بهبود کارایی اندیس گذاری باروز-ویلر برای هم‌ترازسازی توالی‌های خوانش کوتاه با انتخاب زیرمجموعه‌ای از ماتریس پسوند

ربابه شریفی و اسدا... شاه بهرامی

گروه مهندسی کامپیوتر، دانشکده‌ی مهندسی، دانشگاه گیلان، sharifi_r14@msc.guilan.ac.ir

shahbahrami@guilan.ac.ir

چکیده

هم‌ترازسازی توالی‌ها یکی از وظایف مهم در بیوانفورماتیک است. الگوریتم‌های هم‌ترازسازی توالی‌ها در دو دسته‌ی کلی مبتنی بر برنامه نویسی پویا و الگوریتم‌های ابتکاری قرار می‌گیرند. در الگوریتم‌های نوع دوم، اندیس گذاری ژنوم‌ها یک مرحله‌ی پیش نیاز مهم است. تبدیل باروز-ویلر یک روش اندیس گذاری پرکاربرد است که علاوه بر مصرف حافظه‌ی کم، ساختار مناسبی برای جستجوی سریع و دقیق در توالی‌ها فراهم می‌کند. این اندیس در سه مرحله ساخته می‌شود؛ ساختن ماتریس پسوند، مرتب سازی پسوندها و ساختن داده‌های کمکی مربوط به اندیس. بررسی‌ها نشان می‌دهد که مرحله‌ی مرتب سازی پسوندها دارای بیشترین زمان اجرا است بطوریکه برای یک توالی به طول ۲۵۶۰۰ نماد، بیش از ۳ ساعت طول می‌کشد. در این مقاله یک روش برای بهبود زمان اندیس گذاری باروز-ویلر با استفاده از تغییری کوچک در مرتب سازی ماتریس پسوندها معرفی شده که بر اساس ویژگی‌های الگوریتم جستجوی دقیق عقبگرد پیشنهاد شده است. این الگوریتم جستجو یکی از الگوریتم‌های ابزار هم‌ترازسازی باروز-ویلر است که برای جستجوی توالی‌های خوانش کوتاه تولید شده توسط فناوری‌های تعیین توالی جدید (حداکثر ۱۰۰ نماد)، در ژنوم‌ها به کار می‌رود. ایده‌ی اصلی، کاهش اندازه‌ی مسئله‌ی مرتب سازی با انتخاب پیشوندی از تمام سطرهای ماتریس پسوند بر اساس نیازهای الگوریتم جستجوی دقیق عقبگرد است؛ به طوریکه، در درستی الگوریتم جستجو تأثیر منفی نداشته باشد. نتایج حاصل از اجرای الگوریتم نشان می‌دهد که با انتخاب طول ۱۰۰ برای پیشوندها، زمان اندیس گذاری یک توالی ۲۵۶۰۰ نمادی از حدود ۳/۴ ساعت به ۳/۵ دقیقه کاهش می‌یابد. با توجه به اینکه فناوری‌های تعیین توالی جدید، خوانش‌هایی با طول کوتاه تولید می‌کنند، می‌توان با انتخاب طول پیشوند متناسب با این فناوری‌ها، روش پیشنهادی را بدون از دست دادن درستی الگوریتم جستجو به کار برد.

واژه‌های کلیدی

اندیس گذاری ژنوم، تبدیل باروز-ویلر، بهبود کارایی اندیس گذاری، جستجوی دقیق توالی، هم‌ترازسازی توالی.

۱- مقدمه

هم‌ترازسازی توالی‌های بیولوژیکی به معنای مقایسه و هم‌ردیف کردن این توالی‌ها به گونه‌ای است که نواحی بسیار مشابه آنها در یک راستا قرار گیرند. این فرایند از وظایف مهم در بیوانفورماتیک است و دارای کاربردهای متعددی در زمینه‌های مختلف است. نخستین کشف موفق دانشمندان در زمینه‌ی مقایسه‌ی توالی‌ها، یافتن رابطه‌ی بین ژن ایجاد کننده‌ی سرطان و ژن مؤثر در رشد طبیعی بود [۱]. به طور کلی برای هم‌ترازسازی توالی‌ها دو نوع رویکرد وجود دارد. الگوریتم‌های نوع اول بر اساس برنامه نویسی پویا عمل می‌کنند و الگوریتم‌های نوع دوم یا روش‌های ابتکاری، مبتنی بر جستجوی اندیس هستند. روش‌های نوع دوم در مرحله‌ی نخست سعی در یافتن نواحی یکسان یا بسیار مشابه بین توالی‌ها دارند. سپس این نواحی با روش‌های پیچیده تر بسط داده می‌شوند تا هم‌ترازی نهایی به دست آید. اندیس‌ها ساختارهایی هستند که از روی توالی‌های ژنوم ساخته شده و امکان جستجوی سریع را برای یافتن یک زیرتوالی کوچک در پایگاه‌های بیولوژیکی که حجم بسیار بالایی دارند، فراهم می‌کنند. چهار نوع اندیس تاکنون در این زمینه به کار رفته‌اند که عبارتند از جداول درهم سازی، درخت پسوندی، آرایه‌ی پسوند و تبدیل باروز-ویلر. در این بین تبدیل باروز-ویلر که مبتنی بر آرایه‌ی پسوندی است، نسبت به اندیس‌های دیگر از نظر مصرف حافظه کاراتر بوده و با ساختار ویژه‌ای که ایجاد می‌کند، امکان جستجوی سریع را برای یک زیرتوالی در یک توالی بلند فراهم می‌کند [۲]. به عنوان نمونه الگوریتم جستجوی دقیق عقبگرد که در [۲] به عنوان یک الگوریتم سریع معرفی شده است، از این اندیس استفاده می‌کند. اما بر اساس پژوهش‌های صورت گرفته نظیر [۳]، ساختن این اندیس ساعت‌ها طول می‌کشد. در الگوریتم‌های ابتکاری، اندیس گذاری یک پیش نیاز مهم است. بنابراین فراهم کردن سریع‌تر اندیس برای به کارگیری در این الگوریتم‌ها یک نیاز مهم محسوب می‌شود.

