Archive of SID

پیدا کردن موتیف در نواحی بالادست ژن‌های هم‌بیان بر اساس الگوریتم بهینه‌سازی فاخته و سرمایش تدریجی

مهری ملالو[۱]، کارشناسی ارشد، فاطمه زارع میرک آباد[۲]، عضو هیات علمی

۱- دانشکده ریاضی و علوم کامپیوتر - دانشگاه صنعتی امیرکبیر - تهران - ایران - m.mollalo@aut.ac.ir

۲- دانشکده ریاضی و علوم کامپیوتر - دانشگاه صنعتی امیرکبیر - تهران - ایران - f.zare@aut.ac.ir

چکیده: در این مقاله برای حل مسئله کشف موتیف یک روش ترکیبی جدید بر اساس الگوریتم بهینه‌سازی فاخته، روش سرمایش تدریجی و بیشینه‌سازی زمان انتظار به‌نام SA-COAMF ارائه می‌شود. این روش ترکیبی در همگرایی بهینه سراسری بسیار کارآمد است. یکی دیگر از ویژگی‌های شاخص این الگوریتم، بهره بردن از هر دو مدل نمایش موتیف (توالی اجماع و ماتریس احتمالاتی) است. عملکرد الگوریتم پیشنهادی بر روی یک مجموعه از داده‌های زیستی (پایگاه داده SCPD) تست شده و با تعدادی از الگوریتم‌های معروف کشف موتیف (GA-DPAF، PSO+ و MEME) مقایسه می‌گردد. نتایج به‌دست‌آمده نشان‌دهنده توانایی بالای الگوریتم پیشنهادی است.

واژه‌های کلیدی: الگوریتم بهینه‌سازی فاخته، سرمایش تدریجی، ژن‌های هم‌بیان، کشف موتیف، ماکزیمم‌سازی زمان انتظار.

Motif Finding in upstream regulatory regions of co-expressed genes using Cuckoo optimization Algorithm and Simulated Annealing

M. Mollalo, Master Of Science[1], F. Zare-Mirakabad, Assistant Professor[2]

1- Faculty of Computer Science & Mathematics, Amirkabir University of Technology, Tehran, Iran, Email: m.mollalo@aut.ac.ir
2- Faculty of Computer Science & Mathematics, Amirkabir University of Technology, Tehran, Iran, Email: f.zare@aut.ac.ir

Abstract: In this paper, a novel hybrid algorithm is represented named SA-COAMF by using cuckoo optimization algorithm, simulated annealing and expected maximization for motif finding problem. SA-COAMF is very efficient in global optimal convergence. In the algorithm, two models of motif representation, consensus and probability matrix representations, are applied to take the advantage of them. SA-COAMF is run on experimental datasets (SCPD database). The results are compared with some well-known algorithms (GA-DPAF, PSO+ and MEME) to show that our algorithm is efficient.

Keywords: Cuckoo optimization algorithm, simulated annealing algorithm, coexpressed genes, motif finding, expected maximization.

تاریخ ارسال مقاله: ۱۳۹۴/۱/۲۳
تاریخ اصلاح مقاله: ۱۳۹۴/۰۳/۰۵
تاریخ پذیرش مقاله: ۱۳۹۴/۰۶/۲۴
نام نویسنده مسئول: فاطمه زارع میرک آباد
نشانی نویسنده مسئول: ایران - تهران- خیابان حافظ شماره ۴۲۴-دانشگاه صنعتی امیرکبیر -دانشکده‌ی ریاضی و علوم کامپیوتر

www.SID.ir

Archive of SID

۱- مقدمه

یک مجموعه از زیرتوالی‌های حفاظت‌شده در توالی‌های زیستی (ساختار اول مولکول‌های RNA، DNA و پروتئین) را موتیف می‌گویند به‌طوری که این زیرتوالی‌ها دارای ساختار و عملکردهای مشابهی هستند. موتیف‌ها نقش کارایی در تعیین ساختار و عملکرد یک مولکول دارند. مسئله کشف موتیف یکی از مسئله‌های مهم و با اهمیت در زمینه زیست‌شناسی مولکولی شناخته شده است [۱] به‌طوری که حل این مسئله می‌تواند کمک شایانی به زیست‌شناس‌ها در تشخیص بیماری‌هایی مانند آلزایمر بنماید [۲]. یک نوع از موتیف‌های مهم در زیست‌شناسی، موتیف‌های موجود در نواحی پروموتور ژن‌های هم‌بیان است. تعدادی از عوامل الگوبرداری (TF[1]) می‌توانند این موتیف‌ها را به‌صورت خاص شناسایی کنند و این نواحی را برای ورود سایر TFها و الگوبرداری ژن‌ها آماده نمایند [۳]. برای کشف موتیف، روش‌های آزمایشگاهی مختلفی وجود دارد، اما این روش‌ها بسیار زمان‌بر و پر هزینه هستند. ازاین‌رو زیست‌شناس‌ها بر آن شدند تا از محاسبات کامپیوتری در تحقیقات خود بهره گیرند.

در مسئله محاسباتی کشف موتیف، هدف یافتن یک موتیف ناشناخته به طول ℓ است که در t توالی زیستی با بیش‌ترین انطباق رخ داده باشد [۴]. به هر یک از این رخدادهای تقریبی در هر یک از توالی‌های زیستی یک مصداق از موتیف گفته می‌شود. دو روش استاندارد توالی اجماع[2] و ماتریس احتمالاتی (PFM[3]) برای نمایش یک مجموعه از مصداق‌های موتیف وجود دارد [۵, ۶].

در نمایش توالی اجماع، موتیف به‌صورت یک توالی نمایش داده می‌شود که در هر موقعیت آن غالب‌ترین نوکلئوتید از مصداق‌های موتیف در آن موقعیت، قرار می‌گیرد [۷]. در مدل نمایش ماتریس احتمالاتی، موتیف به‌صورت یک ماتریس نمایش داده می‌شود که هر عنصر از موقعیت i و زام این ماتریس نشان‌دهنده احتمال رخداد نوکلئوتید i در موقعیت زام است.

اولین بار در سال ۱۹۹۵، مسئله کشف موتیف به‌عنوان یک مسئله محاسباتی، تعریف شد [۸] و امروزه به‌عنوان یکی از مسئله‌های چالش برانگیز در حوزه بیوانفورماتیک معرفی می‌گردد. با توجه به این که در سال ۲۰۰۰ ثابت شد این مسئله NP- کامل است [۹]، محققان علوم کامپیوتر در پی یافتن یک الگوریتم هستند که بتواند با زمان مناسب یک فضای وسیع از زیر توالی‌های موجود را برای یافتن موتیف اصلی جستجو کند. در راستای رسیدن به این هدف، تا به امروز الگوریتم‌های قطعی و غیرقطعی متنوعی برای حل این مسئله ارائه شده است. الگوریتم‌های Weeder [۱۰]، YMF [۱۱]، MultiProfiler [۱۲] و Projection [۱۳] را می‌توان به عنوان الگوریتم‌های قطعی و الگوریتم‌های EM [۱۴]، MEME [۱۵]، AlignACE [۱۶]، GibbsSampler [۵] و BioProspector [۱۷] را به‌عنوان الگوریتم‌های غیرقطعی معرفی نمود. واضح است که NP-کامل بودن این مسئله باعث شده که الگوریتم‌های غیرقطعی کارآمدتر باشند.

الگوریتم‌های هیوریستیکی GibbsSampler [۵] و MEME [۱۵]، جز اولین الگوریتم‌های غیرقطعی هستند که برای این مسئله ارائه شد. یکی از مشکلات مشترک در هر دو روش، گیرافتادن در بهینه‌های محلی است. جهت غلبه بر این مشکل، اخیرا الگوریتم‌های متاهیوریستیکی در حل مسئله کشف موتیف بسیار مورد توجه قرار گرفته است. زیرا این الگوریتم‌ها فضای جستجو را به‌صورت وسیع‌تری بررسی می‌کنند [۲۳-۱۸].

هدف اصلی اکثر الگوریتم‌های ارائه شده، زمان محاسباتی پایین و کشف موتیف‌های با کیفیت بالا است. نکته قابل توجه این است که دو هدف، محاسبات پایین و پیش‌گویی موتیف با کیفیت بالا، در تضاد با هم هستند.

