملحق می‌شوند و از تکرار گره‌هایی با لبه‌های مشابه جلوگیری می‌کنند (Inenaga et al. 2005).

به‌عنوان مثال، اگر s= aabcabcaac باشد ساختار نمودار مارپیچ مستقیم کلمهٔ آن به‌شکل زیر خواهد بود:

1. Directed Acyclic Word Graph (DAWG)
2. Bulmer

S= a a b c a b c a a c
1 2 3 4 5 6 7 8 9 10

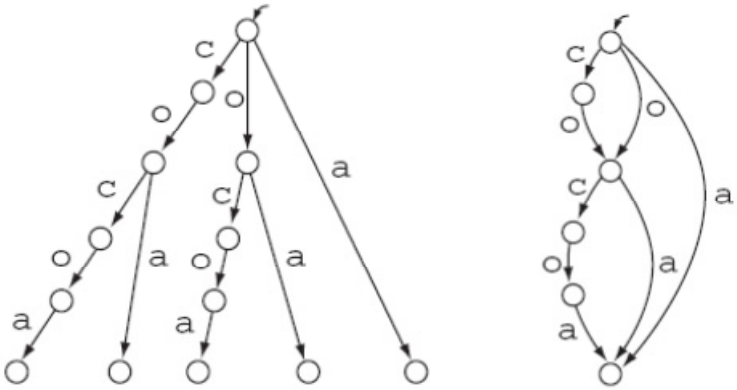
پسوندهای رشته S



aabcabcaac
abcabcaac
bcabcaac
cabcaac
abcaac
bcaac
caac
aac
ac
c

شکل ۹. نمودار مارپیچ مستقیم کلمه برای رشته aabcabcaac (Ehrenfeucht & McConnell 2002)

گره ابتدایی در سمت چپ با پیکان مشخص شده است که گرهی خالی و بدون نویسه می‌باشد. به تعداد حروف الفبای رشتهٔ S، لبه‌هایی با نویسه‌های متفاوت گره ابتدایی را ترک می‌کنند. لبه‌ها به گره‌هایی می‌رسند که مکان‌های نویسه مربوط به برچسب لبهٔ قبل را نشان می‌دهند. سپس نویسهٔ بعدی پسوندها، به ساختار اضافه می‌شود و در صورتی که بخشی از نویسه‌های پسوندها با هم مشابهت داشته باشند، لبه‌ها ملحق شده و یک مسیر واحد را به‌وجود می‌آورند. با این بررسی، تفاوت‌های این ساختار با *تری پسوندی* مشخص خواهد شد. شکل ۹ تفاوت بین ساختار نمودار مارپیچ مستقیم کلمه را با *تری پسوندی* برای رشتهٔ cocoa نشان می‌دهد



شکل ۱۰. تری پسوندی (شکل سمت چپ) و نمودار مارپیچ مستقیم کلمه (شکل سمت راست) برای رشته cocoa (Miyamoto et al. 2004)

نمودار مارپیچ مستقیم کلمه ساختار *تری پسوندی* را به شکل افقی، درخت پسوندی آن را به‌صورت عمودی فشرده می‌کند (Jain 2004). نوع جدیدی از نمودار مارپیچ مستقیم کلمه با نام نمودار مارپیچ مستقیم کلمهٔ فشرده،[1] عملکردی مشابه درخت‌های پسوندی را در مورد نمودار مارپیچ مستقیم کلمه اعمال می‌کند (Jain 2004). به این ترتیب که گره‌های متوالی بدون انشعاب، با هم ادغام شده و یک گره را ایجاد می‌کنند و به‌صورت باز هم تعداد گره‌ها کاهش خواهد یافت. نمودار مارپیچ مستقیم کلمه با زمان $O(n)$ ساخته شده و جست‌وجوی زیررشته در این ساختار نیز با زمان $O(m + S)$ خطی می‌باشد. اما برای ذخیره‌سازی داده‌هایی با طول n به فضایی به اندازهٔ ۱۵n نیاز است (Navarro 2003). ساختار نمودار مارپیچ مستقیم کلمه در مقایسه با درخت پسوندی کاربرد کمتری داشته است (Jain 2004).

