مقاله پژوهشی

ارائه یک رویکرد نگاشت در شبکه روی تراشه مبتنی بر الگوریتم جستجوی هارمونی

زهرا باقری، فاطمه وردی و علیرضا محجوب

چکیده: در پیاده‌سازی مبتنی بر شبکه روی تراشه، نگاشت را می‌توان گامی مهم در اجرای برنامه کاربردی دانست. وظایف یک کاربرد، اغلب در قالب یک گراف هسته نمایش داده می‌شود. هسته‌ها با استفاده از یک بستر ارتباطی و غالباً شبکه روی تراشه، بین خود پیوند برقرار می‌کنند و به این منظور، توسعه‌دهندگان الگوریتم‌های گوناگونی را پیشنهاد داده‌اند. در اغلب موارد به‌دلیل پیچیدگی از روش‌های جستجوی دقیق برای یافتن نگاشت استفاده می‌شود. با این حال این روش‌ها برای شبکه‌های با ابعاد کوچک مناسب هستند. با افزایش ابعاد شبکه، زمان جستجو نیز به‌طور نمایی افزایش می‌یابد. این مقاله از دیدگاه یک رویکرد فراابتکاری با استفاده از روش جستجوی هارمونی به تصمیم‌گیری زمانی برای اتصال هسته‌ها به روترها می‌پردازد. رویکرد ما نوعی بهبودیافته از الگوریتم جستجوی هارمونی را با تمرکز روی کاهش توان مصرفی و تأخیر به کار می‌گیرد. تحلیل پیچیدگی الگوریتم، آشکارکننده راه حل مناسب‌تر در مقایسه با الگوریتم‌های مشابه با توجه به الگوی ترافیکی برنامه کاربردی است. الگوریتم در مقایسه با روش‌های مشابه به ۳۹٫۹۸٪ تأخیر کمتر و ۶۱٫۱۱٪ صرفه‌جویی در توان مصرفی دست می‌یابد.

***کلیدواژه:* شبکه‌های روی تراشه، نگاشت، جستجوی هارمونی، فراابتکاری.**

۱- مقدمه

با افزایش قدرت پردازشی تراشه‌ها، پیچیدگی و قابلیت برنامه‌های کاربردی نیز افزایش یافته و این افزایش پیچیدگی سخت‌افزار و نرم‌افزار در سیستم‌های روی تراشه و پردازنده‌های چندهسته‌ای، به نوبه خود افزایش حجم و پیچیدگی ترافیک ارتباطی را داخل تراشه موجب می‌شود. از سوی دیگر، کاهش اندازه[1] ترانزیستورها، مشکلات و چالش‌های دیگری را در سطح مدار به ویژه برای ساختارهای ارتباطی درون تراشه به همراه دارد. مواجهه با این پیچیدگی ارتباطات و همچنین مسائل موجود در فناوری‌های جدید VLSI، نیاز به بازنگری روش‌های سنتی ارتباطی درون تراشه را ایجاد کرد و شبکه روی تراشه به عنوان یک طرح ارتباطی درون تراشه‌ای نوین برای رفع و کاهش این مشکلات مورد توجه قرار گرفت [۱].

این مقاله در تاریخ ۱۵ تیر ماه ۱۴۰۱ دریافت و در تاریخ ۱۰ دی ماه ۱۴۰۱ بازنگری شد.

زهرا باقری، گروه مهندسی کامپیوتر، دانشگاه آزاد اسلامی واحد پرند، تهران، ایران، (email: bagheri.ati@gmail.com).

فاطمه وردی (نویسنده مسئول)، گروه مهندسی کامپیوتر، دانشگاه آزاد اسلامی واحد پرند، تهران، ایران، (email: f.vardi@piau.ac.ir).

علیرضا محجوب، گروه مهندسی کامپیوتر، دانشگاه آزاد اسلامی واحد کرج، کرج، ایران، (email: alirezamahjoub.a@gmail.com).

1. Feature Size

افزایش کارایی سیستم‌های روی تراشه چندپردازنده‌ای به‌عنوان یکی از چالش‌های مهم در محیط شبکه‌های روی تراشه است. از مشکلات دیگر سیستم‌های چندپردازنده‌ای مبتنی بر شبکه روی تراشه می‌توان تخصیص مناسب و کارآمد منابع به وظایف یا همان نگاشت و تعیین ارتباط مسیریاب‌ها با هسته‌های یک برنامه کاربردی را نام برد.

نگاشت هسته‌های پردازشی مورد نیاز یک کاربرد خاص بر روی گره‌های یک شبکه روی تراشه، یکی از مؤثرترین راه‌های بهینه‌سازی شبکه‌های روی تراشه برای آن کاربرد است. هدف بیشتر الگوریتم‌های نگاشت، کنار هم قرار دادن هسته‌های پردازشی است که حجم ارتباطی زیادی با یکدیگر دارند. از یک دیدگاه می‌توان نگاشت را به دو دسته کلی تقسیم کرد. در دسته اول، نگاشت بر روی گره‌های یک شبکه با همبندی معلوم و اغلب توری انجام می‌گیرد. در حالی که در دسته دوم، ابتدا یک همبندی خاص‌منظوره طبق نیازمندی‌های ترافیکی برنامه کاربردی ساخته می‌شود. سپس عملیات نگاشت بر اساس این همبندی و یا هم‌زمان با ساخت آن انجام می‌پذیرد [۲].

نگاشت می‌تواند ماهیت ایستا یا پویا داشته باشد [۳]. در نگاشت ایستا تخصیص وظایف به هسته‌ها به‌صورت آفلاین و قبل از اجرای برنامه کاربرد انجام می‌گیرد. این نوع نگاشت سعی دارد که همیشه بهترین مکان وظایف را در زمان طراحی مشخص کند و از آنجایی که نگاشت قبل از اجرای برنامه کاربرد انجام می‌گیرد، الگوریتم نگاشت فقط یک بار اجرا می‌شود. برای شبکه روی تراشه، نگاشت ایستا پیشنهاد می‌شود، زیرا سربار محاسبات به صورت چشم‌گیری بر روی عملکرد سیستم تأثیر می‌گذارد و تأخیر کلی سیستم را افزایش می‌دهد. در حالی که استراتژی نگاشت پویا، آنلاین است. بنابراین مکان وظایف در شبکه روی تراشه می‌تواند در طول اجرای برنامه کاربرد تغییر کند. الگوریتم‌های نگاشت در شبکه‌های روی تراشه با دیدگاه بهبود شاخص اصلی آن یعنی کارایی، توان مصرفی، نرخ گذردهی و تأخیر به‌عنوان تابعی متغیر از تزریق داده‌ها، توزیع مکانی مبدأ و مقصد، تعداد گره‌ها بر حسب پروتکل ارتباطی، الگوریتم مسیریابی، اندازه بافر و تأخیر قابل بررسی است [۴]. به این منظور قابلیتی در تراشه طراحی شده که بتواند با برقراری تعادل بین معیارهای دارای اهمیت، کیفیت سرویس مورد نظر را فراهم آورد. اما از نظر عملکردی انتظار می‌رود که نگاشت ایستا بهتر باشد، زیرا می‌تواند دید جامع‌تری از تمام هسته‌ها، محاسبات و الزامات ارتباطی آنها داشته باشد.

این مقاله بر استراتژی‌های نگاشت ایستا متمرکز است. بسیاری از رویکردها از جمله روش‌های دقیقی مانند برنامه‌ریزی خطی، روش‌های اکتشافی، استراتژی‌های فراجستجو مانند الگوریتم‌های ژنتیک و غیره این استراتژی را دنبال می‌کنند. ویژگی‌های بارز رویکرد پیشنهادی به شرح ذیل است:

۱) یک الگوریتم نگاشت مبتنی بر ساختار گراف و الگوریتم جستجوی هارمونی (HS)[1] [۳] برای کاهش هزینه ارتباطات پیشنهاد گردیده است.
۲) مقادیر متریک هزینه ارتباطات در راه حل پیشنهادی با برخی رویکردهای موجود مقایسه شده است. هم در کیفیت و هم در زمان اجرا، نگاشت پیشنهادی بهبود خوبی را نشان می‌دهد.
۳) مقایسه عملکرد پویا از نظر میانگین تأخیر شبکه و توان عملیاتی نیز انجام شده است.
۴) برای مشاهده بهینه‌بودن تکنیک نگاشت پیشنهادی، محاسبه تابع هزینه با امکان کاهش هم‌زمان تأخیر و توان مصرفی فرموله شده است.

ادامه مقاله به شرح زیر سازماندهی شده است. در بخش دوم از مقاله، راهکارهای موجود در زمینه نگاشت وظایف در شبکه‌های روی تراشه مورد بررسی قرار می‌گیرد. در بخش سوم، روش پیشنهادی، ارائه و ساختار مربوط به روش جدید بیان می‌شود. هدف از رویکرد پیشنهادی، ارائه یک راهکار نگاشت وظایف کارا در شبکه‌های روی تراشه به‌منظور بهبود مؤلفه‌های کیفیت سرویس است. در بخش چهارم، شبیه‌سازی روش پیشنهادی ارائه شده و نتایج به‌دست‌آمده از شبیه‌سازی، ارزیابی می‌گردند. در انتها و در بخش پنجم پس از یک نتیجه‌گیری کلی، پیشنهاد کارهای آتی و منابع ارائه می‌شوند.

۲- مرور ادبیات

روش‌های نگاشت در شبکه‌های روی تراشه با توجه به زمان اختصاص‌یافته به وظایف یک برنامه کاربردی به‌منظور پردازش در هسته‌های شبکه روی تراشه طبقه‌بندی می‌شوند. برنامه کاربردی با نگاشت ایستا در زمان طراحی مشخص شده و نمی‌تواند به‌طور پویا تغییر کند. برای یک کاربرد و زیرساخت‌های ارتباطی آن با توجه به در دسترس بودن تمامی اطلاعات لازم، الگوریتم نگاشت ایستا سعی می‌کند که بهترین مکان وظایف را در زمان طراحی مشخص کند [۵]. در این روش چون نگاشت قبل از اجرای برنامه کاربرد کامل می‌شود، الگوریتم نگاشت تنها یک بار در زمان کامپایل اجرا می‌شود؛ بنابراین روی کارایی مربوط به کاربرد در زمان اجرا تأثیر نمی‌گذارد. در نگاشت ایستا با اختصاص یک بستر برای اجرای کاربرد مورد نظر، هیچ گونه رفتار پویایی از قبیل اضافه‌شدن، حذف‌شدگی و یا مهاجرت وظایف در طول زمان اجرا قابل قبول نیست. بنابراین انتظار می‌رود که الگوریتم‌های نگاشت ایستا، راه حل‌های نزدیک به راه حل بهینه بیشتری را ایجاد کنند.

روش‌های نگاشت ایستا به دو دسته نگاشت دقیق[2] و نگاشت مبتنی بر جستجو[3] تقسیم می‌شوند. نگاشت دقیق، نگاشتی است که بر مبنای کدنویسی و فرموله‌بندی ریاضی به راه حل بهینه دست پیدا می‌کند. برنامه‌نویسی خطی عدد صحیح (ILP)[4]، برنامه‌نویسی خطی غیر عدد صحیح[5] و برنامه‌نویسی خطی ترکیبی (MILP)[6] سه نوع از مهم‌ترین الگوریتم‌های نگاشت دقیق هستند [۶].

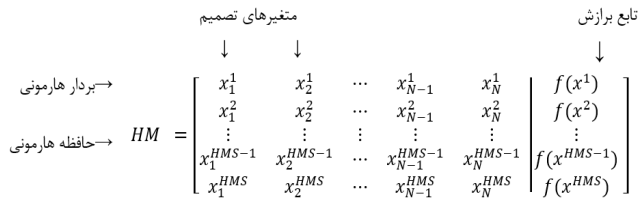
در راهکار [۷] یک روش MILP جهت نگاشت وظایف برای سیستم‌های چندپردازنده ناهمگن ارائه شده است. در این روش ابتدا به‌طور حریصانه، هسته‌ها بر روی یک همبندی خاص نگاشت شده و سپس در مرحله بهبود، موقعیت‌های نسبی هسته‌ها توسط جستجوی ممنوع[7] ثابت می‌شوند. در [۸] از روش MILP برای ساخت یک معماری شبکه روی تراشه استفاده شده که هدف آن بهینه‌سازی و کمینه‌کردن توان مصرفی با درنظرگرفتن محدودیت‌های کارایی است. گلوگاه اصلی در ILP [۹]، زمان اجراست و برای کاهش آن، گراف وظیفه کاربرد مورد نظر به تعدادی خوشه تقسیم‌بندی شده است.

