University of Qom
Engineering Management and Soft Computing
JEMSC.QOM
Online ISSN: 2538-2675

# Present a new method for increasing the intelligence and speed of the Firefly algorithm

**Fariba Solaymani[1], Narges Salehpour[2]✉, and Mohammad Nazari Farokhi[3]**

1. MSc, Department of Computer Engineering, Faculty of Engineering, Azad University of Lorestan, Lorestan, Khorramabad. Email: fariba.solaimani92@yahoo.com
2. Correspnding author, MSc, Department of Computer Engineering, Faculty of Engineering, Azad University of Lorestan, Lorestan, Khorramabad. Email: salehpour_narges@yahoo.com
3. PHD student, Department of Computer Engineering, Faculty of Engineering, Azad University of Lorestan, Lorestan, Khorramabad. Email: m_kasitman@yahoo.com

**Article Info**

**Article type:**
Research Article

**Article history:**
Received 2022 May 27
Received in revised form 2022 Jul 29
Accepted 2022 Aug 3
Published online 2022 Sep 16

**Keywords:**
CPU, FireFly,
GPU,
optimization,
Particle Swarm,
Swarm Intelligenc.

**ABSTRACT**

Today, Most important issues in the industry of non-linear and multi-parametric are considered optimization problems. On the other hand, the attractiveness of the behavior and interaction of animals has led the computer scientists, inspired by these interactions to create algorithms for optimization problems, which in many cases provide quick and acceptable solutions to complex problems. One of the propagation intelligence algorithms is firefly algorithm, which is bace on the exposure of luminous worms and their absorption into more light. The main problem with algorithms Such as firefly is that takes a lot of time to convege the desired answers. So if the number of firefly worms is more than 128, their run time with CPU is 2.5820 milliseconds but with using GPU 1.5090 milliseconds. In this paper, we intend to use a pc graphics unit to provide a version of the firefly algorithm that converages to the desired solutions more quickly while maintaining accuracy

**Cite this article:** Solaymani, F., Salehpour, N & Mohammadi, A. (2022). Present a new method for increasing the intelligence and speed of the Firefly algorithm. *Enginiiring Management and Soft Computing, 8* (1). 1-20. **DOI**: https://doi.org/10.22091/jemsc.2018.753.1033


DOI: https://doi.org/10.22091/jemsc.2018.753.1033
**Publisher**: University of Qom

JEMSC.QOM
شاپا الکترونیکی: ۲۶۷۵-۲۵۳۸

مدیریت مهندسی و رایانش نرم

دانشگاه قم

# ارائه یک روش جدید برای افزایش هوشمندی و سرعت الگوریتم کرم شب‌تاب

**فریبا سلیمانی[۱]، نرگس صالح پور[۲]✉ و محمد نظری فرخی[۳]**

۱. کارشناس ارشد مهندسی کامپیوتر، دانشکده فنی مهندسی، دانشگاه آزاد، لرستان، خرم آباد. رایانامه: fariba.solaimani92@yahoo.com

۲. نویسنده مسئول، کارشناس ارشد مهندسی کامپیوتر، دانشکده فنی مهندسی، دانشگاه آزاد، لرستان، خرم آباد. رایانامه: Salehpour_narges@yahoo.com

۳. دانشجوی دکتری مدیریت فناوری اطلاعات، دانشگاه آزاد اسلامی، واحد علوم و تحقیقات تهران. رایانامه: m_kasitman@yahoo.com

**اطلاعات مقاله**

**نوع مقاله:** مقاله پژوهشی

**تاریخ دریافت:** ۱۴۰۱/۰۳/۰۶

**تاریخ بازنگری:** ۱۴۰۱/۰۵/۰۷

**تاریخ پذیرش:** ۱۴۰۱/۰۵/۱۲

**تاریخ انتشار:** ۱۴۰۱/۰۶/۲۵

**کلیدواژه‌ها:**
ازدحام ذرات و بهینه‌سازی،
جی پی یو،
سی پی یو،
کرم شب‌تاب،
هوش ازدحامی

**چکیده**

امروزه اغلب مسائل مهم در صنعت از نوع مسائل بهینه‌سازی غیرخطی و چندپارامتری محسوب می‌شوند. از طرفی، جذابیت رفتار و تعامل جانوران در طبیعت باعث شده است تا دانشمندان علوم رایانه با الهام از این تعاملات، الگوریتم‌هایی برای مسائل بهینه‌سازی ایجاد نمایند که در خیلی از موارد راه‌حل‌های سریع و قابل قبولی برای مسائل پیچیده به همراه دارند. یکی از الگوریتم‌های هوش ازدحامی، الگوریتم کرم شب‌تاب است که بر اساس نوردهی کرم‌های شب‌تاب و جذب آن‌ها به سمت نور بیشتر شکل گرفته است. ایراد اصلی الگوریتم‌هایی مانند کرم شب‌تاب این است که برای همگرا شدن به جواب‌های موردنظر، نیاز به زمان زیاد دارد. بنابراین، در صورتی که تعداد کرم‌های شب‌تاب بیش از ۱۲۸ باشد، زمان اجرای آن‌ها با استفاده از سی پی یو ۲/۵۸۲۰ میلی‌ثانیه اما با استفاده از جی پی یو ۱/۵۰۹۰ میلی‌ثانیه است. در این مقاله قصد داریم با استفاده از امکانات واحد پردازش کارت گرافیک، نسخه‌ای از الگوریتم کرم شب‌تاب را ارائه دهیم که همگام با حفظ دقت، با سرعت بیشتری به جواب‌های موردنظر همگرا شود.

**استناد:** سلیمانی، فریبا؛ صالح‌پور، نرگس و نظری فرخی، محمد؛. (۱۴۰۱). «ارائه یک روش جدید برای افزایش هوشمندی و سرعت الگوریتم کرم شب‌تاب». *مدیریت مهندسی و رایانش نرم*، دوره ۸ (۱). صص: ۲۰-۱. https://doi.org/10.22091/jemsc.2018.753.1033

ناشر: دانشگاه قم

## ۱) مقدمه

تلاش انسان برای حل مسائل پیرامون خود را می‌توان به‌نوعی جستجو در جهت تکامل راه‌حل‌های موجود دانست، که در آن با یک تعریف کلی می‌توان تکامل را نوعی جستجو برای به دست آوردن بهترین‌ها تعبیر کرد. مشاهدات انسان از محیط پیرامون باعث شده است که انسان برای زیستن بهتر، دست به درک و شبیه‌سازی رفتار حیوانات، گیاهان و حتی اشیاء بی‌جان بزند.از این نظر رفتارهای اجتماعی بین جانداران به‌خصوص رفتارهای اجتماعی بین حشراتی مانند مورچه‌ها و کرم شب‌تاب بیشتر موردتوجه انسان بوده است. معمولاً حل مسائل بهینه‌سازی صنعتی با روش‌های ریاضی، فرآیندی زمان‌گیر است. درحالی‌که الگوریتم‌های هوش ازدحامی[1] معمولاً سریع‌تر عمل می‌کنند. الگوریتم‌های هوش ازدحامی با استفاده از فرآیندهای تصادفی و البته هوشمندانه سعی می‌کنند تا به بهترین جواب‌های یک مسئله بهینه‌سازی برسند. نتایج بررسی‌ها و تحقیقات مختلف نشان می‌دهد، الگوریتم‌های هوش ازدحامی در بیشتر مواقع با بهینه‌های سراسری[2] همگرا[3] می‌شوند. ولی بااین‌وجود این احتمال نیز وجود دارد که در بهینه محلی[4] گرفتار شوند. هر الگوریتم تکاملی به علت غیرقطعی[5] بودن ماهیت آن‌ها، مستعد گرفتار شدن در بهینه‌های محلی می‌باشد. باوجود این‌که افزایش تکرار و جمعیت اولیه الگوریتم‌های تکاملی[6]، شانس یافتن بهینه سراسری را افزایش می‌دهد، زمان اجرای الگوریتم تکاملی نیز به شکل ناخوشایندی افزایش می‌یابد. به‌طور طبیعی تکرار زیاد این‌گونه الگوریتم‌ها بخصوص در جمعیت‌های انبوه، زمان اجرای الگوریتم را افزایش می‌دهد (ایکس.یانگ،۲۰۱۰). در این مقاله با استفاده از امکانات واحد پردازش کارت گرافیک، نسخه‌ای از الگوریتم کرم شب‌تاب را ارائه خواهیم داد که در حین حفظ دقت با سرعت بیشتری به جواب‌های موردنظر همگرا می‌شود. الگوریتم پیشنهادی در این تحقیق با الگوریتم‌هایی مانند الگوریتم ذرات، ژنتیک و الگوریتم استاندارد کرم شب‌تاب ازنظر زمان اجرا و دقت مقایسه شده است. نتایج نشان می‌دهد که الگوریتم پیشنهادی موفق و کارآمد است.

