**تولید جدول کلمات متقاطع فارسی با استفاده از الگوریتم ژنتیک**

**\*میلاد چراغی1**

**دکتر جواد سلیمی2**

1. **دانشجو رشته نرم‌افزار دانشگاه دولتی کاشان** [miladchraghi@gmail.com](mailto:miladchraghi@gmail.com)
2. **استاد رشته‌نرم افزار دانشگاه کاشان** salimi.sartakhti@gmail.com

چکیده

مسائل ان پی سخت (که در زمان چند جمله‌ای حل نمی‌شوند) مانند تولید جدول متقاطع کلمات همواره کاندیدای مناسبی برای الگوریتم‌های ژنتیک بوده‌اند. در این مقاله نتایج حاصل از پیاده‌سازی این الگوریتم و تست و آزمایش آن برای سه نوع مختلف از جدول کلمات متقاطع، برای کلمات فارسی بررسی شده‌است. همچنین در این کار برای جایگذاری کلمات از فرهنگ لغات دهخدا استفاده شده‌است. نتایج حاصل برای جدول کلمات متقاطع از نوع آمریکایی و انگلیسی و ژاپنی بررسی شده‌است و نشان داده می‌شود که تعداد متوسط لغات صحیح پیشنهاد شده پیشرفت حدوداً 20 درصدی داشته است.

**کلمات کليدي: الگوریتم ژنتیک، جدول کلمات متقاطع**

**1-مقدمه**

الگوریتم ژنتیک یکی از الگوریتم‌های مهم و پرکاربرد در زمینه علوم کامپیوتر می باشد، مخصوصا برای مسائلی که پیچیدگی زمانی زیادی دارند. این روش که شامل مراحل متعددی است، کار خود را با یک جمعیت اولیه شروع می‌کند و در هر بار تکرار با توجه به شایستگی نمونه‌ها، به انتخاب آن‌ها برای تولید فرزندان می‌پردازد. این فرزندان به جمعیت فعلی افزوده می‌شوند و مجددا با توجه به معیار شایستگی، نمونه‌هایی برای انتقال به نسل بعدی انتخاب می‌شوند.

در الگوریتم‌های ژنتیک، کلمه جمعیت به تعدادی از پاسخ‌های بالقوه مسئله اشاره دارد که با توجه به روش بازنمایی پاسخ، می تواند فرم و اشکال مختلفی داشته باشد. زاد و ولد جمعیت نیز اشاره به تکثیر ژن ها و بهره‌برداری از آنان اشاره دارد. جهش نیز تضمین کننده کاوش و گوناگونی در جمعیت است. همچنین کارایی الگوریتم ژنتیک به تنظیم پارامتر‌هایی از آن بستگی دارد، نظیر تغیر احتمال جهش، تغیر احتمال زاد و ولد و همچنین تغیر سایز جمیعت.

تولید جدول کلمات متقاطع، یک مسئله از نوع ان پی سخت است که حل آن شامل پر نمودن یک اسکلت خالی با کلمات معنی‌داری است که در دیکشنری کلمات موجود است. در این کار، به عنوان دیکشنری از کلمات داخل فرهنگ لغات دهخدا استفاده شده است که فایل آن در اینترنت موجود و قابل دریافت هست.

در بخش دوم به بررسی کار‌های گذشته می‌پردازیم. بخش سوم به توضیح روش کار و الگوریتم آن می‌پردازد، در بخش چهارم به ارزیابی مدل با توجه به انواع جدول کلمات متقاطع پرداخته شده‌ است. بخش پنجم و پایانی نیز به نتیجه‌گیری و مرز‌های تحقیق می‌پردازد. بخش ششم نیز به مراجع مورد استفاده بیان شده‌اند.

**2- تحقیق‌های پیشین**

تولید و ایجاد جدول کلمات متقاطع با ابعاد بیشتر از 4 در 4 با استفاده از روش‌هایی نظیر جستجوی فراگیر[[1]](#footnote-1) و جستجو اول عمق به دلیل پیچیدگی زمانی تقریبا غیر‌ممکن می باشد ]1[. همچنین به این مسئله در بسیاری از تحقیق ها به عنوان یک مسئله خوب ارضای محدودیت نگاه شده‌است و طی آن راه حلی ارائه شده‌است ]2[.