ازاین‌رو در این مقاله، برای ایجاد یک مصالحه بین این دو هدف، یک الگوریتم ترکیبی جدید بر اساس ترکیب الگوریتم بهینه‌سازی فاخته[4]، تکنیک سرمایش تدریجی (SA[5]) و بیشینه‌سازی زمان انتظار(EM[6]) بنام SA-COAMF[7] ارائه می‌شود. روش بیشینه‌سازی زمان انتظار ابتدا هر فاخته را به بهینه محلی‌اش نزدیک می‌کند و سپس با روش سرمایش تدریجی هر فاخته از بهینه محلی‌اش خارج می‌شود تا به جواب بهینه سراسری دست پیدا کند. در این الگوریتم از مدل ماتریس احتمالاتی برای نمایش موتیف‌ها استفاده می‌شود، زیرا این مدل حامل اطلاعات بیش‌تری نسبت به توالی اجماع است. با توجه به اینکه در الگوریتم ارائه‌شده هدف یافتن یک ماتریس احتمالاتی بهینه است و فضای جستجوی این ماتریس‌ها بسیار وسیع است، برای مهاجرت فاخته‌ها، از نمایش توالی اجماع استفاده می‌گردد. وجود این دو مدل نمایش، باعث می‌شود که الگوریتم از دقت مدل ماتریس احتمالاتی و سرعت بالای توالی اجماع بهره‌مند گردد.

یکی دیگر از مزایای الگوریتم پیشنهادی این است که الگوریتم فاخته استاندارد را به‌صورتی توسعه می‌دهد که بتواند یک مسئله با فضای گسسته را حل نماید.

نتایج اجرای الگوریتم پیشنهادی بر روی ۹ عامل الگوبرداری موجود در پایگاه داده SCPD، نشان‌دهنده قابل رقابت بودن الگوریتم SA-COAMF با الگوریتم‌های MEME [۲۴]، GA-DPAF [۲۵] و PSO+ [۲۱] است.

در بخش ۲ از این مقاله، ابتدا تعاریف اولیه موردنیاز برای مسئله کشف موتیف ارائه و سپس الگوریتم بهینه‌سازی فاخته در حالت کلی شرح داده می‌شود. الگوریتم SA-COAMF به همراه جزئیات در بخش سوم مورد بررسی قرار می‌گیرد. در بخش چهارم، معیار ارزیابی الگوریتم‌های پیداکردن موتیف و پایگاه داده SCPD معرفی می‌شود. سپس در بخش پنجم الگوریتم پیشنهادی با الگوریتم فاخته استاندارد بر روی ۹ مجموعه داده‌ی منتخب از پایگاه داده SCPD، اجرا و مقایسه می‌گردد و همچنین عملکرد SA-COAMF با الگوریتم‌های MEME، PSO+ و GA-DPAF مقایسه می‌شود. سپس زمان پردازش و پیچیدگی زمانی الگوریتم پیشنهادی با دو الگوریتم تکاملی GA-

www.SID.ir

Archive of SID

DPAF و +PSO مقایسه می‌گردد. درنهایت در بخش نتیجه‌گیری، یک جمع‌بندی از الگوریتم پیشنهادی و کارهای آتی ارائه می‌شود.

۲- تعاریف اولیه

در این بخش مفاهیم اولیه مورد نیاز برای حل مسئله کشف موتیف معرفی می‌شوند.

یک توالی DNA، دنباله‌ای از نوکلئوتیدهای $\sum = \{A, C, G, T\}$ است و یک مجموعه از t توالی DNA به‌صورت $S = \{s_1, \dots, s_t\}$ تعریف می‌گردد. طول هر توالی DNA $s_i = s_i[1] \dots s_i[n_i]$ که $s_i[j] \in \sum$ است، برابر $|s_i| = n_i$ است. یک زیرتوالی از توالی s_i که در موقعیت j به طول ℓ رخ می‌دهد، به‌صورت $s_{ij} = s_i[j] \dots s_i[j + \ell - 1]$ نمایش داده می‌شود.

در مسئله کشف موتیف هدف پیداکردن یک مجموعه از زیرتوالی‌ها به‌طول ℓ از مجموعه S است که به هر یک از زیرتوالی‌های به‌دست‌آمده یک مصداق از موتیف گفته می‌شود. یک مجموعه از مصداق‌های موتیف را می‌توان بر اساس رابطه زیر نمایش داد:

$$U = \{u_1, \dots, u_t\} \qquad (۱)$$

که u_i یک زیرتوالی بنام مصداق موتیف از توالی s_i در موقعیت p_i به‌طول ℓ است. مجموعه $P = \{p_1, \dots, p_t\}$ نمایش‌دهنده موقعیت‌های شروع مصداق‌های موتیف در مجموعه توالی S است.

ارزش هر مجموعه مصداق U (برازش) بر اساس تابع محتوای اطلاعاتی ($IC^{\wedge}$) که در رابطه زیر تعریف شده است، به دست می‌آید:

$$IC = \sum_{\alpha \in \sum} \sum_{i=1}^{\ell} W[\alpha, i] \times \log_2 \frac{W[\alpha, i]}{b[\alpha]}, \qquad (۲)$$

به‌طوری که $W[\alpha, i]$ احتمال رخداد کاراکتر α در موقعیت iام از مجموعه مصداق‌های موتیف و $b[\alpha]$ احتمال رخداد کاراکتر α در مجموعه توالی S است [۲۶].

با توجه به تعاریف ارائه شده می‌توان مسئله کشف موتیف را به این صورت تعریف کرد که هدف، پیدا کردن یک مجموعه مصداق U با بیش‌ترین مقدار IC است.

اگر توالی اجماع $e = e[1], \dots, e[\ell]$ نماینده موتیف اصلی از مجموعه S باشد، میزان شباهت زیرتوالی s_{ij} با توالی اجماع e بر اساس رابطه زیر محاسبه می‌شود:

$$MS(s_{ij}, e) = \sum_{k=1}^{\ell} \chi(s_i[j + k - 1], e[k]), \qquad (۳)$$

که تابع M میزان شباهت دو کاراکتر a و b را بر اساس رابطه زیر محاسبه می‌نماید:

$$\chi(a, b) = \begin{cases} 1 & if\ a = b, \\ 0 & otherwise. \end{cases} \qquad (۴)$$

مثال ۱: با توجه به توالی اجماع داده‌شده e = ATTTGAA، از هر رشته ورودی s_i در جدول ۱ یک زیرتوالی با حداکثر میزان شباهت بر اساس روابط (۳) و (۴) استخراج می‌شود. سپس مجموعه مصداق $U = \{u_1, \dots, u_7\}$ که متناظر با جدول ۲ است و مجموعه موقعیت شروع P={۵، ۲، ۵، ۷، ۴، ۶، ۳} مشخص می‌گردند.

جدول ۱: توالی‌های ورودی مثال ۱

s_1 = aa AGTGAAA taataa	s_2 = gtgga ATTGGAA ttg
s_3 = tct AGTTTGA aaaca	s_4 = tttcta TATTGAA ag
s_5 = tgac AGTTGTA acaa	s_6 = a ATATGCT gtcaaca
s_7 = tgtg ATTTCTT gcaa	

جدول ۲: مجموعه مصداق‌های موتیف مثال ۱

u_1 = AGTGAAA,	u_2 = ATTGGAA,
u_3 = AGTTTGA,	u_4 = TATTGAA,
u_5 = AGTTGTA,	u_6 = ATATGCT,
	u_7 = ATTTCTT

اگر موتیف اصلی بر اساس ماتریس احتمالاتی $W_{4\times\ell}$ تعریف گردد، یک زیرتوالی s_{ij} از توالی s_i به‌عنوان مصداق معتبر u_i برای موتیف اصلی انتخاب می‌شود اگر دارای بیش‌ترین شباهت به ماتریس احتمالاتی W باشد. میزان شباهت هر زیرتوالی s_{ij} با W بر اساس رابطه زیر به دست می‌آید:

$$Score(s_{ij}, W) = \sum_{k=1}^{\ell} \log_2 \frac{W[s_i[j + k - 1], k]}{b[s_i[j + k - 1]]}, \qquad (۵)$$

که $W[\alpha, i]$ و $b[\alpha]$ به ترتیب احتمال رخداد نوکلئوتید α در ستون iام از ماتریس W و احتمال رخداد نوکائوتید α در مجموعه S است. جدول ۳ ماتریس احتمالاتی مربوط به مجموعه مصداق‌های مثال ۱ که در جدول ۱ مشخص شده است را نشان می‌دهد.

جدول ۳: ماتریس احتمالاتی مثال ۱

	۱	۲	۳	۴	۵	۶	۷
A	۰/۸۶	۰/۱۴	۰/۸۶	۰	۰/۱۴	۰/۴۴	۰/۷۱
C	۰	۰	۰	۰	۰/۱۴	۰/۱۴	۰
G	۰	۰/۴۳	۰	۰/۲۹	۰/۵۸	۰/۱۴	۰
T	۰/۱۴	۰/۴۳	۰/۱۴	۰/۷۱	۰/۱۴	۰/۲۸	۰/۲۹

۲-۱- الگوریتم بهینه‌سازی فاخته

الگوریتم بهینه‌سازی فاخته یکی از الگوریتم‌های تکاملی جدید است که توسط رامین رجیبون در سال ۲۰۱۱ برای مسئله‌های بهینه‌سازی خطی پیشنهاد شد [۲۷]. این الگوریتم از زندگی یک دسته پرنده که فاخته نامیده می‌شود الهام گرفته شده است.

www.SID.ir

Archive of SID

فاخته‌ها پرندگانی هستند که به‌صورت زیرکانه پرورش تخم‌ها و جوجه‌های خود را برعهده پرندگان دیگر قرار می‌دهند، بدون آنکه پرندگان میزبان متوجه شوند تخم‌های درون لانه، مربوط به خودشان نیست.