## ۲) پیشینه پژوهش

با توجه به این که اجرای الگوریتم‌های هوش ازدحامی برای حل مسائل بهینه‌سازی بر روی سی‌پی‌یو[7] به شدت زمان‌گیر و هزینه‌بر است، این الگوریتم‌ها برای همگرا شدن به جواب‌های مطلوب و با دقت مناسب، نیاز به تکرار زیادی دارند. الگوریتم کرم شب‌تاب یک الگوریتم هوش ازدحامی موفق (ایکس.اس، یانگ ۲۰۱۳) و دارای دقت مناسبی برای رسیدن به جواب‌های بهینه است اما برای رسیدن به جواب‌های بهینه و دقیق نیاز است تا تعداد جمعیت اولیه و تکرارهای الگوریتم افزایش یابد. افزایش جمعیت اولیه و تعداد تکرارها در الگوریتم کرم شب‌تاب فرآیندی بسیار زمان‌بر است. واحد پردازش کارت گرافیک دارای مزایای بسیار زیادی است. تعداد هسته‌های فراوان این پردازنده‌ها و همچنین پهنای باند زیاد حافظه کارت‌های گرافیکی، قابلیت اجرای تعداد نخ‌های موازی زیادی را در مقایسه با سی‌پی‌یو به این پردازنده داده است.

---

[1] Swarm intelligence algorithms
[2] Global Optimization
[3] Convergent
[4] Local optimization
[5] non-deterministic
[6] Evolutionary Algorithms
[7] CPU

یکی از انگیزه‌های این تحقیق استفاده از سخت‌افزارهای جانبی کامپیوتر در پردازش موازی و بهبود دقت و سرعت الگوریتم کرم شب‌تاب است.

هوش ازدحامی حاصل جمع یا برآیند حاصل از هوش تعدادی عامل ساده می‌باشد (اس. سی. چو، ۲۰۱۱). هوش ازدحامی، در برگیرنده گستره وسیعی است که سعی در حل مسائل با تقلید از شیوه رفتار جمعیت‌های موجود در طبیعت را دارد. در دسته‌هایی که دارای هوش ازدحامی هستند، اطلاعات زیادی وجود دارد که ناشی از تجمع هوش گروه است و این اطلاعات، گروه را قادر می‌سازد تا مشکلات خود را حل کنند. هوش ازدحامی، منبع الهام‌بخشی برای محققان و الگوی جدیدی برای حل مسئله است. گروه‌ها، برای حل مسائل از عوامل متعددی استفاده می‌کنند. رفتار گروه مرتبط با یکدیگر و مشابه به هم است و همین ویژگی‌ها، گروه را قادر به حل مشکلاتی می‌کند که حل بهینه آن برای روش‌های سنتی، در بازه زمانی مشخص مشکل است. از جمله الگوریتم‌های مطرح در این زمینه می‌توان به الگوریتم‌های بهینه‌سازی ازدحام ذرات اشاره کرد (ای. شوکلا، ۲۰۱۰). این دسته از الگوریتم‌ها دارای دو مؤلفه اصلی می‌باشند:

۱- خود ترتیبی[8]، به معنی تعریف یکسری قوانین و توابع که تمامی اعضای گروه ملزم به رعایت و اجرای این قوانین می‌باشند.

۲- جریان اطلاعاتی[9]، در هر مرحله پس از اجرای قوانین توسط اعضای گروه، اطلاعات حاصل می‌بایست بین تمامی اعضای گروه مبادله شود.

مهم‌ترین ویژگی الگوریتم هوش ازدحامی به تولید یک سیستم مستقل، تطبیقی، مقیاس‌پذیر، انعطاف‌پذیر، خودسازمانده، مقرون‌به‌صرفه، مبتنی بر جمعیت و برای بهینه‌سازی غیرخطی می‌توان اشاره کرد (اس. ایکس. وو، ۲۰۱۰).

## ۳) الگوریتم ازدحام ذرات

الگوریتم بهینه‌سازی ازدحام ذرات یک الگوریتم بهینه‌سازی فرااکتشافی با روش جستجوی جمعیت است که از حرکت گروهی پرندگان و دیگر حیواناتی که به شکل گروهی حرکت می‌کنند الگو گرفته است (اس. سی. چو، ۲۰۱۱).

در علوم کامپیوتر، بهینه‌سازی ازدحام ذرات یک روش محاسباتی است که سعی در بهینه‌سازی روش حل مسئله دارد. برای رسیدن به پاسخ بهینه، الگوریتم مورد نظر پس از تکرار چندین‌باره حل مسئله و به دست آوردن جواب‌های کاندید، پس از ارزیابی کیفی پاسخ‌های کاندید، بهترین پاسخ از میان راه‌حل‌های به دست آمده انتخاب می‌شود (اس. سی. چو، ۲۰۱۱، آر.پولی).

در الگوریتم ازدحام ذرات، هر پاسخ مسئله به صورت یک ذره می‌توان مدل کرد که دارای یک مقدار سرعت و همچنین میزان تناسب است. طبق ایده مطرح در الگوریتم بهینه‌سازی ازدحام ذرات، موارد زیر تعریف می‌شوند (اس. سی. چو، ۲۰۱۱؛ آر. شوکلا، ۲۰۱۰؛ ام. ایمران، ۲۰۱۳؛ جی. کندی، ۱۹۹۵):

- هر عضو گروه یک ذره نامیده می‌شود. هر ذره در حال جستجو برای نقطهٔ بهینه است.
- برای یافتن نقطه بهینه، هر ذره در حال جابجایی است.

---

[8]. Self-Organization
[9]. Information Flow

- به دلیل جابه‌جایی ذرات، هر ذره دارای سرعت است.
- الگوریتم بهینه‌سازی ذرات، بر مبنای حرکت و هوش ذرات عمل می‌کند.
- الگوریتم بهینه‌سازی ذرات، مفهوم تعامل اجتماعی را برای حل مسائل بهینه‌سازی به کار می‌گیرد.
- ذرات در فضای جستجو حرکت می‌کنند.