با توجه به تعداد زیاد توالی‌های بیولوژیکی و اهمیت اندیس گذاری آنها تاکنون تلاش‌های متعددی برای بهبود زمان روش‌های مختلف اندیس

گذاری صورت گرفته است. در [۳-۹] نمونه‌هایی از این روش‌ها نشان داده شده‌اند. اما در این روش‌ها برای بهینه سازی به ویژگی‌های الگوریتم جستجوی دقیق عقبگرد که برای جستجوی زیرتوالی‌های با طول کم استفاده می‌شود، توجه نشده است.

هدف این مقاله بررسی زمان اندیس گذاری باروز-ویلر برای توالی‌های با طول‌های مختلف و بهبود آن با استفاده از ویژگی‌های الگوریتم جستجوی دقیق عقبگرد برای جستجوی توالی‌های کوتاه است. روش پیشنهادی بر اساس انتخاب پیشوندی با طول مناسب از تمامی پسوندهای یک توالی و مرتب سازی پسوندها بر اساس این پیشوندها عمل می‌کند. انتخاب طول پیشوندها نیازمند بررسی طول توالی‌های خوانشی [۱] است که توسط فناوری‌های تعیین توالی نسل بعد تولید می‌شوند. با انتخاب طول ۱۰۰ برای پیشوندها، میزان بهبود به دست آمده در زمان اندیس گذاری برای یک توالی با طول ۲۵۶۰۰ نماد، ۵۷ است. بطوریکه زمان از ۳/۴ ساعت به حدود ۳/۵ دقیقه کاهش می‌یابد.

این مقاله به صورت زیر سازمان دهی می‌شود. در بخش دوم اندیس گذاری باروز-ویلر و نحوه‌ی استفاده از آن در الگوریتم جستجوی دقیق عقبگرد بررسی می‌شود. بخش سوم به شرح روش پیشنهادی اختصاص داشته و در بخش چهارم نتایج حاصل از پیاده سازی ارائه خواهد شد. در بخش پنجم کارهای مرتبط به اختصار معرفی می‌شود و بخش پایانی شامل نتیجه گیری مقاله است.

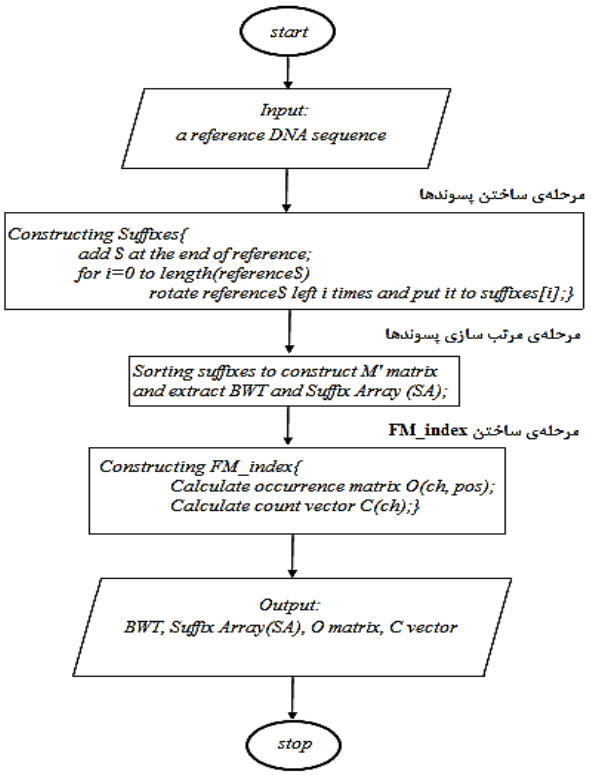
۲- اندیس گذاری باروز-ویلر

۲-۱- محاسبه‌ی تبدیل باروز-ویلر

تبدیل باروز-ویلر[۲] که در ابتدا به عنوان یکی از مراحل فشرده سازی متن معرفی شد [۱۰] جایگشتی خاص از نمادهای موجود در یک توالی و یک نماد $ است. در نتیجه طول آن تنها یک واحد بیشتر از طول توالی اولیه است. ویژگی‌های این تبدیل موجب شده است که از آن در بسیاری از برنامه‌های هم‌ترازسازی که مبتنی بر جستجوی اندیس هستند به عنوان اندیس استفاده شود [۲]. مراحل ساخت این اندیس در شکل ۱ نشان داده شده است. این مراحل عبارتند از ساخت ماتریس پسوند، مرتب سازی پسوندها و ایجاد اندیس FM با استفاده از اندیس باروز-ویلر.

الفبای توالی‌های بیولوژیکی شامل چهار نماد A، C، G و T است. برای ساخت اندیس باروز-ویلر برای یک توالی، ابتدا به آخر آن یک علامت $\$$ اضافه می‌شود. سپس این توالی به تعداد نمادهای توالی به سمت چپ چرخش داده شده و با هر چرخش یک پسوند از آن ساخته می‌شود. هر پسوند برای یک توالی یک زیرتوالی از آن است که از یک مکان در توالی آغاز شده و تا انتهای آن ادامه می‌یابد. پس از ساختن ماتریس پسوند (M) با چرخش‌های متوالی، این پسوندها باید مرتب سازی شوند. در مرتب سازی، علامت $\$$ از سایر نمادها کوچکتر درنظر گرفته می‌شود. پس از مرتب سازی ماتریس پسوند، ستون آخر آن برابر است با تبدیل باروز-ویلر (BWT) برای توالی مورد نظر.