در رویکرد [۱۰]، یک روش ILP دومرحله‌ای را برای تخصیص وظایف و نگاشت داده روی سیستم چندپردازنده‌ای متقارن[8] ارائه داده‌اند و به دنبال آن در راهکار [۱۱]، رقابت در شبکه مورد بررسی قرار گرفته است. در این روش از یک فرمول‌بندی LP برای کاهش رقابت در دستیابی به منبع با استفاده از کاربردهای نگاشت رقابت- آگاه از تراکم[9] استفاده می‌شود. کاهش رقابت در شبکه موجب کاهش تأخیر بسته‌ها می‌شود؛ با این حال برخی از این روش‌ها زمان پردازش بالایی دارند که اتلاف انرژی ارتباطی را افزایش می‌دهد. همچنین ازدحام ترافیک در پیوندها ممکن است موجب کاهش عملکرد سیستم شود. برای غلبه بر این مشکل، راهکار [۱۲] گراف وظایف کاربرد را خوشه‌بندی کرده و بر اساس تعداد خوشه‌ها، معماری توری را به شبکه‌های توری با ابعاد کوچک‌تر تقسیم می‌کند. برای نگاشت خوشه‌ها به زیرشبکه‌های توری مربوط از فرمول‌بندی ILP استفاده شده است. در انتها همه زیرشبکه‌های توری برای تعیین راه حل نهایی ادغام می‌شوند و بنابراین زمان پردازنده، بهبود پیدا می‌کند اما هزینه ارتباطی خوبی به دست نمی‌آید.

الگوریتم‌های نگاشت ایستای مبتنی بر جستجو نیز به دو گونه تقسیم می‌شوند: ۱) نگاشت مبتنی بر جستجوی قطعی[10] و ۲) نگاشت مبتنی بر جستجوی اکتشافی[11]. الگوریتم‌های نگاشت قطعی تمام فضای جستجو را بررسی می‌کنند که الگوریتم BB[12] جزء این دسته از الگوریتم‌های نگاشت است. این الگوریتم با جستجوی راه حل در شاخه‌های درخت و محدودکردن راه حل‌های غیرمجاز، نگاشت مناسب را پیدا می‌کند [۶]. اما این الگوریتم برای حل مسائل کوچک‌تر مناسب است زیرا زمان جستجو با افزایش اندازه مسأله به‌طور نمایی افزایش می‌یابد [۵]. در راهکار [۱۳] یک نگاشت انرژی آگاه از تراکم و در [۱۴] یک نگاشت آگاه از انرژی و کارایی[13] با استفاده از الگوریتم شاخه و حد برای معماری شبکه روی تراشه با هدف کمینه‌کردن انرژی ارتباطی ارائه شده که محدودیت‌های طراحی را از طریق ذخیره پهنای باند برآورده می‌کند. الگوریتم به‌طور هم‌زمان یک نگاشت بهینه و یک تابع مسیریابی را ارائه می‌کند که علاوه بر رعایت محدودیت پهنای باند، انرژی ارتباطی را نیز بهبود می‌بخشد.

در [۱۵] از محدودیت پهنای باند و روش شاخه و حد برای نگاشت استفاده شده است. در این راهکار ابتدا با استفاده از درخت جستجو، یک مجموعه نگاشت با کمترین هزینه ارتباطی جستجو می‌شود و در گام بعدی، نگاشت‌هایی که کمترین تأخیر و توان مصرفی را داشته باشند به عنوان راه حل نهایی باقی می‌مانند. با این حال روش‌های جستجوی

1. Harmony Search
2. Exact Mapping
3. Search Based Mapping
4. Integer Linear Programming
5. Non-Integer Linear Programming
6. Mixed Integer Linear Programming
7. Tabu Search
8. Symmetric Multi-Processing
9. Contention-Aware
10. Deterministic Search
11. Heuristic Search
12. Branch-and-Bound
13. Energy and Performance Aware Mapping



شکل ۱: ماتریس حافظه، متغیر تصمیم و تابع برازش.

اکتشافی تنها راه حل‌های کارا در حل مسائل NP-hard و در اندازه‌های واقعی هستند زیرا در اندازه‌های کوچک‌تر می‌توان با روش‌های قطعی، راه حل بهینه را ایجاد کرد [۶].

هنگامی که جستجوی کامل فضای راه حل و روش‌های قطعی به دلیل بزرگ‌بودن اندازه شبکه و پیچیدگی زمانی الگوریتم قادر نباشد تا راهکار مناسب را در زمانی مناسب ایجاد کند، رویکردهای فرااکتشافی پاسخ مناسب را فراهم می‌آورند [۱۶]. این گروه از روش‌ها خود به دو دسته اکتشافی قابل تغییر[1] و روش‌های سازنده[2] تقسیم می‌شوند. این رویکردها رویکردها آن قدر راه حل‌های نگاشت را تغییر می‌دهند تا به راه حل بهینه برسند. به عنوان نمونه می‌توان به روش‌های تکاملی مانند الگوریتم ژنتیک، الگوریتم بهینه‌سازی ازدحام ذرات (PSO)[3]، الگوریتم کلونی مورچه (ACO)[4] و موارد دیگر اشاره کرد [۱۴]. در [۱۷] و [۱۸] نویسندگان با استفاده از الگوریتم ژنتیک بر اساس حذف ایزومورفیسم، روشی را برای بهینه‌سازی شبکه روی تراشه تعریف می‌کنند. در این رویکرد برای حذف هم‌شکلی توالی نگاشت و تسریع همگرایی جمعیت با انتخاب خودکار عملگرها در طول فرایند نگاشت، پایداری و سرعت همگرایی الگوریتم بهبود می‌یابد. در پژوهش ارائه‌شده توسط خیونگ و همکارانش [۱۹]، یک الگوریتم کوانتومی کلونی مورچه ارائه شده تا مسأله نگاشت در شبکه‌های روی تراشه را حل کند. این رویکرد از بیت کوانتومی برای جایگذاری فرومون کلونی مورچه استفاده می‌کند [۲۰]. روند به‌روزرسانی فرومون در این ساختار به‌صورت پویا برای کاهش همگرایی زودهنگام الگوریتم کلونی مورچه به‌طور مؤثر و بر اساس استراتژی چرخش فاز تطبیقی است. بهینه‌سازی هسته‌های موجود در شبکه‌های روی تراشه یکی از مهم‌ترین گام‌های طرح ارائه‌شده است.

در راهکار گروه تحقیقاتی کانولو [۲۱]، یک روش سلسله‌مراتبی تکرارشونده برای نگاشت وظایف بر روی شبکه بر تراشه ناهمگن در زمان اجرا ارائه و کاربرد مورد نظر با مجموعه‌ای از جریان‌ها یا وظایف ارتباطی مدل شده است. برای کاهش انرژی مصرفی، الگوریتم نگاشت سعی می‌کند که وظایف نزدیک را به یکدیگر مرتبط کند.

در روش لی و همکارش [۲۲] یک الگوریتم ژنتیک دومرحله‌ای برای نگاشت گراف کاربرد بر روی هسته‌های پردازشی شبکه بر تراشه، پیشنهاد شده است؛ به‌طوری که زمان اجرای کلی گراف وظایف کمینه گردد. برای تخمین زمان اجرا از یک مدل تأخیر بر مبنای تأخیر سیستم و تأخیر یال استفاده شده است. تأخیر سیستم گراف وظیفه برابر تأخیر مسیر بحرانی در زمان اجرا و تأخیر یال است. تأخیر یال برابر است با میزان تأخیر انتقال داده بین دو رأس که بر روی گره‌های مختلف شبکه نگاشت شده‌اند.

در [۲۳] یک الگوریتم با عنوان UMARS[5] ارائه شده که نگاشت، تخصیص مسیر و تخصیص برش‌های زمانی را برای کمینه‌کردن انرژی ارتباطی هم‌زمان انجام می‌دهد. در این روش، هسته‌ها روی همبندی شبکه روی تراشه، نگاشت شده و ارتباط‌های آنها مسیریابی می‌شود. برش‌های زمانی TDMA[6]، محدودیت‌های کاربرد را برآورده می‌سازد.

در روش پیشنهادی شن و همکارانش [۲۴] یک الگوریتم بهینه‌سازی و نگاشت دوجمله‌ای برای کاهش هزینه سخت‌افزاری شبکه روی تراشه به نام BMAP ارائه شده است. این الگوریتم خیلی سریع و مؤثر می‌باشد و پیچیدگی محاسباتی کمتری نسبت به NMAP [۲۵] و PSMAP [۲۶] دارد.

الگوریتم BMAP از ۳ مرحله تشکیل شده است: رتبه‌بندی هسته‌های پردازشی، ادغام مجموعه هسته‌ها و دوباره‌سازی مجموعه هسته‌ها. رتبه‌بندی هسته‌ها به پهنای باند ارتباطی بین آنها بستگی دارد. رتبه هر هسته برابر با جمع پهنای باند ارتباطی آن هسته با سایر هسته‌ها و از آن هسته‌ها به هسته هدف است [۲۷].

ONMAP [۲۸] یک روش اکتشافی نگاشت تقریباً بهینه مبتنی بر جستجو پیشنهاد می‌کند که از روش بهینه‌سازی دقیق ماژولار NMAP بهره‌برداری می‌نماید. الگوریتم ترکیبی پیشنهادی، مصرف انرژی ارتباط بین پردازنده روی تراشه را به حداقل می‌رساند و پارامترهای عملکرد شبکه اتصال را بهینه می‌کند. اخیراً دو الگوریتم BEMAP [۲۹] و SBMAP [۳۰] معرفی شده‌اند که مبتنی بر یک الگوریتم نگاشت دقیق مبتنی بر تکنیک ترکیبی BB نگاشت برنامه‌های کاربردی تعبیه‌شده در زمان واقعی را بهبود می‌بخشند. این دو روش با بهره‌گیری از تکنیک‌های بهینه‌سازی سیستماتیک ماژولار، یک راه حل بهینه‌شده چندهدفه را برای مسئله نگاشت طرح‌های NOC به کار می‌گیرند. نتایج بهبود عملکرد هر دو مکانیزم تحت معیارهای چندین برنامه جاسازی شده در زمان واقعی، بهبود در تأخیر و صرفه‌جویی در مصرف برق شبکه را برای همبندی‌های دوبعدی نشان می‌دهد.

در این مقاله به‌منظور بهینه‌سازی مسأله نگاشت با درنظرگرفتن هزینه‌های محاسباتی، یک مدل بهبودیافته از الگوریتم جستجوی نگاشت با استفاده از رویکرد جستجوی هارمونی در شبکه‌های روی تراشه معرفی می‌شود.

۳- الگوریتم جستجوی هارمونی

جستجوی هارمونی، از ساده‌ترین و جدیدترین روش‌های فراابتکاری است که در فرایند جستجوی جواب بهینه در مسائل بهینه‌سازی مورد استفاده قرار گرفته است. این روش اولین بار توسط گیم در سال ۲۰۰۱ میلادی ارائه شد [۳۱]. در منطق این رویکرد، تلاش برای به‌دست‌آمدن یک هماهنگی بین نوازنده‌های گروه ارکستر در یک هم‌نوازی مانند پیداکردن حل بهینه در مسائل بهینه‌سازی است. از مزایای این الگوریتم، همگرایی سریع به دلیل ساختار مناسب آن است. با این حال دارای معایبی نیز می‌باشد. از آن جمله گیرافتادن در نقاط بهینه محلی به دلیل جستجو با تنوع کم در تکرارهای پایانی الگوریتم که برای رفع آن از تکنیک شروع دوباره و تغییر در قواعد الگوریتم به خصوص در تکرارهای پایانی استفاده می‌شود. فرایند الگوریتم جستجوی هارمونی در شکل ۱ آمده است.

