استفاده از نرخ جهش پویا در تکامل مدارات دیجیتال ترکیبی

چکیده

سختافزارهای تکاملپذیر یکی از روشهای جدید برای طراحی مدارات دیجیتال ترکیبی میباشد. در این مقاله به طراحی یک واحد برای انتخاب خودکار نرخ جهش در مرحله تغییر از سختافزارهای تکاملپذیر پرداخته شده است. برای این منظور از سخت - افزارهای تکاملپذیر در سطح گیت به صورت خارجی و غیربلادرنگ استفاده شده است. سیستم ارایه شده با تغییر نرخ جهش با توجه به مقدار تابع کارایی، کارایی تکامل مدارات را افزایش میدهد. همچنین با استفاده از این روش، وابستگی تکامل به پارامتر نرخ جهش کاهش پیدا میکند. کارایی این روش با استفاده از تکامل مدارات جمعکننده ۱ و ۲ بیتی، تشخیص بیت توازن فرد ۲، ۳، ۴، ۵ و ۶ بیتی و ضرب کننده ۲ و ۳ بیتی آزموده شده است. همچنین از دو پارامتر متوسط تعداد نسلهای لازم برای تکامل و متوسط تعداد گیتهای مصرفی برای تکامل، به عنوان پارامترهای کارایی برای مقایسه استفاده شده است. نتایج شبیهسازی نشان میدهند که استفاده از این روش علاوه بر افزایش کارایی تکامل، باعث کاهش وابستگی کارایی به نرخ جهش اولیه میشود.

کلمات کلیدی

سختافزار تکاملپذیر، مدارات ترکیبی، تکامل غیربلادرنگ، تکامل در سطح گیت، نرخ جهش پویا، تابع کارایی.

Dynamic Mutation Rate for Evolution of Digital Combinational Circuits

M.Bagheri[۱], M.R. Mosavi[۲], K.Mohammadi[۳]

Department of Electrical Science, Iran University of Science and Technology
Narmak, Tehran ۱۶۸۴۶-۱۳۱۱۴, Iran

1-M.Sc. Student: m-bagheri@elec.iust.ac.ir
2-Associate Professor: m_mosavi@iust.ac.ir
3-Full Professor: mohammadi@iust.ac.ir

ABSTRACT

Evolvable Hardware is a new approach for design digital circuits. In this paper, the effect of using dynamic mutation rate for evolving digital combinational circuits has been analyzed. For this purpose, an extrinsic evolvable hardware at gate level has been used. The proposed system calculates mutation rate as a function of current fitness value and maximum fitness value. By using dynamic mutation rate, it has been shown an increase in performance parameters of evolved circuits. Another effect of dynamic mutation rate is a reduction on dependence of performance parameters to mutation rate. The performance of proposed method is evaluated by evolving ۳ and ۲ bit multipliers, ۱ and ۲ bit adders and ۷, ۶, ۵, ۴, and ۳ bit parity bit generator circuits. Average number of generation and average number of active gates in evolved circuits have been used as performance parameters. Simulation results shows that using dynamic mutation rate can increase performance parameters and also can decrease dependence of performance parameters to mutation rate.

KEYWORDS

Evolvable Hardware, Combinational Circuits, Extrinsic, Gate Level, Dynamic Mutation Rate, Fitness.

۱- دانشجوی کارشناسی ارشد دانشکده مهندسی برق دانشگاه علم و صنعت ایران، نارمک، تهران ۱۳۱۱۴-۱۶۸۴۶، ایران، M-bagheri@elec.iust.ac.ir
۲- استاد دانشکده مهندسی برق دانشگاه علم و صنعت ایران، نارمک، تهران ۱۳۱۱۴-۱۶۸۴۶، ایران، Mohammadi@iust.ac.ir
۳- دانشیار دانشکده مهندسی برق دانشگاه علم و صنعت ایران، نارمک، تهران ۱۳۱۱۴-۱۶۸۴۶، ایران، M_Mosavi@iust.ac.ir

۱. مقدمه

سختافزارهای تکاملپذیر، شاخه جدیدی از علم است که از قابلیتهای چهار زمینه سختافزارهای باز پیکرهبندیپذیر، هوش مصنوعی، سیستمهای تحملپذیر خطا و سیستمهای خودکار استفاده میکند. در ابتدا سخت افزارهای تکاملپذیر برای استفاده در کاربردهای واقعی طراحی شده بود. ولی مشکل مقیاسپذیری باعث شده است که این روش را نتوان به سیستم های پیچیده اعمال کرد. یکی از روشها برای بهبود مشکل مقیاسپذیری، استفاده از پارامترهای پویا در فرآیند تکامل است.

