دانشگاه علوم کشاورزی و منابع طبیعی ساری
پژوهشنامه اصلاح گیاهان زراعی

انتخاب کوچکترین ابر رشته در DNA با استفاده از الگوریتم ازدحام ذرّات

غلامعلی رنجبر[۱] و فاطمه خادمی آقمشهدی[۲]

۱- دانشیار، دانشگاه علوم کشاورزی و منابع طبیعی ساری
۲- کارشناس فناوری اطلاعات، دانشگاه علوم کشاورزی و منابع طبیعی ساری، (نویسنده مسوول: khademi@sanru.ac.ir)
تاریخ دریافت: ۹۳/۱/۲۶ تاریخ پذیرش: ۹۳/۲/۲۴

چکیده

یک رشته DNA را می‌توان رشته‌ای بسیار طولانی روی الفبایی با ۴ حرف در نظر گرفت. تعداد زیادی از دانشمندان سعی در رمزگشایی این رشته دارند. از آنجایی‌که این رشته بسیار طولانی است، ابتدا بخش‌های کوتاه‌تری از آن که با هم همپوشانی دارند رمزگشایی می‌شود. البته مکان اصلی این بخش‌ها در DNA اصلی مشخص نیست. به‌نظر می‌رسد کوتاه‌ترین رشته‌ای که این بخش‌ها زیر رشته‌ای از آن می‌باشند تقریب مناسبی برای رشته DNA اصلی باشد. لذا این پژوهش بر آن است به ارائه یک الگوریتم تکاملی جهت انتخاب کوتاهترین ابررشته در یک DNA بپردازد. مسئله عملی مورد بحث در این پژوهش، مسئله کوتاه‌ترین ابررشته SSP[۱] است. در این راستا، با استفاده از الگوریتم بهینه‌سازی ازدحام ذرات PSO[۲] که در رده الگوریتم‌های تکاملی قرار دارد و با استفاده از زبان برنامه‌نویسی متلب نسخه R2011a به حل این مسئله پرداخته شد. در مقایسه با مسئله حل شده توسط الگوریتم ژنتیک، نتایج الگوریتم ازدحام ذرات روش برتری است.

واژه‌های کلیدی: DNA، کوتاه‌ترین ابررشته مشترک، الگوریتم بهینه‌سازی ازدحام ذرات

مقدمه

پیشینه علم بیوانفورماتیک به سال‌های ۱۹۶۰ میلادی باز می‌گردد، که در آن اولین پایگاه اطلاعاتی توالی‌های پروتئینی توسط آقای دی‌هوف مطرح گردید (۹). امروزه علم بیوانفورماتیک عمدتاً به حل مسایلی می‌پردازد که در گسترش مطالعات زیست‌شناختی می‌باشند (۳). عمده‌ترین چالش موجود در بیوانفورماتیک چگونگی تبدیل یک مسئله زیستی به یک مسئله کمی و محاسباتی است. مسئله کمی و محاسباتی باید از یک طرف به اندازه کافی انتزاعی باشد تا با الگوریتم‌ها حل شود و از طرف دیگر محدودیت‌های کاربردی و خطاهای واقع شده در آزمایشات را به‌منظور قابلیت اجرا به‌وسیله زیست‌شناسان بررسی کند. یکی از مسائل محاسباتی در زمینه زیست‌شناسی مولکولی توالی‌یابی DNA است که شامل مسئله‌ای به نام کوتاه‌ترین ابررشته می‌شود (۱۵).

پیشرفت‌های زیست مولکولی در چند دهه گذشته توالی‌یابی سریع تعداد زیادی از ژنوم موجودات مختلف را امکان‌پذیر کرده است. چنان که تا به امروز برخی از ژنوم‌های تک‌سلولی و موجودات پیچیده‌تر گیاهی و همچنین ژنوم انسان، توالی‌یابی شده‌اند (۹).

توالی‌های DNA به‌صورت رشته‌هایی با حداکثر ۴ نوع حرف (بر اساس ۴ نوع باز موجود در DNA) و توالی‌های پروتئینی به‌صورت رشته‌هایی (بر اساس ۲۰ نوع اسیدآمینه موجود در سطح پروتئین) قابل دسترس، ثبت و استفاده به‌وسیله کامپیوتراند (۱۸). لزوم ذخیره‌سازی، تجزیه و تحلیل و بهره‌گیری از این اطلاعات منجر به ایجاد رشته‌ای جدید با نام بیوانفورماتیک شد. تمام تلاش دانشمندان در این زمینه به رمزگشایی از رشته طویل DNA معطوف شده است. از آنجایی‌که این رشته بسیار طولانی است، ابتدا بخش‌های کوتاه‌تری از آن که با همدیگر همپوشانی دارند رمزگشایی می‌شوند. البته مکان اصلی این بخش‌ها در DNA اصلی مشخص نیست. به‌نظر می‌رسد که کوتاه‌ترین رشته‌ای که این بخش‌ها زیررشته‌ای از آن می‌باشند تقریب مناسبی برای طول واقعی و محل استقرار این رشته کوتاه روی رشته DNA اصلی می‌باشد (۷).