بر مبنای منطق الگوریتم جستجوی هارمونی، هر راه حل، یک هارمونی نامیده می‌شود و با یک بردار N بعدی نمایش داده می‌شود. الگوریتم دارای ۵ گام به شرح زیر است:

1. Transformative Heuristics
2. Constructive Heuristics
3. Particle Swarm Optimization
4. Ant Colony Optimization
5. Unified Mapping, Routing and Slot Allocation Algorithm
6. Time Division Multiple Access

۱) مقداردهی اولیه مسأله بهینه‌سازی و پارامترهای اولیه
۲) مقداردهی حافظه هارمونی
۳) ایجاد یک هارمونی جدید بهبودیافته
۴) به‌روزآوری حافظه هارمونی
۵) تکرار گام‌های ۳ و ۴ تا زمانی که شرط پایانی فرارسد یا تکرارها پایان پذیرد.

اکنون به توضیح مراحل اجرای گام‌های الگوریتم جستجوی هارمونی برای دستیابی به مقادیر بهینه جواب‌های مسأله می‌پردازیم.

گام اول شامل ساخت یک ماتریس حافظه هارمونی خام و اعمال تکثیر بر روی آن است. متغیرهای تصمیم تصادفی، درایه‌های این ماتریس را تشکیل می‌دهند. بردارهای حل ذخیره‌شده در حافظه هارمونی بر اساس مقدار تابع برازش مرتب می‌شوند. این هارمونی قادر است تا اطلاعات مربوط به هر پاسخ را ذخیره کند. الگوریتم تابع هدف $f(x^i)$ و بردار جواب هارمونی x^i در این مرحله تعیین می‌شود. پارامتر $HMCR$ میزان احتمال استفاده از حافظه هارمونی و پارامتر PAR میزان احتمال ایجاد تغییرات جزئی در یک مؤلفه را نشان می‌دهد و این ویژگی تقریباً با ویژگی جهش در الگوریتم ژنتیک همسان می‌باشد.

گام دوم ایجاد یک هارمونی جدید از حافظه هارمونی و شامل حلقه اصلی الگوریتم است. ماتریس حافظه هارمونی، آرایه‌ای است که برای هارمونی‌های جدید ساخته می‌شود. الگوریتم جستجوی هارمونی در انتخاب مقادیر متغیرهای تصمیم از سه قاعده بازبینی حافظه، تعدیل گام و آرایش تصادفی پیروی می‌کند. در بازبینی حافظه، مقادیر متغیرهای تصمیم می‌توانند با احتمال $HMCR$ از مقادیر متغیرهای تصمیم در ماتریس حافظه انتخاب شوند. به هنگام تعدیل گام، مقادیر متغیرهای تصمیم می‌توانند با احتمال $HMCR-۱$ از دامنه تعریف مقادیر به صورت تصادفی انتخاب شوند. متغیرهایی که بر اساس قاعده بازبینی حافظه تولید شده‌اند می‌توانند با احتمال $HMCR\times PAR$ مقادیری در همسایگی مقادیر فعلی اختیار کنند. این بدان معناست که مقادیر این دسته از متغیرها به احتمال $HMCR\times(۱-PAR)$ بدون تغییر باقی می‌ماند. ()ran مقداری تصادفی است که در بازه [۰,۱] انتخاب شده و bw یک پهنای باند دلخواه و ماکسیمم مقدار تغییر ایجادشده در متغیر انتخابی می‌باشد. آرایه فوق به میزان nNew، ظرفیت نگهداری هارمونی‌های جدید را دارد. شکل ۱ ماتریس حافظه را نشان می‌دهد. پارامترهای x، N و $f(x)$ به ترتیب نشان‌دهنده متغیر تصمیم، تعداد متغیرهای تصمیم و تابع برازش هستند.

گام سوم یک راه حل جدید را با توجه به تعداد عملگرهای مورد نیاز ایجاد می‌کند. در هر بار تکرار حلقه مرحله قبل، موقعیت هارمونی به شکل تصادفی پر می‌شود. برای ایجاد مقدار برای متغیر iام، ابتدا یک عدد تصادفی بین صفر و یک تولید و با $HMCR$ مقایسه می‌شود. اگر عدد انتخاب‌شده، کوچک‌تر از $HMCR$ باشد، متغیر iام از ماتریس حافظه مقدار می‌گیرد و در غیر این صورت یک مقدار تصادفی از فضای جستجو به آن اختصاص داده می‌شود. در حالت انتخاب مقدار از ماتریس حافظه، عدد تصادفی دیگری تولید و با PAR مقایسه می‌شود. در صورتی که عدد تصادفی کوچک‌تر از PAR باشد برای تعیین مقدار تغییر بر روی متغیر انتخاب‌شده، پارامتر دیگری به نام bw تعریف می‌شود. مقدار جدید با توجه به (۱) به‌دست می‌آید. به این ترتیب متغیر انتخاب‌شده از ماتریس حافظه به اندازه‌ای کوچک تغییر پیدا می‌کند

$$X^{New}X^{old}+bw+\varepsilon \qquad (۱)$$

در گام چهارم، حافظه هارمونی به‌روزآوری می‌شود، هارمونی جدید در حافظه HM قرار می‌گیرد و تمام متغیرهای یک جواب یا هارمونی ایجاد می‌شود. سپس ارزش هارمونی با توجه به تابع برازش، محاسبه و با بدترین هارمونی موجود در حافظه ماتریس مقایسه می‌شود. اگر بردار هارمونی جدید از بدترین بردار هارمونی موجود در حافظه هارمونی HM بر مبنای تابع هدف انتخابی بهتر باشد، بدترین هارمونی از مجموعه HM کنار گذاشته می‌شود.

در گام پنجم، شرط توقف الگوریتم کنترل می‌شود. اگر معیار توقف ارضا شود، محاسبه متوقف می‌گردد و در غیر این صورت، مراحل ۴ و ۵ تکرار می‌شوند. دستیابی به مقدار بهینه، عدم تغییر در بهترین هارمونی ماتریس حافظه یا رسیدن به تعداد تک‌سازهای مشخص از شروط پایان کار الگوریتم هستند. با تکرار این پنج مرحله، الگوریتم به مرور به مقادیر بهینه نزدیک‌تر شده و جواب‌های مسأله بهبود می‌یابد.

در سال ۲۰۱۱، Z. W. Geem، خالق الگوریتم جستجوی هارمونی، نسخه جدیدی از الگوریتم را منتشر کرد (IHS)[1] که نیازی به تنظیم پارامترها ندارد. IHS شامل یک تغییر دینامیکی از نرخ درنظرگرفتن حافظه هارمونی ($HMCR$) [۳۲] بود که تنها مقدار مورد نیاز برای آن، حداکثر تعداد تکرار است. این ویژگی می‌تواند نیاز به دانش نظری در مورد جستجوی هارمونی را مرتفع سازد. در ابتدا الگوریتم پس از اجرا با تعداد کمی تکرار، ماتریس دیگری را با عنوان حافظه عملکردی و با ابعاد مشابه ماتریس حافظه هارمونی ایجاد می‌کند. ماتریس جدید، منشأ عناصر حافظه هارمونی است. اگر عنصری از حافظه هارمونی از مکانیسم بداهه‌سازی حاصل شود، حافظه عملکردی این اطلاعات را ذخیره می‌کند. هر بار که یک راه حل جدید، بهتر از راه حل‌های ذخیره‌شده در حافظه هارمونی، تولید می‌شود، حافظه کاربردی تجدید می‌گردد و در همان زمان، مقادیر پارامترهای $HMCR$ و PAR تغییر می‌کنند. همچنین از آنجایی که پارامترها به تعداد عناصر موجود در حافظه هارمونی مربوط به یک مبدأ خاص بستگی دارند، در طول فرایند حل به هر یک از متغیرهای تصمیم‌گیری، مقدار متفاوتی از این پارامترها اختصاص می‌یابد [۳۳]. اگر پارامتر $HMCR$ به مقدار ۱ و یا پارامتر PAR به مقدار صفر برسد، در طول فرایند حل ثابت می‌مانند، زیرا الگوریتم باید بتواند از همگرایی زودهنگام در زمانی که این پارامترها به ترتیب مقادیر نزدیک به ۱ و ۰ دارند، جلوگیری کند. این امر با استفاده از معادلات کنترلی (۲) و (۳) انجام می‌شود

$$HMCR=[HMCR+\phi\times ran(-۱,۱)] \qquad (۲)$$

$$PAR=[PAR+\phi PAR\times ran(-۱,۱)] \qquad (۳)$$

عبارات $\phi HMCR$ و ϕPAR ضرایب نویز با مقادیر پیشنهادی ۰٫۰۵ و ۰٫۱ هستند. اگر عبارات فوق بین [۰,۱] نباشد، PAR و یا $HMCR$ مقدار خود را حفظ می‌کند. البته باید توجه داشت که حتی اگر نیازی به تخصیص مقادیر به پارامترهای مشکل نباشد، باز هم باید ضرایب نویز ϕ را تنظیم کرد. با این حال، [۳۴] احتمال موفقیت برای مقادیر کم $HMCR$ و مقادیر بالای PAR را بسیار پایین می‌داند. نویسندگان در این مقاله با پیاده‌سازی مجدد الگوریتم، همه ترکیبات ممکن برای اندازه حافظه هارمونی، میزان درنظرگرفتن حافظه هارمونی و نرخ تنظیم گام را در مجموع ۳۶ حالت مختلف بررسی کرده‌اند و نتایج نشان داده که تنها با

1. Improved Harmony Search

نرخ درنظرگرفتن حافظه هارمونی ۰/۹ همراه با نرخ تنظیم زیر و بم ۰/۰۱ یا ۰/۱ گام می‌تواند راه حل‌های بهینه را ایجاد کند.

۳-۱ فرمول‌سازی مسأله نگاشت

مسأله نگاشت در دسته‌بندی مسائل NP-hard است و بنابراین برای حل آن در اندازه‌های عملی با قابلیت اجرا، اغلب از شیوه‌های جستجوی فرااکتشافی استفاده می‌شود. در روش‌های فرااکتشافی، شیوه‌های بهینه‌سازی و جستجو اغلب شبه‌تصادفی هستند که کشف و استخراج فضای راه حل بر اساس تجربه‌های به‌دست‌آمده در گذشته انجام می‌شود. در تکنیک‌های مکاشفه‌ای سازنده، راه حل‌های جزئی به‌صورت پشت سر هم تولید می‌شوند و در پایان، راه حل نهایی نگاشت به دست می‌آید. در روش‌های فرااکتشافی برای تصمیم‌گیری در مورد نگاشت هسته‌های شرکت‌کننده در یک برنامه کاربردی، آن را می‌توان به شکل یک گراف اصلی نشان داد که به شرح زیر تعریف می‌شود:

تعریف ۱) گراف $G(C,E)$، گرافی جهت‌دار است و هر رأس $c_i \in C$ نشان‌دهنده یک هسته و لبه مستقیم $e_{i,j} \in E$ نشان‌دهنده ارتباط بین هسته‌های c_i و c_j است. وزن لبه $e_{i,j}$ که با $comm_{i,j}$ نشان داده می‌شود، پهنای باند مورد نیاز ارتباط از c_i تا c_j را نشان می‌دهد.

تعریف ۲) گراف توپولوژی NoC یک گراف جهت‌دار $P(U,F)$ است. هر رأس $u_i \in U$ نشان‌دهنده یک گره در همبندی و لبه جهت‌دار $f_{i,j} \in F$ نشان‌دهنده یک پیوند فیزیکی بین رئوس i_u و u_j است. وزن لبه $f_{i,j}$ که به صورت $bw_{i,j}$ نشان داده می‌شود، نشان‌دهنده پهنای باند موجود در عرض لبه $f_{i,j}$ است.

نگاشت از گراف هسته $G(C,E)$ بر روی نمودار همبندی $P(U,F)$ توسط تابع نگاشت $C \to U$ تعریف می‌شود؛ به‌طوری که تابع $\forall c_i \in C \exists u_j \in U \,\&\, map(c_i)$، هسته c_i را به روتر u_i مرتبط می‌کند. با فرض اتصال هر مسیریاب به یک هسته، نگاشت تنها با شرط $|C| \le |U|$ تعریف می‌شود.