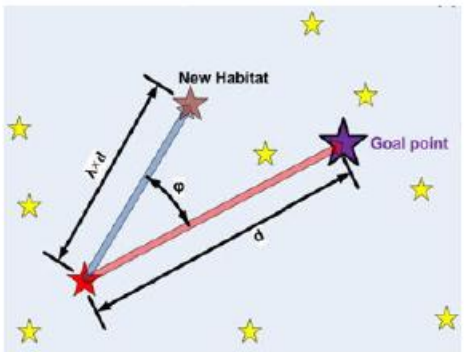
مشابه سایر الگوریتم‌های تکاملی، الگوریتم بهینه‌سازی فاخته با یک جمعیت اولیه شروع می‌شود. این پرندگان تخم‌های خود را در لانه سایر پرندگان قرار می‌دهند و صبر می‌کنند تا آن‌ها در کنار تخم‌های خود تخم‌های این پرندگان را نیز نگه‌داری کنند. برخی از پرندگان میزبان، تخم‌های فاخته‌ها را در لانه‌های خود تشخیص داده و آن‌ها را از لانه بیرون می‌اندازند [۲۸]. بنابراین واضح است، تخم‌های با شباهت بیش‌تر به تخم‌های پرنده میزبان، شانس بیش‌تری برای زنده ماندن و تبدیل به فاخته بالغ دارند. از طرف دیگر، جوجه‌های فاخته زودتر از تخم‌های پرنده میزبان از تخم بیرون می‌آیند و زودتر هم رشد می‌کنند. در اکثر موارد جوجه فاخته، تخم‌ها و یا جوجه‌های پرنده میزبان را از لانه بیرون می‌اندازد و درنهایت در هر لانه یک تخم شانس رشد پیدا می‌کند.

در طبیعت هر فاخته بین ۱ تا ۲۰ تخم را در یک دامنه مشخص می‌گذارد که حداکثر دامنه تخم‌گذاری (ELR[۹]) می‌نامند. هر فاخته دارای ELR مخصوص به خود است.

وقتی جوجه‌های فاخته‌ها رشد کردند و تبدیل به فاخته بالغ می‌شوند، پرنده‌های فاخته با میزانی فاصله و انحراف به‌سمت بهترین منطقه فعلی در بین تمام فاخته‌ها که شانس زنده ماندن تخم‌ها بیش‌تر است، مهاجرت می‌کنند. همان‌طور که در شکل ۱ دیده می‌شود هر فاخته فقط درصدی از کل مسیر را به‌سمت هدف ایده‌آل فعلی طی می‌کند و میزانی انحراف نیز دارد. این دو پارامتر به فاخته‌ها کمک می‌کنند تا محیط بیش‌تری را جستجو کنند و از گیرافتادن در بهینه‌های محلی اجتناب نمایند.

وقتی تمام فاخته‌ها به‌سمت نقطه هدف مهاجرت کردند و نقاط سکونت جدید هرکدام مشخص شد، دوباره مراحل تخم‌گذاری و محاسبه ELR برای هر یک از فاخته‌ها انجام می‌شود.

با توجه به این واقعیت که همیشه تعادلی بین جمعیت پرندگان در طبیعت وجود دارد، N_{max} معادل با حداکثر تعداد فاخته‌هایی که می‌توانند در یک محیط زندگی کنند؛ تعریف می‌گردد.



شکل ۱: مهاجرت یک نمونه فاخته به‌سمت محل اقامت هدف

۳- الگوریتم پیشنهادی SA-COAMF

در این مقاله یک الگوریتم ترکیبی جدید بنام SA-COAMF بر اساس روش بهینه‌سازی فاخته، تکنیک سرمایش تدریجی و بیشینه‌سازی زمان انتظار جهت کشف موتیف ارائه می‌شود که این الگوریتم توانایی جستجوی سراسری و همگرایی به جواب بهینه را داشته و از ترکیب فواید هر دو روش‌های توالی اجماع و ماتریس احتمالاتی جهت نمایش موتیف استفاده می‌نماید. این الگوریتم، یک مجموعه توالی(S) و طول موتیف (ℓ) را به‌عنوان ورودی دریافت می‌کند. مراحل الگوریتم پیشنهادیSA-COMF در شبه‌کد (۱) بیان می‌گردد.

۳-۱- جمعیت اولیه

در گام اول الگوریتم، یک جمعیت اولیه از فاخته‌ها به تعداد $\text{numCuckoos} = \frac{\sum_{i=1}^{t} n_i}{(15 \times t)}$، که n_i طول توالی iام و t تعداد توالی‌های مجموعه S است، ساخته می‌شود. عدد ۱۵ بیش‌ترین طول موتیف در توالی‌ها است، زیرا طول موتیف‌ها در نواحی پروموتور معمولاً بین ۱۵-۱۰ است.

هر فاخته j، در الگوریتم SA-COAMF یک مجموعه موقعیت $P_j = \{p_{j1}, \ldots, p_{jt}\}$ است که هر p_{ji}، $1 \le i \le t$، موقعیت شروع مصداق iام در توالی s_i را نشان می‌دهد. به هر p_{ji}، یک موقعیت تصادفی در محدوده $(n_i - \ell) \ldots 1$ نسبت داده می‌شود. برای هر فاخته‌ی j، مجموعه P_j $1 \le j \le \text{numCuckoos}$، بر اساس روش EM به‌صورت زیر به‌روز می‌گردد:

مجموعه مصداق U_j بر اساس بردار موقعیت فاخته jام، P_j، ساخته می‌شود به‌طوری که طول هر مصداق برابر ℓ است. سپس بر اساس مجموعه مصداق‌های U_j، ماتریس احتمالاتی W_j حاصل می‌گردد. با استفاده از ماتریس احتمالاتی W_j در تمامی توالی‌های مجموعه S جستجو می‌شود تا در هر توالی s_i، یک زیرتوالی مانند u_{ji} به طول ℓ پیدا گردد که نسبت به ماتریس احتمالاتی W_j و با توجه به رابطه (۵) دارای بیش‌ترین میزان شباهت باشد. به‌این‌ترتیب بردار موقعیت جدیدی بنام P_j^* به دست می‌آید. به ازای P_j^* و P_j دو مجموعه مصداق U_j^* و U_j ساخته می‌گردد و سپس برازش هر یک از آن‌ها بر اساس رابطه (۲) محاسبه می‌شود. اگر برازش P_j^* از P_j بهتر است، P_j^* به‌جای P_j قرار می‌گیرد و مجدداً الگوریتم EM اجرا می‌گردد، در غیر این‌صورت الگوریتم خاتمه می‌یابد.

جمعیت اولیه فاخته‌ها و محاسبه نتایج برازش در یک لیست بنام Cuckoos ثبت می‌شود.

در ادامه G فاخته با بیش‌ترین مقدار برازش، به‌عنوان جواب هدف در لیست Goals نگهداری می‌شود.

www.SID.ir

Archive of SID

شبه کد (۱): مراحل الگوریتم **SA-COAMF**

```
SA_COAMF() {
  Inputs:
        Sequences : S = {s, ..., s_t}
        ℓ : the length of the motif
  Parameters:
        T: The temperature for Simulated Annealing;
        α: amount of decrees in temperature;
        maxIter: maximum iteration of the algorithm;
        N_max: maximum number of cuckoos that can live at the same time;
        G: maximum number of goal points;
1.  (Cuckoos, numCuckoos) = Generate Initial Population;
2.  Create a list called Goal to save G numbers of Cuckoos with the highest fitness values;
3.  for iter=1 to maxIter do begin
4.      for j=1: numCuckoos do begin
5.          Cuckoos[j] lay eggs:
                A. Initialize number of eggs for each cuckoo P_j and called Egg;
                B. Calculate ELR for each egg of P_j and called E_k;
                C. Based on computed E_k s', find a nest for each egg P_j^k of P_j ;
                D. If P_j^k ∉ Cuckoos, compute fitness function for P_j^k and add it to the list Cuckoos;
        end
6.      while (numCuckoos > N_max) do begin
            i. Remove each cuckoo with the worst fitness values from Cuckoos list;
            ii. numCuckoos = numCuckoos − 1;
        end
7.      Update Goal list based on Cuckoos list
8.      Immigration of cuckoo;
    end

9.  Goal List is declared as a list of predicted motifs
```

پیچیدگی زمانی محاسبه هر فاخته برابر $O\left(nt^2\ell\right)$ است، زیرا برای جستجوی هر مجموعه مصداق از موتیف $O\left(nt\ell\right)$ زمان نیاز است و در ضمن هر فاخته t بار بر اساس الگوریتم EM به‌روزرسانی می‌شود.