مسئله SSP درصدد جستجوی کوتاه‌ترین رشته‌ای است که شامل تمام رشته‌ها با توجه به یک مجموعه داده شده است که به آن ابررشته گفته می‌شود. یک روش معمول برای یافتن یک ابررشته ادغام رشته‌ها است. ابررشته SSP، یکی از مسائل شناخته شده و مشکل بهینه‌سازی ترکیبی است که تعدادی از مسائل دنیای واقعی در زمینه آنالیز رشته‌ها را فرموله می‌نماید. مایر، غیرچندجمله‌ای از نوع سخت[۳] بودن مسئله کوتاه‌ترین ابررشته را اثبات نمود (۲).

1- Shortest Superstring Problem 2- Particle Swarm Optimization 3- NP-HARD

پیدا کردن کوتاهترین ابررشته مشترک در فشرده‌سازی داده به‌دلیل اینکه ممکن است به‌طور مؤثر به‌عنوان یک ابررشته ذخیره شود کاربرد دارد. ابررشته SSP همچنین در زیست‌شناسی محاسباتی کاربردهای مهمی دارد، کـه در آن مسئلـه تـوالـی DNA بـه یک رشته از DNA نگاشت می‌شود. (۷).

حل این مسئله در زمینه‌های مختلفی از رده ابررشته‌ها و ابرتوالی‌ها، مانند الگوریتم‌های فشرده‌سازی و خلاصـه‌سازی داده‌هـا، روش‌هـای زمـان‌بـندی، فشرده‌سازی ماتریس‌های اسپارس، رمزنگاری اطلاعات محرمانه و محاسبات پزشکی و بیولوژیکی مانند تولید توالی رشته‌های DNA، و جستجوی درصد تکرار ژن‌ها در مولکول‌های DNA به‌منظور فشرده‌سازی محتوای DNA موجودات سودمند و مورد استفاده است (۲).

بلوم و همکاران (۴) برای حل مسئله SSP، یک الگوریتم تقریب با نام TGREEDY معرفی نمودند، که یک حل بهینه با فاکتور ۳ تولید نمود. بهترین فاکتور در حال حاضر ۲/۵ شناخته شده است و توسط سوئیدیک (۱۶) به‌دست آمده است.

آرمن (۱) در مقاله‌ای با عنوان الگوریتم تقریب $2\frac{2}{3}$ برای مسئله کوتاهترین ابـررشته، یک الگوریتم جدید بـا عنـوان
G-SHORT STRING را برای حل مسئله کوتاهترین ابررشته مطرح کرد و نشان داد که این الگوریتم یک نرخ تقریب از $2\frac{2}{3}$ را به‌دست می‌آورد (۱).

ربای (۱۱) برای حل مسئله تقریب کوتاه‌ترین ابررشته مشترک (SACS) الگوریتم تقریب حریصانه SACS را پیشنهاد کرد. الگوریتم پیشنهادی دارای تقریب $\frac{1}{2}$ برای مسئله SACS با پیچیـدگـی زمـان اجـرای $O(n2 \times (L2 + \log(n))$ می‌باشد. که در آن n تعداد رشته‌ها و L طول رشته می‌باشد. الگوریتم SACS پیشنهادی بر مبنای محاسبه طول بزرگترین همپوشانی تقریب LALO[1] می‌باشد. که یک الگوریتم برای محاسبه LALO نیز پیشنهاد داده است (۱۱).

زو و همکاران (۱۹) الگوریتم برنامه‌نویسی پویا را که سریع و به‌صورت ساده محاسبه می‌شود برای حل مسئله کوتاه‌ترین ابررشته مشترک پیشنهاد دادند. این الگوریتم می‌تواند برای مدیریت مسئله کوتاه‌ترین ابررشته مشترک در دامنه‌های خاص، مانند علوم کامپیوتر، فناوری اطلاعات و بیوانفورماتیک تطبیق یا توسعه داده شود. همچنین الگوریتم می‌تواند در بودن نویزها نیز مؤثر باشد. اگر در عوض یک گراف معتبر از گراف همپوشانی استفاده شود الگوریتم کارایی بهتری را دارا می‌باشد. در این پیشنهاد گراف همپوشانی به‌صورت کامل و جهت‌دار است. یک یال از رأس v_a و v_b، $\frac{\mu(a,b)}{min(|a|,|b|)}$ می‌باشد که a و b رشته‌هایی هستند که توسط v_b و v_a به‌ترتیب نمایش داده شده است (۱۲).

گلار و همکاران (۶) یک الگوریتم DNA برای حل یک مسئله NP کامل (مسئله کوتاه‌ترین ابررشته مشتـرک) را پیشنهاد دادند. آنهـا با یـکسری محدودیت‌های عملی رو به رو بودند که برای پیاده‌سازی الگوریتم یک سیستم کدینگ را به‌عنوان یک راه‌حل برای این محدودیت‌های عملی پیشنهاد نمودند (۶).