با توجه به [۳۵] هزینه کل ارتباط برای کاربرد تحت این نگاشت به‌صورت d^k تعریف می‌شود؛ به‌طوری که اگر $d^k, k = ۱,۲,\ldots,|E|$ باشد، پهنای باند مورد نیاز به‌صورت (۴) ارزیابی می‌شود

$$bw = d^k \left| value\ d^k = comm_{i,j} \text{ and } e_{i,j} \in E \right| \qquad (۴)$$

پیوند بین دو مسیریاب مجزای u_i و u_j دارای حداکثر پهنای باند $bw_{i,j}$ است. کل آنچه که از طریق چنین پیوندی دریافت می‌شود نباید از این پهنای باند تجاوز کند. مقداری با عنوان $x_{i,j}^k$ توسط (۵) تعریف می‌شود که نشان‌دهنده ارزش پهنای باند مورد نظر از طریق پیوند $u_i u_j$ است

$$x_{i,j}^k = \begin{cases} value\ d^k \text{ if } link(u_i u_j) \in \\ \quad path(source\ d^k, \sin k\ d^k) \\ \cdot \qquad \text{otherwise} \end{cases} \qquad (۵)$$

در غیر از این موارد، مقدار آن صفر در نظر گرفته می‌شود. با این حال تمام راه حل‌های نگاشت باید (۶) را برآورده سازند

$$\sum_{k=۱}^{|E|} x_{i,j}^k \le bw_{i,j} \text{ for } i,j \in \{۱,۲,\ldots,|U|\} \qquad (۶)$$

اگر همه محدودیت‌های پهنای باند برآورده شوند، هزینه ارتباطی یک راه حل نگاشت برابر (۷) خواهد بود

$$T = \sum_{k=۱}^{|E|} value\ d^k . hopcount(source\ d^k . \sin k\ d^k) \qquad (۷)$$

که در آن $hopcount$ تعداد پرش بین گره‌های همبندی است.

تعریف ۳) توان مصرفی شبکه در فرایند محاسبه و پردازش و انتقال داده در لینک‌های شبکه با (۸) تعریف می‌شود و (۹) پردازش داده‌ها را در اندازه یکسان Δ و تنها برای هسته‌های IP نشان می‌دهد

$$E_{NOC} = \sum_{i}^{ip} E_i^{ip} + \sum_{j}^{path} E_j^{path} \qquad (۸)$$

$$E_{\Delta}^{IP} = \rho_{IP} \times \Delta \qquad (۹)$$

با فرض همگن‌بودن هسته‌ها، توان مصرفی ناشی از اجرای وظایف بر روی هر هسته با توجه به [۱۲] به‌صورت (۱۰) خلاصه می‌شود

$$E_{bit} = E_{Sbit} + E_{Bbit} + E_{Wbit} + E_{Lbit} \qquad (۱۰)$$

پارامترهای ثابت E_{bit}، E_{Sbit}، E_{Bbit} و E_{Wbit} به‌ترتیب بیانگر انرژی مصرفی برای انتقال یک بیت داده بین دو گره شبکه، انرژی مصرفی سوئیچ، انرژی مصرفی در ناحیه بافر و انرژی مصرفی در سیم‌های ارتباطی درون یک مسیریاب هستند. E_{Lbit} انرژی مصرفی بین لینک‌های ارتباطی بین دو مسیریاب در همبندی مورد نظر است. از پارامتر انرژی مصرفی در انتقال داده در لینک‌های مرتبط صرف نظر می‌شود زیرا در محیط داخلی تغییر نخواهد کرد. به‌منظور ساده‌سازی با جایگزینی E_R به عنوان انرژی مصرفی توسط مسیریاب به‌صورت (۱۱) خواهیم داشت

$$E_{i,j}bit = n_{i,j} \times E_R + (n_{i,j} - ۱) \times E_{Lbit} \qquad (۱۱)$$

$E_{i,j}bit$ انرژی مصرفی برای انتقال یک بیت از گره i به j و $n_{i,j}$ تعداد گره‌های مسیریابی‌شده برای انتقال یک بیت از گره مبدأ است. بنابراین انرژی مصرفی بین گره‌های Ti به Tj در همبندی مورد نظر از (۱۲) به‌دست می‌آید

$$E_{i,j} = V_{i,j} \times E_{i,j}bit \qquad (۱۲)$$

که $V_{i,j}$ ترافیک بین دو گره مبدأ و مقصد در نظر گرفته می‌شود.

تعریف ٤) تأخیر پردازش داده‌ها در هسته IP یا زمان لازم برای انتقال از گره $a(xa,ya)$ به گره $b(xb,yb)$ از طریق مسیر d_{ab} در کوتاه‌ترین مسیر با توجه به [۳۶] به‌صورت (۱۳) محاسبه می‌شود

$$D = \sum_i D_i^{core} + \sum_j D_j^{path} \qquad (۱۳)$$

که D_i^{core} زمان صرف‌شده برای پردازش کار در iامین هسته IP و D_j^{path} زمان مصرفی برای انتقال داده از طریق پیوند j را نشان می‌دهد. در (۱۲) پردازش Δ بیت داده بر روی یک هسته IP خاص با (۱۴) تعریف شده است

$$D_{\Delta}^{ip} = \rho_{cd} . \Delta \qquad (۱۴)$$

که ρ_{cd} یک ثابت مربوط به نوع هسته IP است. زمان تلف‌شده برای انتقال داده، عمدتاً از تأخیر فیزیکی در خط اتصال و تأخیر تجزیه بسته داده ناشی می‌شود [۳۶]. بنابراین در شرایط استفاده از کوتاه‌ترین مسیر برای محاسبه زمان صرف‌شده در فرایند انتقال از (۱۵) استفاده می‌شود

$$D_{\Delta}^{path} = \Delta . [۲\rho_{NI} + \rho_L . d_{ab} + \rho_R (d_{ab} + ۱)] \qquad (۱۵)$$

که ρ_{NI}، ρ_L و ρ_R به ترتیب ثابت‌های NI، خط اتصال و مسیریاب در مورد زمان انتقال هستند.

۳-۲ بهینه‌سازی تابع هزینه

پارامترهای توان مصرفی و تأخیر هم‌واحد نیستند. بهینه‌سازی، این نیاز

را ایجاد می‌کند که این دو پارامتر هم‌ارزش شوند. تابع هزینه بر مبنای وزن‌دهی به لینک‌های ارتباطی در شبکه روی تراشه است. مقدار پارامتر محاسبه‌شده در هر جفت گره از کمترین عدد به‌دست‌آمده برای هر مؤلفه کسر و بر کمترین عدد به‌دست‌آمده تقسیم می‌شود. یکی از جنبه‌های نوآوری در رویکرد پیشنهادی، محاسبه تابع هزینه با امکان کاهش هم‌زمان تأخیر و توان مصرفی است. زیرا اگر تنها توان مصرفی مد نظر باشد، ممکن است منجر به ازحام در شبکه و تأخیر در اجرای برنامه شود و در نتیجه بر کیفیت خدمات تأثیر می‌گذارد. عکس این حالت نیز امکان‌پذیر است. بنابراین در رویکرد ما یک وزن نرمال‌شده در توان و تأخیر جهت یکسان‌سازی ضرب می‌شود. تابع هزینه مورد استفاده در (۱۶) نشان داده شده است

$$cost = \sum_{i=1}^{n} distance \times w_{np} \times E_{i,j} + w_{nd} \times D_{\Delta}^{path} \qquad (۱۶)$$

در این تابع، D_{Δ}^{path} از (۱۱) و $E_{i,j}$ از (۸) محاسبه می‌شود. w_{np} وزن نرمال‌شده توان مصرفی و w_{nd} وزن نرمال‌شده تأخیر است. فرض گردیده که شبکه مورد نظر دارای n لینک ارتباطی است که برای هر لینک مقادیر توان و تأخیر مشخص است. مسافت بین دو گره در شبکه هم با توجه به مسیریابی X-Y محاسبه می‌شود. برای گره‌هایی که با هم در ارتباط هستند، این مقدار ۱ و برای گره‌هایی که با یکدیگر ارتباطی ندارند، این مقدار ۰ در نظر گرفته می‌شود. این مقدار در توان و تأخیر با وزن‌های مشخص ضرب شده و مجموع آنها تابع هزینه را می‌سازد. وزن‌های نرمال‌شده مذکور به‌منظور یکی‌سازی توان و تأخیر است؛ به نحوی که بتوان این دو پارامتر را با یکدیگر در تابع هزینه جمع نمود.

۳-۳ فرایند نگاشت با الگوریتم جستجوی هارمونی

چگونگی دستیابی سریع و مؤثر به نتایج نگاشت، بهینه‌سازی مستمر، محاسبات ریاضیاتی کم، مفهوم ساده، پارامترهای کمتر و اجرای آسان‌تر، الگوریتم جستجوی هارمونی را برای حل مسائل بهینه‌سازی گسسته و پیوسته متمایز می‌کند. استفاده از دو بردار حل در هر نسل و استفاده از همه راه حل‌های موجود در حافظه، انعطاف‌پذیری الگوریتم را در جستجوی فضاهای حل افزایش می‌دهد. الگوریتم در مدت زمان مناسبی فضاهای حل را با محدوده عملکرد بهتر شناسایی می‌کند. با این حال همان گونه که قبلاً نیز بیان شد، در صورتی که مسأله مورد مطالعه از بهینگی محلی[1] برخوردار باشد دچار مشکل می‌شود و نمی‌تواند به بهینگی سراسری[2] برسد. دلیل این مشکل، عدم کارایی مناسب الگوریتم در اجرای جستجوی محلی در مسائل بهینه‌سازی گسسته است.

در رویکرد پیشنهادی بر اساس راهکار [۳۶] و نقش پارامتر PAR در ایجاد تنوع برای نقطه شروع جستجو و ضریب تأثیر bw در نرخ همگرایی و جستجوی محلی، با روشی مبتنی بر الگوریتم جستجوی هارمونی پس از یافتن نگاشت‌های قبلی، نگاشت بهینه جایگزین می‌شود. در این قسمت به توضیح گام‌های راهکار حل مسأله جستجوی نگاشت پیشنهادی با استفاده از الگوریتم جستجوی هارمونی می‌پردازیم. فلوچارت جریان الگوریتم جستجوی هارمونی با الگوی مسأله نگاشت در شکل ۲ نشان داده شده است. در ذیل به توضیح گام‌های اجرای الگوریتم پرداخته شده است.

گام اول) در مرحله راه‌اندازی مسأله بهینه‌سازی و پارامترهای کنترل، a مجموعه‌ای از متغیر تصمیم (a_x) فرض شده است. مقدار هر متغیر تصمیم باید در محدوده $a_{x1} \le a_x \le a_{xh}, x = ۱,۲,۳,\ldots$ قرار گیرد و مقادیر HMS، $HMCR$، PAR و NI نیز مقداردهی اولیه شوند. در ابتدا فرض می‌شود که x^i برداری از جواب‌های هارمونی است که در مسأله نگاشت بر روی همبندی شبکه روی تراشه در اولین گام وجود دارد. هر بردار شامل تعدادی متغیر تصمیم‌گیری x_j^i است که در آن i بین ۱ تا حداکثر، اندازه حافظه هارمونی HMS و j تعداد گره‌های گراف وظایف است. در هر مرحله از الگوریتم تابع هدف باید مینیمم شود. مسأله بهینه‌سازی با (۱۷) تعریف شده است

$$\begin{aligned} &find\ \vec{x} = (x_1, x_2, \ldots, x_n) \text{ to minimize } f(\vec{x}), \\ &\text{subject to : } g_i(\vec{x}) \le 0,\ i = ۱,۲,\ldots,M, \\ &h_j(\vec{x}) \le 0,\ i = ۱,۲,\ldots,N \end{aligned} \qquad (۱۷)$$

که $f(\vec{x})$ تابع هدف، $g_i(\vec{x})$ تابع قیود مساوی، $h_j(\vec{x})$ تابع قیود نامساوی و X دسته‌ای از گره‌های x_i در محدوده ممکن مقادیر برای تصمیم‌گیری است که در این راهکار شامل تمام گره‌های قابل نگاشت است. در این مرحله پارامترهای الگوریتم هارمونی نیز تعریف می‌شوند که شامل اندازه حافظه هارمونی HMS و یا تعداد بردارهای جواب در حافظه هارمونی، نرخ درنظرگرفتن از حافظه هارمونی[3] $HMCR$، نرخ تنظیم زیر و بمی[4] PAR، تعداد متغیرهای تصمیم‌گیری (N) و تعداد بداهه‌سازی[5] NI یا ناحیه توقف الگوریتم هستند. نتیجه، اختصاص یک جمعیت اولیه از بردارهای هارمونی به محلی از حافظه است؛ به‌طوری که تمام بردارهای جواب نگاشت وظیفه در آن ذخیره شوند. اندازه حافظه هارمونی نیز برابر با تعداد بردارها در حافظه هارمونی در نظر گرفته شده است.