محققین در این کار ]3[ برای حل پازل استفاده از روش‌های کلاسیک هوش مصنوعی را پیشنهاد کرده‌اند. همچنین استفاده از روش‌هایی نظیر استفاده از خرد جمعی[[2]](#footnote-2) در کنار روش‌های دیگر نظیر الگوریتم ژنتیک برای حل این نوع پازل ها مورد استفاده قرار گرفته است. آقای یامپولیسکی[[3]](#footnote-3) از این روش برای بهبود مسائل ان پی سخت در علوم کامپیوتر استفاده کرده است ]4[. همچنین آقای بونومو[[4]](#footnote-4) از روش ترکیب الگوریتم‌های ژنتیک و خرد جمعی برای تولید جدول کلمات متقاطع استفاده کرده‌اند ]5[، در این کار نشان داده شده‌ بود که برای جدول کلمات از نوع آمریکایی و انگلیسی و ژاپنی حداکثر به ترتیب 35، 10 و 30 کلمه درست و با معنی جایگزاری شده بود.

در این کار و در بخش چهارم نشان داده خواهد شد که با استفاده از الگوریتم ژنتیک می توان برای همان جدول کلمات به طور متوسط تعداد 50 کلمه صحیح و با معنی یافت.

**3- ایجاد جدول کلمات متقاطع با الگوریتم ژنتیک**

این کار بر اساس مراحل زیر انجام شده است:

مقداردهی اولیه: در این مرحله یک جمعیت اولیه به سایز N ایجاد می‌شود. نمونه‌هایی که در این مرحله ایجاد می‌شوند کاملا تصادفی هستند.

زاد و ولد: در این مرحله بهترین نمونه‌ها بر اساس مقدار شایستگی آنها انتخاب می‌شوند و دو‌به‌دو با یکدیگر آمیزش داده می‌شوند و به ازای هر جفت والدین از جمعیت فعلی، دو فرزند به جمعیت افزوده می‌شود.

جهش: به احتمال C% هر کدام از فرزندان تولید شده امکان جهش دارند.

پیرایش جمعیت: در این بخش مقدار شایستگی هر نمونه محاسبه می‌شود و در انتخاب بازماندگان برای نسل بعدی استفاده می‌شود.

شرط پایانی الگوریتم: در این بخش ضمن بررسی شروطی تعین می‌شود که آیا تولید نسل و ادامه الگوریتم انجام شود یا خیر.

در زیر به توضیح هر کدام از این بخش ها می‌پردازیم:

**1-3- مقدار**‌**دهی اولیه**

جدول کلمات متقاطع می‌توانند سایز‌های مختلفی داشته باشند. برای شروع کار یک اسکلت خام و بدون کاراکتر خاصی از یک فایل متنی خوانده می‌شود. فرمت این اسکلت به صورتی است که تعداد سطر‌ها و ستون‌های آن نشان‌دهنده تعداد سطر‌ها و ستون‌های جدول کلمات هستند، همچنین این اسکلت از تعدادی 0 و 1 تشکیل شده است. 0 ها نشان دهنده خانه های سیاه و 1 ها نشان دهنده مکان‌های قرار گرفتن کاراکتر‌ها می‌باشند.

در مرحله بعدی، اسکلت خوانده شده از فایل متنی به یک آرایه دوبعدی تبدیل می‌شود و به غیر از خانه‌هایی که 0 در آن‌ها قرار دارد، در بقیه مکان‌ها یک کاراکتر فارسی به صورت تصادفی وارد می‌شود. سپس این کار برای تمام افراد جمعیت انجام می‌شود. سایز جمعیت اولیه در این قسمت تعین شده است، می توان برای آن مقادیر مختلفی را در نظر گرفت که در این تحقیق مقدار آن برابر با 50 در نظر گرفته شده‌است.

**2-3- زاد**‌**و**‌**ولد**

در این بخش به ترتیب نزولی و بر اساس میزان شایستگی، نمونه‌ها به صورت جفت انتخاب می‌شوند و به تابع زاد و ولد فرستاده می‌شوند. در این تابع یکی از نمونه‌ها به عنوان پدر و دیگری به عنوان مادر نام‌گذاری می‌شوند و سپس هر کدام آن‌ها به صورت ستونی و سطری پیمایش می‌شوند.

در این پیمایش تمام لغات جدول کلمات بررسی می‌شوند و در صورتی که در دیکشنری کلمات، وجود داشته باشند، به عنوان یک لغت با معنی در نظر گرفته می‌شوند و تضمین می‌شود که این لغات به فرزندان منتقل شود. این کار باعث می‌شود که فرزندان دارای شایستگی برابر یا بیشتر از والدین خود داشته باشند و الگوریتم از حالت جستجوی تصادفی خارج شود و به سمت فضای پاسخ بهینه همگرا شود.