در این مقاله از نرخ جهش پویای جبری برای تسریع روند تکامل استفاده شده است. برای این منظور از سختافزارهای تکاملپذیر در سطح گیتهای اولیه به صورت خارجی و غیربلادرنگ استفاده شده است. برای تکامل از توابع اولیهای AND، OR، NOR و NAND استفاده شده که به سادگی قابل پیادهسازی بر روی ادوات باز پیکربندیپذیر مانند FPGA ها هستند. مدارات تکامل داده شده با استفاده از پارامترهای متوسط تعداد نسلهای لازم برای تکامل و متوسط تعداد گیتهای مصرفی برای تکامل، به عنوان پارامترهای کارایی برای مقایسه استفاده شده است. روش ارایه شده با نتایج حاصل از نرخ جهش ثابت مقایسه شده است. این نتایج نشان میدهد که در تکامل مدارات ترکیبی، نرخ جهش پویا همیشه بهتر از از نرخ جهش ثابت عمل میکند.

ادامه مقاله به این صورت است: بخش دوم به خلاصه کارهای مرتبط انجام شده میپردازد. روش استفاده شده برای تکامل مدارات در بخش سوم بیان میشود. بخش چهارم نحوه استفاده از نرخ جهش پویا و نتایج شبیه سازی را توضیح میدهد. نتیجه گیری کلی در بخش پنجم آورده شده است.

۲. کارهای مرتبط

در این قسمت به مقدمهای در مورد کارهای مرتبط انجام گرفته درمورد سختافزارهای تکاملپذیر و نحوه استفاده از نرخ جهش پویا در الگوریتمهای هوشمند میپردازیم.

در سال ۱۹۹۷ هینتردینگ و همکارانش یک دستهبندی کلی برای انواع تکامل ارایه کردند که به توضیح مختصر آنها میپردازیم[۱ و ۲]. اولین نوع تکامل، تکامل استاتیک است که در آن همه پارامترهای الگوریتم تکاملی ثابت در نظر گرفته میشود. در این نوع تکامل، برای بهینه کردن کارایی تکامل به یک عامل بیرونی نیاز داریم. همچنین کارایی این نوع تکامل، به شدت به مقدار پارامترها بستگی دارد. دومین نوع تکامل، تکامل پویا است که در آن بدون دخالت یک عامل خارجی، یک یا چند پارامتر الگوریتم تکاملی تغییر میکند. تکامل پویا با توجه به کلاس الگوریتم تکاملی استفاده شده به سه زیر قسمت تقسیم می شود که در زیر به توضیح در مورد آن میپردازیم.

اولین نوع تکامل پویا ، تکامل پویای جبری است که در آن تغییر یک یا چند پارامتر الگوریتم تکاملی بر اساس یک رابطه جبری است و هیچ گونه فیدبکی از الگوریتم تکاملی برای تغییر پارامترها استفاده نمیشود. از این روش برای تغییر پارامترهای الگوریتم تکاملی به صورت تابعی از زمان یا شماره نسل کنونی میتوان استفاده کرد. در سال ۱۹۸۹ آقای فوگارتی از نرخ جهش پویای جبری استفاده کرد [۳]. او برای این کار از رابطه زیر که بصورت تجربی بدست آورده بود استفاده کرد:

$$P_m(t) = \frac{1}{40} + \frac{0.11375}{2^t} \qquad (۱)$$

که در این رابطه t بیانگر زمان سپری شده از شروع تکامل است و با گذشت زمان نرخ جهش از ۰.۱۱۷۹ تا ۰.۰۰۴۱۶ کاهش پیدا می کند. در سالهای بعد رابطههای جبری جدیدی برای این روش ارایه شد که تابعی از شماره نسل کنونی بود. برای مثال در سال ۱۹۹۶ آقایان اسکوتز و باک از رابطه جبری زیر برای کاهش نرخ جهش استفاده کردند :

$$P_m(t) = (2 + \frac{l-2}{T-1}t)^{-1} \qquad (۲)$$

که در آن t شماره نسل کنونی، l بیانگر طول رشته و T بیانگر حداکثر شماره نسل است[۴].

دومین نوع تکامل پویا ، تکامل پویای تطبیقی است که در آن از یک نوع بازخور از الگوریتم تکاملی برای تغییر یک یا چند پارامتر الگوریتم تکاملی استفاده میشود. ریچنبرگ در سال ۱۹۷۳ از این روش برای تکامل تطبیقی استفاده کرد. این روش به قانون ۱/۵ ریچنبرگ معروف شد. در این روش او نسبت تعداد جهشهای موفق به کل جهش را ۱/۵ در نظر گرفت و در صورتی که نسبت فوق بیشتر از ۱/۵ بود، نرخ جهش را افزایش میداد و در صورتی که نسبت فوق از ۱/۵ کمتر بود، نرخ جهش را کاهش میداد [۵]. با وجود اینکه این روش خیلی مفید به نظر میرسد، ولی تاکنون کار زیادی روی این روش انجام نشده است.

سومین نوع تکامل پویا ، تکامل پویای خود تطبیقی است. در این روش، ایده الگوریتم تکاملی تکاملپذیر، برای تطبیق خودکار یک یا چند پارامتر از الگوریتم تکاملی استفاده میشود. در این روش پارامترهایی که قرار است تکامل داده شوند، به صورت یک ژن در کروموزوم وارد میشوند

و همراه با تکامل سیستم، تکامل مییابند. در سال ۲۰۰۴ آقای هارتونو روشی به نام الگوریتم ژنتیکی برچسب گذاری شده ارایه کرد که پارامترهای مورد تغییر بجای وارد شدن در کروموزوم، وارد ژنها میشد.