گام دوم) در گام دوم حافظه هارمونی (HM) به‌طور دلخواه با (۱۸) و در محدوده $a_{x1} \le a_x \le a_{xh}, x = ۱,۲,۳,\ldots$ مقداردهی اولیه می‌شود

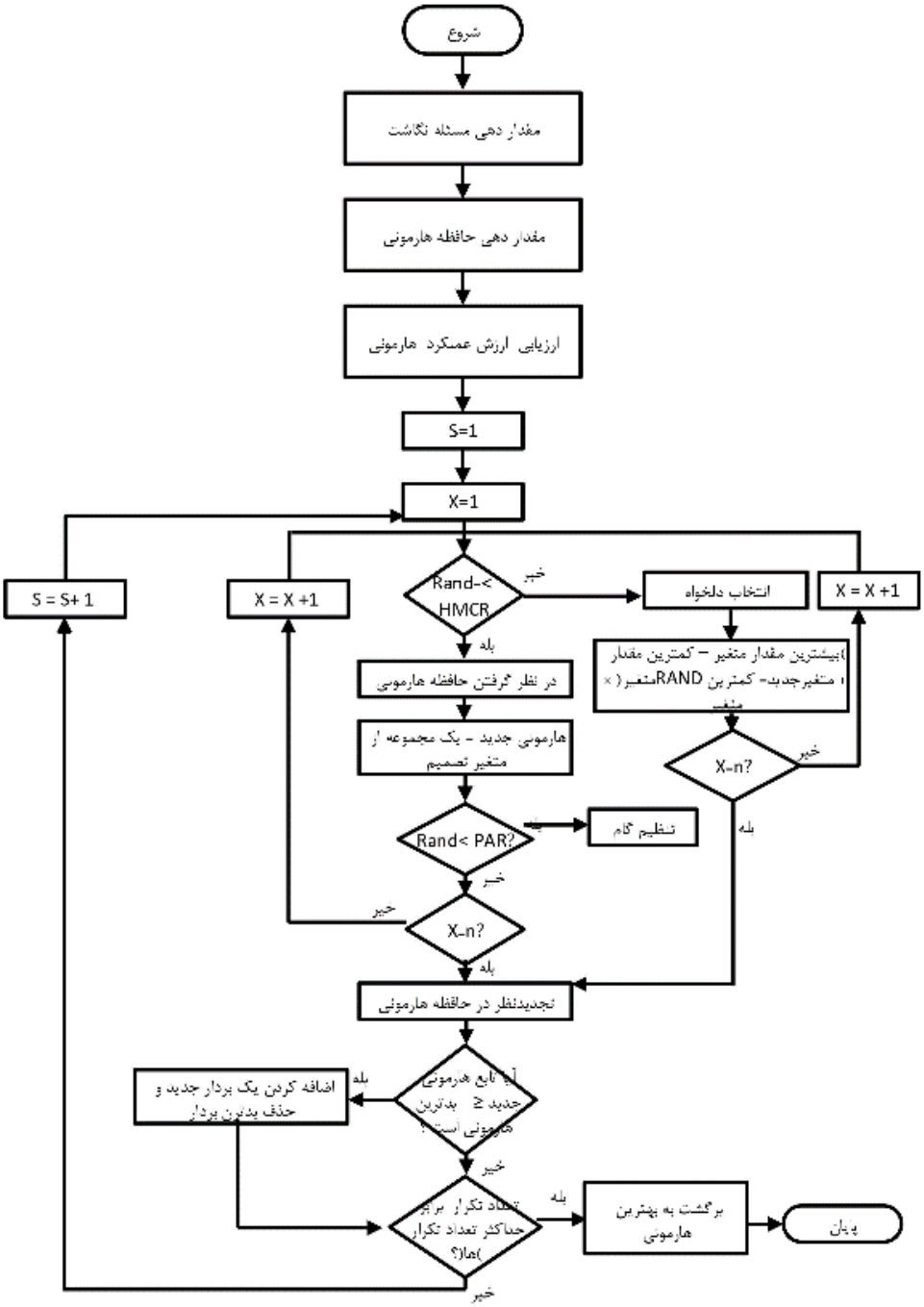
$$a_{xi} = a_{x1} + RAND \times (a_{xh} - a_{x1}),\ x = ۱,۲,۳,\ldots,HMS \qquad (۱۸)$$

$RAND$ تابعی تصادفی است که هر مقدار دلخواه را از [۰,۱] می‌گیرد.

در شکل ۳ نشان داده شده که چگونه در طول فرایند اولیه‌سازی، تعداد معینی از راه حل‌ها به‌طور یکنواخت به صورت تصادفی تولید و در حافظه هارمونی ذخیره می‌شوند. تعداد دقیق راه حل‌های تولیدشده با اندازه حافظه هارمونی مشخص می‌شود.

گام سوم) برای ایجاد یک هارمونی جدید از بردار هارمونی $a' = (a_1', a_2', a_3', \ldots, a_n')$ از سه پارامتر کنترلی تنظیم گام، انتخاب دلخواه و درنظرگرفتن حافظه و پهنای باند استفاده می‌شود. پهنای باند، فاصله لازم برای اصلاح بردار هارمونی می‌باشد که نقش مهمی را در مرحله تنظیم گام ایفا می‌کند. به دلیل تأثیر دو مکانیزم انتخاب تصادفی و تنظیم گام، مقادیر جداگانه‌ای برای هر یک از متغیرهای تصمیم تولید می‌شوند. این مکانیزم با احتمال $HMCR - ۱$ خواهد بود و در موارد غیر، با احتمال $HMCR$، مقداری از حافظه هارمونی گرفته می‌شود. پس از آن، این مقدار با تنظیم گام با احتمال PAR اصلاح می‌شود. تنظیمات گام، مقدار را با احتمال ۵۰٪ به یکی از دو مقدار همسایه تغییر می‌دهد و در صورتی که فقط ۱ مقدار همسایه وجود داشته باشد، با احتمال ۵۰٪ بدون تغییر باقی می‌ماند و در غیر این صورت به مقدار همسایه یکتا تغییر می‌کند.

1. Optimum Local
2. Optimum Global
3. Harmony Memory Size
4. Pitch Adjustment Rate
5. Number of Improvisations



شکل ۲: فلوچارت جریان الگوریتم جستجوی هارمونی با الگوی مسأله نگاشت.

X_1^1	X_1^2	X_1^3	X_1^4	X_1^5	X_1^6	X_1^7	X_1^8	X_1^9

شکل ۴ : بردار جواب بعد از درنظرگرفتن نرخ $HMCR$.

X_1	X_2	X_3
X_4	X_5	X_6
X_7	X_8	X_9

۶	۳	۱
۴	۲	
۷	۵	۸

X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9
۶	۳	۱	۴	۲		۷	۵	۸

شکل ۳: نمایش اختصاص یک جمعیت اولیه از بردارهای هارمونی به محلی از حافظه.

توجه به این نکته مهم است که این احتمال مستقل از راه حل‌های ذخیره‌شده در حافظه هارمونی است و بنابراین احتمال تولید مقدار صحیح برای یک متغیر تصمیم را می‌توان از روش (۱۹) محدود کرد

$$X^{NEW} \leftarrow x_j^i . x_i^j \in \{۱,۲,۳,\ldots,۹\} w.p(۱-HMCR) \qquad (۱۹)$$

می‌توان به‌جای انتخاب تصادفی، حافظه را برای انتخاب هر مقدار ذخیره‌شده در HM با احتمال $HMCR$ از طریق (۲۰) در نظر گرفت

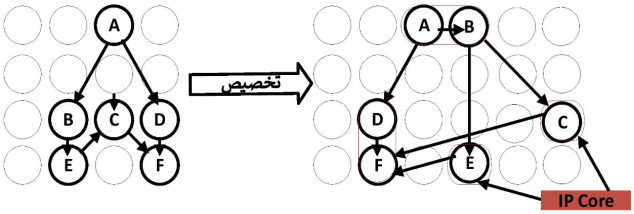
$$X^{NEW} \leftarrow x_j^i . x_i^j \in \{x_{ij}^۱, x_{ij}^۲, \ldots, x_{ij}^۹\} w.p(HMCR) \qquad (۲۰)$$

شیوه نمایش جواب مطابق شکل ۴ با توجه به ماهیت پیوسته الگوریتم بین مقادیر حد پایین صفر و حد بالای یک و به‌طور تصادفی برای هر مؤلفه بردار خواهد بود.

گاهی ممکن است که با درنظرگرفتن حافظه هارمونی به جای انتخاب تصادفی، مقدار به‌دست‌آمده به مقادیر همسایه با احتمال $HMCR \times PAR$ منتقل شود. در حالی که با انتخاب تصادفی، مقدار اصلی به‌دست‌آمده با احتمال $HMCR \times (۱-PAR) \times PAR$ $(۰ \le PAR \le ۱)$ به سمت همسایه انتقال پیدا نمی‌کند. به این حالت تنظیم گام گفته می‌شود. در اینجا $x_{i,j}^{NEW}$ مقداری است که از بررسی حافظه به‌دست آمده و Δ مقدار افزایش است. اگر $x_{i,j}^{NEW}$ حد بالایی (۸) یا حد پایینی (۱) نباشد، Δ برابر

$X^1_{1-y^1}$	$X^2_{1-y^2}$	$X^3_{1-y^3}$	$X^4_{1-y^4}$	X_5	X_6	X_7	X_8	X_9

شکل ۵: اصلاح بردار جواب الگوریتم هارمونی با مقادیر پارامترهای PAR و bw.



شکل ۶: اولین مرحله، نگاشت از وظیفه به هسته IP.

یک و در غیر این صورت برابر با صفر است. در به‌روزرسانی HM، هر جزء بردار هارمونی جدید، مورد بررسی قرار می‌گیرد تا مشخص شود که آیا باید آن را تنظیم کرد یا خیر. در فرآیند در به‌روزرسانی HM، از پارامتر PAR استفاده می‌شود که بر اساس نرخ تنظیم گام انتخاب‌شده عمل کرده و توسط آن HM تنظیم می‌گردد. فرآیند تنظیم $Pitch$ تنها پس از انتخاب HM با (۲۱) انجام می‌گیرد

$Pitch$ adjusting decision for:

$$x_{ij}^{NEW} \Leftarrow \begin{cases} \text{Yes with probability } PAR \\ \text{No with probability } (1-PAR) \end{cases} \qquad (۲۱)$$

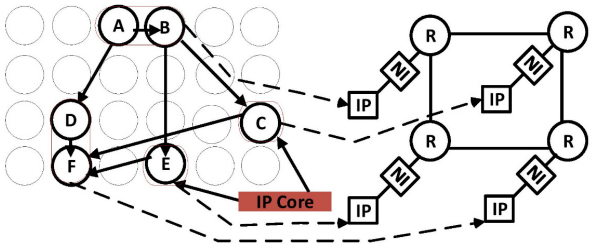
به منظور رفع عملکرد ضعیف الگوریتم هارمونی به دلیل استفاده از مقادیر ثابت PAR و bw و تنظیم مناسب این پارامترها باید مقدار تعداد بهبودها[1] (NI) را افزایش دهیم تا به حل بهینه دست یابیم. هرچه مقدار پارامتر bw در تکرارهای ابتدایی بزرگ‌تر باشد موجب می‌شود الگوریتم، تنوع جستجو را در کل فضای مسأله افزایش دهد؛ در حالی که در تکرارهای بعدی که با هدف جستجوی محلی انجام می‌شود برای جستجوهای محدودتر مناسب است. بنابراین مقادیر بزرگ PAR و مقادیر کوچک bw در تکرارهای پایانی منجر به رسیدن به فضای بهینه و همگرایی به بهینگی می‌گردد. بنابراین الگوریتم هارمونی با مقادیر پارامترهای PAR و bw به‌صورت پویا در هر تکرار و جداگانه مطابق با (۲۲) و (۲۳) اصلاح می‌گردد

$$PAR(t) = PAR_{\min} + \frac{PAR_{\max} - PAR_{\min}}{NI} \times t \qquad (۲۲)$$

$$bw(t) = bw_{\max} \times e^{\frac{in\left(\frac{bw_{\min}}{bw_{\max}}\right)}{NI} \times t} \qquad (۲۳)$$

که $bw_{\max}$، $bw_{\min}$، $PAR_{\max}$ و $PAR_{\min}$ مقادیر ثابتی هستند که در ابتدای الگوریتم تعریف می‌شوند. t شماره تکرار ایجاد بردار حل جدید الگوریتم و NI شماره تعداد بهبودها برای رسیدن به توقف است. شکل ۵ اصلاح بردار جواب الگوریتم هارمونی را با پارامترهای PAR و bw و نگاشت گره‌های وظایف بر روی کاشی‌های همبندی در شبکه توری نشان می‌دهد.