نکته دیگر در مورد زاد و ولد جفت نمونه‌ها این است که بدنه مادر در فرزند اول کپی می‌شود و بدنه پدر در فرزند دوم کپی می‌شود. سپس در مرحله بعدی تمام لغات با معنی و صحیح افقی یا عمودی از مادر در فرزند دوم و همچنین تمام لغات با معنی و صحیح افقی یا عمودی از پدر به فرزند اول منتقل می‌شوند.

**3-3- جهش**

احتمال اینکه در فرزندان تولید شده در مرحله قبل جهش صورت گیرد برابر با 30% در نظر گرفته شده است. همچنین مقدار زیاد جهش می‌تواند به همگرایی زودرس منجر شود. مشاهده می‌شود که برای جدول کلمات با سایز بزرگ نظیر 15 در 15، چنانچه الگوریتم تا 200 نسل ادامه پیدا کند و به پاسخ صحیح نرسیم، در بهینه های محلی گیر می‌کنیم. می توان احتمال جهش را در این موارد افزایش داد تا الگوریتم سریع‌تر بتواند از این بهینه‌های محلی خارج شود.

همچنین جهش در فرزندان به گونه‌ای انجام می‌شود که به صورت تصادفی یکی از سطر‌ها و یا یکی از ستون‌ها انتخاب می‌شود. سپس یکی از کلمات داخل آن سطر یا ستون مجددا انتخاب شده و طول آن مورد بررسی قرار می‌گیرد. در ادامه به دیکشنری مراجعه می‌شود و باز به صورت تصادفی کلمه‌ای انتخاب می‌شود و طول آن بررسی شده و در صورتی که طول کلمه داخل دیکشنری و طول کلمه داخل جدول با یکدیگر مساوی بودند، جایگذاری این کلمات انجام می‌شود. در غیر این صورت حروفی تصادفی به جای آن کلمه در جدول جایگذاری می‌شود.

**4-3- پیرایش جمعیت**

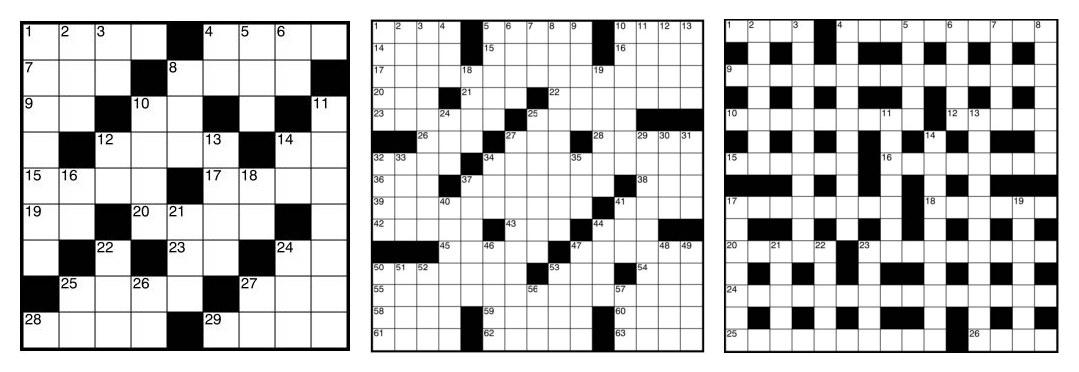
این بخش به انتخاب بازماندگان می‌پردازد. لازم به ذکر است که فرزندان و همچنین والدین در هر مرحله با یکدیگر ادغام می‌شوند و جمعیت نهایی را ایجاد می‌کنند. سپس در پایان همه نمونه ها بر اساس مقدار شایستگی که دارند به صورت نزولی مرتب می‌شوند و نمونه‌هایی با کم‌ترین مقدار شایستگی از جمعیت حذف می‌شوند. حذف نمونه‌ها تا جایی انجام می‌شود که سایز جمعیت به مقدار 100 نمونه برسد.

**5-3- شرط پایانی**

این الگوریتم تا تعداد 90 نسل را تولید می‌نماید و پس از رسیدن در یک حلقه به 90 امین نسل متوقف می‌شود. در این حین اگر پاسخ کاملی یافت شود نیز، الگوریتم باز می‌ایستد.

**4- ارزیابی نتایج**

برای ارزیابی نتیجه بدست آمده در کار‌های گذشته همواره از سه نوع جدول کلمات متقاطع استفاده شده است. نوع آمریکایی جدول کلمات و نوع انگلیسی آن و نوع ژاپنی. این جدول کلمات رایج‌ترین جدول کلمات هستند که اصلی‌ترین تفاوت آن‌ها در تعداد لغاتی است که در یک سطر یا ستون جای می‌گیرند و با یکدیگر همپوشانی دارند تا لغات با معنی را بسازند. شکل 1 نمایانگر انواع جدول کلمات متقاطع می‌باشند.