زاریسکی (۱۸) در مقاله‌ای با عنوان راه‌حل‌های تکاملی برای مسئله کوتاه‌ترین ابررشته مشترک، ۴ رویکرد برای یافتن راه‌حل‌هایی برای مسائل کوتاه‌ترین ابررشته مشترک را مقایسه کرده است. شامل: الگوریتم ژنتیک استاندارد، الگوریتم مشارکتی Coevolving، الگوریتم حریصانه BenchMark و یک رویکرد موازی حریصانه-coevolving. نتایج نشان می‌دهد که coevolving مشارکتی پرقدرت‌ترین ابزار می‌باشد که از هر دو الگوریتم ژنتیک و حریصانه بهتر است و نشان داد که رویکرد coevolving بهترین نتایج را تولید می‌کند (۱۸).

صفائی و همکاران (۱۳)، راه‌حلی بهینه برای مسئله SSP با استفاده از الگوریتم ممتیک ارائه کردند. وی از راه‌کارهای متفاوتی در بخش‌های مختلف الگوریتم ممتیک به‌منظور بهبود کیفیت پاسخ‌ها و تسریع روند همگرایی استفاده نمود. وی از ساختار جدیدی برای شبیه‌سازی ابررشته کروموزوم استفاده نمود. در ساختار پیشنهادی وی، هر کروموزوم از دو قسمت ژن (Gene) و مم (Meme) تشکیل شده است. ژن‌هـا به ترتیب قـرارگیری
زیررشته‌ها در امتداد یکدیگر را مشخص نموده و مم‌ها نیز میزان همپوشانی هر زیرشته را نسبت به زیررشته قبل از آن تعیین می‌کنند (۱۳).

هدف از این پژوهش ارائه یک الگوریتم تکاملی جهت انتخاب صحیح، دقیق و آسان کوتاهترین ابررشته در یک رشته طویل DNA می‌باشد. با توجه به قدمت کوتاه معرفی الگوریتم PSO به‌عنوان یک روش مؤثر و کارا در حل مسائل بهینه‌سازی و همچنین زمینه تخصصی مبتکران این روش، در ابتدا از PSO بیشتر در حل مسائل علوم کامپیوتر و مهندسی برق و الکترونیک استفاده می‌شد. با نتایج قابل قبولی که از تحقیق روی این الگوریتم حاصل گردید، PSO به مرور قدم به دنیای سایر علوم نهاد. مقالات ارائه شده و رساله‌هایی که در علوم متعدد در زمینه کاربرد و یا توسعه الگوریتم PSO در حل مسائل بهینه‌سازی نوشته شده‌اند، خود بیانگر این مطلب است. لذا در این پژوهش الگوریتم بهینه‌سازی مورد استفاده PSO می‌باشد که در محیط

1- Length of the Approximate Longest Overlap

نرم‌افزار متلب نسخه Matlab R2011a پیاده‌سازی شده است (۱۰).

مواد و روش‌ها

این پژوهش به حل مسئله کوتاه‌ترین ابررشته با استفاده از الگوریتم ازدحام ذرات خواهد پرداخت. پیاده‌سازی انجام شده با استفاده از نرم‌افزار متلب نسخه Matlab R2011a است.

الگوریتم بهینه‌سازی ازدحام ذرات (PSO)

روش بهینه‌سازی PSO در سال ۱۹۹۵ توسط جیمز کندی و راسل ابرهارت معرفی گردید. آنها در ابتدا قصد داشتند که با بهره‌گیری از مدل‌های اجتماعی و روابط موجود اجتماعی، نوعی از هوش محاسباتی را به وجود بیاورند که به توانایی‌های فردی ویژه نیازی نداشته باشد. کار آنان، منجر به ایجاد الگوریتمی قوی برای بهینه‌سازی، به نام الگوریتم بهینه‌سازی ازدحام ذرات یا به اختصار PSO شد. این روش از عملکرد دسته‌جمعی گروه‌های حیوانات مانند پرندگان و ماهی‌ها اقتباس شده است (۱۷،۲).

در الگوریتم PSO تعدادی از موجودات وجود دارند که آنها را ذره می‌نامیم و در فضای جستجو پخش شده‌اند. هر ذره مقدار تابع هدف را در موقعیتی از فضا که در آن قرار گرفته است محاسبه می‌کند. سپس با استفاده از ترکیب اطلاعات محل فعلی آن و بهترین محلی که قبلاً در آن بوده است و همچنین اطلاعات یک یا چند ذره از بهترین ذرات موجود در جمع، جهتی را برای حرکت انتخاب می‌کنیم. پس از انجام حرکت جمعی، یک مرحله از الگوریتم به اتمام می‌رسد. این مراحل چندین بار تکرار می‌شوند تا آنکه جواب مورد نظر به‌دست آید.