در سال ۲۰۰۲ آقای کرولینگ یک الگوریتم تکاملی با نرخ جهش پویای تطبیقی طراحی کرد که برای کنترل یک ربات با استفاده از FPGA مناسب بود[۶]. نحوه کار بدین صورت بود که برای اعضا با تابع کارایی بالا، از نرخ جهش کمتر و برای اعضای جامعه با تابع کارایی پایینتر از نرخ جهش بزرگتر استفاده میکرد. با این کار او موفق شد تعداد نسلهای لازم برای رسیدن به هدف را کاهش دهد.

با وجود اینکه تاکنون روشهای زیادی برای تغییر پارامترهای تکاملی ارایه شده است، ولی کارایی این روشها در تکامل مدارات ارزیابی نشده است. در این مقاله به بررسی کارایی نرخ جهش پویای جبری در تکامل مدارات ترکیبی پرداخته شده است.

۳. طراحی مدارات ترکیبی با استفاده از سختافزارهای تکاملی

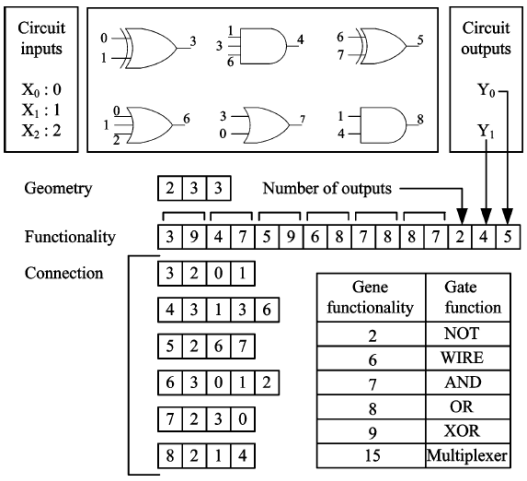
در این قسمت روش معمول برای تکامل یک مدار ترکیبی به صورت کامل آورده شده است. استراتژی تکامل استفاده شده، نحوه نمایش کروموزوم، تابع کارایی و عملگرهای ژنتیکی استفاده شده بیان شده است[۷و۸]. در این مقاله برای آزمون کارایی نرخ جهش پویا، از استراتژی توضیح داده شده در این بخش برای تکامل مدارات استفاده شده است.

۱.۳. الگوریتم تکاملی

الگوریتم تکاملی استفاده شده، الگوریتم تکاملی شناخته شده (λ+۱) است که λ بیانگر اندازه جامعه است. ابتدا همه کروموزومها به صورت تصادفی تولید میشوند. در دومین مرحله، مقدار کارایی هر یک از کروموزومها محاسبه میشود. سومین مرحله، شامل انتخاب کروموزوم با بهترین اندازه کارایی است. در چهارمین مرحله، کروموزوم انتخاب شده در مرحله سوم، برای بررسی شرایط اتمام تکامل، ارزیابی میشود. این شرایط عبارتند از: مقدار کارایی کروموزوم ٪۱۰۰ است و یا تعداد نسل های تولید شده، بزرگتر از تعداد نسل های مجاز تعیین شده توسط کاربر برای آن آزمایش است. اگر یکی از این شرایط برقرار باشد، تکامل به پایان رسیده است. در غیر این صورت جمعیت جدیدی با استفاده از λ بار جهش بهترین کروموزوم انتخاب شده در مرحله سوم ایجاد میشود که تعداد اعضای نسل جدید برابر λ+۱ است. در مرحله بعدی همه اعضای جدید ایجاد شده، مورد ارزیابی قرار میگیرند و مقدار کارایی این اعضا با مقدار کارایی بهترین کروموزوم مرحله قبل، مقایسه میشوند و بهترین کروموزوم نسل جدید انتخاب میشود. این فرآیند تا برقراری شرط اتمام تکامل، ادامه مییابد.

۲.۳. رمز گذاری کروموزوم

کروموزوم ساختار مدار منطقی را مشخص میکند و ارتباطات بین گیتها را بیان میکند. در این رویکرد، مدار منطقی به عنوان آرایهای مستطیلی از سلولهای منطقی در نظر گرفته شده است. نوع هر سلول منطقی به صورت تصادفی، از مجموعه توابع اولیهای AND، OR، NOR و NAND انتخاب میشود. کروموزوم توسط یک ساختار سه لایه توصیف میشود.