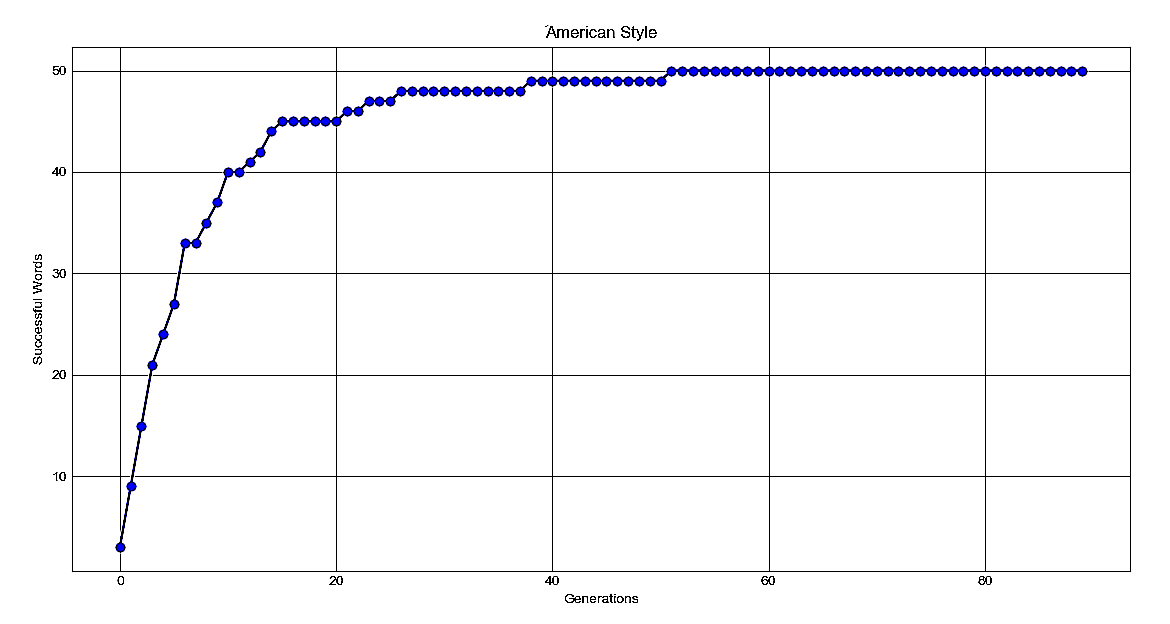


**شکل 1: شکل سمت راست جدول کلمات نوع انگلیسی را نشان می**‌**دهد و جدول وسط نوع آمریکایی آن است و جدول سمت چپ نیز از نوع ژاپنی است.**

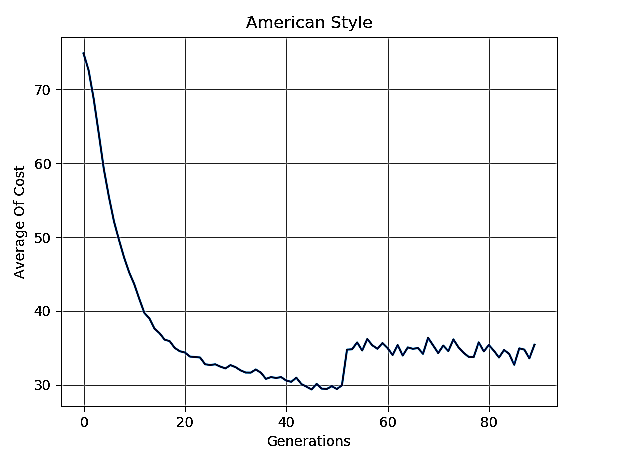
برای سه نوع جدول کلمات مطرح شده در بالا این الگوریتم اجرا شده است و نتایج را با استفاده از نمودار‌‌هایی که یکی از محور‌های آن تعداد نسل‌های تولید شده توسط الگوریتم ژنتیک است و محور دیگر مربوط به تعداد لغات درست تشخیص داده شده است، نمایش می‌دهیم، همچنین منظور از محور متوسط هزینه همان تعداد لغات اشتباهی است که به صورتی سطری یا ستونی در جدول وجود دارد. (به ازای هر لغت اشتباه، هزینه یک واحد افزایش می‌یابد و سپس هزینه محاسبه شده برای تمامی نمونه‌ها، میانگین گرفته می‌شود.)