## ۴) الگوریتم کرم شب‌تاب

امروزه مسائل بهینه‌سازی یکی از مسائل پیشرو در صنعت است. مسائل بهینه‌سازی معمولاً با یک تابع غیرخطی نشان داده می‌شوند. هدف، پیدا کردن نقاط اکسترمم این تابع است. روش‌های ریاضی برای حل این مسائل معمولاً توصیه نمی‌شوند بدین دلیل که حل آن‌ها سخت و زمان‌بر است (جی. اگروال، ۲۰۱۳). الگوریتم کرم شب‌تاب، الگوریتم بهینه‌سازی است که از طبیعت الهام گرفته است. هدف اصلی، نور ساطع شده کرم شب‌تاب است که به‌عنوان یک سیستم سیگنال برای جذب کرم شب‌تاب دیگر عمل می‌کنند. در این الگوریتم، تنوع نور در شدت و تدوین جذابیت دو موضوع مهم هستند (ای. وی. هاسولمن، ۲۰۱۲).

### ۴-۱) فرموله کردن الگوریتم کرم شب‌تاب

در الگوریتم کرم شب‌تاب نیاز است که روشنایی کرم‌های شب‌تاب را در یک مکان مشخص با استفاده از یک رابطه ریاضی مدل کرد. رابطه (۱) حالت کلی سازی را نشان می‌دهد:

$$I(x) \propto f(x)$$ رابطه ۱)

در رابطه (۱)، $I(x)$ مقدار شدت روشنایی کرم شب‌تاب در نقطه x و f(x) تابعی است که روشنایی را مدل می‌کند.

در رابطه قبل فرض شده است که منبع نور کرم شب‌تاب نقطه‌ای باشد و توان نقطه‌ای بودن به‌صورت m=2 در نظر گرفته‌شده است. در حالت کلی منبع نور کرم‌های شب‌تاب کاملاً نقطه‌ای نیست، بلکه تا حدودی نقطه است برای اینکه رفتار نور پراکنی کرم‌های شب‌تاب کاملاً نقطه‌ای فرض نشود. همچنین توان را می‌توان بزرگ‌تر از ۲ در نظر گرفت. مقدار m می‌تواند بین مقادیر ۲ تا بی‌نهایت متغیر باشد. مقدار m=۲ منبع نورهای کرم شب‌تاب را نقطه‌ای و m=∞ منبع نورهای کرم شب‌تاب را از حالت نقطه‌ای خارج می‌کند و منبع نور را مسطح فرض می‌کند. در حالت کلی شدت جذب کرم‌های شب‌تاب در رابطه (۲) نشان داده شده است:

$$\beta(r) = \beta_0 e^{-\gamma^m} \cong \frac{\beta_0}{1+\gamma r^m}$$ رابطه ۲)

در الگوریتم کرم شب‌تاب، برای جابه‌جای کرم‌های شب‌تاب از نقطه‌ای با جذابیت کمتر به نقطه‌ای با جذابیت بیشتر از روابط استفاده می‌شود. در رابطه (۲) نحوه انتقال و حرکت یک کرم شب‌تاب بین دونقطه خاص، نشان داده است (ایکس.یانگ، ۲۰۱۰):

$$x_i^t + \beta.\exp\left(-\gamma r_y^2\right).\left(x_j^t - x_i^t\right) + \alpha\epsilon_t = x_i^{t+1}$$ رابطه۳)

در رابطه (۳)، $\epsilon_t$ پارامتر کنترل اندازه گام است که بر اساس توزیع گوسی یا یکنواخت تولید می‌شود. $\alpha_i$ را ضریب القا می‌گویند که باعث می‌شود حرکت کرم‌های شب‌تاب لزوماً روی یک خط راست صورت نگیرد.

### ۴-۲) مشکلات الگوریتم کرم شب‌تاب

الگوریتم‌های تکاملی و هوش ازدحامی از گرفتار شدن در نقاط بهینه محلی و زمان اجرای طولانی رنج می‌برند. الگوریتم کرم شب‌تاب نیز به‌عنوان یک الگوریتم هوش ازدحامی از این چالش مستثنی نیست. یکی از روش‌های بهبود دقت و فرار از نقاط بهینه محلی، افزایش اندازه جمعیت اولیه کرم‌های شب‌تاب و تعداد تکرار الگوریتم کرم شب‌تاب می‌باشد. با افزایش تعداد کرم‌های شب‌تاب و پراکنش آن‌ها در فضای مسئله درواقع شانس یافتن بهینه‌های سراسری بیشتر می‌شود. تکرار زیاد نیز منجر به همگرا شدن جمعیت کرم‌های شب‌تاب به جواب‌های بهینه می‌شود. اعمال چنین روشی، زمان اجرای الگوریتم تکاملی را بیشتر می‌نماید با این تفاوت که در اکثر موارد قابل‌قبول نیست.

## ۵) معماری کودا

امروزه جی‌پی‌یو[10] به عنوان یک پردازنده با منابع محاسباتی بسیار زیاد مورد توجه قرارگرفته است (جی.دی. اوونس، ۲۰۰۸). تعداد هسته‌های فراوان این پردازنده و همچنین پهنای باند زیاد حافظه کارت‌های گرافیکی، قابلیت اجرای تعداد نخ‌های موازی بسیار زیادی را در مقایسه با سی‌پی‌یو به این پردازنده داده است. به طور کلی ساختار جی‌پی‌یو به گونه‌ای است که با پردازش دسته‌ای داده‌ها، بازده محاسبات را افزایش می‌دهد.

جی‌پی‌یوها در ابتدا به‌صورت سخت‌افزارهایی با کارکرد ثابت و برای محاسبات گرافیکی به وجود آمدند. تبدیل مسئله هدف به یک مسئله گرافیکی، خود موضوعی چالش‌برانگیز بود که استفاده از جی پی یوها را مشکل و در نتیجه محدود می‌کرد. در نهایت سازندگان این تراشه‌ها به فکر ارائه یک معماری عام به همراه یک زبان سطح بالای برنامه‌نویسی افتاده و مفهوم جی‌پی‌یو همه‌منظوره یا جی‌پی‌جی‌پی‌یو[11] را معرفی کردند. ازجمله مهم‌ترین این معماری‌ها کودا است که در آن هر جی‌پی‌یو شامل چندین واحد پردازشی مستقل به نام جریان‌های چندپردازنده‌ای[12] است. هر یک از این واحدها نیز به‌نوبه خود حاوی چندین هسته اجرایی هستند (زد. یو،۲۰۱۱).

برنامه‌ای که مطابق با معماری کودا نوشته شده باشد شامل دو جز سری و موازی است که قسمت سری آن بر روی سی‌پی‌یو و قسمت موازی آن بر روی جی‌پی‌یو در قالب توابع خاصی به نام کرنل اجرا می‌گردند. در حین اجرا، ابتدا برنامه توسط سی‌پی‌یو شروع به کارکرده و به صورت سری و خط به خط جلو می‌رود. اگر یک خط خاص شامل یک فراخوانی کرنل باشد. درنهایت جی‌پی‌یو وارد عمل شده و تعداد زیادی نخ موازی را که تمامی آن‌ها تابع کرنل را اجرا می‌کنند؛ ایجاد می‌گردد. در معماری کودا فراخوانی یک تابع کرنل در قالب ایجاد یک تور از نخ‌ها صورت می‌گیرد که هر تور حاوی چندین بلوک بوده که هر بلوک نیز حاوی چندین نخ است.