۳-۲- تخم‌گذاری هر فاخته

در گام پنجم الگوریتم، هر فاخته j در سه مرحله تخم‌گذاری می‌کند که در ادامه هر یک از این مراحل به‌صورت کامل شرح داده می‌شود.

در مرحله A، تعداد تخم‌های هر فاخته j، Egg، در محدوده $1,..,\log_2 t$ به‌صورت تصادفی تعریف می‌شود. این شرط باعث می‌شود که احتمال قرار گرفتن یک تخم از چند فاخته را در یک لانه کاهش دهد.

در مرحله B، به ازای هر تخم از فاخته j، شعاع تخم‌گذاری، E_k، در محدوده $1, \ldots, t$ به‌صورت تصادفی مشخص می‌گردد به‌طوری که $k = 1 \ldots Egg$ است. این محدودیت باعث می‌شود که احتمال قرار گرفتن چندین تخم از یک فاخته را در یک لانه کاهش دهد.

در مرحله C، برای هر تخم k از فاخته‌ی j، P_j^k معادل P_j تعریف می‌گردد، سپس E_k عنصر از P_j^k تصادفی انتخاب می‌شوند و هر یک از آن‌ها به‌صورت زیر تغییر می‌کنند:

فرض کنید p_{ji}^k یک عنصر انتخاب شده از P_j^k باشد. در این شرایط، با احتمال ۰/۵ از سمت چپ، $p_{ji}^k \ldots 1$ ، یا راست، $p_{ji}^k \ldots n_i$، توالی s_i یک موقعیت به تصادف مشخص می‌گردد و با p_{ji}^k جایگزین می‌شود.

در پایان، P_j^k حاصل بر اساس روش EM به‌روز می‌شود. در مرحله D، برازش فاخته تولیدشده که معادل هیچیک از فاخته‌های لیست Cuckoos نیست بر اساس رابطه (۲) محاسبه و به‌همراه فاخته‌اش به لیست اضافه می‌شود.

در گام ششم، فاخته‌هایی که برازش کم‌تری دارند تا رسیدن به سقف N_{max}، حداکثر تعداد فاخته در طبیعت، از لیست Cuckoos حذف می‌شوند.

در گام هفتم، لیست Goal بر اساس لیست Cuckoos جدید به‌روز می‌گردد.

پیچیدگی زمانی تخم‌گذاری هر فاخته برابر $O\left(nt^2\ell\log_2 t\right)$ است، به‌طوری که $\log_2 t$ تعداد تخم‌های هر فاخته و به‌روزرسانی هر تخم $O\left(nt^2\ell\right)$ زمان نیاز دارد.

۳-۳- مهاجرت فاخته‌ها

در گام هشتم الگوریتم SA-COAMF، عمل مهاجرت فاخته‌ها انجام می‌شود. با توجه به اینکه در این الگوریتم چندین هدف وجود دارد که فاخته‌ها می‌توانند به‌سمت آن‌ها مهاجرت کنند، برای پیدا کردن هدف برای هر فاخته j، مراحل زیر انجام می‌شود:

برای هر فاخته j، بر اساس بردار موقعیت P_j و مجموعه مصداق‌های U_j، توالی اجماع محاسبه می‌گردد. سپس عمل ساخت توالی اجماع برای هریک فاخته‌های لیست Goal نیز انجام می‌شود. میزان شباهت توالی اجماع فاخته jام، با هر یک از توالی‌های اجماع لیست Goal بر اساس رابطه (۳) محاسبه می‌گردد. از لیست Goal فاخته d که توالی اجماع آن شباهت بیش‌تری با توالی اجماع فاخته jام دارد، انتخاب می‌شود و فاخته jام، P_j، با میزانی انحراف به‌سمت فاخته dام، P_d ، حرکت می‌کند تا مجموعه موقعیت جدید P_j^*

www.SID.ir

Archive of SID

تولید شود. جابه‌جایی هر p_{ji} به‌سمت p_{di} که $1 \le i \le t$، بر اساس رابطه زیر انجام می‌شود:

$$P_{ji}^{*} = \begin{cases} int\left(p_{ji} - \left(r(1) \times (p_{ji} - p_{di})\right)\right), & p_{di} < p_{ji} \\ int\left(p_{di} - \left(r(1) \times (p_{di} - p_{ji})\right)\right), & else \end{cases} \quad (۶)$$

به‌طوری‌که $\mathbf{r(1)}$ یک عدد تصادفی بین صفر تا یک است.

بعد از انجام عمل مهاجرت، با استفاده از تکنیک سرمایش تدریجی P_j^* با P_j در لیست Cuckoos جایگزین می‌شود. اگر مقدار برازش P_j^* بیش‌تر از P_j باشد، بردار موقعیت P_j^* جایگزین بردار P_j می‌شود و به‌عنوان فاخته نسل بعد در جمعیت قرار می‌گیرد. در غیر این صورت، روش SA با احتمال کم‌تر از $e^{(fit(P_j^*)-fit(P_j))/t}$ بردار P_j^* را جایگزین P_j در جمعیت نسل بعد می‌نماید که $\mathbf{fit}(P)$ میزان برازش بردار موقعیت P از یک فاخته را نشان می‌دهد.

مقدار کاهش تدریجی دما در هر نسل باعث می‌شود که الگوریتم در نسل‌های بالاتر از پذیرش P_j^* با مقدار تابع برازش پایین اجتناب نماید و الگوریتم به سمت جواب بهینه سراسری همگرا شود. پیچیدگی زمانی مهاجرت هر فاخته معادل $O(G\ell t)$ است.

۳-۴- اعلام جواب نهایی

پس از اجرای maxIter نسل، الگوریتم خاتمه یافته و لیست $\boldsymbol{Goal}$ به‌عنوان جواب‌های نهایی انتخاب می‌شوند. بر اساس بردارهای موقعیت هر یک از فاخته‌های لیست $\boldsymbol{Goal}$ یک پروفایل احتمالاتی مانند W تولید می‌شود. به هر زیرتوالی s_{ij} به‌طول ℓ از توالی $s_i \in S$ بر اساس رابطه زیر امتیاز داده می‌شود:

$$\mathrm{NScore}\left(s_{ji}, \mathrm{W}\right) = \frac{\mathrm{Score}\left(s_{ji}, \mathrm{W}\right) - \mathrm{Min}}{\mathrm{Max} - \mathrm{Min}}. \quad (۷)$$

مقادیر Min و Max به ترتیب بر اساس رابطه‌های زیر تعریف می‌شوند:

$$\mathrm{Min} = \sum_{j=1}^{\ell} \min_{\alpha \in \Sigma} \log \frac{\mathrm{W}[\alpha, j]}{\mathrm{b}[\alpha]}, \quad (۸)$$

$$\mathrm{Max} = \sum_{j=1}^{\ell} \max_{\alpha \in \Sigma} \log \frac{\mathrm{W}[\alpha, j]}{\mathrm{b}[\alpha]}. \quad (۹)$$

زیرتوالی‌هایی به‌عنوان مصداق‌های معتبر انتخاب می‌شوند که مقدار آن‌ها با توجه به رابطه (۷) از یک حد آستانه‌ای بیش‌تر باشد. از نظر زیست‌شناسی، موتیف‌های قوی موتیف‌هایی هستند که مصداق‌های آن‌ها در موقعیت‌های بیش‌تری با هم تطابق دارند و برعکس موتیف‌های ضعیف در موقعیت‌های کم‌تری باهم تطابق دارند. حد آستانه برای موتیف‌های قوی ۰/۹[۱۰] و برای موتیف‌های ضعیف ۰/۸[۱۱] در نظر گرفته می‌شود [۲۵].

۴- ارزیابی الگوریتم SA-COAMF

۴-۱- معیار ارزیابی الگوریتم‌های پیدا کردن موتیف

برای مطالعه صحت پیش‌بینی نتایج الگوریتم‌های پیدا کردن موتیف از معیار ضریب کارایی نوکلئوتید (PC[۱۲]) که بر اساس رابطه زیر تعریف می‌شود، استفاده می‌گردد:

$$PC = \frac{TP}{TP + FP + FN} \quad (۱۰)$$

مقادیر FN[۱۳]، TP[۱۴] و FP[۱۵] بر اساس موقعیت‌های روی مجموعه توالی S که در شرایط آزمایشگاهی ثابت شده موتیف هستند و موقعیت‌هایی که توسط الگوریتم پیش‌گویی می شوند، متناظر با جدول ۴ قابل تعریف است.