گام چهارم) در گام تجدید نظر در حافظه هارمونی، ارزش تناسب بردار هارمونی جدید a'_x با ارزش تناسب بدترین بردار هارمونی ذخیره‌شده در حافظه مقایسه می‌شود. اگر اولی مناسب‌تر از دومی باشد، بدترین مقدار هارمونی در حافظه با بردار هارمونی تازه ایجادشده جایگزین می‌شود و در غیر این صورت، بردار هارمونی تازه ایجاد شده نادیده گرفته می‌شود.



شکل ۷: دومین مرحله، نگاشت از هسته IP به پلتفرم NoC.

اگر بردار هارمونی جدید X^{NEW} از نظر مقدار تابع هدف، $Z(X^{NEW})$ از بدترین هارمونی در HM بهتر باشد، هارمونی جدید در HM لحاظ و بدترین هارمونی موجود از HM حذف می‌شود.

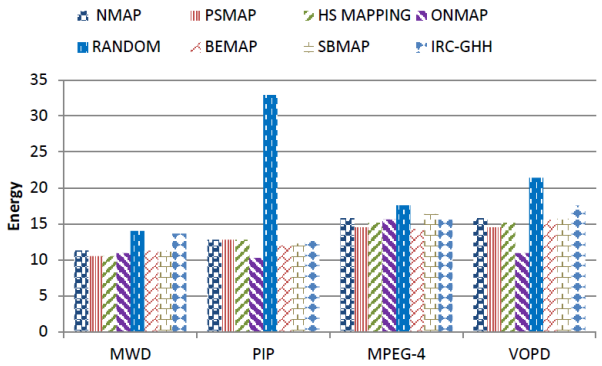
گام پنجم) گام پنجم ارزیابی شرایط فسخ است. در صورت رسیدن به شرایط خاتمه، فرایند تکامل HSA متوقف می‌شود. شرط خاتمه را می‌توان با تعداد کل بداهه‌هایی که تا کنون انجام شده است تعیین کرد و در غیر این صورت به مرحله ۳ بازمی‌گردد. مرحله ۳ یک مرحله حیاتی برای فرایند بداهه‌سازی است. اگر مدل HS به حداکثر تعداد جواب برسد، محاسبات خاتمه می‌یابد و در غیر این صورت مجدداً با درنظرگرفتن یکی از سه مکانیسم بیان‌شده، هارمونی جدید دیگری جستجو می‌شود.

کل نیاز ارتباطی هر هسته با دیگر هسته‌ها با جمع‌بندی برچسب‌های لبه‌های مرتبط با این گره در نمودار هسته تعیین می‌شود. در مرحله بعد هسته‌ها به ترتیب نزولی ارتباطشان مرتب می‌شوند. در اولین مرحله با شروع از یک نقطه اولیه که در آن هیچ تصمیمی برای ارتباط هسته‌ای با مسیریاب‌ها گرفته نشده است، الگوریتم تلاش می‌کند هسته‌ها را انتخاب و به مسیریاب‌ها نگاشت کند. مشکل اصلی چنین رویکردی این است که جفت‌هسته‌هایی که ارتباط بالایی بین خود دارند همیشه نزدیک به یکدیگر قرار نمی‌گیرند. با این حال، الگوریتم با کار بر روی ترتیب لبه‌ها به جای هسته‌ها پس از نگاشت دو هسته روی یک یال، لبه دیگری را برای نگاشت انتخاب می‌کند. در دومین مرحله و در صورت وجود مطابقت بین گره‌های مسیریابی و گره منبع، گره مسیریابی و گره منبع، داده‌ها را از طریق رابط شبکه (NI)[2] منتقل می‌کنند. شکل‌های ۶ و ۷ مراحل نگاشت را نشان می‌دهند.

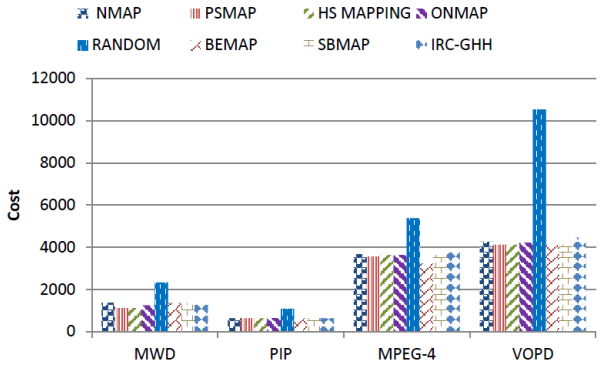
اکثر رویه‌های نگاشت اکتشافی، یک مرحله پس از پردازش را نیز انجام می‌دهند تا از بهبود حاصل از تغییرات کوچک محلی اطمینان حاصل کنند. آنها با محاسبه هزینه برخی از راه حل‌های نیمه‌نگاشت‌شده، یک مسیریاب را برای نگاشت هسته مورد نظر انتخاب می‌کنند و غالباً تغییر در صورتی پذیرفته می‌شود که به راه حل بهتری منجر گردد. با این حال زمان اجرای کلی الگوریتم افزایش می‌یابد. الگوریتم پیشنهادی ماهیت جامع‌تری دارد و نیاز به انجام چنین تلاش‌هایی برای بهبود تکراری ندارد. این الگوریتم عمیق‌تر به فضای جستجو می‌پردازد که اغلب منجر به صرفه‌جویی در زمان CPU و راه حل‌های نگاشت با کیفیت بهتر در مقایسه با سایر روش‌های مشابه می‌شود. تولید و مرتب‌سازی حافظه اولیه HM اولیه دارای پیچیدگی زمانی $O(HMS \times M + (HMS \times \log HMS))$ می‌باشد و M تعداد راه حل‌ها است. هزینه تولید هر بردار هارمونی جدید $O(M \times (1 + HMCR \times PAR))$ است. با تکرار جستجو، بردارهای هارمونی بیشتری برای جستجوی راه حل بهینه تولید می‌شوند و هر بار موارد جدید بررسی می‌گردد تا بدترین بردار را در HM با پیچیدگی $O(\log(HMS) + M)$ جایگزین کند. فرایند جستجوی تکراری تا رسیدن

1. Number of Improvements

2. Network Interface



شکل ۸: انرژی مصرفی در الگوریتم‌های NMAP، ONMAP، PSMAP، BEMAP، IRC-GHH، SBMAP و HS MAPPIN در کاربردهای مختلف.



شکل ۹: تابع هزینه در الگوریتم‌های NMAP NMAP، ONMAP، PSMAP، BEMAP، IRC-GHH، SBMAP و HS MAPPIN در کاربردهای مختلف.

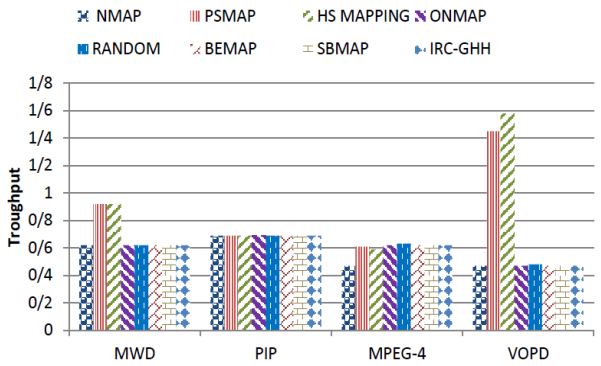
به عدد متناسب تکرار می‌شود و نهایتاً پیچیدگی کلی الگوریتم جستجوی هارمونی، O است.

٤- شبیه‌سازی و ارزیابی نتایج

در این بخش، نتایج شبیه‌سازی رویکرد نگاشت پیشنهادی بر روی همبندی توری و الگوریتم مسیریابی XY و سپس مقایسه نتایج با سایر تکنیک‌های نگاشت موجود برای تعدادی از کاربردها ارائه می‌شود. هزینه ارتباط به عنوان معیار شاخص در مورد عملکرد یک شبکه است. در این مقاله، معیار هزینه در الگوریتم‌های متفاوت نگاشت گردیده و رویکرد پیشنهادی بر روی همبندی توری دوبعدی مورد بررسی قرار گرفته است. هر هسته با استفاده از الگوی ترافیک مطابق با وزن یال گراف جهت‌دار $G(C,E)$، پیام‌ها را ارسال می‌کند. برای ارزیابی روش پیشنهادی از شبیه‌ساز دقیق ناکسیم[1] [۳۷] استفاده کرده‌ایم. برای هر هسته از یک عامل بدون بعد برای تخمین ویژگی خودشباهتی استفاده شده است. پارامتر Hurst [۳۸] تعداد زیادی منابع پیام را به کمک توزیع پارتو و با فرض $H = ۰/۷۵$ جمع‌آوری می‌کند. تمام وظایف به‌صورت انحصاری تعریف شده و هر وظیفه در هر لحظه فقط در یک پردازنده پردازش می‌شود. انرژی مصرف‌شده و تأخیر در مسیریاب و لینک برابر با یک ژول در نظر گرفته شده و همچنین هزینه انتقال یک بیت داده بین دو همسایه برابر عدد ثابت ۱ می‌باشد. این روش بر روی ۴ گراف استاندارد پیاده‌سازی شده است، به‌طوری که تعداد گره‌های گراف وظایف بر اساس تعداد گره‌های موجود در آن تعیین می‌شوند.

نتایج مربوط با هزینه ارتباط در راه‌حل‌های نگاشت برای تعدادی از

1. Noxim



شکل ۱۰: گذردهی در الگوریتم‌های NMAP، ONMAP، PSMAP، BEMAP، IRC-GHH، SBMAP و HS MAPPIN در کاربردهای مختلف.

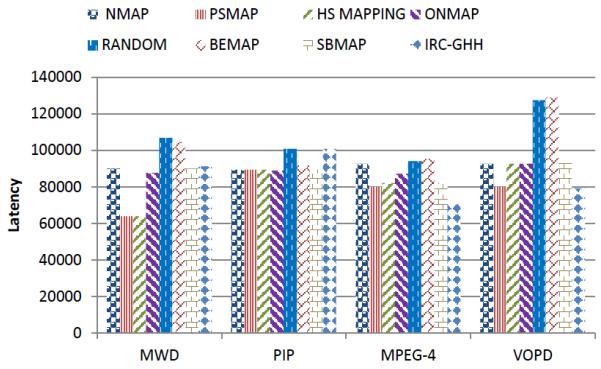
جدول ۱: هزینه ارتباط در الگوریتم‌های نگاشت برای کاربردهای مختلف بر روی شبکه روی تراشه مبتنی بر توری.