مراحل اجرای الگوریتم PSO

مراحل اجرای الگوریتم PSO به‌صورت ذیل می‌باشد:

۱- ایجاد جمعیت اولیه و ارزیابی آن

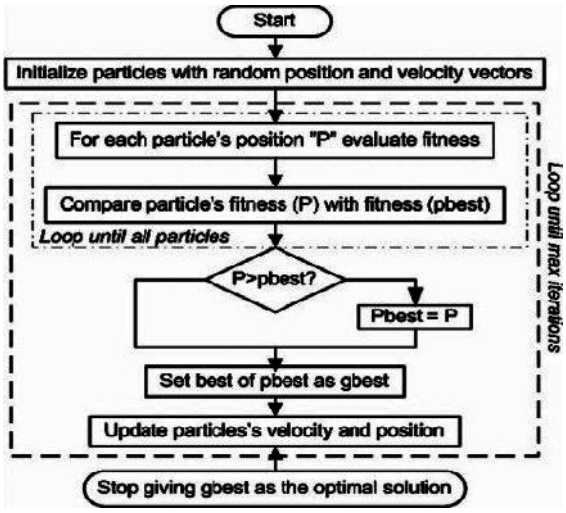
۲- تعیین بهترین خاطره‌های شخصی و بهترین خاطره جمعی

۳- به‌روز رسانی سرعت و موقعیت و ارزیابی پاسخ‌های جدید

۴- در صورت برآورده نشدن شرایط توقف به مرحله ۲ می‌رویم

۵- پایان

ساختار فلوچارت مربوط به الگوریتم PSO در شکل ۱ آمده است:



شکل ۱- فلوچارت الگوریتم PSO [۱۹]

به‌روز رسانی ذرات

در الگوریتم PSO سرعت ذره در هر گام از دو قسمت تشکیل می‌شود که قسمت اول سرعت فعلی ذره و قسمت دوم مربوط به دنبال کردن بهترین تجربه شخصی و بهترین تجربه گروه است. بدون قسمت دوم الگوریتم حالت جستجوی سراسری کورکورانه را خواهد داشت و بدون قسمت اول، الگوریتم به جستجویی محلی در نزدیکی بهترین ذره تبدیل خواهد شد که در رسیدن به قسمت‌های زیادی از فضای جستجو ناتوان خواهد بود. الگوریتم PSO با ترکیب این دو قسمت، سعی می‌کند که به نوعی تعادل را بین جستجوی سراسری و محلی ایجاد نماید (۱۹،۵).

برای به‌روز رسانی سرعت از رابطه زیر استفاده می‌شود (۱۹):

$$Velocity[t+1] = W * Velocity[t] + c1 * rand(0,1) * (pbest[t] - Position[t]) + c2 * rand(0,1) * (gbest - Position[t])$$

W: وزن اینرسی است.

$Velocity[t]$: سرعت ذره در لحظه t است.

$Position[t]$: موقعیت فعلی ذره در لحظه t است.

$c1$ و $c2$: فاکتورهای یادگیری می‌باشند.

$rand(0,1)$: یک عدد تصادفی در بازه (۱،۰) است.

وزن اینرسی (W): وزن اینرسی W در رابطه بالا برای تضمین همگرایی در PSO به‌کار می‌رود. وزن اینرسی جهت کنترل تأثیر سوابق سرعت‌های پیشین بر سرعت‌های جاری مورد استفاده قرار می‌گیرد. مقدار مناسب W توازنی بین توانایی الگوریتم برای جستجوی کلی و محلی ایجاد می‌کند. یک مقدار مناسب وزن اینرسی، معمولاً تعادلی را بین قابلیت اکتشاف فراگیر و محلی گروه ایجاد می‌نماید. با انتخاب مناسب مقدار وزن اینرسی، میزان تکرار برای یافتن جواب بهینه کاهش می‌یابد. وزن ثابت اینرسی بزرگ‌تر از یک، هر چند که سبب می‌شود فضای جستجوی الگوریتم وسیع‌تر گردد اما الگوریتم را ناپایدار می‌کند، زیرا اثر سرعت پیشین را افزایش می‌دهد (۸).

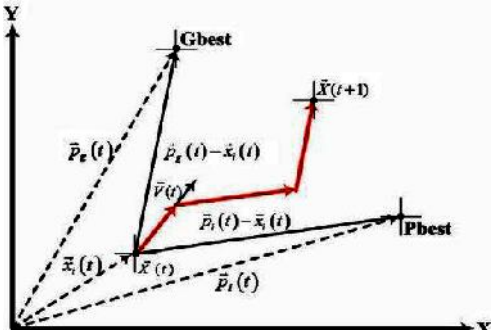
پارامتر $c1$ و $c2$ در رابطه بالا برای همگرایی PSO چندان بحرانی نیستند. مقدار مناسب ممکن است جواب را زودتر همگرا کند و از احتمال قرار گرفتن در کمینه محلی جلوگیری می‌کند. در ابتدا مقدار $c1 = c2 = 2$ پیشنهاد شد، اما آزمایش‌های مختلف نشان داد که $c1 = c2 = 0.5$ می‌تواند در رسیدن به جواب بهتر مفیدتر باشد. در حالت کلی $c1$ و $c2$ می‌توانند بسته به مورد، متفاوت انتخاب شوند؛ اما باید همواره شرط $c1 + c2 \leq 4$ رعایت شود (۳).

پارامتر $rand$ برای حفظ تنوع و گوناگونی گروه به‌کار می‌رود، مقدار مناسب پارامتر $rand$ به‌صورت تصادفی در بازه بین صفر و یک انتخاب می‌شود. این مقادیر به ذرات این اجازه را می‌دهد که در گام‌های تصادفی در محدوده بین gbest و pbest حرکت کنند (۱۰).