در طی اجرای این طرح، دست‌یابی به اهداف ذیل مورد انتظار است:

---

[10]. GPU
[11]. GPGPU
[12]. Streaming Multi processor

۱- افزایش سرعت حل مسائل بهینه‌سازی

۲- افزایش سرعت و دقت الگوریتم هوش دسته‌جمعی کرم شب‌تاب

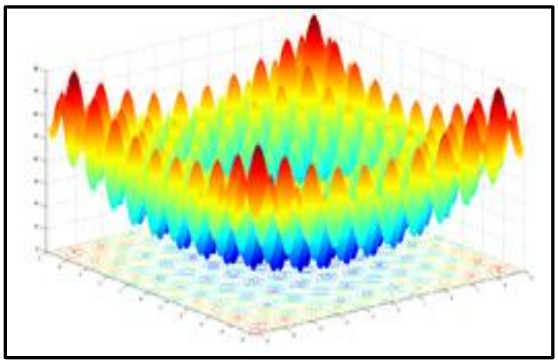
۳- استفاده از سطح بالایی از موازی‌سازی پردازش‌ها از طریق اجرای هزاران رشته پردازشی سخت‌افزاری جهت حل مشکل زمان طولانی محاسبات به عنوان یکی از عوامل بازدارنده در حل مسائل بهینه‌سازی

۴- از آنجا که حافظه کارت‌های گرافیکی با سرعت زیادی افزایش می‌یابد، استفاده از حافظه آن‌ها به‌عنوان حافظه موقت، موجب کاهش ترافیک اطلاعات در گذرگاه کارت PC می‌شود و درنتیجه کارایی افزایش می‌یابد.

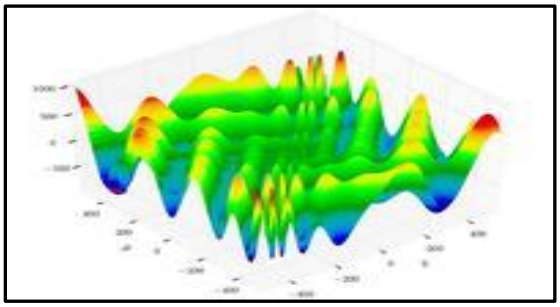
۵- فراهم نمودن امکان تبادل ناهمگام اطلاعات بر پایه پردازنده‌های گرافیکی

## ۵-۱) روش‌شناسی پژوهش

در این مقاله، نسخه الگوریتم پیشنهادی با الگوریتم‌های مانند ژنتیک ، ذرات و الگوریتم کرم شب‌تاب از نظر دقت، سرعت اجرا و همگرایی مقایسه شده است. برای ارزیابی و تحلیل جواب‌ها از توابع بهینه‌سازی موسوم به توابع بنچمارک استفاده شده است. توابع بنچمارک، نوزده تابع ریاضی با اشکال پیچیده هستند. تابع بنچمارک با علائم F1 تا F19 مشخص‌شده‌اند و از این توابع برای سنجش همگرایی الگوریتم‌های تکاملی استفاده می‌شود (جی.ژووا،۲۰۱۰). در شکل (۱) تابع بنچمارک راستریجن[13] و در شکل (۲) تابع بنچمارک اگهولدر[14] نشان داده شده‌اند.



**شکل ۱. تابع بنچمارک راستریجن در مسائل بهینه‌سازی**



**شکل ۱. تابع بنچمارک اگهولدر در مسائل بهینه‌سازی**

[13] Rastrigin
[14] Eggholder

برای محاسبه نمودارهای زمانی در الگوریتم پیشنهادی و رقیب، ابتدا جمعیت اولیه کم در نظر گرفته شده است و برای اجرا یک تابع دلخواه بنچمارک انتخاب شده است الگوریتم پیشنهادی در واحد پردازش کارت گرافیک اجرا می‌شود و زمان اجرا محاسبه و یادداشت می‌شود. همین تعداد جمعیت اولیه را به الگوریتم‌های ژنتیک، ذرات و کرم شب‌تاب غیرموازی تخصیص داده می‌شود. و زمان اجرا آن‌ها در سی‌پی‌یو یادداشت می‌شود. این فرآیند را ۵۰ بار برای این جمعیت اولیه اجرا شده است و میانگین زمان الگوریتم‌ها محاسبه می‌شود. در مرحله بعد مقدار جمعیت اولیه را افزایش داده و مرحله قبلی مجدد طی می‌شود. این مراحل تا افزایش بیشتر جمعیت و اندازه داده‌ها مرتب انجام می‌شود و نمودار مقایسه‌ای بین الگوریتم پیشنهادی، ژنتیک، ذرات و کرم شب‌تاب استاندارد برحسب زمان – اندازه داده (یا جمعیت اولیه در هر آزمایش) رسم می‌شود این نمودار مربوط به یک تابع بنچمارک است و برای تعدادی از این توابع انجام می‌شود که یکی دیگر از روش‌های تحقیق ما محاسبه شتاب الگوریتم پیشنهادی نسبت به الگوریتم غیرموازی کرم شب‌تاب است.

### ۵–۲) روش‌هایی جهت بهبود الگوریتم کرم شب‌تاب

یکی از روش‌های اجرای الگوریتم‌های تکاملی با تعداد جمعیت اولیه انبوه و تکرار زیاد، اجرای آن‌ها به شکل موازی است. روش‌های موازی‌سازی امروزه بسیار متنوع هستند. از جمله آن‌ها می‌توان به محاسبات شبکه‌های گرید اشاره کرد. اخیراً امکان پردازش موازی در واحد پردازش کارت گرافیک که دارای صدها و هزاران هسته محاسباتی است، فراهم شده است. واحد پردازش کارت گرافیک در ابتدا برای پردازش تصاویر و عملیات گرافیکی تدارک دیده شده بود. اما اخیراً امکانات موازی‌سازی کودا باعث شده است تا برنامه‌نویسان از پردازنده‌های داخل کارت گرافیک جهت پردازش موازی از الگوریتم‌های خود استفاده نمایند. در این حالت، محاسبات پردازشی به‌جای اینکه روی سی‌پی‌یو اجرا شوند و کل سیستم مرکزی را کند نمایند، بر روی هسته‌های تقریباً بیکار جی‌پی‌یو اجرا می‌شوند. مزایای برنامه‌نویسی کودا بسیار زیاد است؛ از جمله نیاز به خریدن سخت‌افزار بسیار گران‌قیمت نیست، مهم‌تر این که از کارت گرافیک کاربر به‌عنوان یک ابررایانه کوچک در جهت پردازش‌های محاسباتی کاربران استفاده می‌شود و در مصرف انرژی صرفه‌جویی می‌شود.