شکل۱: مثالی از کدگذاری کروموزوم برای مدار با سه ورودی و دو خروجی[۷]

درشکل ۱، یک مثال از کدگذاری کروموزوم برای یک مدار نمونه با سه ورودی و دو خروجی نشان داده شده است. طرح کلی مدار به

صورت یک آرایه با دو ردیف و سه ستون است و پارامتر درجه اتصالات داخلی، سه در نظرگرفته شده است. در نتیجه، کروموزوم درلایه هندسی، به صورت ۲، ۳ و ۳ است. کروموزوم در لایه عملکردی، آرایه سلولهای منطقی را از نظر عملکردی مشخص میکند. به عنوان مثال، گیت منطقی مشخص شده با خروجی شماره ۷، یک گیت OR است. یک گیت OR با استفاده از شماره ۸ رمزگذاری شده است. بنابراین دو داده اول مربوط به کروموزوم در لایه عملکردی، ۷ و ۸ است. دو داده انتهایی در کروموزوم عملکردی، خروجی های نهایی مدار را مشخص میکند. به عنوان مثال در این مورد خاص، خروجیهای نهایی مدار، از خروجیهای گیتهای منطقی ۴ و ۵ گرفته شدهاند. گیت منطقی با خروجی شماره ۴، یک گیت AND است و گیت AND با عدد شماره ۷ کدگذاری شده است. بنابراین دو داده بعدی مربوط به کروموزوم درلایه عملکردی، ۴ و ۷ است. همه گیتهای منطقی دیگر، با همین روش کد گذاری شدهاند.

کروموزوم در لایه اتصالات، شماره گیت منطقی، تعداد ورودیهای آن و اینکه این ورودیها از خروجی کدام گیتهای دیگر گرفته شده است را مشخص میکند. کروموزوم اولین گیت منطقی، ستون اول، ردیف اول شامل عدد ۳ است که بیانگر شماره گیت منطقی است. عدد ۲ بعد از آن نشان دهنده این است که این گیت دو ورودی دارد. اعداد ۰ و ۱ بعد از آن بیانگر این است که ورودی های این گیت از x_0 و x_1 گرفته شدهاند. بقیه گیت های منطقی نیز مانند این گیت توصیف شدهاند.

۳.۳. تابع کارایی

تابع کارایی عملکرد مدار تکامل یافته را ارزیابی میکند. تابع کارایی برای معیار طراحی، از رابطه زیر محاسبه می شود:

$$f_1 = \sum_{f_c=0}^{2^n-1} \sum_{i=0}^{m-1} | y_i - d_i | \qquad (۳)$$

که در آن m و n، به ترتیب تعداد خروجیها و ورودیهای تابع منطقی داده شده هستند، y_i، i امین رقم ترکیبات خروجی است که توسط ارزیابی مدار تکامل داده شده به دست آمده است، d_i خروجی مطلوب برای f_c مورد نظر است و در نهایت $| y_i - d_i |$ اختلاف مطلق بین خروجی ایجاد شده توسط مدار منطقی تکامل داده شده و مقدار مطلوب است. حداکثر مقدار تابع کارایی برابر $2^n \times m$ است.

۴.۳. عملگر ژنتیکی

در این سیستم نمونه، از دو عملگر ژنتیکی جهش و نخبه سالاری استفاده شده است. نخبه سالاری، انتقال بهترین کروموزوم نسل فعلی را به نسل بعدی تضمین میکند. عملگر جهش برای تغییر بعضی از دادههای موجود در کروموزومها استفاده میشود. هدف اصلی این عملگر، ایجاد تغییر در جامعه است. با افزایش نرخ جهش، جستجوی ژنتیکی، به یک جستجوی تصادفی تبدیل میشود. اما کمک میکند که راه حل های گم شده را بتوانیم دوباره پیدا بکنیم. از عملگر تقاطع استفاده نشده است. روش تعیین نرخ جهش در بخش بعدی توضیح داده شده است.

۴. نتایج شبیه سازی

در این قسمت به بررسی پارامترهای استفاده شده برای تکامل، نحوه استفاده از نرخ جهش پویا و نتایج بدست آمده از شبیهسازیها پرداخته شده است. برای بررسی کارایی این روش، از دو پارامتر متوسط تعداد نسلهای لازم برای تکامل و متوسط تعداد گیتهای مصرفی برای تکامل، به عنوان پارامترهای کارایی برای مقایسه استفاده شده است.

۱.۴. نرخ جهش پویا

همانطور که در بخش کارهای مرتبط بیان شد، برای پویا کردن نرخ جهش سه روش متفاوت وجود دارد. در این مقاله از روش نرخ جهش پویای جبری استفاده شده است. در این روش معمولا نرخ جهش به صورت تابعی از زمان یا تعداد نسلهای سپری شده، تغییر میکند. استراتژی استفاده شده در این مقاله به این صورت است که نرخ جهش به صورت تابعی از مقدار تابع کارایی و مقدار حداکثر تابع کارایی تغییر میکند. رابطه استفاده شده برای تغییر نرخ جهش، به صورت زیر است:

$$p_m = p_{m.start} \times (1 - \frac{F}{F_{\max}}) \qquad (۴)$$

که دراین رابطه $p_{m.start}$ برابر مقدار اولیه تعیین شده برای نرخ جهش است. $F_{\max}$ برابر حداکثر مقدار تابع کارایی است که وقتی تابع کارایی به آن میرسد، روند تکامل متوقف میشود. F هم برابر حداکثر تابع کارایی در نسل جاری است. استفاده از رابطه بالا نرخ جهش را به صورت تابعی خطی از زمان کاهش میدهد. به دلیل اینکه استفاده از نرخ جهش خیلی کوچک در سختافزارهای تکاملپذیر باعث کاهش کارایی تکامل