آرایه‌ی پسوندی (SA) که یکی از شیوه‌های اندیس گذاری توالی‌ها است، آرایه‌ای از تمام پسوندهای مرتب شده‌ی یک توالی است. اما در اندیس باروز-ویلر، آرایه‌ی پسوند، شامل خود پسوندها نیست؛ بلکه محل هر یک از پسوندهای مرتب را در توالی مورد نظر در بر دارد. پس از ساخت اندیس باروز-ویلر ساختمان داده‌ی کمکی آن که موسوم به اندیس FM [۱۱] است ساخته می‌شود. این ساختمان داده‌ی کمکی شامل ماتریس رخداد (O) و بردار شمارش (C) است که در الگوریتم جستجوی دقیق برای محاسبه‌ی محل‌های رخداد یک زیرتوالی خاص در یک توالی مرجع استفاده می‌شوند. هر سطر از ماتریس O و هر عنصر از بردار چهار عضوی C متناظر با یکی از نمادهای الفبای توالی‌ها است. هر عنصر $O(a, i)$ برابر است با تعداد دفعاتی که نماد a در اندیس باروز-ویلر قبل از مکان i تکرار شده است. هر عنصر $C(a)$ نیز برابر است با تعداد نمادهایی از اندیس باروز-ویلر که از نظر الفبایی از a کوچک‌ترند (بدون در نظر گرفتن $\$$).



شکل ۱: نمودار جریان اجرای اندیس گذاری باروز-ویلر.

۲-۲- استفاده از اندیس باروز-ویلر در الگوریتم جستجوی دقیق عقبگرد

جستجوی دقیق عقبگرد الگوریتمی است که برای یک زیرتوالی مانند W تمامی رخدادهای دقیق آن را در صورت وجود در توالی مرجعی مانند X می‌یابد. کاربرد این الگوریتم هنگامی است که یافتن یک زیرتوالی بیولوژیکی (مانند یک پروتئین شناخته شده) در یک توالی ژنوم مد نظر

---

[۱] Read

[۲] Burrows-Wheeler Transform (BWT)

باشد. نوع تعمیم یافته‌ی آن، الگوریتم جستجوی نادقیق است که در آن هنگام جستجوی یک زیرتوالی، تعداد محدودی عدم تطابق نیز مجاز است. الگوریتم جستجوی دقیق عقبگرد که در ابزار هم‌ترازسازی باروز-ویلر (*BWA*)[۳] پیاده سازی شده است، می‌تواند تمامی رخدادهای یک زیرتوالی مانند W را در یک توالی مرجع در زمانی متناسب با طول توالی W بیابد. با توجه به اینکه طول W برای این الگوریتم هم‌ترازساز باروز-ویلر کوتاه (حداکثر ۱۰۰ نماد [۲]) است، این الگوریتم بسیار سریع اجرا می‌شود.

خروجی الگوریتم جستجوی دقیق برای زیرتوالی W یک بازه به صورت $[k, l]$ موسوم به بازه‌ی پسوند است. بازه‌ی پسوند، بازه‌ای شامل اندیس‌هایی از SA است که با مراجعه به آن اندیس‌ها و استخراج مقادیر موجود در آنها، محل رخدادهای W در توالی مرجع X به دست می‌آید. k، اولین و l، آخرین اندیس از SA است که در آنها به یکی از محل‌های رخداد W در X اشاره شده است و اندیس بقیه‌ی رخدادها در بین این بازه قرار خواهد گرفت. زیرا رخدادهای یک زیرتوالی در اندیس باروز-ویلر به نحو مناسبی مرتب سازی و گروهبندی شده‌اند. عدم وجود W در X منجر به تولید بازه‌ی پسوند تهی خواهد شد. این جستجو عقبگرد انجام شده و با شروع از آخرین نماد W و به کمک اندیس FM بازه‌ی پسوند با استفاده از روابط بازگشتی (۱) و (۲) محاسبه می‌شود.

$$k(aW) = C(a) + O(a, k(W)) + 1 \quad (۱)$$

$$l(aW) = C(a) + O(a, l(W) + 1) \quad (۲)$$

مقدار اولیه برای بازه‌ی پسوند برابر با $[0, n]$ است که در آن n طول توالی مرجع است. این بازه به این معنی است که یک زیرتوالی تهی، در سراسر توالی وجود دارد. سپس آخرین نماد W بررسی شده و بازه‌ی پسوند برای آن با استفاده از روابط (۱) و (۲) به دست می‌آید (یعنی پسوندهایی که با آخرین نماد توالی شروع شده‌اند). سپس دو نماد آخر با استفاده از این روابط و مقدار به دست آمده در مرحله‌ی قبل برای k و l، جستجو شده و k و l جدیدی محاسبه می‌شوند. در هر مرحله، بازه‌ی پسوند کوچک‌تر می‌شود. این روند ادامه می‌یابد تا اینکه بازه‌ی پسوند برای تمام توالی W به دست آید. در صورتی که توالی aW در X موجود باشد، همواره نامساوی (۳) برقرار است و هرجا این نامساوی نقض شد، الگوریتم پایان می‌یابد.

$$k(aW) \le l(aW) \quad (۳)$$

روند اندیس گذاری باروز-ویلر و به کارگیری آن در جستجوی دقیق عقبگرد در ادامه برای توالی $X=CAGGTAG\$$ نشان داده شده است.