### ۵–۳) تغییر پارامترهای الگوریتم کرم شب‌تاب به شکل پویا

در الگوریتم استاندارد کرم شب‌تاب ، مقدار $\epsilon_t$ در همه شرایط یک عدد تصادفی در بازه صفر و یک است. اگر کرم‌های شب‌تاب، به نقاطی منتقل شوند که مقدار تابع هدف در آنجا از میزان فعلی کمتر باشند، باید این امکان را داشته باشند که بتوانند پرش و جهش بیشتری داشته باشند.

$$=x_i^t+\beta.\exp(-\gamma r_{ij}^2).(x_j^t-x_i^t)+(\alpha_i+T\alpha_j)\epsilon_t x_i^{t+1} \quad \text{رابطه ٤)}$$

T=0 or T=1

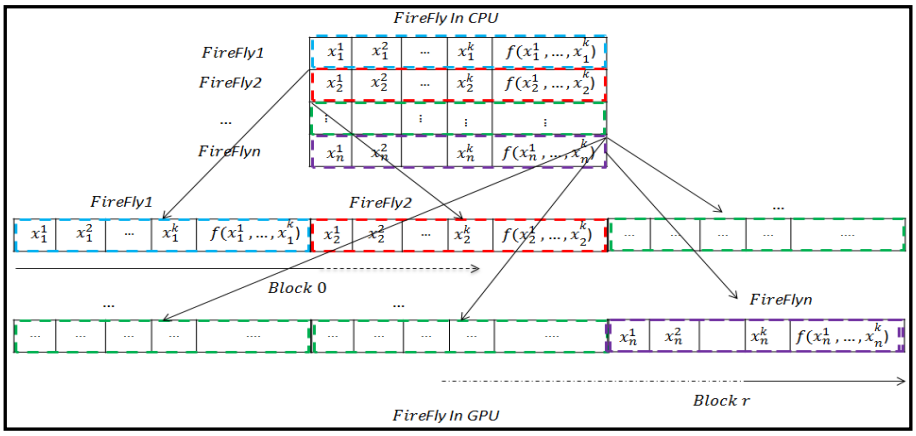
در رابطه (۴) مقدار $\alpha_j$ و $\alpha_i$ دو مقدار ثابت عددی هستند و مقدار T معرف انتقال خوب یا بد خواهد بود. اگر کرم شب‌تابی حرکتی انجام دهد که نسبت به حرکت قبلی شایستگی آن را کاهش دهد، برای حرکت فعلی T=۱ را در نظر می‌گیرد و در غیر این صورت مقدار T=۰ منظور می‌شود. اگر در رابطه (۴) فرض شود که $\alpha_j = \alpha_i$، رابطه (۴) ساده‌تر شده و به رابطه (۵) حاصل خواهد شد.

رابطه ۵)

$$x_i^{t+1}=x_i^t+\beta.\exp(-\gamma r_{ij}^2).(x_j^t-x_i^t)+\alpha_i(1+T)\epsilon_t$$
$$T=0 \quad or \quad T=1$$

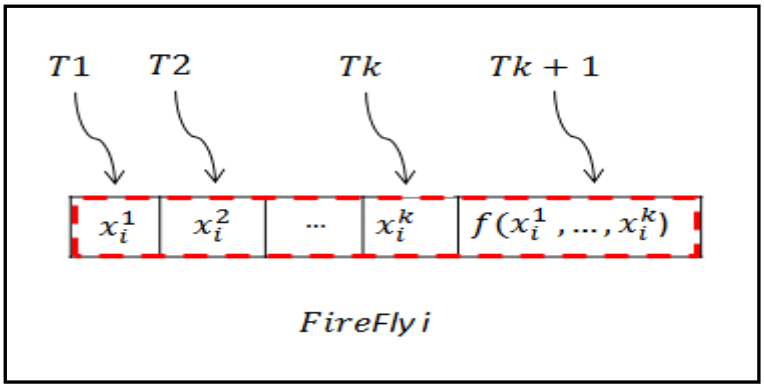
## ۵-۴) اختصاص هسته‌های واحد پردازش کارت گرافیک به کرم‌های شب‌تاب

در الگوریتم کرم شب‌تاب، هر کرم شب‌تاب در واقع یک بردار به‌اندازه k می‌باشد که k، اندازه بعد مسئله است. اگر تعداد کرم‌های شب‌تاب n باشد، روش مرسوم در برنامه‌نویسی، استفاده از یک ماتریس n در k می‌باشد که هر سطر معرف یک کرم شب‌تاب و هر ستون معرف یک بعد از مسئله است. در این روش برای پردازش ماتریس در سی‌پی‌یو باید ردیف‌های ماتریس به شکل سریالی پردازش شوند. در این فرآیند اگر تعداد ابعاد مسئله و جمعیت اولیه کرم‌های شب‌تاب زیاد باشد دارای زمان اجرای بسیار بالایی خواهد بود این مسئله زمانی حادتر خواهد شد که الگوریتم کرم شب‌تاب دارای تکرار زیادی باشد. شکل (۳) نحوه قرار گرفتن کرم‌های شب‌تاب در سی‌پی‌یو و جی‌پی‌یو را نشان داده است.




**شکل ۳. نحوه قرار گرفتن کرم‌های شب‌تاب در سی‌پی‌یو و جی پی یو**

در این روش، هر نخ یک بعد از هر کرم شب‌تاب را زمان‌بندی می کند و مجموعه‌ای از نخ‌ها مسئول زمان‌بندی یک کرم شب‌تاب هستند. هر بلاک در روش پیشنهادی شامل تعدادی کرم شب‌تاب است. هم‌چنین تعداد بلاک‌ها توسط برنامه‌نویس تعیین می‌شود. شکل (۴) نحوه مدیریت یک کرم شب‌تاب توسط تعدادی نخ را نشان می‌دهد.




**شکل٤. مدیریت اجزاء یک کرم شب‌تاب توسط تعدادی نخ**

## ۶) شبه کد روش پیشنهادی

شکل (۵)، شبه کد روش پیشنهادی را نشان می‌دهد. ورودی‌های الگوریتم در شبه کد مواردی مانند آرایه‌ها، پارامترها، ثابت‌ها، تابع ارزیابی و شایستگی، تعداد تکرار، تعداد جمعیت اولیه کرم‌های شب‌تاب و غیره می‌باشند. در روش پیشنهادی ابتدا تمام متغیرها و آرایه‌ها در حافظه سراسری سیستم مرکزی مقداردهی اولیه می‌شوند و مکان اولیه کرم‌های شب‌تاب نیز به شکل تصادفی انتخاب می‌شوند. در مرحله‌ی بعد آرایه‌ها را از سی‌پی‌یو به جی‌پی‌یو انتقال داده می‌شوند. برای انتقال آرایه‌ها از سی‌پی‌یو به جی‌پی‌یو از توابع موجود کتابخانه کودا استفاده می‌شود. سپس کرنل در یک حلقه تکرار قرارگرفته می‌شود تا در هر گام حلقه، بهترین کرم شب‌تاب حاصل شود. بنابراین کرم شب‌تاب به‌دست‌آمده از حلقه‌ی تکرار را با بهترین کرم شب‌تاب سراسری مقایسه می‌کنیم. در صورتی که شایستگی بهترین کرم شب‌تاب فعلی از کرم شب‌تاب‌های سراسری بهتر بود آن را جایگزین کرم شب‌تاب سراسری می‌کنیم. درصورتی‌که کرم‌های شب‌تاب موجود در حلقه بخواهند مقدار تابع ارزیابی کمتری حاصل نمایند، T=۱ می‌شود و در غیر این صورت T=۰ خواهد شد. درنهایت بهترین کرم شب‌تاب سراسری از واحد پردازش کارت گرافیک خارج و در پردازنده مرکزی کپی می‌شود و نتایج به‌عنوان خروجی قابل‌دسترس است.

```
Algorithm GPU-FireFly
Input : arrays , parameter, const , fitness  function , iterNum,NumFA,etc
Initialize the States of FAs  randomly
Initialize the random arrays
Initialize the parameter of  FA algorithms
Transfer data from CPU to GPU
foriter ← 1    toiterNum
do
        Calculate  fitness  values  of  all  FAs by __device__ fitness  function
         Update  the  optimization  result by
Kernel<<<BlockNum,ThreadNum>>>
         If  Current fitness  FA > Next  fitness  FA
 Set   T←0
elseif
                 Set   T←1
Set Best FireFly
         If  Best  FA >Global  FA  : Set   Best  FA instead of  Global  FA
end for
 Transfer  device_best_global  FA  From  GPU  to  CPU
Output : best global FA
```