جدول ۴: تعریف FN، TP و FP

FN:	تعداد موقعیت‌هایی که موتیف هستند و پیش‌گویی نشده است.
TP:	تعداد موقعیت‌هایی که صحیح پیش‌گویی می‌شوند.
FP:	تعداد موقعیت‌هایی که به‌صورت ناصحیح، به‌عنوان موتیف پیشگویی می‌شوند.

مقدار ضریب کارایی همیشه بین صفر و یک است و هرچه مقدار این معیار بیش‌تر باشد نشان‌دهنده صحت بیش‌تر پیش‌بینی نوکلئوتیدهای پیش‌گویی‌شده به‌عنوان مصداق‌های موتیف است.

۴-۲- پایگاه داده

در این مقاله برای بررسی و ارزیابی صحت و دقت عملکرد الگوریتم SA-COAMF، ۹ عامل الگوبرداری از پایگاه داده SCPD استخراج شده است. پایگاه داده SCPD توسط ژو[۱۶] و ژانگ[۱۷] در سال ۱۹۹۹ ارائه شد [۲۹]. این پایگاه داده شامل عوامل الگوبرداری و جایگاه پیوند خوردن آن‌ها در مخمر است. جزئیات مربوط به هر یک از عوامل الگوبرداری انتخاب شده در جدول ۵ مشاهده می شود. ستون اول این جدول نام ۹ عامل الگوبرداری استخراج‌شده از پایگاه داده SCPD را نشان می‌دهد. برای هر یک از عوامل الگوبرداری در ستون دوم یک توالی اجماع از مصداق‌های موتیف آزمایشگاهی نشان داده شده است. ستون سوم از این جدول تعداد توالی‌های هر عامل الگوبرداری را نشان می‌دهد به‌طوری که هر توالی نشان‌دهنده منطقه‌ای از ۸۰۰- تا ۵۰+ از ناحیه بالادست ژن‌های هم‌بیان است. ستون چهارم طول هر توالی را مشخص می کند.

۵- تحلیل و نتایج آزمایش‌ها

در این بخش ابتدا الگوریتم SA-COAMF را با الگوریتم فاخته استاندارد (COA) بر روی عوامل الگوبرداری جدول ۵ مقایسه می کنیم. در ادامه، نتایج حاصل از شبیه‌سازی و اجرای الگوریتم SA-COAMF بر روی ۹ عامل الگوبرداری معرفی‌شده در جدول۵ با الگوریتم‌های MEME [۲۴]، GA-DPAF [۲۵] و PSO+ [۲۱] مقایسه و مورد ارزیابی قرار می‌گیرد. سپس به تحلیل زمان پردازش و پیچیدگی زمانی الگوریتم پیشنهادی با دو الگوریتم تکاملی GA-DPAF و PSO+ پرداخته

www.SID.ir

Archive of SID

می‌شود و برقراری توازن بین زمان اجرا و کیفیت موتیف‌های کشف‌شده توسط الگوریتم SA-COAMF مورد بحث و بررسی قرار می‌گیرد.

تمام نتایج حاصل که در ادامه شرح داده می‌شود بر روی سیستمی با ۱۶ گیگابایت حافظه‌اصلی و پردازنده‌ای با سرعت ۳/۴ گیگاهرتز به دست آمده است.

جدول ۵: مشخصات عوامل الگوبرداری پایگاه داده **SCPD**

طول	تعداد توالی‌ها	اجماع	عامل الگوبرداری
۸۵۰	۶	-	BAS1
۸۵۰	۳	TCACGTG	CPF1
۸۵۰	۶	CWTCC	GCR1
۸۵۰	۴	-	HAP2
۸۵۰	۶	WCGCGW	MCB
۸۵۰	۷	TCCGYGGA	PDR3
۸۵۰	۳	-	PHO2
۸۵۰	۳	CNCGAAA	SCB
۸۵۰	۳	GTSAACAA	SFF

۵-۱- تحلیل نتایج **SA-COAMF** در مقابل **COA**

در این بخش SA-COAMF با الگوریتم فاخته استاندارد، COA، مقایسه می‌شود. در الگوریتم فاخته استاندارد یک فاخته سراسری وجود دارد که تمام فاخته‌ها به‌سمت آن حرکت می‌کنند. مسئله کشف موتیف با این ویژگی الگوریتم هم‌راستا نیست، زیرا در توالی‌های زیستی ممکن است چندین موتیف وجود داشته باشد که هدف الگوریتم یافتن همه آن سیگنال‌ها است. درنهایت زیست‌شناس تشخیص می‌دهد که موتیف (سیگنال) موردنظر کدام است. به‌همین دلیل، این الگوریتم به‌گونه‌ای توسعه یافته است که فاخته‌ها قادر هستند از بین چندین فاخته سراسری، فاخته موردنظر را برای مهاجرت به‌سمت آن انتخاب نمایند. بنابراین G فاخته از بهترین فاخته‌ها به‌عنوان فاخته‌های سراسری انتخاب می‌گردند.

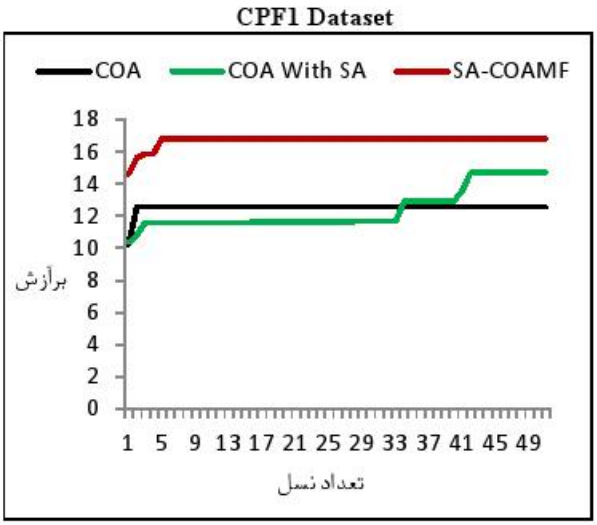
با توجه به تعریف زیستی از پروموتور حدوداً ممکن است بین ۵ تا ۱۰ سیگنال در این نوع از توالی‌ها باشد. ما در الگوریتم پیشنهادی خود، در هر نسل از اجرای الگوریتم، ۱۰ فاخته با بهترین برازش، G=۱۰، را به‌عنوان فاخته هدف انتخاب می‌کنیم. همچنین با توجه به اینکه طول هر توالی ۸۵۰ و طول موتیف‌ها بین ۱۰ تا ۱۵ است جمعیت اولیه از فاخته‌ها، به‌صورت میانگین به تعداد ۶۰ فاخته تعریف می‌شود.

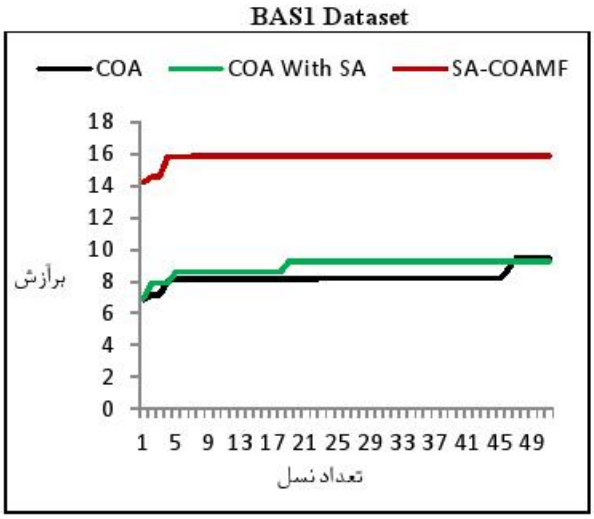
درادامه SA-COAMF بر روی عوامل الگوبرداری جدول ۵ اجرا می‌شود. سپس الگوریتم COA را که بدون تکنیک‌های سرمایش تدریجی (SA)، بیشینه‌سازی زمان انتظار (EM) و استفاده از یک فاخته هدف برای مهاجرت است نیز بر روی عوامل الگوبرداری جدول ۵ اجرا می‌گردد. در هر نسل از اجرای این دو الگوریتم، برازش بهترین موتیفی که الگوریتم توانسته است پیدا کند، به‌عنوان خروجی اعلام می‌شود.

نتایج حاصل از اجرای هر دو الگوریتم COA و SA-COAMF بر روی تمامی عوامل الگوبرداری جدول ۵ در ۵۰ نسل، در شکل ۲ قابل مشاهده است. همه الگوریتم‌ها در ۵۰ نسل اجرا شده‌اند.