$$M = \begin{bmatrix} CAGGTAG\$ \\ AGGTAG\$C \\ GGTAG\$CA \\ GTAG\$CAG \\ TAG\$CAGG \\ AG\$CAGGT \\ G\$CAGGTA \\ \$CAGGTAG \end{bmatrix} \quad M' = \begin{bmatrix} \$CAGGTAG \\ AG\$CAGGT \\ AGGTAG\$C \\ CAGGTAG\$ \\ G\$CAGGTA \\ GGTAG\$CA \\ GTAG\$CAG \\ TAG\$CAGG \end{bmatrix} \quad BWT = (GTC\$AAGG)$$

$$SA = (7, 5, 1, 0, 6, 2, 3, 4)$$

$$C = (0, 2, 3, 6) \qquad O = \begin{bmatrix} 0\,0\,0\,0\,0\,1\,2\,2 \\ 0\,0\,0\,1\,1\,1\,1\,1 \\ 0\,1\,1\,1\,1\,1\,2\,3 \\ 0\,0\,1\,1\,1\,1\,1\,1 \end{bmatrix}$$

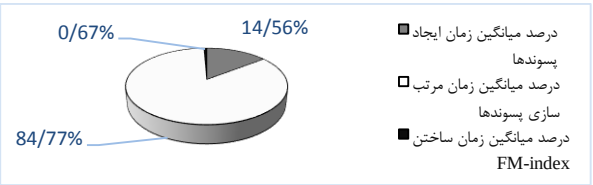
جستجوی زیرتوالی $W=AG$ در توالی مرجع X در ادامه نشان داده شده است. ناحیه‌هایی از ماتریس پسوند که در هر مرحله به دست می‌آیند، به صورت رنگی مشخص شده است.

$$M' = \begin{bmatrix} \$CAGGTAG \\ AG\$CAGGT \\ AGGTAG\$C \\ CAGGTAG\$ \\ \mathbf{G\$CAGGTA} \\ \mathbf{GGTAG\$CA} \\ \mathbf{GTAG\$CAG} \\ TAG\$CAGG \end{bmatrix} \; k = 4, l = 6 \qquad M' = \begin{bmatrix} \$CAGGTAG \\ \mathbf{AG\$CAGGT} \\ \mathbf{AGGTAG\$C} \\ CAGGTAG\$ \\ G\$CAGGTA \\ GGTAG\$CA \\ GTAG\$CAG \\ TAG\$CAGG \end{bmatrix} \; k = 1, l = 2$$

بازه‌ی پسوند به دست آمده برابر با $[1, 2]$ است. با مراجعه به اندیس ۱ و ۲ در SA، به ترتیب مقادیر ۵ و ۱ به دست می‌آید که این اعداد، اندیس محل شروع W در توالی X هستند.

۳- روش پیشنهادی برای بهبود زمان اندیس گذاری

برای بهینه سازی اندیس گذاری، ابتدا باید دانست که کدام مرحله‌ی آن بیشترین زمان اجرا را صرف می‌کند. به این منظور برای توالی‌هایی با طول‌های مختلف، زمان اندیس گذاری به تفکیک مراحل مختلف آن اندازه گیری شده است. نتایج نشان داده شده در شکل ۲ حاکی از آن است که مرحله‌ی مرتب سازی ماتریس پسوند نسبت به دو مرحله‌ی دیگر ساخت اندیس بیشترین زمان اجرا را به خود اختصاص می‌دهد. سهم مرتب سازی پسوندها در مصرف زمان سپری شده برای توالی‌هایی با طول ۴۰۰، ۸۰۰، ۱۶۰۰، ۳۲۰۰، ۶۴۰۰، ۱۲۸۰۰ و ۲۵۶۰۰ نماد اندازه گیری شده و میانگین مقادیر به دست آمده حدوداً ۸۵ درصد است.



شکل ۲: زمان اجرای هر یک از مراحل اندیس گذاری باروز-ویلر.

رویکرد استفاده شده برای بهبود زمان مرتب سازی، با بررسی الگوریتم جستجوی دقیق عقبگرد برای توالی‌هایی که توسط فناوری‌های تعیین توالی نسل بعد تولید می‌شوند، انتخاب شده است. با پیشرفت فناوری‌های تعیین توالی، ابزارهایی در این زمینه توسعه یافته‌اند که قادرند در یک بار اجرا میلیاردها توالی با طول میانگین ۱۰۰ تولید کنند که به آنها خوانش‌های کوتاه گفته می‌شود [۱۲]. طول خوانش‌های تولیدی برخی فناوری‌های تعیین توالی در جدول ۱ نشان داده شده است [۱۳، ۱۴].

ابزار BWA شامل سه الگوریتم BWA-MEM [۱۵]، BWA-SW [۴] [۱۶] و BWA-backtrack [۲] است. دو الگوریتم اول برای هم‌ترازسازی

[۳] Burrows-Wheeler Aligner (BWA)

[۴] BWA-Maximal Exact Match

[۵] BWA-Smith Waterman

توالی‌های طولانی تولید شده توسط ماشین‌های توالی‌ساز از ۷۰ تا ۱ میلیون جفت باز به کار می‌روند. الگوریتم سوم برای هم‌ترازسازی توالی‌های کوتاه تولیدی توالی ساز ایلومینا [۶] به طول حداکثر ۱۰۰ جفت باز طراحی شده و الگوریتم جستجوی دقیق عقبگرد یکی از توابع پرکاربرد آن است.

جدول ۱: فناوری‌های تعیین توالی مختلف و طول خوانش آنها [۱۳، ۱۴].