میشود[۸]، یک حد پایین برای نرخ جهش در نظر گرفته شده است. برای بدست آوردن این حد پایین، از روش آزمون و خطا استفاده گردید و به این نتیجه رسیدیم که حد پایین برابر ۱ در نظر گرفته شود. در نتیجه خواهیم داشت:

$$\text{if } p_m < 1 \Rightarrow p_m = 1 \quad (۵)$$

نتایج بدست آمده از شبیه سازی نشان میدهد با استفاده از این رابطه در کمترین تعداد نسلهای ممکن مدار تکامل مییابد.

جدول ۱: پارامترهای استفاده شده برای تکامل مدارات

پارامتر	مقدار
تعداد عضوها	**۵**
نرخ جهش اولیه	**٪۲۰**
حد پایین نرخ جهش	**٪۱**
حداکثر تعداد نسلها	**۳,۰۰۰,۰۰۰**
حداکثر تعداد نسلها برای اثر ایستایی	**۹۰۰,۰۰۰**
تعداد ردیفهای آرایه تکاملپذیر	**۱**
تعداد ستونهای آرایه تکاملپذیر	**۱۰۰**
سطح اتصالات	**۱۰۰**

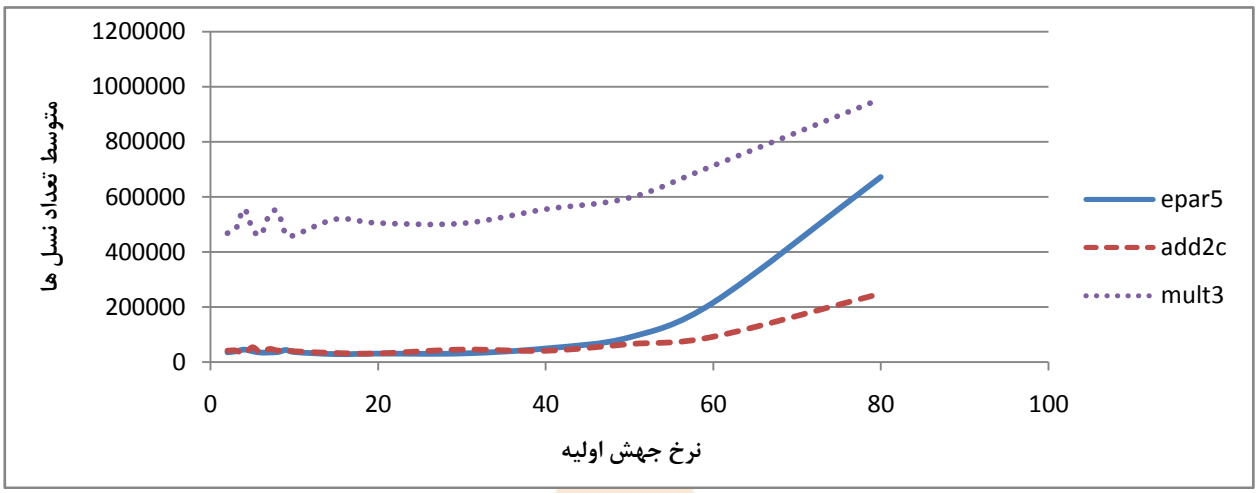
۲.۴. مدارات تکامل داده شده

پارامترهای استفاده شده برای تکامل مدارات در جدول۱ آورده شده است. در تکامل مدارات از گیتهای اولیه AND، OR، NOR و NAND استفاده شده است. به علت ماهیت تصادفی الگوریتمهای تکاملی، هر یک از مدارات ۵۰ بار تکامل داده شدهاند و از پارامترهای کارایی آنها میانگین گرفته شده است. جدول ۲ نتایج حاصل از تکامل مدارات را نشان میدهد. در این جدول دو پارامتر متوسط تعداد نسلهای لازم برای تکامل و متوسط گیتهای لازم برای مدارات تکامل داده شده با نرخ جهش ثابت ٪۳، نرخ جهش ثابت ٪۲۰ ، نرخ جهش پویای ٪۳ و نرخ جهش پویای ٪۲۰ مقایسه شده است. پارامتر متوسط زمان لازم برای تکامل، به علت وابستگی به سیستم مورد استفاده برای شبیهسازی در این جدول آورده نشده است.