اولین جدول کلمات، جدول کلمات آمریکایی مورد بررسی قرار می‌گیرد. در شکل 2 تعداد لغاتی که به درستی تشخیص و جایگذاری شده‌اند آورده شده است. همچنین در شکل 3 متوسط مقدار هزینه (عکس مقدار شایستگی) طی نسل‌های مختلف ترسیم شده است.

تعداد کل لغات معنی‌داری که در این جدول جای میگیرد مقدار 76 است که در کار‌های گذشته موفق به پیش بینی 32 لغت با استفاده از الگوریتم ژنتیک و روش خرد جمعی، شده بودند. در این تحقیق تعداد 50 لغت به درستی پیش‌بینی و جایگذاری شده اند.



**شکل 2: تعداد 90 نسل و تعداد لغات با معنی و صحیح پیش بینی شده با استفاده از الگوریتم ژنتیک برای جدول کلمات آمریکایی**

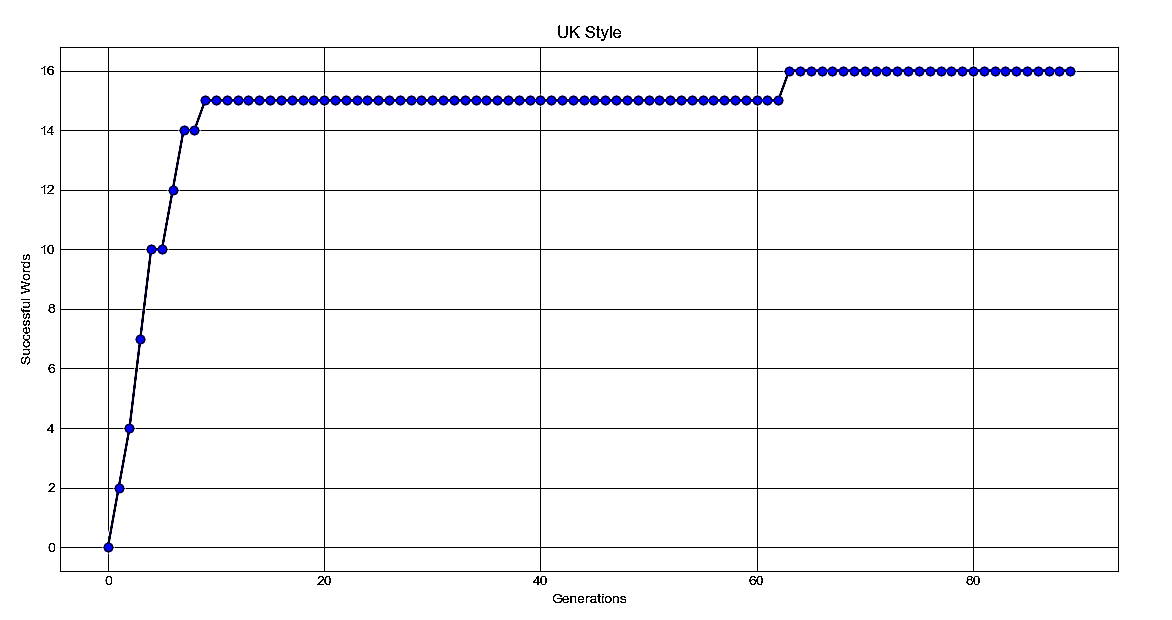


**شکل 3: میانگین هزینه (عکس شایستگی) نمونه**‌**ها طی 90 نسل تکرار الگوریتم برای جدول کلمات آمریکایی**

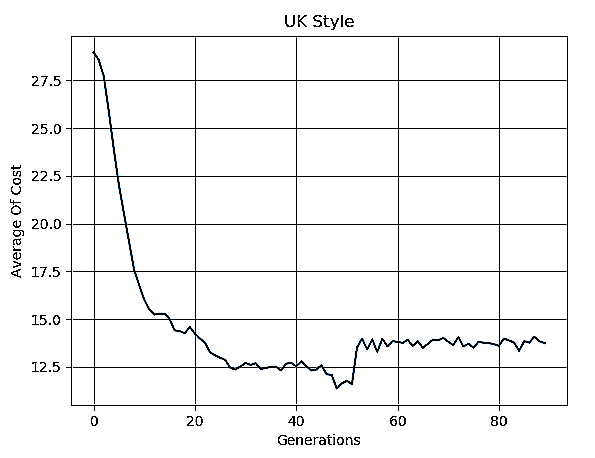
نمودار‌های رسم شده برای جدول کلمات آمریکایی به ترتیب برای جدول کلمات انگلیسی و همچنین ژاپنی در شکل های 4 الی 7 ترسیم شده است.