با بررسی شکل ۲ می‌توان نتیجه گرفت که الگوریتم COA حتی بعد از ۵۰ نسل اجرا به جواب بهینه و یا نزدیک به بهینه نخواهد رسید و در بهینه محلی گیر می‌افتد. درحالی که ترکیب روش سرمایش تدریجی با الگوریتم فاخته استاندارد (COA With SA) کمک شایانی به الگوریتم می‌کند که از بهینه‌های محلی خارج شود. در روش سرمایش تدریجی دمای اولیه $T=100$ و ضریب کاهش دما در هر نسل $\alpha = 0/9$ درنظر گرفته می‌شود.

همان‌طور که در شکل۲ دیده می‌شود، با افزودن تابع EM به الگوریتم COA With SA (SA-COAMF) سرعت همگرایی الگوریتم به جواب بهینه اصلی رشد چشمگیری دارد. نتایج حاصل از اجرای الگوریتم SA-COAMF بر روی تمامی عوامل الگوبرداری جدول ۵ در شکل ۲ قابل مشاهده است.

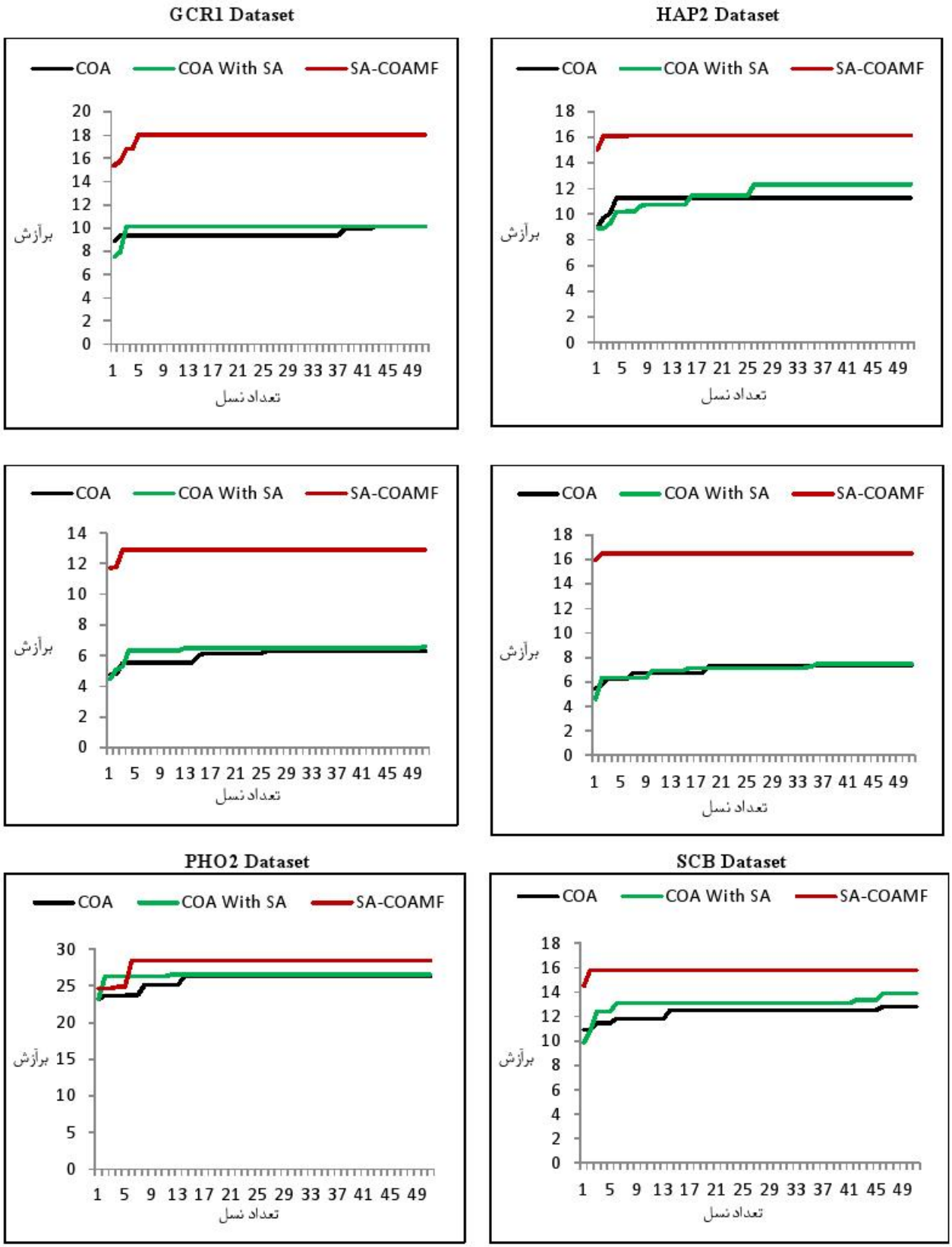




شکل ۲: نتایج اجرای سه الگوریتم **COA**، **COA With SA** و **SA-COAMF** بر روی عوامل الگوبرداری جدول ۵

www.SID.ir

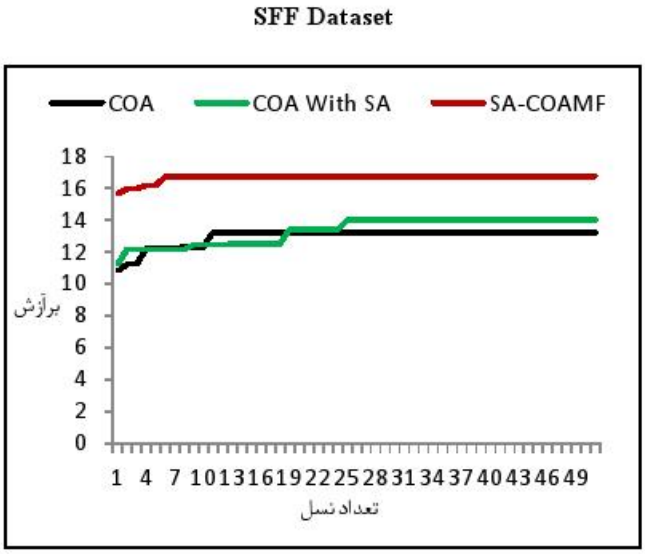
Archive of SID



ادامه شکل ۲: نتایج اجرای سه الگوریتم **COA**، **COA With SA** و **SA-COAMF** بر روی عوامل الگوبرداری جدول ۵.

www.SID.ir

Archive of SID



ادامه شکل ۲: نتایج اجرای سه الگوریتم **COA**، **COA With SA** و **SA-COAMF** بر روی عوامل الگوبرداری جدول ۵

۵-۲- بررسی صحت و دقت SA-COAMF

همان‌طور که در بخش قبل گفته شد، SA-COAMF نسبت به الگوریتم‌های COA و COA With SA دقیق‌تر عمل کرده و توانایی قابل‌ملاحظه‌ای در پیدا کردن جواب بهینه از خود نشان می‌دهد. از این‌رو در این بخش، SA-COAMF با سه الگوریتم GA-DPAF، PSO+ و MEME با توجه به فرمول ضریب کارایی بر روی عوامل الگوبرداری جدول ۵ مقایسه می شود و نتایج در جدول ۶ قابل مشاهده است (مقادیر الگوریتم MEME از [۳۰]، GA-DPAF از [۲۵] و PSO+ بر اساس پیاده‌سازی [۲۱] استخراج شده است).

الگوریتم‌ها بر روی مجموعه داده‌های زیستی جدول ۵ تست شده‌اند. الگوریتم پیشنهادی در ۱۰ نسل اجرا شده است زیرا همان‌طور که در شکل ۲ مشاهده می‌شود، الگوریتم پس از ۱۰ نسل تقریباً به شرایط همگرایی می‌رسد.

با توجه به اینکه نتایج الگوریتم MEME از [۳۰] و GA-DPAF از [۲۵] استخراج شده است و در این مقاله‌ها ذکر شده که هر الگوریتم ۳ بار روی هر داده اجرا شده است و سپس در هر اجرا، از بین ۱۰ فرد برتر، فردی که ضریب کارایی بالاتری دارد (PC) به‌عنوان موتیف اصلی برگزیده می‌شود، بنابراین الگوریتم SA-COAMF با همین شرایط اجرا می‌گردد و نتایج در جدول ۶ قابل مشاهده است.