۱۰. بحث و نتیجه‌گیری

اگر تعریف ذیل برای نمایه‌سازی و مدرک را بپذیریم، نمایه راهنمایی برای مدرک، بخشی از مدرک، یا تعدادی از مدارک موجود در پایگاه اطلاعات است که برای آشکارساختن ماهیت آنها ساخته می‌شود (اشرفی‌ریزی و کاظم‌پور ۱۳۸۶) و مدرک عبارت است از هر نوع نوشتهٔ خطی، چاپی، عکسی، و یا به‌صورت‌های دیگر و هر شیء مادی که بتوان از محتوای آن اطلاعی به‌دست آورد (سلطانی ۱۳۷۹). از سوی دیگر، می‌دانیم که نمایه‌سازی روش بنیادین سازماندهی اطلاعات است (نیازی ۱۳۸۱–۱۳۸۵) و سازماندهی اطلاعات از ارکان مهم علم کتابداری و اطلاع‌رسانی است. پس، می‌توان گفت که علم کتابداری و اطلاع‌رسانی محدود به حوزهٔ اطلاعات متنی وابسته به زبان‌های گفتاری بشر نیست و وظیفهٔ سازماندهی همهٔ اشکال اطلاعات بشر را بر عهده دارد. این مقاله سعی داشته است به نمایه‌سازی ماشینی تمام‌متن نویسه‌محور[2] بپردازد و برخی ساختارهای ذخیره و بازیابی داده‌ها در این نوع نمایه‌سازی را معرفی نماید.

در این مقاله سه نوع نمایهٔ توالی‌های ژنوم، درخت پسوندی، آرایهٔ پسوندی و نمودار مارپیچ مستقیم کلمه معرفی و برخی ویژگی‌های آنها مورد بررسی قرار گرفت.

به‌طور کلی، ارزیابی ساختارهای داده از چهار منظر بیشتر حائز اهمیت است:

- ◇ کم‌حجم‌بودن ساختار داده که با میزان کارایی آن در ارتباط مستقیم است؛
- ◇ پویایی و امکان به‌روزرسانی اطلاعات؛
- ◇ امکان ذخیره‌سازی در حافظه‌های جانبی؛ و

1. Compact Directed Acyclic Word Graph
2. character level

◇ سرعت جست‌وجوی اطلاعات.

با بررسی ساختارهای دادهٔ معرفی‌شده در این مقاله، ارزیابی ساختارها به این ترتیب خواهد بود که درخت پسوندی و نمودار مارپیچ مستقیم کلمه ساختارهایی با حجم بالا هستند، حال آنکه آرایهٔ پسوندی با توجه به خطی‌بودن ساختار، حجم کمتری را در حافظه اشغال خواهد نمود. درخت پسوندی و نمودار مارپیچ مستقیم کلمه نسبتاً پویا بوده و امکان به‌روزرسانی داده‌ها در این ساختارها وجود دارد، اما آرایهٔ پسوندی ساختاری مرتب شده بوده و تغییر داده‌ها در این ساختار به‌سختی صورت می‌گیرد. آرایهٔ پسوندی می‌تواند بر روی حافظه‌های جانبی ذخیره پیاده‌سازی شود؛ با وجود اینکه بازیابی داده‌ها در آن به‌کندی صورت می‌گیرد؛ اما در مورد نمودار مارپیچ مستقیم کلمه امکان ذخیره‌سازی در حافظهٔ جانبی امکان نداشته و این کار در مورد درخت‌های پسوندی نیز ناکارآمد می‌باشد. از سوی دیگر، با توجه به اینکه آرایهٔ پسوندی ساختاری مرتب‌شده است، سرعت جست‌وجوی اطلاعات در آن از دیگر ساختارها بیشتر خواهد بود.

فهرست منابع

اشرفی‌ریزی، حسن، و زهرا کاظم‌پور. ۱۳۸۶. ارزیابی نمایه و نمایه‌سازی: معیارها و استانداردها. *علوم و فناوری اطلاعات* ۲۳ (۱ و ۲). (دسترسی در اردیبهشت ۱۳۹۴) از http://jipm.irandoc.ac.ir/files/site1/user_files_e1671e/admin-A-10-1-44-aae09b7.pdf

پائو، میراندالی. ۱۳۷۸. *مفاهیم بازیابی اطلاعات*. ترجمهٔ اسدالله آزاد و رحمت‌الله فتاحی. مشهد: دانشگاه فردوسی مشهد.

تنها، جعفر، ناصر آیت. ۱۳۸۷. *ساختمان داده‌ها و الگوریتم‌ها*. تهران: دانشگاه پیام نور.

جباریه، علیرضا. ۱۳۷۹. *ساختمان داده‌ها ۱ و ۲*. تهران: جهان نو.

جلیلی ایرانی، امیر. ۱۳۹۱. *ساختمان داده‌ها*. http://amirjalili.ir/?cat=10 (دسترسی در خرداد ۱۳۹۴)

سلطانی، پوری، و فروردین راستین. ۱۳۷۹. *دانشنامهٔ کتابداری و اطلاع‌رسانی*. تهران: فرهنگ معاصر.