تعداد لغاتی که در جدول کلمات انگلیسی جای می‌گیرد مقدار 29 است، در کار‌های گذشته تنها موفق به پیش‌بینی 8 لغت از این مقدار شده بودند، در حالی که در کار فعلی با استفاده از الگوریتم ژنتیک تعداد 18 لغت به درستی پیش‌بینی و جایگذاری شده است.

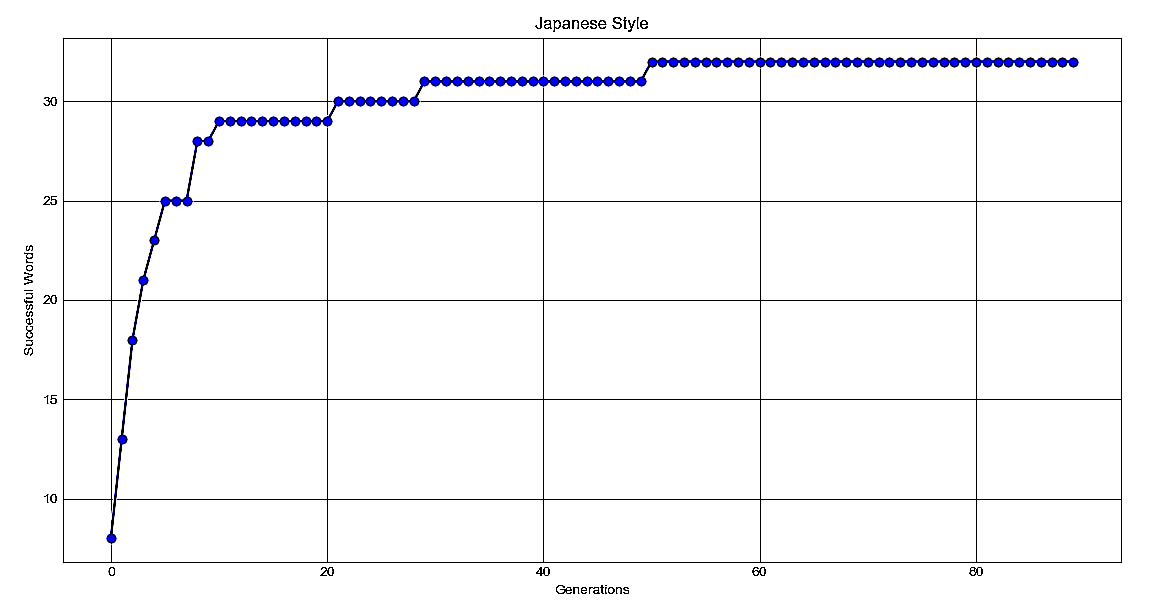
همچنین جدول کلمات ژاپنی نیز تعداد 38 کلمه افقی یا عمودی را در خود جای می‌دهد که در کار‌های گذشته موفق به پیش‌بینی تعداد 26 لغت آن شده بودند در حالی که در تحقیق فعلی موفق به پیش‌بینی 32 کلمه از آن شده‌ایم.



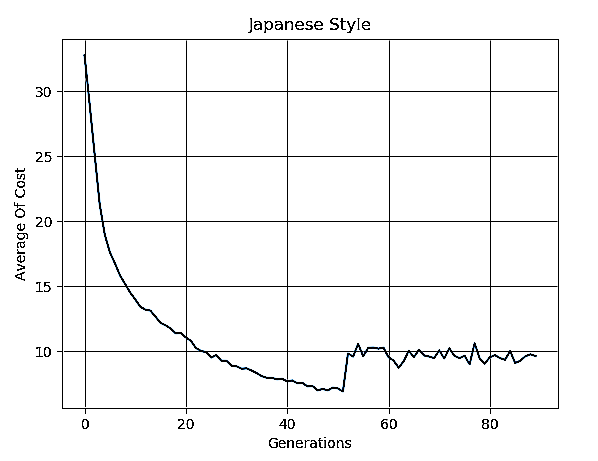
**شکل 4: تعداد 90 نسل و تعداد لغات با معنی و صحیح پیش بینی شده با استفاده از الگوریتم ژنتیک برای جدول کلمات انگلیسی**