برای به‌روز رسانی موقعیت نیز از رابطه ذیل استفاده می‌شود:

$$Position_i(t+1) \leftarrow Position_i(t) + Velocity_i(t) \quad (۲)$$

در معادله بالا $Position_i(t)$ موقعیت ذره iام در زمان t می‌باشد و $Velocity_i(t)$ سرعت ذره iام در زمان t می‌باشد. به‌روز رسانی ذرات در فضای دو بعدی در شکل ۲ نشان داده شده است.



شکل ۲- به روزرسانی ذرات در فضای دو بعدی [۱۰]

حل مسئله کوتاه‌ترین ابررشته با استفاده از الگوریتم PSO

در مسئله کوتاه‌ترین ابررشته مشترک، تعدادی زیررشته وجود دارند که هدف از این مسئله یافتن رشته‌ای است که تمامی این زیررشته‌ها را در بر داشته و طول آن مینیمم باشد.

تعریف ذرات

برای حل این مسئله با استفاده از الگوریتم PSO، هر ذره را به صورت یک ساختار در نظر می‌گیریم، که هر ذره دارای یک موقعیت (متغیر مستقل)، مقدار تابع هدف برای آن موقعیت (متغیر وابسته)، سرعت و یک خاصیت مرکب Best که خود دارای دو خاصیت موقعیت و تابع هدف برای آن موقعیت است می‌باشد. مقادیر مربوط به موقعیت و سرعت را به‌ترتیب با بردار مکان و بردار سرعت مدل‌سازی می‌نماییم.

$$Particle \begin{cases} Position \\ Objective\ Value \\ Velocity \\ Best \begin{cases} Position \\ Objective\ Value \end{cases} \end{cases}$$

برای حل این مسئله برای ایجاد ذره اولیه از ادغام رشته‌ها به‌صورت تصادفی به ایجاد ابررشته پرداختیم. برای پیاده‌سازی ذره در این مسئله با الگوریتم PSO، می‌توان از ساختار رشته‌ای یا آرایه عددی حاوی شماره حروف استفاده نمود. اما نکته‌ای که باید مورد توجه قرار گیرد این است که استفاده از یک ساختار ذره‌ای با طول متغیر، دشواری‌هایی در دیگر بخش‌های الگوریتم به‌وجود می‌آورد. به‌عنوان مثال ممکن است که در طی عمل به‌روز رسانی، ذرات با طول غیرمجاز به‌وجود بیایند. نکته دیگر این است که تعیین تابع هزینه ذرات با طول متغیر، زمان بیشتری را می‌طلبد. بهترین راه برای حل این مشکل، ارائه روشی است که بتوان پاسخ مسئله را در یک ذره با طول ثابت فرموله کرد. در اینجا برای این مسئله از یک آرایه عددی به طول تعداد زیررشته‌ها استفاده شده است که هر خانه آن نماینده مکان شروع یکی از زیررشته‌های مورد نظر در ابررشته است. به‌عبارتی در ذره P به ازای هر زیررشته S_i در مجموعه S، $P[i]$ برابر با مکان شروع زیررشته S_i در ابررشته مورد نظر است. مثلاً برای مجموعه $S = \{ABC, BAAB, ABBA, BAB, ABAA, AAA\}$، ذره مورد نظر یک آرایه ۶ خانه‌ای عددی است و یکی از پاسخ‌های قبول و صحیح این مسئله ابررشته $AAABABAABC$ است که ذره متناظر آن $P = \{3,5,6,1,\dots\}$ خواهد بود.

تابع هزینه

در این مسئله برای رسیدن به هدف مسئله، تابع هزینه براساس میزان همپوشانی رشته‌ها به‌صورت زیر تعریف شده است:

$$CostFunction = 1 \Big/ \sum_{i=1}^{numSubString - 1} Overlap(S_i, S_{i+1})$$

به این معنا که برای تعیین تابع هزینه هر ذره، مجموع همپوشانی رشته‌های مجاور موجود در ابررشته را به‌صورت دوبه‌دو به‌دست می‌آوریم.

به‌روزرسانی ذرات

برای اجرای الگوریتم PSO، نیاز به تعیین پارامترهای موقعیت و سرعت ذرات، مقدار بهینه محلی برای هر ذره و مقدار بهینه سراسری در هر اجتماع می‌باشد. موقعیت هر ذره در واقع همان مقادیر ذره است که در بالا به آن اشاره شد و نحوه قرارگیری هر ذره را در فضای مسئله تعیین می‌کند.

سرعت ذره‌برداری به ابعاد خود ذره است که تعیین می‌کند هر یک از این موقعیت‌ها در فضای مسئله با چه سرعت و جهتی حرکت کنند. سرعت ذره در ابتدا به‌صورت یک آرایه با مقادیر صفر و به ابعاد خود ذره در نظر گرفته شده است و در طول الگوریتم مقادیر این آرایه به‌روز رسانی می‌شود.