شریف‌زاده، مناف. ۱۳۸۵. *آشنایی با طراحی الگوریتم*. روزنامه ایران آنلاین، ۱۴ بهمن http://www.magiran.com/npview.asp?ID=1336456 (دسترسی در اردیبهشت ۱۳۹۴)

مقسمی، حمیدرضا. ۱۳۸۶. *ذخیره و بازیابی اطلاعات*. تهران : گسترش علوم پایه.

نیازی، سیمین. ۱۳۸۱-۱۳۸۵. نمایه‌سازی. در *دائرةالمعارف کتابداری و اطلاع‌رسانی*. (ج. ۲). تهران: کتابخانه ملی جمهوری اسلامی ایران. http://portal.nlai.ir/daka/Wiki%20Pages/ (دسترسی در اردیبهشت ۱۳۹۴)

Abouelhoda, Mohamed Ibrahim, Stefan Kurtz, & Enno Ohlebusch. 2002. The enhanced suffix array and its applications to genome analysis. Proceedings of the 2nd Workshop on Algorithms in Bioinformatics, LNCS 2452, Springer-Verlag, (pp. 449-463). http://www.dbis.informatik.huberlin.de/dbisold/lehre/SS05/genome_syntax/AboKurOhl2002.pdf

(accessed 2015)

Aoe, Jun -Ichi, Katsushi Morimoto, & Takashi Sato. 1992. An Efficient Implementation of Trie Structures. *Sofiware -Practice and Experience*, 22 (9): 695–721. http://sc.snu.ac.kr/~xuan/spe777ja.pdf (aaccessed 2015)

Bell College (n.d.). *Software Development 2*, *Graph data structures.* http://hamilton.bell.ac.uk/swdev2/notes/notes_18.pdf (accessed 2015)

Bioinformatics. 2007. *Encyclopedia of Wikipedia*. http://en.wikipedia.org/wiki/Bioinformatics (accessed 2008)

Black, Paul E. 2004. external index, in Dictionary of Algorithms and Data Structures [online], Paul E. Black, ed., U.S. National Institute of Standards and Technology. 17 December 2004. http://www.nist.gov/dads/HTML/externalindx.html (accessed 2015)

Crochemore, Maxime, and Thierry Lecroq (n.d.). Suffix Tree. http://www-igm.univ-mlv.fr/~lecroq/articles/suffix.pdf (accessed 2008)

Dementiev, R., Kärkkäinen J, & P Sanders. 2006. Better external memory suffix array construction, Journal of Experimental Algorithmics (JEA), 12(3-4): 1-24. http://dl.acm.org/citation.cfm?id=1402296 (accessed 2015).

Ehrenfeucht, Andrzej, & Ross M. McConnell. 2001. String Searching. In Mehta, Dinesh P; sahni, Sartaj, (ed), *Handbook of data structures and applications* (pp. 30-54) Chapman & hall/CRC: Boca Raton. http://www.cs.colostate.edu/~rmm/dawgs.pdf (accessed 2015)

Generalised suffix tree (n.d.). Encyclopedia of Wikipedia. http://en.wikipedia.org/wiki/Generalised_suffix_tree (accessed 2015)

Gent, Ian P., Christopher Jefferson, Ian Miguel, and Peter Nightingale. 2007. Data Structures for Generalised Arc Consistency for Extensional Constraints. In AAAI 2007: 191-197. http://www.cs.st-andrews.ac.uk/~pn/AAAI07.pdf (accessed 2015)

Grossi, Roberto. 2004. *Recent Advances on Text Indexing and String Algorithms*. Paper presented at the bioinformatics Seminar, University of Rennes. http://www.di.unipi.it/~grossi (accessed 2015)

_____, and Giuseppe F Italiano. 1996. Suffix trees and their applications in string algorithms. Rapporto di Ricerca CS-96-14, Università “Ca' Foscari” di Venezia, Italy. http://www.di.unipi.it/~grossi/IND/survey.pdf (accessed 2015)

_____, and Jeffrey Scott Vitter. 2005. Compressed suffix arrays and suffix trees with applications to text indexing and string matching. *SIAM journal on computing* 35 (2): 378-405. http://www.di.unipi.it/~grossi/PAPERS/sicomp05.pdf (accessed 2015)

Hon, Wing-Kai, Tak-Wah Lam, and Kunihiko Sadakane. 2007. A Space and Time Efficient Algorithm for Construction Compressed Suffix Arrays. *Algorithmica* 48 (1): 28-36. http://www.cs.nthu.edu.tw/~wkhon/papers/HLSSY05.pdf (accessed 2015)

Inenaga, Shunsuke et al. (eds). (2005). On-Line Construction of Compact Directed Acyclic Word Graphs? *Discrete Applied Mathematics* 146 (2): 156-179. http://www.shino.ecei.tohoku.ac.jp/~ayumi/papers/SPIRE2001_SCDAWG.pdf (accessed 2015)