**شکل 5: میانگین هزینه (عکس شایستگی) نمونه ها طی 90 نسل تکرار الگوریتم برای جدول کلمات انگلیسی**

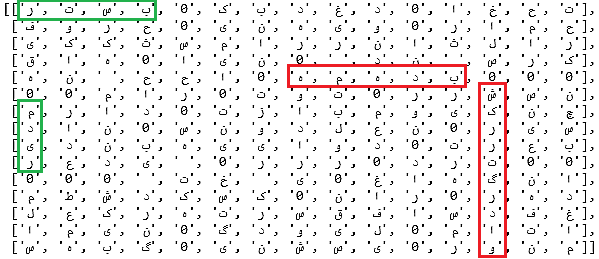


**شکل 6: عداد 90 نسل و تعداد لغات با معنی و صحیح پیش**‌**بینی شده با استفاده از الگوریتم ژنتیک برای جدول کلمات ژاپنی**

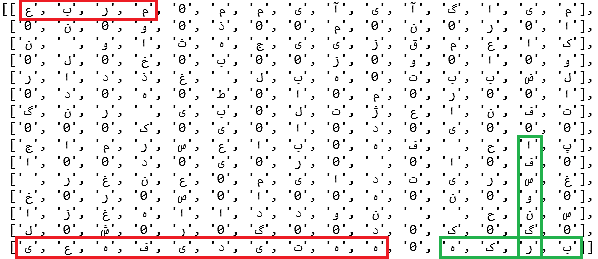


**شکل 7: میانگین هزینه (عکس شایستگی) نمونه ها طی 90 نسل تکرار الگوریتم برای جدول کلمات ژاپنی**

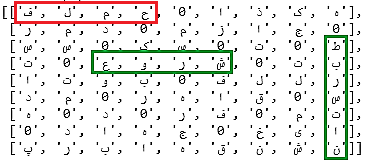
در انتها جدول کلمات بدست آمده برای سه فرمت متداول ذکر شده، در شکل‌های 8 الی 10 آورده شده است.



**شکل 8: جدول کلمات آمریکایی بدست آمده، مستطیل**‌**های سبز رنگ نشان**‌**دهنده کلمات صحیح هستند و مستطیل**‌**های قرمز رنگ نشان**‌**دهنده کلمات اشتباه می**‌**باشند.**

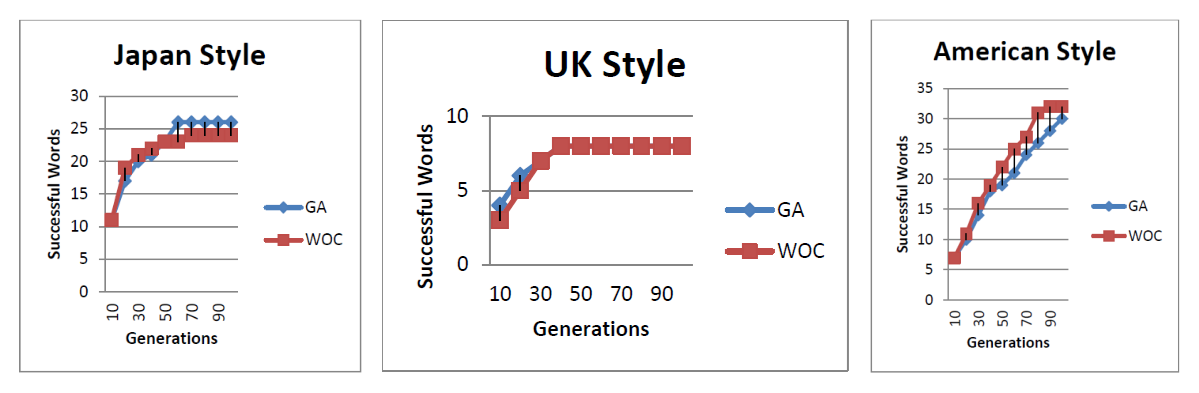


**شکل 9: جدول کلمات انگلیسی بدست آمده، مستطیل**‌**های سبز رنگ نشان دهنده کلمات صحیح هستند و مستطیل های قرمز رنگ نشان دهنده کلمات اشتباه می**‌**باشند.**



**شکل 10: جدول کلمات آمریکایی بدست آمده، مستطیل های سبز رنگ نشان دهنده کلمات صحیح هستند و مستطیل های قرمز رنگ نشان دهنده کلمات اشتباه می**‌**باشند.**

همچنین در شکل 11 به نمایش عملکرد الگوریتم های مورد استفاده در کار های گذشته اشاره شده است.[5]



**شکل 11: عملکرد کار**‌**های گذشته انجام شده بر روی تولید جدول کلمات متقاطع، منحنی آبی رنگ اشاره به روش الگوریتم ژنتیک دارد و منحنی قرمز رنگ اشاره به روش خرد جمعی دارد.**

**5- نتیجه**‌**گیری**

با توجه به بخش ارزیابی نتایج شاهد آن بودیم که در بخش جدول کلمات متقاطع آمریکایی، در مورد تعداد لغات درستی که جایگذاری می‌شدند، به پیشرفت 23 درصدی رسیدیم و در بخش جدول کلمات متقاطع انگلیسی، مقدار پیشرفت 35 درصد بود و در بخش جدول کلمات متقاطع ژاپنی نیز پیشرفت 16 درصدی داشتیم.