**شکل ۵. شبه کد الگوریتم پیشنهاد**

### ۶-۱) ابزار شبیه‌سازی

برای شبیه‌سازی و پیاده‌سازی الگوریتم پیشنهادی از سیستم‌عامل ویندوز ۷[۱۵] و با حافظه ۶۴ بیتی، کامپایلر ویژوال استادیو۲۰۱۲[۱۶] محصول شرکت مایکروسافت و فریم ورک ۵/۵ محصول شرکت انویدیا[۱۷] استفاده‌شده است. زبان مورد استفاده در شبیه‌سازی‌ها ویژوال سی پلاس پلاس[۱۸] است. یکی از مزایای استفاده از زبان ویژوال سی پلاس پلاس دسترسی

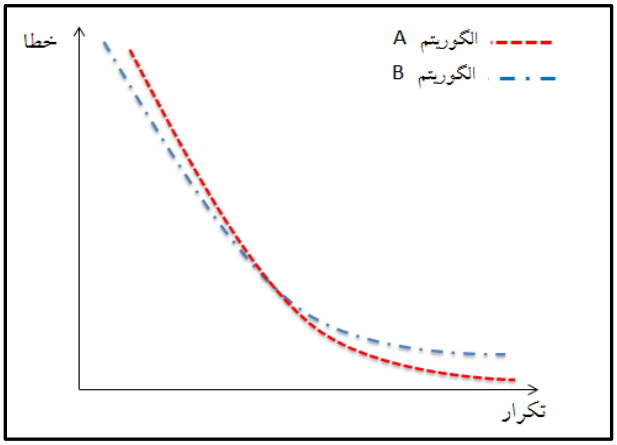
15. Windows7
16. VisualStudio 2012
17. Nvidia
18. VC++

بالا به حافظه، سرعت بیشتر و وجود مستندات فراوان در مورد این زبان در برنامه‌نویسی کودا است. زبان ویژوال سی پلاس پلاس زبان استاندارد در برنامه‌نویسی موازی کودا است.

## ۷) یافته‌های پژوهش

برای ارزیابی الگوریتم پیشنهادی ازنظر سرعت و دقت به معیارهای استاندارد برای ارزیابی نیاز است. ازجمله معیارهای که برای سنجش الگوریتم‌های هوش ازدحامی در واحد پردازش کارت گرافیک بکار می‌رود می‌توان به موارد زیر اشاره کرد:

**۱- سرعت هم‌گرایی:** معیار سرعت هم‌گرایی نشان می‌دهد که کدام الگوریتم تمایل دارد سریع‌تر به جواب‌های موردنظر همگرا شود. در واقع، هم‌گرایی بدین معنا است که بعد از تعداد مشخصی تکرار، خطای بین جواب واقعی و تقریبی از یک مقدار آستانه کمتر است. در شکل (۶)، یک مقایسه بین هم‌گرایی الگوریتم‌های A و B صورت گرفته است. با توجه به این که، با ادامه تکرار میزان خطای نمودار A کمتر از نمودار B می‌باشد. بنابراین هم‌گرایی الگوریتم A دریافتن جواب‌های مسئله، بهتر از هم‌گرایی نمودار B است.




**شکل ٦. مقایسه همگرایی دو الگوریتم مختلف ، به‌عنوان معیار ارزیابی دقت و سرعت**

**۲- شتاب کارت گرافیک:** این معیار به‌صورت رابطه (۶) تعریف می‌شود :

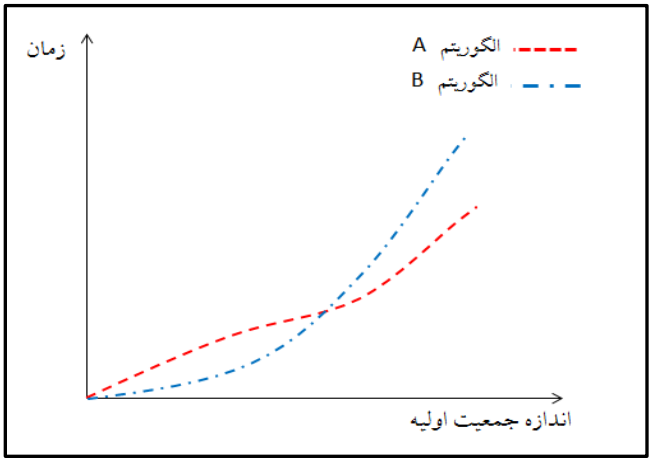
$$SpeedUp = \frac{Time_{CPU}}{Time_{GPU}}$$ رابطه٦)

$Time_{CPU}$ و $Time_{GPU}$ در رابطه (۶) به ترتیب زمان اجرای الگوریتم در واحد پردازش مرکزی و واحد پردازش کارت گرافیک است. هر چه مقدار شتاب بیشتر باشد نشان می‌دهد الگوریتم موازی کارآمدتر بوده است.

**۳- تفاوت کارت‌های گرافیک:** این معیار، برای بیان تفاوت سخت‌افزار و شتاب به کار می‌رود در این ارزیابی الگوریتم را روی چند کارت گرافیک مختلف، از نظر توان پردازشی، اجرا می‌کنند و تأثیر توان پردازشی سخت‌افزارهای مختلف را بر روی سرعت و شتاب الگوریتم‌ها بررسی می‌کنند.

**٤- زمان اجرای الگوریتم A و B:** یکی دیگر از معیارهای مقایسه‌ای، مقایسه زمان اجرای دو الگوریتم است برای مقایسه زمانی و سرعت دو الگوریتم روش‌های مختلفی وجود دارد در این تحقیق زمان اجرا برحسب جمعیت اولیه در نظر

گرفته شده است. به طور کلی جمعیت اولیه را به طور معمول توانی از عدد ۲ در نظر گرفته می‌شود. در شکل (۷)، یک مقایسه زمانی بین دو الگوریتم A و B نشان داده شده است. با توجه به شکل (۷) با افزایش جمعیت اولیه، الگوریتم A در زمان کمتری محاسبات خود را بر روی این جمعیت انجام می‌دهد. بنابراین، نسبت به الگوریتم B سریع‌تر می‌باشد و زمان اجرای کمتری دارد. به طور کلی الگوریتم‌هایی که محاسبات آن‌ها در زمان کمتری نسبت به سایر الگوریتم‌ها انجام می‌شوند، دارای سرعت بیشتری می‌باشند.