Jain, Anoop. 2004. Performance Analysis of Horizontally Compacted String Indexes. masters thesis, Indian institute of science, Bangalore, MA. http://dsl.serc.iisc.ernet.in/publications/thesis/anoop.pdf (accessed 2015)

Kovacs, Daniel. 1998. Tutorial on Linked Lists. http://www.fortunecity.com/skyscraper/false/780/linklist.html (accessed 2008)

Mäkinen Veli, Gonzalo Navarro 2005. Succinct suffix arrays based on run-length encoding Combinatorial Pattern Matching, 45-56. Berlin: pringer http://personales.dcc.uchile.cl/~gnavarro/ps/njc05.pdf(accessed 2015)

Manber, Udi, and Gene Myers. 1993. Suffix arrays: A new method for on-line string searches. *SIAM Journal on Computing* 22 (5): 935-948. http://www.webglimpse.net/pubs/suffix.pdf (accessed

2008)

McGill University: School of Computer Science. 1997. Data structures and algorithms. http://www.cs.mcgill.ca/~cs251/OldCourses/1997/topic24/ - 22k (accessed 2008)

Mehta, Dinesh P, and Sartaj Sahni (ed). 2005. *Handbook of data structures and applications*. Chapman & hall/CRC: Boca Raton.

Miyamoto, Satoru, Shunsuke Inenaga, Masayuki Takeda, and Ayumi Shinohara. 2004. Ternary directed acyclic word graphs. *Theoretical Computer Science*: *Implementation and Application of Automata* 328 (1-2): 97-111. http://www.cpe.ku.ac.th/~anan/courses/ phd-seminar/S16/S16-paper.pd (accessed 2008)

Navarro, Gonzalo. 2003. Current Challenges in Textual Databases. Presented at the 4th Mexican International Conference on Computer Science Computer Society. http://www.infor.uva.es/Investigacion/ Descargas/navarro.Sep04.pdf (accessed 2008)

Parlante, Nick. 2001. *Linked list Basics*. http://cslibrary.stanford.edu/103/LinkedListBasics.pdf (accessed 2008)

Puglisi, Simon J., William F. Smyth, and Andrew Turpin. 2005. suffix Arrays: What Are They Good For? In *ACM International conference proceeding series*; Vol. 170. Proceedings of the Seventeenth Australasian Database Conference (ADC2006), 49, (pp. 17-18). Hobart, Australia: Australian Computer Society, Inc. http://www.crpit.com/confpapers/CRPITV49Puglisi.pdf (accessed 2008)

Sadakane, Kunihiko, and Tetsuo Shibuya. 2001. Indexing huge genome sequences for solving various problems. In H. Matsuda, et al. (eds), Proceedings of the 12th Genome Informatics (GIW01) 2001 (Universal Academy Press, 2001) 175-183. Proceedings of the 12th Genome Informatics (GIW01), (pp. 175-183): Universal Academy Press. http://www-math.mit.edu/~lippert/18.417/papers/indexing-huge-genome-sequences.pdf (accessed 2008)

Schurmann, Klaus-Bernd, and Jens Stoye. 2005. An Incomplex Algorithm for Fast Suffix Array Construction. In *Proceedings* of the 7th Workshop on Algorithm Engineering and Experiments, http://www.siam.org/meetings/alenex05/papers/07kschuermann.pdf (accessed 2008)

Seki, Kazuhiro. 2005. Text Retrieval and Analysis Suffix Trees and Suffix Arrays. http://www.lair.indiana.edu/courses/textir/l590-9.ppt (accessed 2008)

Skiena, Steven. 2008. The Stony Brook Algorithm Repository, Graph Data Structures. http://www.cs.sunysb.edu/~algorith/ (accessed 2008)

Stoye, Jens. 2006. Suffix trees, Affix trees, and some of their applications. Bielefeld University, Jens Stoye's Recent Talks, Birkenfeld, May 17. http://www.techfak.uni-bielefeld.de/ ~stoye/talks/20060517birkenfeld.pdf (accessed 2008)

Sung, Wing-Kin. 2005. Suffix Tree and Suffix Array, Lecture 4 from Combinatorial methods in bioinformatics. http://www.comp.nus.edu.sg/~ksung/ cs5238/note/Lect4-suffixtree_2005.pdf (accessed 2008)

عادله اسعدی شالی

متولد سال ۱۳۶۰، دارای مدرک کارشناسی ارشد در رشته علم اطلاعات و دانش‌شناسی از دانشگاه تهران است. ایشان هم‌اکنون دانشجوی دکتری علم اطلاعات و دانش‌شناسی دانشگاه شیراز است.

ارتباطات و اطلاعات، رفتار اطلاع‌یابی، ذخیره و بازیابی اطلاعات از جمله علایق پژوهشی وی است.