با بررسی نتایج اعلام شده در جدول ۶، می‌توان به برتری الگوریتم پیشنهادی نسبت به سه الگوریتم دیگر پی برد، زیرا SA-COAMF بهتر و دقیق‌تر از دیگر الگوریتم‌ها در کشف موتیف عمل می‌کند. همان‌طور که پیش‌تر اشاره شد، دلیل برتری SA-COAMF نسبت به الگوریتم‌های GA-DPAF، PSO+ و MEME به‌علت بهره بردن از فواید الگوریتم فاخته جهت افزایش سرعت همگرایی، روش بیشینه‌سازی زمان انتظار در پیداکردن بهینه‌های محلی برای هر فاخته، روش سرمایش تدریجی در اجتناب کردن از گیرافتادن در بهینه‌های محلی است. یکی دیگر از ویژگی‌های بارز این الگوریتم در نظر گرفتن G تا فاخته هدف است درحالی‌که الگوریتم‌های GA-DPAF و PSO+ به‌سمت یک سیگنال (موتیف) حرکت می‌کنند و در مواردی ممکن است بهترین سیگنالی که پیدا می‌کنند موتیف موردنظر زیست‌شناس نباشد. همچنین در الگوریتم پیشنهادی با استفاده از روش EM در هر لحظه هر فاخته به بهترین خودش (بهینه محلی) نزدیک می‌شود، درصورتی‌که دو الگوریتم GA-DPAF و PSO+ چنین کاری را انجام نمی‌دهند.

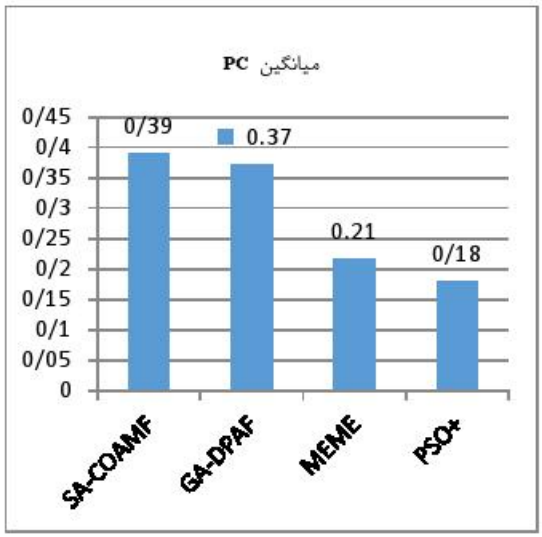
در شکل ۳، میانگین نتایج مقادیر موجود در جدول ۶ بر روی پایگاه داده SCPD قابل مشاهده است. در این شکل می‌توان به صحت و دقت SA-COAMF نسبت به سه الگوریتم MEME، PSO+ و GA-DPAF پی برد.

جدول ۶: نتایج الگوریتم‌های **SA-COAMF**، **GA-DPAF**، **MEME** و **PSO+** بر روی عوامل الگوبرداری پایگاه داده **SCPD**

عوامل الگوبرداری	MEME	PSO+	GA-DPAF	SA-COAMF
BAS1	۰/۳۰	۰/۱۱	۰/۰۸	۰/۱۷
CPF1	۰/۴۹	۰/۲۷	۰/۷۴	۰/۷۴
GCR1	۰/۲۰	۰/۱۴	۰/۲۲	۰/۲۶
HAP2	۰/۰۰	۰/۰۵	۰/۱۴	۰/۰۹
MCB	۰/۱۵	۰/۲۴	۰/۶۶	۰/۶۶
PDR3	۰/۴۳	۰/۷۵	۰/۷۳	۰/۸۰
PHO2	۰/۰۰	۰/۰۴	۰/۰۰	۰/۱۵
SCB	۰/۶۱	۰/۰۱	۰/۵۲	۰/۳۸
SSF	۰/۰۳	۰/۰۱	۰/۲۵	۰/۲۶

www.SID.ir

Archive of SID



شکل ۳: میانگین نتایج چهار الگوریتم **SA-COAMF**، **GA-DPAF**، **MEME** و **PSO+** بر روی عوامل الگوبرداری پایگاه داده **SCPD**

۵-۳- تحلیل زمان پردازش الگوریتم پیشنهادی

به دلیل اینکه الگوریتم پیشنهادی جزء دسته الگوریتم‌های تکاملی به حساب می‌آید، سرعت پردازش و پیچیدگی زمانی SA-COAMF با دو الگوریتم تکاملی GA-DPAF و PSO+ که جزء الگوریتم‌های تکاملی هستند، در شرایط یکسان مورد مقایسه و بررسی قرار می‌گیرد. الگوریتم‌ها را به زبان برنامه‌نویسی پرل پیاده‌سازی کردیم.

حداکثر تعداد افراد برای هر سه الگوریتم بر اساس $\frac{\sum_{i=1}^{t} n_i}{(\ell \times t)}$ تعریف می‌شود به‌طوری که ℓ طول موتیف در هر عامل الگوبرداری از جدول ۵ است. بیش‌ترین تعداد نسل تعریف‌شده در این شبیه‌سازی برای هر سه الگوریتم ۱۰ در نظر گرفته شده است که اگر در ۱۰ نسل جواب بهینه عوض نگردد الگوریتم متوقف می‌شود، در غیراین‌صورت مادامی که الگوریتم می‌تواند جواب بهینه‌ای پیدا کند به کار خود ادامه می‌دهد.

پیچیدگی زمانی سه الگوریتم SA-COAMF، GA-DPAF و PSO+ در جدول ۷ بیان شده است.

جدول ۷: پیچیدگی زمانی سه الگوریتم **SA-COAMF**، **GA-DPAF** و **PSO+**.

	PSO+	GA-DPAF	SA-COAMF
پیچیدگی زمانی	$O(nt^2\ell)$	$O(n^2t^2d\ell)$	$O(nt^2\ell\log_2\ell)$

*در الگوریتم GA-DPAF پارامتر d تعداد گپ را مشخص می‌کند.

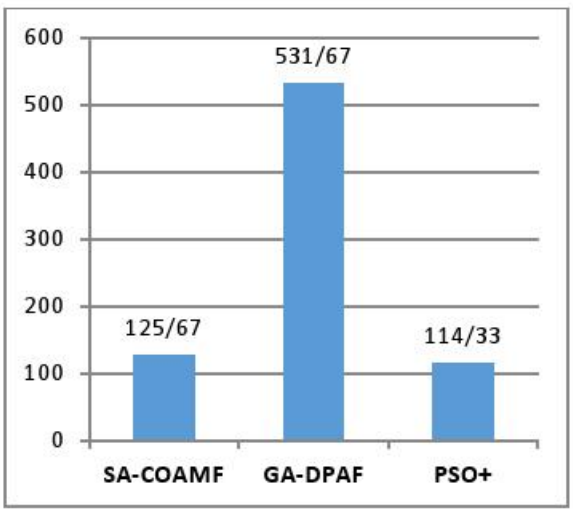
جدول ۸ زمان پردازش سه الگوریتم SA-COAMF، GA-DPAF و PSO+ را بر روی تمامی عوامل الگوبرداری جدول ۵ نشان می‌دهد.

جدول ۸: زمان پردازش سه الگوریتم **SA-COAMF**، **GA-DPAF** و **PSO+** برحسب میلی‌ثانیه

عوامل الگوبرداری	PSO+	GA-DPAF	SA-COAMF
BAS1	۱۷۵	۱۴۷۰	۲۰۳
CPF1	۵۲	۲۳۰	۶۶
GCR1	۱۳۵	۶۰۹	۱۶۹
HAP2	۹۰	۸۳۲	۱۳۵
MCB	۱۷۵	۹۹۴	۱۶۴
PDR3	۱۹۱	۳۴۷	۱۷۶
PHO2	۶۴	۱۷۶	۳۵
SCB	۸۸	۶۵	۸۲
SSF	۵۹	۶۲	۱۰۱

با توجه به نتایج اعلام‌شده در جدول ۶، الگوریتم SA-COAMF در بیش‌تر موارد دارای دقت بالاتری نسبت به الگوریتم GA-DPAF است در حالی که با توجه به نتایج جدول ۸ در تمام موارد زمان پردازش الگوریتم پیشنهادی از الگوریتم GA-DPAF کم‌تر است و دارای سرعت بالاتری نسبت به GA-DPAF است. ولی با توجه به جدول ۸، الگوریتم PSO+ در برخی موارد سرعت پردازش پایین‌تری نسبت به الگوریتم SA-COAMF دارد که در مقابل با توجه به جدول ۶ در تمامی موارد دارای دقت پایین‌تری است. بنابراین، الگوریتم پیشنهادی با کاهش زمان نسبت به GA-DPAF و میزانی افزایش نسبت به PSO+ قادر به کشف جواب‌های بهینه‌تری است.

در شکل ۴، میانگین زمان اجرای هر الگوریتم بر روی تمامی عوامل الگوبرداری قابل مشاهده است. با مقایسه شکل‌های ۳ و ۴ می‌توان پی برد که SA-COAMF در زمانی معقول نسبت به دو الگوریتم PSO+ و GA-DPAF به جواب‌های بهینه دسترسی پیدا می‌کند.