این میزان از پیشرفت به دلیل استفاده درست از پارامتر‌های الگوریتم ژنتیک نظیر احتمال جهش، سایز جمعیت و احتمال زاد و ولد بود. همچنین از مشکلات این روش به دام افتادن در مینیمم‌های محلی بود (همانگونه که در نمودار‌های تعداد نسل - میانگین هزینه آمده بود.) برای رفع این مشکل می‌توان از راهبرد‌هایی نظیر افزایش احتمال جهش استفاده نمود.

همچنین کد تولید شده برای این مقاله در زبان برنامه‌نویسی پایتون نوشته شده است، همچنین خروجی حاصل در محیط ترمینال ویندوز بدست آمده و قابل مشاهده و فهم انسان است و می‌توان به سادگی آنرا ارزیابی نمود.

**6- مراجع**

1. [Ginsberg](https://www.researchgate.net/scientific-contributions/7730708_Matthew_L_Ginsberg), [Matthew L](https://www.researchgate.net/scientific-contributions/7730708_Matthew_L_Ginsberg). [Frank](https://www.researchgate.net/profile/Michael_Frank33), [Michael](https://www.researchgate.net/profile/Michael_Frank33), in AAAI,1990 - Search Lessons Learned from Crossword Puzzles, pp. 210-215.

2. [Beacham](https://dl.acm.org/author_page.cfm?id=81313480840&coll=DL&dl=ACM&trk=0), [Adam](https://dl.acm.org/author_page.cfm?id=81313480840&coll=DL&dl=ACM&trk=0). [Chen](https://dl.acm.org/author_page.cfm?id=81430619646&coll=DL&dl=ACM&trk=0), [Xinguang](https://dl.acm.org/author_page.cfm?id=81430619646&coll=DL&dl=ACM&trk=0). [Sillito](https://dl.acm.org/author_page.cfm?id=81100395045&coll=DL&dl=ACM&trk=0), [Jonathan](https://dl.acm.org/author_page.cfm?id=81100395045&coll=DL&dl=ACM&trk=0). [Beek](https://dl.acm.org/author_page.cfm?id=81100459486&coll=DL&dl=ACM&trk=0), [Petervan](https://dl.acm.org/author_page.cfm?id=81100459486&coll=DL&dl=ACM&trk=0), in Advances in Artificial Intelligence, ed: Springer – Constraint programming lessons learned from crossword puzzles, London, 2001, pp 78-87.

3. [Purdin](https://dl.acm.org/author_page.cfm?id=81339522744&coll=DL&dl=ACM&trk=0), [Titus D. M](https://dl.acm.org/author_page.cfm?id=81339522744&coll=DL&dl=ACM&trk=0). [Harris](https://dl.acm.org/author_page.cfm?id=81100116815&coll=DL&dl=ACM&trk=0), [Geoff](https://dl.acm.org/author_page.cfm?id=81100116815&coll=DL&dl=ACM&trk=0). in Proceedings of the 1993ACM/SIGAPP symposium on Applied computing: states of the art and practice - A genetic-algorithm approach to solving crossword puzzles, Indiana, 1993, pp 263-270.

4. Yampolskiy, Roman. Hughes, Ryan. International Journal of Intelligent Games & Simulation - Solving Sudoku Puzzles

with Wisdom of Artificial Crowds , vol. 7, p. 6, 2013.

5. Bonomo, Douglas. Lauf, Adrian P. Yampolskiy, Roman, [Computer Games: AI, Animation, Mobile, Multimedia, Educational and Serious Games (CGAMES)](https://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7256527) - A Crossword Puzzle Generator Using Genetic

Algorithms with Wisdom of Artificial Crowds, Louisville, KY, USA, 27-29 July 2015.

1. همان جستجوی brute force [↑](#footnote-ref-1)
2. همان روش همان wisdom of artificial crowds (WoAC) [↑](#footnote-ref-2)
3. Yampolskiy [↑](#footnote-ref-3)
4. Bonomo [↑](#footnote-ref-4)