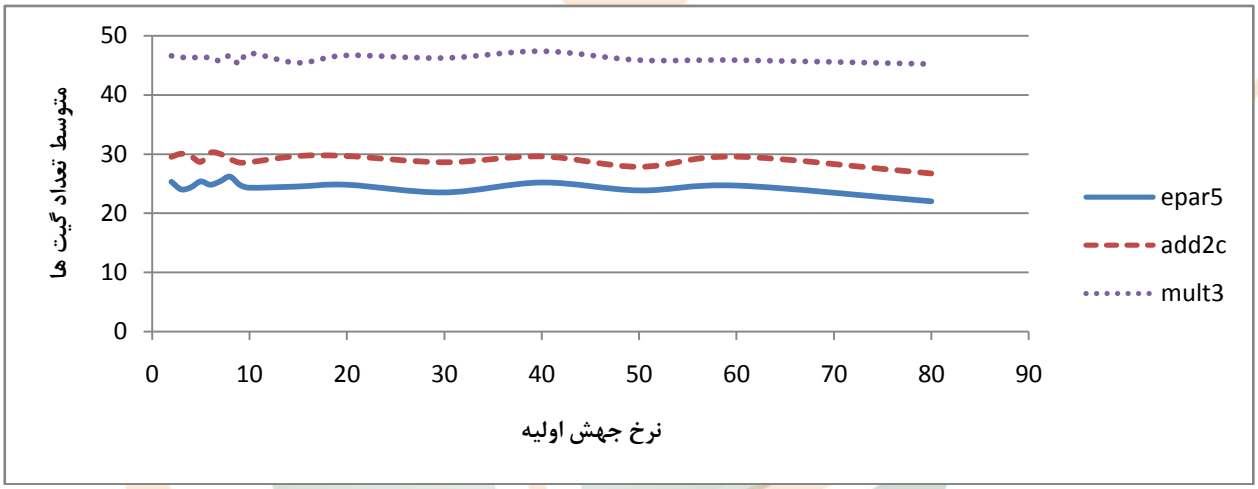
در جدول ۲ بعضی خانهها با علامت "-" مشخص شدهاند. این علامت مربوط به مداراتی است که با پارامترهای تعین شده قادر به تکامل انها نبودهایم. اگر در جدول ۲ دقت شود، دیده میشود که در بعضی مدارات، متوسط نسلهای لازم برای تکامل با استفاده از نرخ جهش ثابت ٪۳ بهتر از نرخ جهش پویا است. برای مثال در تکامل مدار ضرب کننده ۲ بیتی این پارامتر با نرخ جهش ثابت ٪۳ بهتر از سایر حالتها است. اما توجه به این نکته ضروری است که اعمال این روش به مدارات بزرگتر مانند ضرب کننده ۳ بیتی و مدار تولید بیت توازن ۷ بیتی باعث کاهش چشمگیر تعداد نسلها میشود. همچنین استفاده از نرخ جهش پویا باعث شده است که با نرخ جهش اولیه ٪۲۰ نتایج قابل مقایسه با نرخ جهش ثابت ٪۳ بگیریم و این همان کاهش وابستگی کارایی تکامل به پارامتر نرخ جهش است.

جدول ۲ پارامترهای کارایی برای مدارات تکامل داده شده

پارامتر	متوسط تعداد نسل های لازم				متوسط گیت های لازم			
نرخ جهش	ثابت ٪۲۰	پویا ٪۲۰	ثابت ٪۳	پویا ٪۳	ثابت ٪۲۰	پویا ٪۲۰	ثابت ٪۳	پویا ٪۳
ضرب ۳ بیتی	-	۵۰۵,۲۳۷.۰۰	۱,۲۹۵,۳۰۵.۸۱	۴۸۴,۲۴۶.۳۱	-	۴۶.۶۵	۴۵.۲۴	۴۶.۳۱
ضرب ۲ بیتی	۲۸۶,۲۵۲.۷۶	۵,۴۶۴.۰۰	۴,۴۱۲.۶۴	۵,۸۴۳.۵۸	۲۱.۳۲	۲۶.۹۸	۲۶.۰۲	۲۷.۱۲
جمع ۲ بیتی	-	۳۰,۶۶۵.۱۲	۴۲,۸۳۱.۹۲	۴۱,۳۳۸.۶۴	-	۲۹.۶۴	۲۸.۶۲	۳۰.۰۲
جمع ۱ بیتی	۳۷,۲۳۳.۵۲	۴,۹۰۵.۸۲	۲,۲۲۲.۳۰	۵,۶۰۱.۰۴	۱۸.۸۲	۲۱.۲۴	۲۱.۳۰	۲۱.۲۰
بیت توازن ۷ بیتی	-	۳۳۲,۵۱۹.۴۸	۳۳۱,۱۹۰.۱۰	۲۸۳,۰۷۶.۰۸	-	۳۱.۹۸	۳۲.۰۸	۳۲.۲۸
بیت توازن ۶ بیتی	-	۱۳۰,۹۷۱.۴۴	۱۰۳,۳۴۱.۶۲	۱۲۳,۱۲۰.۹۸	-	۲۸.۷۲	۲۵.۹۴	۲۹.۱۸
بیت توازن ۵ بیتی	۱,۲۹۷,۹۹۷.۲۹	۳۱,۰۳۱.۸۰	۲۷,۸۰۹.۱۸	۳۹,۲۲۶.۳۸	۱۹.۸۶	۲۴.۷۸	۲۴.۳۲	۲۴.۰۰
بیت توازن ۴ بیتی	۲۲۱,۲۹۰.۴۷	۷,۷۳۰.۸۲	۶,۴۴۱.۹۶	۱۲,۹۱۶.۰۰	۱۸.۲۴	۲۰.۷۰	۲۰.۲۸	۲۲.۳۲
بیت توازن ۳ بیتی	۴,۱۸۰.۴۲	۸۸۲.۴۶	۱,۱۲۷.۰۴	۲,۹۰۳.۳۶	۱۵.۶۴	۱۷.۵۴	۱۷.۷۶	۱۷.۴۶