مقادیر rand اعداد تصادفی در بازه صفر و یک می‌باشند و مقادیر پارامترهای c_2 و c_1 برابر با مقدار ۲ و W برابر با مقدار ۰/۹ در نظر گرفته شده است. شروطی نیز برای جلوگیری از ایجاد مقادیر نامعتبر در هر ذره در طول الگوریتم استفاده شده است.

مقادیر بهینه محلی و سراسری

مقادیر بهینه محلی برای هر ذره در ابتدای الگوریتم و قبل از شروع، معادل با همان مقادیر اولیه ذرات خواهند بود، در طول الگوریتم مقادیر بهینه محلی برای هر ذره به‌روز رسانی می‌شوند. برای تعیین مقدار بهینه سراسری در جمعیت نیز، ابتدا بهترین ذره به‌عنوان بهینه سراسری در نظر گرفته می‌شود، و در طول الگوریتم چنانچه در جمعیت ذره‌ای با تابع هزینه بهتر نسبت به مقدار بهینه سراسری تولید شود جایگزین این مقدار خواهد شد.

نتایج و بحث

از آنجایی که یکی از مهم‌ترین کاربردهای این مسئله در زمینه بیولوژی مولکولی می‌باشد، معمولاً مهم‌ترین مجموعه داده‌هایی که برای تست این الگوریتم مورد استفاده قرار می‌گیرند الفبای نوکلئوتید هستند. در این مورد هم می‌توان از زنجیره‌های DNA و پروتئین که به صورت تصادفی برای تست این الگوریتم تولید شده‌اند استفاده نمود و هم از DNAهایی که شناخته شده باشد. به‌عنوان مثال زنجیره‌های ویروسی ارائه شده در http://gel.ym.edu.tw/sars/genomes.html .

مسئله برای زنجیره‌های ویروسی SARS گرفته شده از سایت http://gel.ym.edu.tw/sarsl، با ۵۰۰ تکرار و ۲۵۰ ذره حل شده است که رشته ذیل با طول ۳۵۷ حاصل شده است:

مجموعه زنجیره ویروسی SARS برای حل مسئله گرفته شده از http://gel.ym.edu.tw/sarsl:

{ cagagccatg cctaacatgc ttaggataat ggcctctctt gttcttgctc gcaaacataa cacttgctgt aacttatcac accgtttcta caggttagct aacgagtgtg cgcaagtatt aagtgagatg gtcatgtgtg gcggctcact atatgttaaa ccaggtggaa catcatccgg tgatgctaca actgcttatg ctaatagtgt ctttaacatt tgtcaagctg ttacagccaa tgtaaatgca cttctttcaa ctgatggtaa taagatagct gacaagtatg tccgcaatct acaacacagg ctctatgagt gtctctatag aaatagggat gttgatcatg aattcgtgga tgagttttac gcttacctgc gtaaacattt ctccatgatg attct}

Output (Shortest SuperString):

Gttgatcatgtcaagctgcttctttcaattcgtggacgcaagtattagg ataatgttcttgctccatgatgctacagagccatgacaacacaggcta atagtgtaaatgcatcatccggcctctcttcacttgctgtcatgtgtgtt acagccaaatatgttaaagtctctatagcaaacataagatagcttacc tgctccgcaatctaaatagggatcctaacatgctttaacattaagtga gatgattctaacttatcaccaggtggaacgagtgtgcaggttagctg atggtaactctatgagttttacgcggctcactgcttatgacaagtatgt aaacatttaccgtttcta

ارزیابی راهکار پیشنهادی با مجموعه داده‌ها

برای تست الگوریتم پیشنهادی از دو مجموعه داده که در (جدول ۱) معرفی شده‌اند استفاده شده است.

مجموعه اول از [۲۰] انتخاب شده است و مجموعه‌ی بعدی نیز شامل زنجیره‌های ویروسی معرفی شده در بخش قبل است. مجموعه انتخاب شده از [۲۰]:

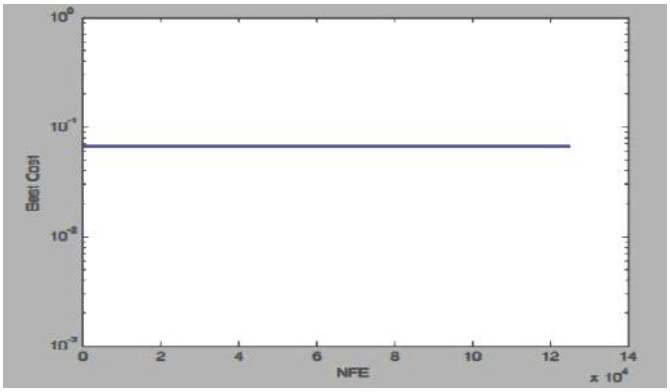
Random={DABRA BRACA CADAB ACADA ABRAC}

جدول ۱- مجموعه داده‌های آزمون

مجموعه		طول کوتاه‌ترین ابررشته
Random	۱	۱۰
SARS	۲	۳۶۵

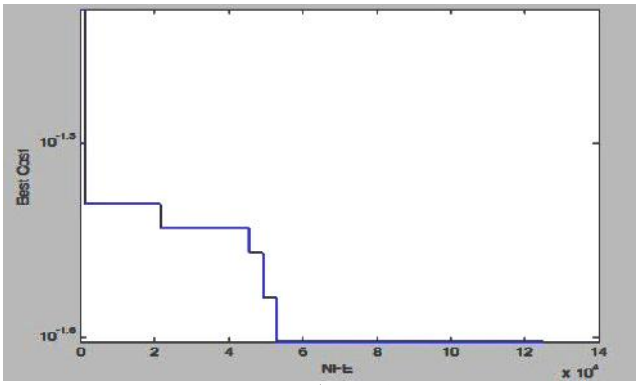
این آزمون با استفاده از ۲۵۰ ذره و در طی ۵۰۰ نسل متوالی از الگوریتم بهینه‌سازی ازدحام ذرات اجرا شده است. شکل‌های ۳ و ۴ نتایج آزمون را برحسب NFE در رابطه با روند بهبود و همگرایی الگوریتم نشان می‌دهند. در الگوریتم PSO معیاری برای مقایسه وجود دارد که در حل مسئله کوتاه‌ترین ابررشته لحاظ گردیده است. این معیار NFE است که بر اساس فرمول ذیل محاسبه می‌شود:

به این معنا که در الگوریتم PSO، اگر به اندازه n_{pop} جمعیت اولیه داشته باشیم یک بار در مرحله ۱ الگوریتم PSO، یعنی مرحله ایجاد جمعیت اولیه و ارزیابی آن، الگوریتم را ارزیابی می‌کنیم و سپس در هر تکرار در مرحله ۳ الگوریتم PSO، یعنی مرحله به‌روز رسانی سرعت و موقعیت و ارزیابی پاسخ‌های جدید آن‌ها را ارزیابی می‌کنیم.



شکل ۳- نمودار مجموعه اول

زمان فاکتور مناسبی برای مقایسه الگوریتم PSO نیست، در اکثر مواقع یک الگوریتم کند است و این به خاطر محاسباتی است که الگوریتم نیاز دارد و آن سخت‌افزار قابلیت جوابگویی سریع به آن الگوریتم را ندارد. بر همین اساس می‌توان از معیار NFE برای مقایسه الگوریتم‌ها استفاده نمود. به‌کار بردن NFE این مزیت را دارد که مستقل از اندازه جمعیت می‌شود و لزومی ندارد اگر اندازه جعیت در مسئله‌ای ۱۰۰ بود در دیگری هم ۱۰۰ باشد. بر اساس NFE مقایسه انجام می‌شود و بررسی می‌شود به‌عنوان مثال بررسی می‌شود در NFE=1000 به چه جوابی رسید. بر همین اساس در حل مسئله کوتاه‌ترین ابررشته مشترک این معیار لحاظ گردیده است.



شکل ۴- نمودار مجموعه دوم

مقایسه با الگوریتم ژنتیک

به‌منظور ارزیابی، با حل انجام شده توسط الگوریتم ژنتیک [۱۴] و [۱۳] مقایسه‌ای بر اساس مجموعه داده ویروس سارس انجام خواهد شد. به این صورت که در سه مجموعه داده به ترتیب محتوی ۱۰، ۱۵ و ۲۰ رشته مقایسه صورت خواهد گرفت.

سه مجموعه داده مورد مقایسه به شرح ذیل است:

مجموعه اول:

{cagagccatg cctaacatgc ttaggataat ggcctctctt gttcttgctc gcaaacataa cacttgctgt aacttatcac accgtttcta caggttagct}

مجموعه دوم:

{cagagccatg cctaacatgc ttaggataat ggcctctctt gttcttgctc gcaaacataa cacttgctgt aacttatcac accgtttcta caggttagct aacgagtgtg cgcaagtatt aagtgagatg gtcatgtgtg gcggctcact}

مجموعه سوم:

{cagagccatg cctaacatgc ttaggataat ggcctctctt gttcttgctc gcaaacataa cacttgctgt aacttatcac accgtttcta caggttagct aacgagtgtg cgcaagtatt aagtgagatg gtcatgtgtg gcggctcact atatgttaaa ccaggtggaa catcatccgg tgatgctaca actgcttatg}

نتایج مقایسه شده بر اساس طول رشته خروجی برای تعداد نسل برابر ۵۰۰ و تعداد ذرات برابر ۲۵۰ برای دو الگوریتم، در جدول ۲ نشان داده شده است؛ همان‌طور که مشخص است (جدول ۲)، نتایج حاصله نشان‌دهنده برتری الگوریتم PSO نسبت به الگوریتم ژنتیک است.

جدول ۲- مقایسه الگوریتم ژنتیک و PSO بر اساس طول کوتاه‌ترین ابررشته

	مجموعه داده اول	مجموعه داده دوم	مجموعه داده سوم
ژنتیک	۹۷	۱۳۸	۱۷۷
PSO	۸۸	۱۳۱	۱۷۵

نتایج مقایسه و ارزیابی روش پیشنهادی با روش‌های بیان شده در فصل گذشته نشان‌دهنده برتری الگوریتم PSO نسبت به الگوریتم ژنتیک برای حل مسئله کوتاه‌ترین ابررشته می‌باشد. نتایج حاصل از مقایسه دو الگوریتم برای حل مسئله کوتاه‌ترین ابررشته در جدول ۲ برای تعداد نسل برابر ۵۰۰ و تعداد ذرات برابر ۲۵۰ نشان داده شده است.