شکل ۴: میانگین زمان پردازش سه الگوریتم **SA-COAMF**، **GA-DPAF** و **PSO+** بر روی عوامل الگوبرداری پایگاه داده **SCPD**.

www.SID.ir

Archive of SID

۶- نتیجه‌گیری

یک مجموعه از زیرتوالی‌های حفاظت‌شده در توالی‌های زیستی را موتیف می‌گویند به‌طوری‌که این زیرتوالی‌ها دارای ساختار و عملکردهای مشابهی هستند. غالباً به دلیل نقش مهم موتیف‌ها در تعیین ساختار و عملکرد یک مولکول، مسئله کشف موتیف یکی از مسئله‌های مهم و با اهمیت در زمینه زیست‌شناسی مولکولی است. در مسئله کشف موتیف هیچ اطلاعاتی درباره موتیف و مصداق‌های آن وجود ندارد و تنها تعدادی توالی زیستی داده می‌شود و هدف یافتن موتیف ناشناخته با رخدادهای تقریبی در موقعیت‌های ناشناخته است. در این مقاله الگوریتم بهینه‌سازی فاخته به‌عنوان روشی برای حل مسئله کشف موتیف پیشنهاد شده است که این الگوریتم از سبک زندگی یک نوع پرنده بنام فاخته الهام گرفته است. یکی از مزایای الگوریتم بهینه‌سازی فاخته سرعت بالای آن در همگرایی به جواب بهینه است. همچنین در روش پیشنهادی از تکنیک‌های مختلفی جهت گریز از بهینه‌های محلی و رسیدن به جواب بهینه سراسری استفاده شده است. برای کارهای آتی می‌توان این الگوریتم را برای کشف موتیف‌های گپ‌دار توسعه داد.

مراجع

[1] B. Brejová, C. DiMarco, T. Vinar, S. R. Hidalgo, G. Holguin, and C. Patten, *Finding patterns in biological sequences*, Unpublished project report for CS798G, University of Waterloo, Fall, 2000.

[2] http://www.hindawi.com/journals/ijad/2011/154325/. Available: http://www.hindawi.com/journals/ijad/2011/154325/

[3] L. Shao, Y. Chen and A. Abraham, "Motif discovery using evolutionary algorithms," in *Soft Computing and Pattern Recognition, (SOCPAR'09)*, pp. 420-425, 2009.

[4] C. Reyes-Rico, "Finding DNA motifs using genetic algorithms," in *Fifth Mexican International Conference on Artificial Intelligence*, pp. 331-339, 2006.

[5] C. E. Lawrence, S. F. Altschul, M. S. Boguski, J. S. Liu, A. F. Neuwald and J .C. Wootton, "Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment," *Science*, vol. 262, no. 5131, pp. 208-214, 1993.

[6] J. van Helden, B. André and J. Collado-Vides, "Extracting regulatory sites from the upstream region of yeast genes by computational analysis of oligonucleotide frequencies," *Journal of molecular biology*, vol. 281, no. 4, pp. 827-842, 1998.

[7] M. Tompa, "An exact method for finding short motifs in sequences, with application to the ribosome binding site problem," *ISMB*, pp. 262-271, 1999.

[8] T. F. Smith and M. S. Waterman, "Identification of common molecular subsequences," *Journal of molecular biology*, vol. 147, no. 1, pp. 195-197, 1981.

[9] J. Kevin Lanctot, M. Li, B. Ma, S. Wang and L. Zhang, "Distinguishing string selection problems," *Information and Computation*, vol. 185, no. 1, pp. 41-55, 2003.

[10] G. Pavesi, G. Mauri and G. Pesole, "An algorithm for finding signals of unknown length in DNA sequences," *Bioinformatics*, vol. 17, no. 1, pp. S207-S214, 2001.

[11] S. Sinha and M. Tompa, "YMF: a program for discovery of novel transcription factor binding sites by statistical overrepresentation," *Nucleic acids research*, vol. 31, no. 13, pp. 3586-3588, 2003.

[12] U. Keich and P. A. Pevzner, "Finding motifs in the twilight zone," in *Proceedings of the sixth annual international conference on Computational biology*, pp. 195-204, 2002.

[13] J. Buhler and M. Tompa, "Finding motifs using random projections," *Journal of computational biology*, vol. 9, no. 2, pp. 225-242, 2002.

[14] C. E. Lawrence and A. A. Reilly, "An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences," *Proteins: Structure, Function, and Bioinformatics*, vol. 7, no. 1, pp. 41-51, 1990.

[15] T. L. Bailey and C. Elkan, "Unsupervised learning of multiple motifs in biopolymers using expectation maximization," *Machine learning*, vol. 21, no. 1, pp. 51-80, 1995.

[16] J. D. Hughes, P. W. Estep, S. Tavazoie and G. M. Church, "Computational identification of Cis-regulatory elements associated with groups of functionally related genes in Saccharomyces cerevisiae," *Journal of Molecular Biology*, vol. 296, no. 5, pp. 1205-1214, 2000.

[17] X. Liu, D. L. Brutlag and J. S. Liu, "BioProspector: discovering conserved DNA motifs in upstream regulatory regions of co-expressed genes," in *Pacific symposium on biocomputing*, 2001, pp. 127-138.

[18] L. Shao and Y. Chen, "Bacterial foraging optimization algorithm integrating tabu search for motif discovery," in *Bioinformatics and Biomedicine*, pp. 415-418, 2009.

[19] D. L. González-Álvarez, M. A. Vega-Rodríguez, J. A. Gómez-Pulido and J. M. Sánchez-Pérez, "Finding motifs in DNA sequences applying a multiobjective artificial bee colony (MOABC) algorithm," in *Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*, pp. 89-100, 2011.

[20] S. Bouamama, A. Boukerram and A. F. Al-Badarneh, "Motif finding using ant colony optimization," in *Swarm Intelligence*, pp. 464-471, 2010.

[21] C. Lei and J. Ruan, "A particle swarm optimization-based algorithm for finding gapped motifs," *BioData mining*, vol. 3, no. 9, pp. 1-12, 2010.

[22] F. Zare-Mirakabad, H. Ahrabian, M. Sadeghi, J. Mohammadzadeh, S. Hashemifar, A. Nowzari-Dalini, et al., "PSOMF: An algorithm for pattern discovery using PSO," in *Proceedings of the Third IAPR International Conferences on Pattern Recognition in Bioinformatics*, pp. 61-72, 2008.

[23] J. M. Keith, P. Adams, D. Bryant, D. P .Kroese, K. R. Mitchelson, D. A. Cochran, et al., "A simulated annealing

www.SID.ir

Archive of SID

algorithm for finding consensus sequences," *Bioinformatics*, vol. 18, no. 11, pp. 1494-1499, 2002.

[24] T. L. Bailey and C. Elkan, "The value of prior knowledge in discovering motifs with MEME," *ISMB*, pp. 21-29, 1995.

[25] F. Zare-Mirakabad, H. Ahrabian, M. Sadeghi, S. Hashemifar, A. Nowzari-Dalini and B. Goliaei, "Genetic algorithm for dyad pattern finding in DNA sequences," *Genes & Genetic Systems*, vol. 84, no. 1, pp. 81-93, 2009.

[26] W. Liu ,H. Chen and L. Chen, "An ant colony optimization based algorithm for identifying gene regulatory elements," *Computers in Biology and Medicine*, vol. 43, no. 7, pp. 922-932, 2013.

[27] R. Rajabioun, "Cuckoo optimization algorithm," *Applied Soft Computing*, vol. 11, no. 8, pp. 5508-5518, 2011.

[28] M. Sarkar, B. Yegnanarayana and D. Khemani, *"A clustering algorithm using an evolutionary programming-based approach," Pattern Recognition Letters*, vol. 18, no. 10, pp. 975-986, 1997.

[29] J. Zhu and M. Q. Zhang, "SCPD: a promoter database of the yeast Saccharomyces cerevisiae," *Bioinformatics*, vol. 15, no. 7-8, pp. 607-611, 1999.

[30] S. Sinha and M. Tompa, "Performance comparison of algorithms for finding transcription factor binding sites," in *Bioinformatics and Bioengineering*, pp. 214-220, 2003.

زیرنویس‌ها

[1] Transcription Factor
[2] Consensus Sequence
[3] Position Frequency Matrix
[4] Cuckoo Optimization Algorithm
[5] Simulated Annealing
[6] Expected Maximization
[7] Simulated Annealing–Cuckoo Optimization Algorithm Motif Finding
[8] Information Content
[9] Egg Laying Radius
[10] Strong Motifs
[11] Weak Motifs
[12] Performance Coefficient
[13] False negative
[14] True Positive
[15] False Positive
[16] Zhu
[17] Zhung

www.SID.ir