شکل ۲: وابستگی متوسط نسلهای لازم برای تکامل به نرخ جهش اولیه



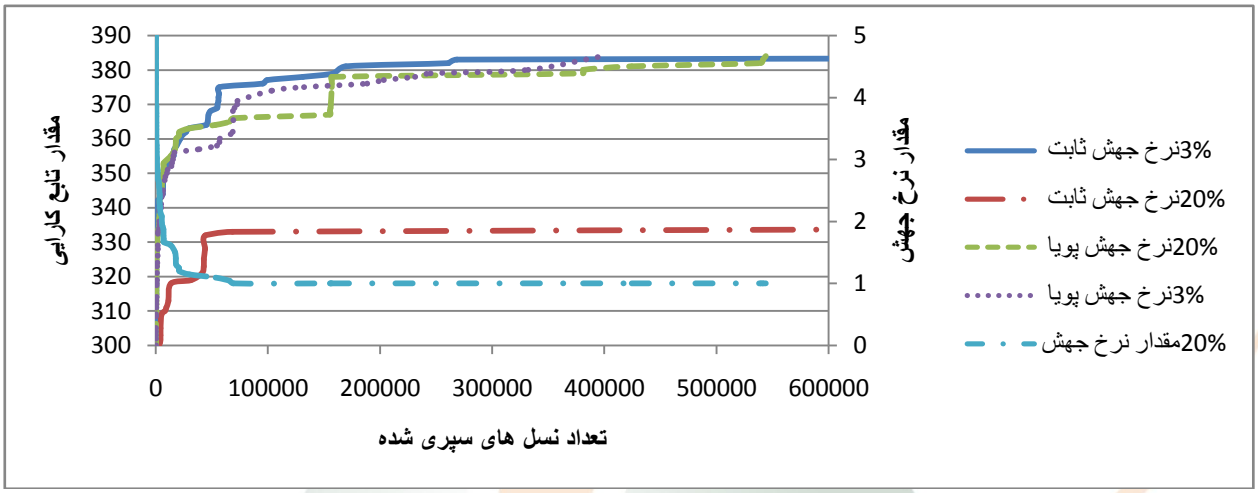
شکل ۳: وابستگی متوسط نسلها ی لازم برای تکامل به نرخ جهش اولیه

شکل ۲ و ۳ به ترتیب نمودارهای مربوط به تعداد متوسط نسلهای لازم و تعداد متوسط گیتهای لازم برای تکامل یک مدار ضرب کننده ۳ بیتی، یک جمع کننده دو بیتی و یک مدار تولید بیت توازن ۵ بیتی را در برابر پارامتر نرخ جهش اولیه نشان میدهد. همانطور که از شکل ۲ کاملا مشخص است، با استفاده از نرخ جهش پویای جبری، وابستگی پارامتر متوسط نسلهای لازم برای تکامل به نرخ جهش اولیه بشدت کم میشود. برای مقایسه باید در نظر داشت که با استفاده از نرخهای جهش ثابت بزرگتر از ۱۰، با استفاده از پارامترهای جدول ۱، موفق به تکامل مدارات ذکر شده نشدیم. به همین دلیل در این نمودار فقط دادههای مربوط به مربوط به تکامل با استفاده از نرخ جهش پویا آورده شده است.

نکته قابل توجه دیگر در شکل ۳، عدم وابستگی تعداد گیتهای استفاده شده برای تکامل مدارات مزبور، به نرخ جهش اولیه است. این در حالی است که آقای استومئو در [۸] به این نتیجه رسیده است که استفاده از نرخ جهش ثابت بزرگ، با افزایش تعداد نسلهای لازم برای تکامل، تعداد گیتهای مصرفی برای تکامل را کاهش میدهد. درصورتی که با استفاده از نرخ جهش پویا، علاوه بر کاهش تعداد نسلهای لازم برای تکامل، وابستگی تعداد گیتهای لازم برای تکامل به پارامتر نرخ جهش، به شدت کم میشود. در نتیجه استفاده از این روش در جاهایی که دانش فنی کارکنان نسبتا پایین است، میتواند باعث کاهش خطای ناشی از اشتباه استفاده کننده در تعیین پارامترهای اولیه تکامل بشود.