فناوری تعیین توالی	طول خوانش تولیدی
روش SMRT (PacBio)	میانگین ۱۵۰۰ جفت باز
روش ختم زنجیره سازی سانگر	۴۰۰ تا ۹۰۰ جفت باز
ایلومینا (sequencing by synthesis)	۵۰ تا ۳۰۰ جفت باز
ایلومینا (MiSeq, GAIIx, Hiseq 2000)	حداکثر ۱۵۰ جفت باز
روش حرارتی (Pyrosequencing (454))	۷۰۰ جفت باز
روش نیمه هادی Ion Torrent	حداکثر ۴۰۰ جفت باز

بررسی الگوریتم جستجوی دقیق عقبگرد، نشان می‌دهد که این الگوریتم برای جستجوی توالی‌های کوتاه با استفاده از اندیس باروز-ویلر، نیازی به اندیس کامل ندارد. به عبارت دیگر برای ساخت اندیس در مرحله‌ی مرتب سازی پسوندها، چنانچه پسوندها بر اساس m کاراکتر ابتدایی خود مرتب باشند، آنگاه اندیس BWT ساخته شده، با اینکه اندیس کامل و درستی نیست، اما می‌تواند در جستجوی توالی‌هایی که طولشان حداکثر برابر با m است مورد استفاده قرار گیرد و در عین حال به درستی جستجو خللی وارد نکند. زیرا طبق این الگوریتم برای توالی‌های مورد جستجو به طول m هر یک از پسوندهای توالی مرجع تنها تا m نماد مورد بررسی قرار می‌گیرند و نیازی به بررسی بقیه‌ی نمادهای موجود در پسوند نیست. در ادامه برای مثال نشان داده شده در بخش قبل، اندیس BWT با استفاده از مرتب سازی پسوندها تنها بر اساس دو نماد ابتدایی ساخته شده و مجدداً جستجوی توالی $W=AG$ با استفاده از اندیس جدید اجرا می‌شود.

$$M = \begin{bmatrix} CAGGTAG\$ \\ AGGTAG\$C \\ GGTAG\$CA \\ GTAG\$CAG \\ TAG\$CAGG \\ AG\$CAGGT \\ G\$CAGGTA \\ \$CAGGTAG \end{bmatrix} \qquad M' = \begin{bmatrix} \$CAGGTAG \\ \mathbf{AGGTAG\$C} \\ \mathbf{AG\$CAGGT} \\ CAGGTAG\$ \\ G\$CAGGTA \\ GGTAG\$CA \\ GTAG\$CAG \\ TAG\$CAGG \end{bmatrix}$$

$BWT = (G\underline{\mathbf{CT}}\$AAGG), \quad SA = (7, \underline{\mathbf{1, 5}}, 0, 6, 2, 3, 4)$

عناصری از SA و BWT که با خط زیر مشخص شده‌اند، تفاوت بین اندیس جدید و اندیس BWT ساخته شده در بخش قبل را نشان می‌دهند. این تفاوت‌ها ناشی از آن است که در ماتریس M'، جای دو پسوند ۱ و ۵ با هم عوض نشده است. در صورتیکه برای مرتب سازی کامل پسوندها این دو با هم تعویض می‌شوند. ماتریس رخداد نیز که در ادامه نشان داده شده، با اندیس قبلی در سطرهایی که به رنگ خاکستری مشخص شده، تفاوت دارد.

$$C = (0, 2, 3, 6), \qquad O = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 2 & 2 & 2 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 3 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

مزیت مرتب سازی جزیی پسوندها نسبت به مرتب سازی کامل، کاهش تعداد جا به جایی‌هاست. برای توالی‌هایی مانند مثال فوق که طولشان کوتاه است، این مسئله چالشی ایجاد نمی‌کند. اما زمانی که توالی‌های ژنوم طول زیاد دارند، ساختن اندیس و به ویژه مرتب سازی پسوندهای آن زمان زیادی را صرف می‌کند. در ادامه نشان داده شده است که جستجوی توالی $W=AG$ با استفاده از اندیس جدید نیز بازه‌ی $[1, 2]$ را نتیجه می‌دهد. می‌توان نشان داد که برای هر توالی به طول حداکثر ۲ می‌توان با استفاده از اندیس جدید بازه‌ی پسوند صحیح را به دست آورد.

$$M' = \begin{bmatrix} \$CAGGTAG \\ AGGTAG\$C \\ AG\$CAGGT \\ CAGGTAG\$ \\ \mathbf{G\$CAGGTA} \\ \mathbf{GGTAG\$CA} \\ \mathbf{GTAG\$CAG} \\ TAG\$CAGG \end{bmatrix} \quad k = 4, l = 6 \qquad M' = \begin{bmatrix} \$CAGGTAG \\ \mathbf{AGGTAG\$C} \\ \mathbf{AG\$CAGGT} \\ CAGGTAG\$ \\ G\$CAGGTA \\ GGTAG\$CA \\ GTAG\$CAG \\ TAG\$CAGG \end{bmatrix} \quad k = 1, l = 2$$

۴- پیاده سازی و نتایج

در این بخش نتایج به دست آمده از بهبود اندیس گذاری با استفاده از مرتب سازی جزیی پسوندها نشان داده شده است.

۴-۱- سیستم مورد استفاده

سیستمی که دو نسخه‌ی اندیس گذاری BWT روی آن پیاده سازی و اجرا شده‌اند دارای پردازنده‌ی Intel(R) Core(TM) i5-3337U CPU @ 1.80 GHz، ۶ گیگابایت RAM و سیستم عامل ویندوز ۸/۱ است.