**شکل ۷. مقایسه زمان اجرای دو الگوریتم مختلف، به‌عنوان یک معیار ارزیابی سرعت**

**۵- توابع ارزیابی:** توابعی که میزان دقت، سرعت هم‌گرایی الگوریتم‌های تکاملی با آن‌ها بررسی می‌شوند، توابع بنچمارک نامیده می‌شود. توابع بنچمارک توابعی با اشکال پیچیده‌ای هستند. در شکل (۸)، توابع بنچمارک به همراه رابطه ریاضی آن‌ها نشان داده شده است. این توابع در فضای یک‌بعدی تا بعدهای بیشتر را شامل می‌شود. در تحقیق از چهار عدد از این توابع بنچمارک برای ارزیابی استفاده شده است. در شکل (۸) متغیر D به ابعاد مسئله اشاره دارد.

| Function Name | Equation | Bounds |
|---|---|---|
| Sphere | $\sum_{i=1}^{D} x_i^2$ | $(-10,10)^D$ |
| Rastrigin | $\sum_{i=1}^{D} [x_i^2 - 10 * \cos(2\pi x_i) + 10]$ | $(-5,5)^D$ |
| Griewangk | $\frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos(\frac{x_i}{\sqrt{i}}) + 1$ | $(-10,10)^D$ |
| Rosenbrock | $\sum_{i=1}^{D-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$ | $(-10,10)^D$ |

**شکل ۸. توابع بنچمارکی که برای ارزیابی الگوریتم پیشنهادی با سایر الگوریتم‌ها در نظر گرفته شده است**

## ۸) ارزیابی و شبیه‌سازی

در این قسمت به ارزیابی و نتایج حاصل از شبیه‌سازی روش پیشنهادی و مقایسه آن با روش‌های دیگر پرداخته می‌شود.

## ۹) مقایسه زمان اجرای الگوریتم پیشنهادی با سایر الگوریتم‌های تکاملی

در این قسمت الگوریتم کرم شب‌تاب پیشنهادی از نظر زمان اجرا با الگوریتم استاندارد کرم شب‌تاب و الگوریتم ذرات مقایسه شده است. برای مقایسه، اول از تابع اسپیر[19] استفاده شده است. در این آزمایش از ۵۰ بار تکرار الگوریتم برای میانگین گیری استفاده شده است. تعداد تکرار الگوریتم پیشنهادی و کرم شب‌تاب استاندارد، ۱۰۰ در نظر گرفته شده است. مشخصه کارت گرافیک برای تمام شبیه‌سازی‌ها GeForce 710M است. در این شبیه‌سازی از تغییر جمعیت اولیه به‌عنوان یک پارامتر متغیر استفاده شده است. شبیه‌سازی و پیاده‌سازی‌ها روی این تابع بنمچارک با D= ۳ یا به عبارتی فضای سه‌بعدی در نظر گرفته‌شده است. در جدول (۱) الگوریتم پیشنهادی که در واحد پردازش کارت گرافیک اجرا می‌شود با الگوریتم کرم شب‌تاب استاندارد که در پردازنده مرکزی اجرا می‌شود مقایسه شده است. در این جدول اندازه جمعیت کرم شب‌تاب‌ها از ۱۶ تا ۱۰۲۴ تغییر کرده است. همچنین تأثیر اندازه جمعیت اولیه کرم‌های شب‌تاب روی زمان اجرای الگوریتم پیشنهادی در واحد پردازش کارت گرافیک و الگوریتم کرم شب‌تاب استاندارد بررسی‌شده است. اطلاعات جدول (۱) نشان می‌دهد اگر تعداد جمعیت اولیه کم باشد زمان اجرا در سی‌پی‌یو بهتر از زمان جی‌پی‌یو است. و اگر اندازه کرم‌های شب‌تاب بیشتر از ۱۲۸باشد شتاب کارت گرافیک بیشتر از یک می‌شود.

**جدول ۱. مقایسه الگوریتم پیشنهادی با الگوریتم کرم شب‌تاب استاندارد، با داده‌های مختلف و تابع اسپیر.**

| سرعت | زمان جی پی یو کرم شب‌تاب(میلی ثانیه) | زمان سی پی یو کرم شب‌تاب (میلی‌ثانیه) | کرم‌های شب‌تاب |
|---|---|---|---|
| ۰/۰۴۰ | ۰/۷۱۳۰ | ۰/۰۳۲۰ | ۱۶ |
| ۰/۱۸۰ | ۰/۸۰۲۰ | ۰/۱۴۴۰ | ۳۲ |
| ۰/۶۳۰ | ۰/۸۹۳۰ | ۰/۵۶۰۰ | ۶۴ |
| ۲/۰۱۰ | ۱/۰۷۹۰ | ۲/۱۶۸۰ | ۱۲۸ |
| ۶/۰۷۰ | ۱/۴۲۷۰ | ۸/۶۶۸۰ | ۲۵۶ |
| ۱۵/۶۲ | ۲/۱۴۶۰ | ۳۳/۵۱۰ | ۵۱۲ |
| ۲۰/۹۳ | ۶/۳۹۲۰ | ۱۳۳/۷۲ | ۱۰۲۴ |

در شبیه‌سازی دوم از تابع بنچمارک راستریجن، در شبیه‌سازی سوم از تابع بنچمارک گریوانک[20] و در شبیه‌سازی چهارم از تابع بنچمارک رزنبرک استفاده شده است. در جداول (۲)، (۳) و (۴) نتایج این شبیه‌سازی‌ها نشان داده شده است.

**جدول ۲. مقایسه الگوریتم پیشنهادی با الگوریتم کرم شب‌تاب استاندارد، با داده‌های مختلف و تابع راستریجن**

| سرعت | زمان جی پی یو کرم شب‌تاب(میلی‌ثانیه) | زمان سی پی یو کرم شب‌تاب(میلی‌ثانیه) | کرم‌های شب‌تاب |
|---|---|---|---|
| ۰/۰۵۰ | ۰/۷۴۱۰ | ۰/۰۳۹۰ | ۱۶ |
| ۰/۱۸۰ | ۰/۹۲۴۰ | ۰/۱۶۲۰ | ۳۲ |
| ۰/۵۷۰ | ۱/۱۱۲۰ | ۰/۶۳۸۰ | ۶۴ |
| ۱/۷۱۰ | ۱/۵۰۹۰ | ۲/۵۸۲۰ | ۱۲۸ |

[19] Sphere
[20] Griewangk

| سرعت | زمان جی پی یو کرم شب‌تاب(میلی ثانیه) | زمان سی پی یو کرم شب‌تاب(میلی ثانیه) | کرم‌های شب‌تاب |
|---|---|---|---|
| ۴/۳۹۰ | ۲/۳۱۲۰ | ۱۰/۱۴۸ | ۲۵۶ |
| ۱۰/۶۳ | ۳/۷۷۶۰ | ۴۰/۱۴۹ | ۵۱۲ |
| ۱۲/۴۳ | ۱۲/۸۱۸ | ۱۵۸/۱۴ | ۱۰۲۴ |