شکل ۴ نمودار تغییرات مقدار تابع کارایی را در برابر نسلهای سپری شده برای یک مدار نمونه ضرب کننده ۳ بیتی نمایش میدهد. برای مقایسه، نمودار مربوط به تکامل با استفاده از نرخ جهش پویا با نرخ جهش اولیه ۲۰٪ و ۳٪ ، نرخ جهش ثابت ۳٪ و ۲۰٪ در این شکل آورده شده است. همانطور که در این شکل مشخص است، نرخ جهش ثابت ۲۰٪ در مقدار تابع کارایی خیلی کم، دچار اثر ایستایی میشود و نمیتواند بر آن غلبه بکند.در مقابل تکامل با نرخ جهش ثابت ۳٪ ، در نسلهای پایین خیلی سریع رشد میکند. ولی دقت شود که تکامل با استفاده از نرخ جهش پویای ۳٪ و ۲۰٪ خیلی سریعتر نسبت به نرخ جهش ثابت ۳٪ به جواب میرسند. در این آزمایش نمونه، تکامل با استفاده از نرخ جهش پویای ۳٪، بعد از حدود ۴۰۰۰۰۰ نسل، نرخ جهش پویای ۲۰٪ بعد از حدود ۵۵۰۰۰۰ نسل و نرخ جهش ثابت ۳٪ بعد از حدود ۲۰۰۰۰۰۰ نسل به تابع کارایی حداکثر دست پیدا میکنند. همچنین تکامل با استفاده از نرخ جهش ثابت ۲۰٪ بعد از حدود ۵۰۰۰۰۰۰ نسل، به مقدار تابع کارایی ۳۳۹ (برابر با ۸۸٪) دست پیدا میکند. درنتیجه استفاده از نرخ جهش پویا، سرعت تکامل را بهبود داده است.

در نمودار شکل ۴ همچنین تغییرات مقدار نرخ جهش در مقابل تغییر تعداد نسلهای سپری شده ترسیم شده است. این منحنی مربوط به تکامل با استفاده از نرخ جهش پویای ۲۰٪ است. همانطور که از این نمودار مشخص است، با افزایش مقدار تابع کارایی، مقدار نرخ جهش کاهش پیدا میکند. این کاهش در نسلهای اولیه، به علت افزایش سریع مقدار تابع کارایی سریع است. اما با رسیدن به نسلهای انتهایی این کاهش کمتر میشود. در این نمودار محدود شدن حد پایین مقدار نرخ جهش به صورت کاملا واضح نشان داده شده است. همانطور که در بخشهای قبلی به آن اشاره شد، کارایی الگوریتم تکاملی در نرخ جهشهای خیلی پایین، کاهش پیدا میکند. محدود کردن حد پایین نرخ جهش برای جلوگیری از این کاهش کارایی انجام گرفته شده است.



شکل۴: مقدار تابع کارایی و نرخ جهش در برابر تعداد نسلهای سپری شده

۵– نتیجه

در این مقاله به بررسی تاثیر نرخ جهش در سختافزارهای تکاملپذیر پرداختیم. ابتدا با تکامل مدارات مختلف با استفاده از دو نوع نرخ جهش پویا و ثابت، نشان دادیم که استفاده از نرخ جهش ثابت باعث کاهش کارایی الگوریتم تکاملی میشود. همچنین استفاده از نرخ جهش ثابت باعث وابستگی شدید کارایی تکامل به پارامتر نرخ جهش ثابت انتخاب شده میشود. سپس برای برطرف کردن این نواقص، استفاده از یک نوع نرخ جهش پویای جبری پیشنهاد شد. با استفاده از شبیهسازیها و تکامل مدارات مختلف به این نتیجه رسیدیم که استفاده از نرخ جهش پویا در تکامل مدارات، علاوه بر افزایش کارایی تکامل، باعث کاهش شدید وابستگی کارایی تکامل به نرخ جهش انتخاب شده میشود.

۶– مراجع

[۱] R. Hinterding, Z. Michalewicz, and A.E. Eiben, "*Adaption in Evolutionary Computation: A Survey*," in Proc. of 1997 IEEE Conf. on Evolutionary Computation (ICEC`97), 1997, pp. 65-69.

[۲] H.G. Beyer, "*The Theory of Evolution Strategies*," Natural Computing Series. Springer, Heidelberg, 2001.

[۳] T. Fogarty; "*Varying the Probability of Mutation in the Genetic Algorithm*," Proc. of the Third International Conference on Genetic Algorithms, pp. 104-109, Morgan Kaufmann, 1989.

[۴] T. Back and M. Schutz, "*Intelligent Mutation Rate Control in Canonical Genetic Algorithms*," Proc. of the International Symposium on Methodologies for Intelligent Systems, pp. 158-167, 1996.

[۵] I. Rechenberg, "*Evolutions Strategie: Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution*," Frommann, 1973.

[۶] A. Renato, Krohling, Yuchao Zhou, and M. Tyrrell, "*Evolving FPGA-based Robot Controllers Using an Evolutionary Algorithm*," 1st International Conference on Artificial Immune, 2002.

[۷] E. Stomeo, T. Kalganova, C. Lambert, "*Generalized Disjunction Decomposition for Evolvable Hardware*", IEEE Transaction on system, Vol.36, NO.5, 2006, PP. 1024-1043.

[۸] E. Stomeo, T. Kalganova, C. Lambert, "*Mutation Rate for Evolvable Hardware*," World Academy of Science, Engineering and Technology, 2005.