۴-۲- داده‌ها و محیط پیاده سازی

در این مقاله دو نسخه کد برای اندیس گذاری BWT پیاده سازی شده‌اند. نسخه‌ی اول با نام BWT، اندیس گذاری با استفاده از مرتب سازی کامل پسوندهاست. نسخه‌ی دوم که در آن از مرتب سازی پیشوند m کاراکتری از پسوندها استفاده شده، به صورت BWT_P (*BWT with Partial sorting*) نام گذاری شده است. برای آزمایش بهبود حاصل شده با استفاده از الگوریتم BWT_P از توالی‌های متشکل از چهار نماد A، C، G و T استفاده شده است که در آنها ترتیب قرارگیری و ترکیب این نمادها به صورت تصادفی است. برای مشاهده‌ی تأثیر طول توالی بر زمان اجرای الگوریتم‌ها طول‌های ۴۰۰، ۱۶۰۰، ۳۲۰۰، ۶۴۰۰، ۱۲۸۰۰ و ۲۵۶۰۰ انتخاب شده‌اند. کدهای نوشته شده برای هر دو الگوریتم BWT و BWT_P به زبان $C++$ و در محیط ویژوال استودیو ۲۰۱۰ پیاده سازی و اجرا شده‌اند.

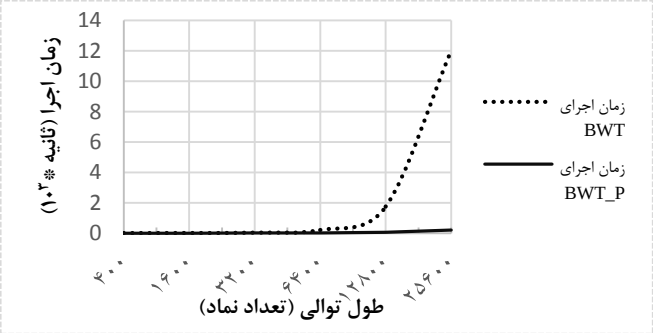
۴-۳- نتایج به دست آمده

با توجه به نمودارهای شکل ۳ و شکل ۴ زمان اجرای الگوریتم اندیس گذاری BWT برای مجموعه‌ی داده‌ی انتخابی به میزان ۱/۵ تا ۵۷ برابر

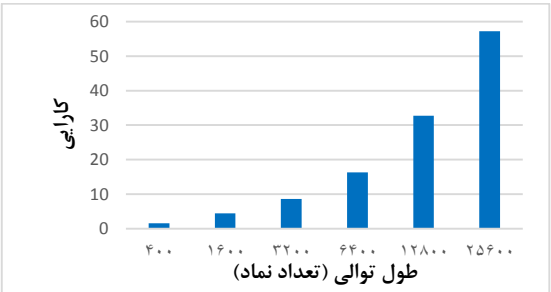
---

[۶] Illumina

بهبود یافته است. دلیل اینکه زمان اندیس گذاری برای توالی‌ها با افزایش طول توالی افزایش می‌یابد، این است که هرچه طول توالی‌ها بیشتر می‌شود، تعداد و طول پسوندهای آن توالی نیز بیشتر خواهد شد. اگر طول توالی n باشد، اجرای الگوریتم مرتب سازی n پسوند با طول n از مرتبه‌ی $O(n^2)$ خواهد بود. می‌توان با به کار گیری روش‌هایی مثل آنچه در [۴] پیشنهاد شده است، از الگوریتم‌های سریع‌تری برای مرتب سازی استفاده کرد و زمان را به $O(n)$ تقلیل داد. با استفاده از روش پیشنهادی نیز می‌توان به زمان $O(n)$ دست یافت. به این صورت که با ثابت نگه داشتن طول پسوندها به یک مقدار ثابت و مشخص مانند m، تعداد مقایسه‌ها برای یک توالی به طول n برابر است با $n \times m$ که در این صورت با افزایش n و ثابت بودن m، می‌توان از m در مقابل n چشم پوشی کرد و بنابراین مرتبه‌ی اجرای الگوریتم $O(n)$ خواهد بود. نمودار شکل ۳ زمان اجرای الگوریتم BWT_P را در مقایسه با BWT نشان می‌دهد. در شکل ۴ نیز میزان بهبود الگوریتم BWT_P نسبت به BWT برای توالی‌های به طول ۴۰۰ تا ۲۵۶۰۰ نماد که پیش‌تر بیان شد، و با ثابت نگه داشتن طول توالی‌های خوانش به مقدار ۱۰۰ نماد، نشان داده شده است.

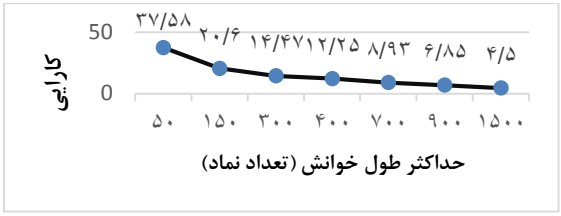


شکل ۳: مقایسه‌ی زمان اجرای الگوریتم *BWT* و *BWT_P* برای مجموعه داده‌ی انتخابی.



شکل ۴: نمودار بهبود کارایی الگوریتم *BWT_P* نسبت به *BWT* برای مجموعه داده‌ی انتخابی و *m=100*.

برای بررسی اینکه طول توالی خوانش تا چه حد بر میزان بهبود تأثیر دارد، توالی به طول ۱۲۸۰۰ برای اندیس گذاری انتخاب شده و طول خوانش‌ها از ۵۰ تا ۱۵۰۰ تغییر یافته است. هفت مقدار در این محدوده با بررسی طول توالی‌های خوانش تولیدی توسط فناوری‌های تعیین توالی نسل بعد که در جدول ۱ نشان داده شده است، انتخاب شده‌اند. طول ۱۵۰۰ برای بررسی میزان بهبود روش پیشنهادی برای خوانش‌های بلند در نظر گرفته و آزمایش شده است. نتایج در شکل ۵ نشان داده شده‌اند.