منابع

1. Armen, C. and C. Stein. 1998. A 2 2/3 superstring approximation algorithm. Science direct, 88: 29-57.
2. Assareh, E., M. Behrang, M. Assari and A. ghanbarzadeh. 2010. Application of PSO and GA techniques on demand estimation of oil in iran. Science direct, 35: 5223-5229. (In Persian)
3. Baba-Abbasi, B. 2012. A look on Bioinformatics, Applications and Roles. Ebook. www.bpi.ir. 9 pp. (In Persian)
4. Blum, A., T. jiang, M. Li, J. Tromp and M. Yannakakis. 1994. Linear approximation of shortest superstring. Journal of the ACM, 41: 630-647.
5. Fahimi, M. 2010. Algorithm of Birds' popular movement. Bahar Institute of Higher Education Press19.
6. Gloor, G., L. Kari, M. Gaasenbeek and S. Yu. 1999. Toward a DNA solution to the shortest common superstring problem. ISBN: 0-8186-8548-4. 140-145 pp.
7. Jensen, F. 2010. Using the traveling salesman problem in bioinformatics algorithms. Master thesis. Arhus University, 181pp.

8. Meraji, H., M. Afshar and A. Afshar. 2009. Optimal designing of flooding control systems using particle Swarm optimization algorithm. International Journal of Engineering Sciences. Iran University of Science and Technology, 8: 41-53. (In Persian)
9. Naghavi, M., M. Malbobi and S. Rashidimonfared. 2012. Bioinformatics. Tehran University Press, 475 pp. (In Persian)
10. Rajabpur, R. and M. Afshar. 2008. Optimal exploitation from multiple pumping stations using PSO algorithm. Water and Wastewater, 66: 56-66. (In Persian)
11. Rebai, A. and M. Elloumi. 2005. Approximation algorithm for the shortest approximate common superstring. International journal of computer system science and engineering, 12: 301-307.
12. Rujeerapaiboon, P., A. Burutarchanai and P. Chongstitvatana. An algorithm for the shortest superstring problem: a dynamic programming approach. Department of Computer Engineering. Chulalongkorn University, 4 pp.
13. Safaei, A. and M. Shamsjavi. 2011. A new methodology for solving problem of shortest superstring using memetic algorithm. National Conference in Soft Accounting and Information Technology. Tehran. 2-3 March, 2011. (In Persian)
14. Shamsjavi, M. 2012. Implementation and solving applied problems using genetic algorithm. Aryapaju Press, Tehran. 214 pp. (In Persian)
15. Shatabda, S. 2007. Algorithms in bioinformatics. Department of Computer Science and Engineering.
16. Sweedyk, Z. 1999. A 2.5-Approximation algorithm for shortest superstring. SIAM Journal of Computing, 29: 954-986.
17. Yin, Z., C. Ye and M. Wen. 2011. Digital encoded DNA sequence set design based on particle swarm optimization algorithm. Journal of Computational Information Systems, 7: 940-947.
18. Zaritsky, A. and M. Sipper. 2004. Coevolving solution to the shortest common superstring problem. Science direct, 76: 209-216.
19. Zhu, H., Y. Wang, K. Wang and Y. Chen. 2011. Particle swarm optimization for the constrained portfolio optimization problem. Science direct, 181: 908-919.
20. Bilo, D., H. Joachim, B. Hauer, D. Komm, R. Kralovic, T. Momke, S. Seibert and A. Zych. 2011. Reoptimization of the shortest common superstring problem. Journal of the ACM, 61: 227-251.

Select the Shortest Superstring using Particle Swarm Algorithm in DNA

Golamali Ranjbar[1] and Fateme Khademi Aghmashhadi[2]

1- Associate Professor, Sari Agricultural Sciences and Natural Resources University
2- IT expert, Sari Agricultural Sciences and Natural Resources University
(Corresponding author: khademi@sanru.ac.ir)
Received: April 15, 2014 Accepted: May 14, 2014

Abstract

A DNA string can be supposed a very long string on alphabet with 4 letters. Numerous scientists attempt in decoding of this string. since this string is very long , a shorter section of it that have overlapping on each other will be decoded. There is no information for the right position of these sections on main DNA string. It seems that the shortest string (substring of the main DNA string) is a proper estimation of the main DNA string. Therefore aims of the present study is demonstrating an evolutionary algorithm for selecting the shortest superstring into a DNA string. The practical problem in current study is the shortest superstring problem (SSP). We solve the problem using particle swarm optimization algorithm (PSO) in evolutionary algorithm level by programming language MATLAB version R2011a. In comparison with solving problem by genetic algorithm, the result of present study was much better than the above mentioned algorithm.

Keywords: DNA, Shortest Common Superstring, Particle Swarm Optimization Algorithm