الگوریتم نگاشت	معیار			
	VOPD	MPEG-۴	PIP	MWD
	هزینه ارتباطات (گام×BW) در MB/s			
NMAP [۲۵]	۴۲۶۵	۳۶۷۲	۶۴۰	۱۳۷۶
PSMAP [۲۶]	۴۱۱۹	۳۵۶۷	۶۴۰	۱۱۲۰
IRC-GHH [۱۷]	۴۴۵۷	۳۷۵۸	۶۴۰	۱۲۸۰
ONMAP [۲۸]	۴۲۱۷	۳۶۳۳	۶۴۰	۱۲۴۸
RANDOM	۱۰۵۵۲	۵۳۸۳	۱۰۸۸	۲۳۳۶
BEMAP [۲۹]	۴۱۱۹	۳۲۳۱	۶۴۰	۱۳۷۰
SBMAP [۳۰]	۴۱۲۵	۳۶۳۵	۶۴۰	۱۳۶۲
Ours HS MAPPING	۴۱۱۹	۳۶۳۱	۶۴۰	۱۱۲۰

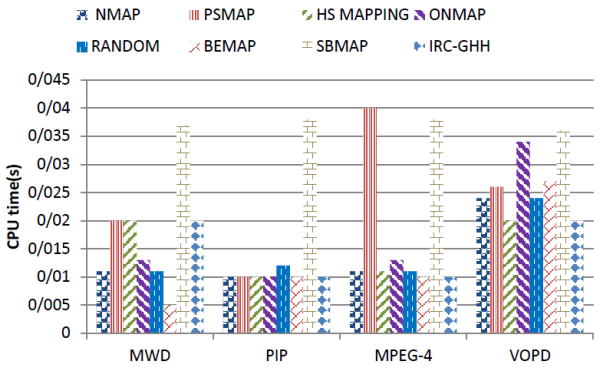
کاربردها در جدول ۱ نشان داده شده است. کاربرد MPEG-۴، عملکرد بهتری را نشان می‌دهد؛ با این حال چنین تکنیک‌هایی به‌طور کلی زمان اجرای بالایی دارند. در مقایسه با عملکرد تکنیک‌های نگاشت، استراتژی HS عملکرد بهتری دارد. از آنجایی که NMAP الگوریتمی است که به‌طور گسترده برای مقایسه نتایج نگاشت برنامه‌ها استفاده می‌شود، نتایج مقایسه با NMAP در جدول ۲ مقایسه شده است.

مقایسه نتایج جدول ۱ نشان می‌دهد که روش نگاشت HS در فضای جستجوی محلی در اغلب معیارها نتایجی مانند PSMAP ایجاد می‌نماید و تنها در مورد معیار MPEG-۴ هزینه ارتباط اندکی افزایش پیدا می‌کند. استراتژی PSMAP در اکتشاف و جایگزینی راه حل‌های محلی مناسب به‌خوبی عمل می‌کند؛ با این حال در اغلب موارد به سختی می‌تواند بین این دو تعادل برقرار کند، زیرا نمی‌تواند فضای جستجو را به‌طور کامل کاوش نماید و بنابراین دارای انعطاف‌پذیری پایین‌تری است. ضمن اینکه در انتخاب فضای جستجو مقیاس‌پذیری کمتری دارد. رویکرد پیشنهادی این مقاله در هر دو مورد بسیار کامل‌تر عمل می‌کند. کاوش بر روی یک جمعیت اولیه کاملاً تصادفی، نتایج امیدوارکننده‌تری در تعداد راه حل‌های کشف‌شده ایجاد می‌کند.

جدول ۲ و شکل‌های ۸ تا ۱۱ نتایج مقایسه تأخیر، توان مصرفی و توان عملیاتی را برای الگوریتم‌های نگاشت مختلف نشان می‌دهد. برای هر طراحی شبکه روی تراشه انتظار می‌رود که توان عملیاتی شبکه بالا و میانگین تأخیر هر بسته در کمترین مقدار باشد. مقایسه توان مصرفی الگوریتم HS MAPPING پیشنهادی در مقایسه با NMAP برای کاربرد MWD تا ۷٪ صرفه‌جویی بیشتر را نشان می‌دهد. این نسبت صرفه‌جویی در توان مصرفی برای کاربرد MPEG-۴ ۳/۹۸٪ بوده است همچنین



شکل ۱۱: نرخ تأخیر در الگوریتم‌های NMAP، ONMAP، PSMAP، BEMAP، IRC-GHH، SBMAP و HS MAPPIN در کاربردهای مختلف.



شکل ۱۲: زمان مورد نیاز CPU در الگوریتم‌های NMAP، ONMAP، PSMAP، BEMAP، SBMAP و HS MAPPIN در کاربردهای مختلف.

جدول ۲: مقایسه گذردهی (µJ/Packe)، تأخیر (Cycle) و توان مصرفی (M Cycles/Packet) در HS MAPPING پیشنهادی در مقایسه با الگوریتم‌های نگاشت مختلف.

الگوریتم نگاشت	معیار											
	VOPD			PIP			MWD			MPEG -۴		
	تأخیر	توان مصرفی	نرخ گذردهی	تأخیر	توان مصرفی	نرخ گذردهی	تأخیر	توان مصرفی	نرخ گذردهی	تأخیر	توان مصرفی	نرخ گذردهی
NMAP [۲۵]	۹۲۶۸۶٫۶	۱۵٫۸۰	۰٫۴۷	۸۹۴۱۵٫۲۰	۱۲٫۸۰	۰٫۶۹	۸۹۹۳۶٫۷۰	۱۱٫۳۰	۰٫۶۲	۹۲۶۸۶٫۹۰	۱۵٫۸۰	۰٫۴۷
PSMAP [۲۶]	۷۹۹۷۲٫۵۰	۱۴٫۵۱	۱٫۴۵	۸۹۴۱۵٫۲۰	۱۲٫۸۰	۰٫۶۹	۶۴۰۵۸٫۱۰	۱۰٫۵۱	۰٫۹۲	۷۹۹۷۲٫۵۰	۱۴٫۵۱	۰٫۶۱
IRC-GHH [۱۷]	۷۹۲۲۱٫۹۸	۱۷٫۵۷	۰٫۴۷	۱۰۰۸۸۰	۱۲٫۸۰	۰٫۶۹	۹۱۲۴۸٫۰۹	۱۳٫۶۴	۰٫۶۲	۷۰۶۳۵٫۸۷	۱۵٫۶۴	۰٫۶۲
ONMAP [۲۸]	۹۳۶۸۶٫۶	۱۰٫۹۵	۰٫۴۷	۸۸۷۳۳٫۸۶	۱۰٫۲۵	۰٫۶۹	۸۷۶۶۵٫۵۷	۱۰٫۹۵	۰٫۶۲	۸۷۲۰۵٫۴۱	۱۵٫۶۲	۰٫۶۲
RANDOM	۱۲۷۴۴۴٫۰۷	۲۱٫۴۳	۰٫۴۸	۱۰۰۸۸۰	۳۲٫۹۱	۰٫۶۹	۱۰۶۷۴۳٫۰۵	۱۴٫۰۲	۰٫۶۲	۹۴۱۸۱٫۱۶	۱۷٫۶۰	۰٫۶۲
SBMAP [۳۰]	۹۳۱۷۱٫۸۷	۱۵٫۷۱	۰٫۴۷	۸۹۴۱۹٫۰۱	۱۰٫۲۸	۰٫۶۹	۸۹۹۳۶٫۷۰	۱۱٫۲۷	۰٫۶۲	۸۲۰۳۸٫۷۸	۱۶٫۳۶	۰٫۶۲
BEMAP [۲۹]	۱۲۸۸۱۰٫۱۸	۱۵٫۵۸	۰٫۴۷	۹۱۸۱۸٫۲۸	۱۲٫۰۱	۰٫۶۹	۱۰۴۳۸۷٫۸۸	۱۱٫۲۵۸	۰٫۶۲	۹۵۳۶۳٫۸۶	۱۴٫۸۲	۰٫۶۲
Ours HS MAPPING	۹۲۶۸۶٫۹۰	۱۵٫۱۷	۱٫۵۸	۸۹۴۱۵٫۲۰	۱۲٫۸۰	۰٫۶۹	۶۴۰۵۸٫۱۰	۱۰٫۵۱	۰٫۹۲	۸۲۰۳۵٫۲۰	۱۵٫۱۷	۰٫۵۹

درصد صرفه‌جویی در توان مصرفی برای HS MAPPING پیشنهادی در مقایسه با رویکرد RANDOM در کاربرد MWD تقریباً ۴٫۸۳٪ و برای کاربرد MPEG -۴ اندکی بیشتر از ۲۵٪ می‌باشد. با این حال رویکرد HS MAPPING پیشنهادی تا ۳۲٫۶٪ گذردهی بیشتری نسبت به NMAP برای کاربرد MPEG -۴ فراهم می‌کند و تأخیر هر بسته را ۱۱٫۴۹٪ کاهش می‌دهد.

از سوی دیگر، مقادیر مقایسه مشابه برای رویکرد RANDOM به ترتیب ۱۲٫۸۹ و ۹۴ درصد است. بیشترین صرفه‌جویی در توان مصرفی الگوریتم پیشنهادی ۲۹٫۲، ۶۱٫۱۱ و ۲۵٫۰۴ درصد به ترتیب برای کاربردهای VOPD، PIP و MWD در مقایسه با الگوریتم‌های NMAP و RANDOM است.

به منظور ارزیابی اثرات بهبود استفاده از الگوریتم جستجوی هارمونی برای نگاشت وظایف و تولید راه حل‌های آن در میانگین هزینه ارتباطی نسبت به سایر روش‌ها در بنچمارک‌های مختلف از نتایج شبیه‌سازی بر روی NoCهایی که تعداد هسته‌های بیشتری دارند، مقایسه‌ای بر روی زمان CPU انجام شده است. اگرچه این موضوع در خصوص اغلب روش‌های دقیق برای شناسایی راه‌حل‌های نگاشت بهینه برای یک برنامه کاربردی به خوبی کار می‌کند، با این حال زمان CPU و فضای حافظه مورد نیاز در چنین استراتژی‌هایی حتی برای برنامه‌هایی با تعداد هسته‌های متوسط، بسیار زیاد می‌شود و این افزایش به ترتیب با مجذور حاصل‌ضرب تعداد هسته‌ها و روترها در نمودار هسته و نمودار توپولوژی متناسب است. بهبود تأخیر الگوریتم HS در مقایسه با روش‌های NMAP و RANDOM به ترتیب ۲۷٫۲۷ و ۳۹٫۹۸ درصد برای کاربردهای VOPD و MPEG -۴ و به‌طور مشابه برای کاربرد MWD، ۱۲٫۸۹ درصد در مقایسه با رویکرد RANDOM و ۱۱٫۴۹ درصد در مقایسه با الگوریتم NMAP است.

با درنظرگرفتن مقادیر NMAP به عنوان واحد، شکل ۱۲ زمان مورد نیاز CPU را در ۷ رویکرد NMAP، ONMAP، PSMAP، BEMAP، IRC-GHH، SBMAP و HS MAPPING پیشنهادی مقایسه می‌کند.

لازم است به این موضوع اشاره شود که PSMAP به‌طور متوسط به ۲٫۸ درصد زمان بیشتر نیاز دارد؛ در حالی که HS MAPPING پیشنهادی تا ۷۲ درصد زمان کمتری می‌گیرد. زمان مورد نیاز CPU برای الگوریتم HS با رویکردهای PSMAP و ONMAP قابل مقایسه است، زیرا برای همه این معیارها کمتر از ۰٫۰۱۵ ثانیه است. شبکه به دلیل نرخ تزریق ترافیک یکنواخت ازدحام ندارد و بنابراین بر اساس شکل ۱۰ گذردهی برای این کاربردها تقریباً ثابت است. تحلیل مقایسه‌ای در این شکل نشان می‌دهد که کیفیت پارامتر گذردهی الگوریتم پیشنهادی و PSMAP از سایر الگوریتم‌های مورد مقایسه برتر است.