شکل ۵: کارایی اندیس گذاری توالی ۱۲۸۰۰ کاراکتری برای طول‌های خوانش مختلف.

با توجه به شکل ۵ می‌توان نتیجه گرفت که این روش می‌تواند برای هم‌ترازسازهایی که با خوانش‌های کوتاه سروکار دارند، نظیر آنهایی که هم‌ترازسازی را برای خوانش‌های کوتاه تولیدی ایلومینا انجام می‌دهند میزان کارایی بالایی داشته باشد. اما برای هم‌ترازسازی خوانش‌های بلند میزان بهبود کمتر است. اما در صورتی که طول توالی‌های مرجع بلند باشد، آنگاه نسبت طول توالی مرجع به طول خوانش، حتی برای خوانش‌های ۱۵۰۰ نمادی بیشتر شده و مجدداً میزان بهبود کارایی افزایش می‌یابد.

۵- کارهای مرتبط

در زمینه‌ی بهبود اندیس گذاری توالی‌ها و نیز بهینه سازی هم‌ترازسازی با استفاده از بهبود اندیس‌ها تاکنون کارهای متعددی صورت گرفته است. کاری که در [۵] انجام شده، استفاده از یک درخت پسوندی توسعه یافته به منظور بهبود هم‌ترازسازی توالی‌ها در ابزار هم‌ترازسازی بلَست است. اطلاعاتی راجع به هر یک از پسوندهایی که در درخت نشان داده شده‌اند درباره‌ی دفعات تکرار و طول آنها به درخت اضافه می‌شود که در محدود کردن فضای جستجوی بلست تأثیر مطلوبی دارد و سبب بهبود سرعت بلست به میزان دو برابر می‌شود. در [۳] از چارچوب MapReduce برای ساختن آرایه‌ی پسوند و BWT به صورت موازی استفاده شده است. MapReduce، چارچوبی در رایانش ابری است که متشکل از دو مرحله‌ی Map و Reduce است. در مرحله‌ی اول زوج‌هایی به صورت <key,value> از یک پایگاه داده‌ی پردازشی بزرگ استخراج می‌شود. سپس این زوج‌ها بر اساس key به هر یک از گره‌های مختلف یک خوشه از کامپیوترها منتسب شده و گره‌ها به صورت موازی عملیات خاصی روی مؤلفه‌ی دوم زوج‌های دریافتی خود انجام می‌دهند. در زمینه‌ی ایجاد آرایه‌ی پسوند و اندیس گذاری BWT تکنیکی که در [۳] معرفی شده، با استفاده از این چارچوب پسوندها را بر اساس پیشوندهایی از پسوندها بین گره‌ها تقسیم می‌کند. به این صورت که هر گره پسوندهایی را مرتب سازی می‌کند که دارای پیشوند k کاراکتری برابری هستند. اندیس BWT با داشتن آرایه‌ی پسوندی قابل محاسبه است [۲، ۱۲]. با این تکنیک موازی سازی میزان زمان اندیس گذاری BWT برای ۵ توالی ژنوم با طول‌های ۳۷۰ میلیون تا ۳/۱ میلیارد، از چندین ساعت به چند دقیقه کاهش یافته است. حداکثر بهبود به دست آمده برای آرایه‌ی پسوندی نسبت به ابزار Vmatch برابر ۹/۱ و برای BWT نسبت به ابزار Bowtie برابر ۱۴/۹۳ است. این چارچوب همچنین در [۶] برای ساختن درخت پسوندی به صورت موازی استفاده شده است. ساخت درخت در صورتی که کل آن در

حافظه‌ی اصلی موجود باشد، در زمان خطی میسر است. اما برای توالی‌های بزرگ این امر با چالش کمبود حافظه‌ی اصلی مواجه می‌شود. در [۶] برای افزایش مقیاس پذیری ساختن درخت پسوندی در زمان خطی از MapReduce و ساختار حافظه‌ی توزیع شده‌ی آن استفاده شده است. درخت پسوندی به صورت عمودی در یک مرحله‌ی پارتیشن بندی به پیشوندهایش تقسیم می‌شود. دو شیوه‌ی پارتیشن بندی پیشوند طول ثابت و طول متغیر استفاده می‌شود. پس از تقسیم، هر گرهِ کاری با استفاده از پیشوندی که به آن منتسب می‌شود، ساختن زیردرختی را انجام می‌دهد که شامل پسوندهای آغاز شده با آن پیشوند هستند. در [۷] برای ساخت درخت پسوندی یک الگوریتم موازی به نام PCF معرفی شده است که در آن مرحله‌ی پیش پردازش ساخت درخت، موازی شده و زیردرخت‌ها در آن با آرایه‌های پسوندی جایگزین شده‌اند. در [۴] دو الگوریتم برای ساخت اندیس BWT معرفی شده‌اند که زمان اجرای آنها از مرتبه‌ی خطی است. این دو الگوریتم از قاعده‌ی تقسیم و غلبه و کاهش برای بهبود زمان اندیس گذاری استفاده می‌کنند. ایده‌ی کاهش به کار رفته در [۴] به این صورت است که تمام پسوندهای یک توالی در دو کلاس کلی به نام S-type و L-type دسته بندی شوند. هر پسوند S-type (L-type) پسوندی است که از پسوند بعدی خود در توالی کوچکتر (بزرگتر) باشد. یک قاعده‌ی تقسیم به این صورت است که یک توالی به دو زیرتوالی تقسیم می‌شود؛ زیرتوالی کاهش یافته و زیرتوالی باقیمانده. سپس آرایه‌ی پسوند برای زیرتوالی کاهش یافته به صورت بازگشتی به همین شیوه ساخته می‌شود. سپس با استفاده از آرایه‌ی پسوندی حاصل، برای قسمت باقیمانده با استفاده از الگوریتمی خاص آرایه‌ی پسوندی ساخته می‌شود و در نهایت این دو آرایه‌ی پسوندی با هم ادغام می‌شوند تا نتیجه‌ی نهایی به دست آید. شیوه‌ی دیگری که برای تقسیم و کاهش اندازه‌ی مسئله استفاده شده، تشخیص زیرتوالی‌هایی از نوع Leftmost S-type است. با این شیوه، اندازه‌ی مسئله‌ی کاهش یافته (توالی شکسته شده) متغیر خواهد بود. برای هر یک از این دو رویکرد الگوریتم متفاوتی به منظور مرتب سازی استفاده شده است. دو الگوریتم معرفی شده در [۴] در بین تمامی روش‌های خطی ساخت آرایه‌ی پسوند از نظر زمان و فضای مصرفی بهترین کارایی را دارد. در [۸] از سکوی واسط انتقال پیام (MPI) برای ساخت آرایه‌ی پسوندی به صورت موازی استفاده شده است. روش معرفی شده در آن پژوهش بسیار مقیاس پذیر است. روشی که در [۹] برای ساخت آرایه‌ی پسوندی استفاده شده، به کارگیری ترکیبی از CPU و سکوی موازی سازی GPU است. در [۹] الگوریتم prefix doubling که از روش‌های ساخت آرایه‌ی پسوندی است به صورت موازی روی GPU پیاده سازی شده و کارایی بالایی نسبت به حالت سریال این الگوریتم به دست آمده است.