۵- نتیجه‌گیری و پیشنهاد کار آتی

این مقاله، یک استراتژی نگاشت بهبودیافته مبتنی بر ساختار گراف و الگوریتم جستجوی هارمونی را برای شبکه روی تراشه مبتنی بر توری ارائه می‌کند. در این روش از همه بردارهای موجود در حافظه برای تولید راه حل جدید استفاده می‌شود. برای مشاهده بهینه‌بودن تکنیک نگاشت پیشنهادی، محاسبه تابع هزینه با امکان کاهش هم‌زمان تأخیر و توان مصرفی فرموله شده است. مقادیر متریک هزینه ارتباطات راه حل‌های نگاشت رویکرد پیشنهادی با برخی رویکردهای موجود از نظر توان

مصرفی، میانگین تأخیر شبکه و گذردهی برای نگاشت مقایسه شده است. از مزایای این روش، همگرایی سریع به دلیل ساختار مناسب آن است. نتایج شبیه‌سازی نشان می‌دهند که در بسیاری از موارد، تکنیک نگاشت پیشنهادی، نتایج مشابهی را با زمان CPU کمتر ایجاد می‌کند. همچنین می‌توان به این نکته اشاره کرد که استراتژی نگاشت پیشنهادی برای شبکه‌هایی که تعداد هسته‌های بیشتری دارند، بهبود بهتری را نشان می‌دهد. کار آینده شامل گسترش رویکرد به شبکه‌های سه‌بعدی و توسعه استراتژی‌های نگاشت با هدف قرار دادن طرح‌های مقاوم به خطا و غیره است.

مراجع

[1] A. Hemani, *et al.*, "Network on chip: an architecture for billion transistor era," in *Proc. of the IEEE NorChip Conf.*, pp. 166-173, Turku, Finland, Nov. 2000.

[2] C. L. Chou and R. Marculescu, "Contention-aware application mapping for network-on-chip communication architectures," in *Proc. IEEE Int. Conf. on Computer Design, ICCD'08*, pp. 164-169, Lake Tahoe, CA, USA, 12-15 Oct. 2008.

[3] X. S. Yang, "Harmony search as a metaheuristic algorithm search algorithm: theory and applications," *Studies in Computational Intelligence,* vol 191, pp.1-14, Springer, Berlin, Heidelberg, 2009.

[4] C. Marcon, T. Webber, and A. A. Susin, "Models of computation for NoC mapping: timing and energy saving awareness," *Microelectronics J.*, vol. 60, pp. 129-143, Feb. 2017.

[5] P. Mesidis, *Mapping of Real-Time Applications on Network-on-Chip Based MPSOCS*, MS Thesis, Department of Computer Science, University of York, Dec. 2011.

[6] P. K. Sahu and S. Chattopadhyay, "A survey on application mapping strategies for network-on-chip design," *J. of Systems Architecture*, vol. 59, no. 1, pp. 60-76, Jan. 2013.

[7] A. Bender, "MILP based task mapping for heterogeneous multiprocessor systems," in *Proc. of International Conf. on Design and Automation*, vol. 96, pp. 190-197, Sep. 1996.

[8] K. Srinivasan, K. S. Chatha, and G. Konjevod, "Linear-programming-based techniques for synthesis of network-on-chip architectures," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 14, no. 4, pp. 407-420, Apr. 2006.

[9] P. Ghosh, A. Sen, and A. Hall, "Energy efficient application mapping to NoC processing elements operating at multiple voltage levels," in *Proc. 3rd ACM/IEEE Intl Symp. on Networks-on-Chip*, pp. 80-85, La Jolla, CA, USA, 10-13 May 2009.

[10] C. Ostler and K. S. Chatha, "An ILP formulation for system-level application mapping on network processor architecture," in *Proc. of Design, Automation and Test in Europe, DATE'07*, pp. 99-104, Nice, France, 16-20 Apr. 2007.

[11] J. Hu and R. Marculescu, "Communication and task scheduling of application-specific networks-on-chip," *IEEE Proc. Computers & Digital Techniques*, vol. 152, no. 5, pp. 643-651, Sept. 2005.

[12] S. Tosun, "Clustered-based application mapping method for network-on-chip," *J. of Advances in Engineering Software*, vol. 42, no. 10, pp. 868-874, Oct. 2011.

[13] R. Marculescu and J. Hu, "Energy-aware mapping for tile-based NoC architectures under performance constraints," in *Proc. Asia and South Pacific Design Automation Conf., ASP-DAC'03*, pp. 233-239, Kitakyushu, Japan,2 4-24 Jan. 2003.

[14] R. Marculescu and J. Hu, "Energy-and performance-aware mapping for regular NoC architectures," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 4, pp. 180-187, Apr. 2005.

[15] M. Reshadi, A. Khademzadeh, and A. Reza, "Elixir: a new bandwidth-constrained mapping for networks-on-chip," *IEICE Electronics Express*, vol. 7, no. 2, pp. 73-79, 2010.

[16] R. Pop and S. Kumar, A Survey of Techniques for Mapping and Scheduling Applications to Network on Chip Systems, Technical Report ISSN 1404-0018 04:4, ING Jönköping, 2004.

[17] C. Xu, *et al*., "Optimization strategy of regular NoC mapping using genetic-based hyper-heuristic algorithm," *Symmetry*, vol. 14, no. 8, Article ID: 1637, Aug. 2022.

[18] C. Xu, Y. Liu, P. Li, and Y. Yang, "Unified multi-objective mapping for network-on-chip using genetic-based hyper-heuristic algorithms," *IET Comput. Digit. Tech*, vol. 12, no. 4, pp. 158-166, Jul. 2018.

[19] H. M. Ali, S. Ashrafinia, and J. Liu, "Wireless mesh network planning using quantum inspired evolutionary algorithm [C]," in *Proc. IEEE Conf. on Vehicular Technology*, 5 pp., San Francisco, CA, USA, 5-8 Sept. 2015.

[20] Y. Xie and Y. Liu, "A research on NoC mapping with quantum ant colony algorithm," in *International Conf. on Wireless Communications, Signal Processing and Networking, WiSPNET'17,* pp. 874-877, Chennai, v20-24 Mar. 2017.

[21] E. Carvalho, N. Calazans, and F. Moraes, "Dynamic task mapping for MPSoCs," *IEEE Design and Test of Computers*, vol. 27, no. 5, pp. 26-35, Sept./Oct. 2010.

[22] T. Lei and S. Kumar, "A two-step genetic algorithm for mapping task graphs to a network on chip architecture," in *Proc. of the Euromicro Symp. on Digital System Design, DSD'03*, pp. 180-187, Belek-Antalya, Turkey, 1-6 Sept. 2003.

[23] A. Hansson, K. Goossens, and A. Radulescu, "A unified approach to constrained mapping and routing on network-on-chip architectures," in *Proc. IEEE/ACM Int Conf. on Hardware/Software Codesign and System Synthesis, CODES+ISSS'05*, pp. 75-80, Jersey City, NJ, USA,19-21 Sept. 2005.

[24] W. T. Shen, C. H. Chao, Y. K. Lien, and A. Y. Wu, "A new binomial mapping and optimization algorithm for reduced-complexity mesh-based on-chip network," in *Proc. of First Int. Symp. on Networks-on-Chip, NOCS'07*, pp. 317-322, Princeton, NJ, USA, 7-9 May 2007.

[25] S. Murali and G. De Micheli, "Bandwidth constrained mapping of cores onto NoC architectures," in *Proc. of Design, Automation, and Test in Europe Conf. and Exhibition*, vol. 2, pp. 896-901, Feb. 2004.

[26] P. K. Sahu, P. Venkatesh, S. Gollapalli, and S. Chattopadhyay, "Application mapping onto mesh structured network-on-chip using particle swarm optimization," in *Proc. IEEE Int. Symp. VLSI, ISVLSI'11*, pp. 335-336, Chennai, India, 4-6 Jul. 2011.

[27] S. Murali and G. De Micheli, "Bandwidth constrained mapping of cores onto NoC architectures," in *Proc. of Design, Automation, and Test in Europe Conf. and Exhibition, DATE'04*, vol. 2, pp. 896-901, Paris, France, 16-20 Feb. 2004.

[28] S. Khan, *et al.*, "An optimized hybrid algorithm in term of energy and performance for mapping real time workloads on 2d based on-chip networks," *Springer Science + Business Media, LLC, Part of Springer Nature*, vol. 48, no. 12, pp. 4792-4804, Dec. 2018.

[29] S. Khan, *et al.*, "An efficient algorithm for mapping real time embedded applications on NoC architecture," *IEEE ACCESS*, vol. 6, pp. 16324-16335, 2018.

[30] S. Khan, S. Anjum, U. A. Gulzari, T. Umer, and B. S. Kim, "Bandwidth-constrained multi-objective segmented brute-force algorithm for efficient mapping of embedded applications on NoC architecture," *IEEE ACCESS*, vol. 6, pp. 11242-11254, 2017.

[31] G. Z. Woo, K. J. Hoon, and G. Loganathan, "A new heuristic optimization algorithm: harmony search," *Simulation*, vol. 76, no. 2, pp. 60-68, Feb. 2001.

[32] Z. W. Geem and Y. H. Cho, "Optimal design of water distribution networks using parameter-setting-free harmony search for two major parameters," *J. of Water Resources Planning and Management*, vol. 137, no. 4, pp. 377-380, Oct. 2010.

[33] N. Theodossiou and I. Kougias, "Harmony search algorithm," *WIT Trans. on State of the Art in Science and Engineering*, vol. 56, Ch. 7, 25 pp., Jul. 2012.

[34] P. K. Sahu, K. Manna, T. Shah, and S. Chattopadhyay, "A constructive heuristic for application mapping onto mesh based network-on-chip," *J. of Circuits, Systems, and Computers*, vol. 24, no. 8, Article ID: 1550126, Aug. 2015.

[35] C. Marcon, *et al.*, "Exploring NoC mapping strategies: an energy and timing aware technique," in *Proc. Design, Automation and Test in Europe*, pp. 502-507, Munich, Germany, 7-11 Mar. 2005.

[36] M. Mahdavi, M. Fesanghary, and E. Damangir, "An improved harmony search algorithm for solving optimization problems," *Appl Math Comput.*, vol. 188, no. 2, pp. 1567-1579, May 2007.

[37] "Noxim, http://noxim.sourceforge.net [available on dated: 01.08.2014]."

[38] D. Fernandes, M. Neto, L. F. B. Soares, M. M. Freire, and P. R. M. Inácio, "Chapter 10-on the self-similarity of traffic generated by network traffic simulators," *Modeling and Simulation of Computer Networks and Systems*, pp. 285-311, 2015.

زهرا باقری مدرک کاردانی خود را در رشته نرم افزار از دانشگاه علم و صنعت اراک در سال ۱۳۸۳ و مدرک کارشناسی خود را در رشته فناوری اطلاعات از جهاد دانشگاهی تهران در سال ۱۳۹۵ دریافت نمود. همچنین طی سال‌های ۱۳۹۶ تا ۱۴۰۰ مشغول تحصیل در مقطع کارشناسی ارشد در رشته معماری سیستم‌های کامپیوتری در دانشگاه

آزاد اسلامی واحد پرند بود. زمینه علمی مورد علاقه ایشان امنیت شبکه‌های کامپیوتری می‌باشد.

فاطمه وردی مدرک کارشناسی خود را در رشته کامپیوتر از دانشگاه آزاد اسلامی واحد تهران جنوب در سال ۱۳۸۴دریافت نمود. همچنین، تحصیلات خود در مقطع کارشناسی ارشد و دکتری را در رشته معماری سیستم‌های کامپیوتری به‌ترتیب در سال ۱۳۸۸و ۱۳۹۵ در دانشگاه آزاد اسلامی واحد علوم و تحقیقات به اتمام رساند و هم‌اکنون استادیار دانشکده کامپیوتر دانشگاه آزاد اسلامی واحد پرند است. زمینه‌های تحقیقاتی مورد علاقه ایشان شبکه‌های میان ارتباطی، معماری سیستم‌های چند پردازنده‌ای و مدیریت توان و کارایی در سیستم‌های نهفته است.

علیرضا محجوب مدرک کارشناسی خود در رشته منابع طبیعی را از دانشگاه آزاد اسلامی واحد نوشهر و چالوس در سال ۱۳۷۳ دریافت کرد. همچنین طی سال‌های ۱۳۹۶ تا ۱۳۹۸ مشغول تحصیل در مقطع کارشناسی ارشد در رشته معماری سیستم‌های کامپیوتری در دانشگاه آزاد اسلامی واحد پرند بود. از سال ۱۳۹۹ تا کنون نیز در مقطع دکتری معماری سیستم‌های کامپیوتری در دانشگاه آزاد اسلامی واحد کرج مشغول به تحصیل است. زمینه‌های علمی مورد علاقه ایشان شامل موضوعاتی مانند محاسبات نانو الکترونیک، یادگیری ماشین کوآنتومی و پردازش هوشمند می‌باشد.