۶- نتیجه گیری

با توجه به اهمیت اندیس گذاری توالی‌های ژنوم، در این مقاله یکی از روش‌های اندیس گذاری به نام تبدیل باروز-ویلر از نظر زمان اجرا مورد استفاده قرار گرفت. سپس روشی برای بهبود کارایی اندیس گذاری باروز-ویلر بر اساس ایده‌ی کاهش حجم مسئله با توجه به ویژگی‌های الگوریتم جستجوی دقیق عقبگرد و طول خوانش‌های فناوری‌های تعیین توالی جدید ارائه و نتایج آن نشان داده شد. این روش برای توالی‌های با طول‌های ۴۰۰ تا ۲۵۶۰۰، بهبودی به میزان ۱/۵ تا ۵۷ برابر دارد و زمان اندیس گذاری را برای توالی ۲۵۶۰۰ نمادی از ۳/۳ ساعت به ۳/۵ دقیقه کاهش می‌دهد.

مراجع

[1] P. Pevzner, Computational Molecular Biology: An Algorithmic Approach, MIT press, 2000.

[2] H. Li and R. Durbin, "Fast and Accurate Short Read Alignment with Burrows-Wheeler Transform," Bioinformatics, vol. 25, no. 14, 1754-1760, 2009.

[3] R. K. Menon, G. P. Bhat, and M. C. Schatz, "Rapid Parallel Genome Indexing with MapReduce," Proceedings of the second international workshop on MapReduce and its applications, ACM, pp. 51-58, 2011.

[4] G. Nong, S. Zhang, and W. Hong Chan, "Two Efficient Algorithms for Linear Time Suffix Array Construction," IEEE Transactions on Computers, vol. 60, no. 10 pp. 1471-1484, 2011.

[5] D. R. Singh and A. N. Arslan, "Using an Extended Suffix Tree to Speed-Up Sequence Alignment," IADIS International Conference Applied Computing, pp. 655-660, 2006.

[6] U. Chandra Satish, P. kondikoppa, S. J. Park, M. Patil, and R. Shah, "MapReduce based Parallel Suffix Tree Construction for Human Genome," IEEE International Conference on Parallel and Distributed Systems (ICPADS), pp. 664-670, 2014.

[7] M. Comin, and M. Farerras, "Parallel Continuous Flow: a parallel suffix tree construction tool for whole genomes," Journal of Computational Biology, vol. 21, no. 4, pp. 330-334, 2014.

[8] F. Kulla, and P. Sanders, "Scalable Parallel Suffix Array Construction," Parallel Computing, vol. 33, no. 9, 605-612, 2007.

[9] W. Sun, "Using GPU to Accelerate Suffix Array Construction," International Conference on Biomedical Engineering and Informatics, pp. 677-682, 2014.

[10] M. Burrows, and D. J. Wheeler, "A Block-Sorting Lossless Data Compression Algorithm," 1994.

[11] P. Ferragina, and G. Manzini, "Opportunistic Data Structures with Applications," Annual Symposium on Foundations of Computer Science, pp. 390-398, 2000.

[12] J. S. Torres, I. B. Espert, A. T. Dominguez, V. Hernendez, I. Medina, J. Terraga, and J. Dopazo, "Using GPUs for the Exact Alignment of Short-Read Genetic Sequences by Means of the Burrows-Wheeler Transform," Journal of Computational Biology and Bioinformatics, vol. 9, no. 4 pp. 1245-1256, 2012.

[13] M. A. Quail, and et al, "A Tale of Three Next Generation Sequencing Platforms: Comparison of Ion Torrent, Pacific Biosciences and Illumina MiSeq Sequencers," BMC Genomics, vol. 13, no. 1, pp. 1-13, 2012.

[14] L. Liu, and et al, "Comparison of Next-Generation Sequencing Systems," Journal of Biomedicine and Biotechnology, 2012.

[15] H. Li, "Aligning Sequence Reads, Clone Sequences and Assembly Contigs with BWA-MEM," rXiv preprint arXiv, 2013.

[16] H. Li and R. Durbin, "Fast and Accurate Long-Read Alignment with Burrows–Wheeler Transform," Bioinformatics, vol. 26, no. 5, pp. 589-595, 2010.