**جدول ۳. مقایسه الگوریتم پیشنهادی با الگوریتم کرم شب‌تاب استاندارد، با داده‌های مختلف و تابع گریوانک**

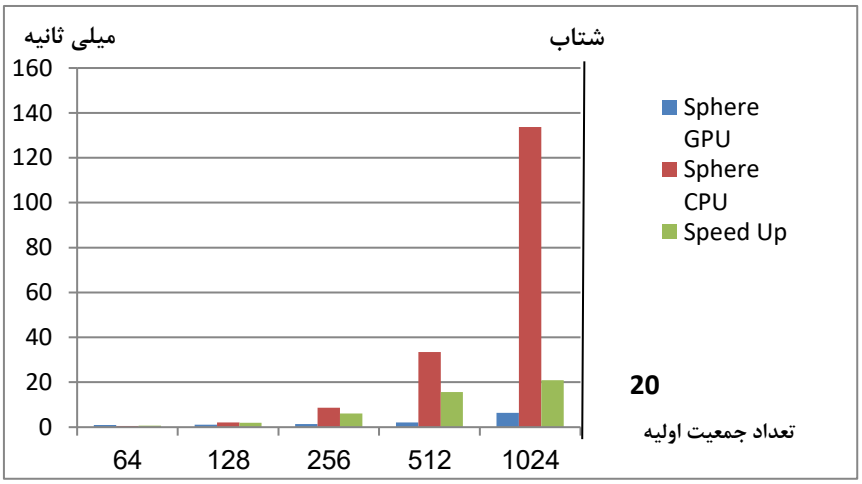
| سرعت | زمان جی پی یو کرم شب‌تاب(میلی ثانیه) | زمان سی پی یو کرم شب‌تاب(میلی ثانیه) | کرم‌های شب‌تاب |
|---|---|---|---|
| ۰/۰۶۰ | ۰/۷۳۸۰ | ۰/۰۴۳۰ | ۱۶ |
| ۰/۱۹۰ | ۰/۹۵۰۰ | ۰/۱۷۸۰ | ۳۲ |
| ۰/۶۰۰ | ۱/۱۴۹۰ | ۰/۶۹۴۰ | ۶۴ |
| ۱/۸۷۰ | ۱/۵۴۰۰ | ۲/۸۷۷۰ | ۱۲۸ |
| ۴/۵۱۰ | ۲/۳۵۴۰ | ۱۰/۶۲۸ | ۲۵۶ |
| ۱۰/۵۷ | ۳/۹۸۱۰ | ۴۲/۰۷۸ | ۵۱۲ |
| ۱۲/۲۳ | ۱۳/۷۳۲ | ۱۶۷/۹۴ | ۱۰۲۴ |

**جدول ٤. مقایسه الگوریتم پیشنهادی با الگوریتم کرم شب‌تاب استاندارد، با داده‌های مختلف و تابع رزنبرک**

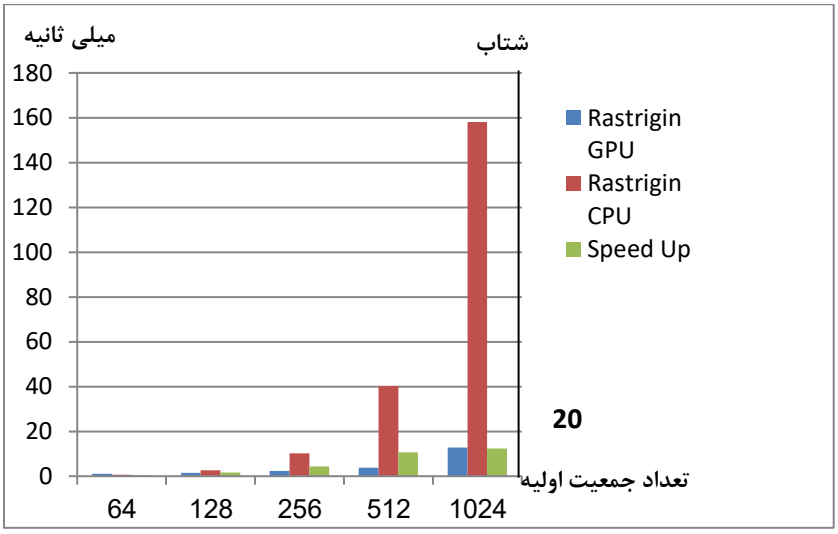
| سرعت | زمان جی پی یو کرم شب‌تاب(میلی ثانیه) | زمان سی پی یو کرم شب‌تاب(میلی ثانیه) | کرم‌های شب‌تاب |
|---|---|---|---|
| ۰/۱۰۰ | ۰/۷۰۷۰ | ۰/۰۶۸۰ | ۱۶ |
| ۰/۲۰۰ | ۰/۷۹۹۰ | ۰/۱۶۱۰ | ۳۲ |
| ۰/۵۷۰ | ۰/۹۸۶۰ | ۰/۵۶۱۰ | ۶۴ |
| ۱/۹۸۰ | ۱/۰۷۲۰ | ۲/۱۲۷۰ | ۱۲۸ |
| ۶/۴۴۰ | ۱/۳۵۳۰ | ۸/۷۱۷۰ | ۲۵۶ |
| ۱۶/۷۳ | ۲/۰۲۲۰ | ۳۳/۸۳۴ | ۵۱۲ |
| ۲۲/۷۶ | ۵/۹۲۷۰ | ۱۳۴/۸۹ | ۱۰۲۴ |

شکل‌های (۱۰)، (۱۱)، (۱۲) و (۱۳) میزان زمان اجرای الگوریتم پیشنهادی در واحد پردازش و الگوریتم استاندارد کرم شب‌تاب، به ترتیب با توابع بنچمارک اسپیر، راستریجن، گریوانک و رزنبرک بررسی شده است. در این نمودارها محور افقی اندازه جمعیت اولیه و محور عمودی دارای دو معیار زمان‌بر حسب میلی‌ثانیه و شتاب کارت گرافیک است. بررسی دقیق نمودارها نشان می‌دهد که اگر اندازه جمعیت اولیه کم باشد در همه موارد زمان اجرا در سی‌پی‌یو کمتر از زمان اجرا در جی‌پی‌یو خواهد بود. اما اگر تعداد جمعیت اولیه افزایش یابد زمان اجرا در جی پی یو کمتر از سی‌پی‌یو خواهد بود. شتاب کارت گرافیک در تمام نمودارها روندی صعودی دارد که با افزایش تعداد جمعیت اولیه کرم‌های شب‌تاب، اندازه شتاب بیشتر می‌شود. این نتایج نشان می‌دهد که افزایش اندازه داده‌ها تأثیر بیشتری روی شتاب کارت گرافیک دارد. در مواردی که برای به دست آوردن جواب‌های دقیق نیاز است تعداد جمعیت اولیه کرم‌های شب‌تاب و

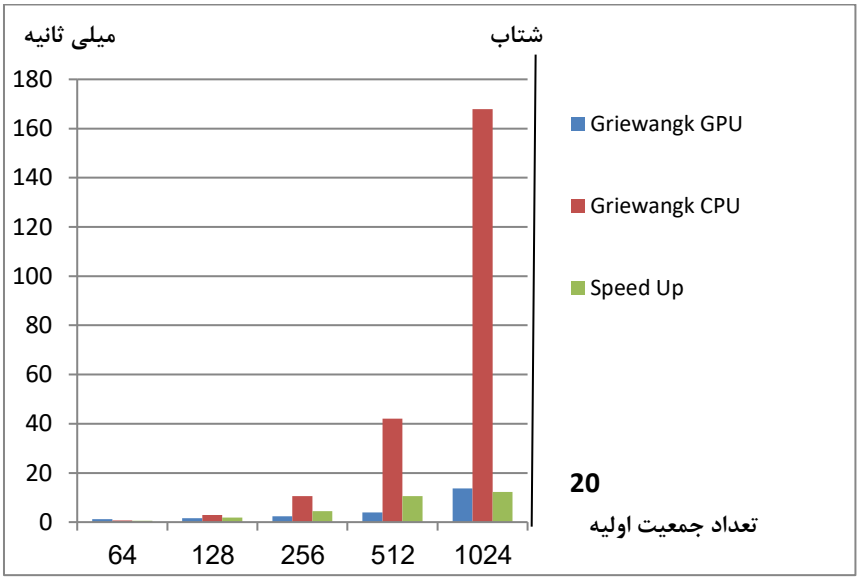
تکرار الگوریتم زیاد باشد. روش پیشنهادی دارای سرعتی تا ۱۶۰ برابر سی‌پی‌یو است. اما در حالی که تعداد جمعیت اولیه و تکرار کم باشد استفاده از سی‌پی‌یو توصیه می‌شود.



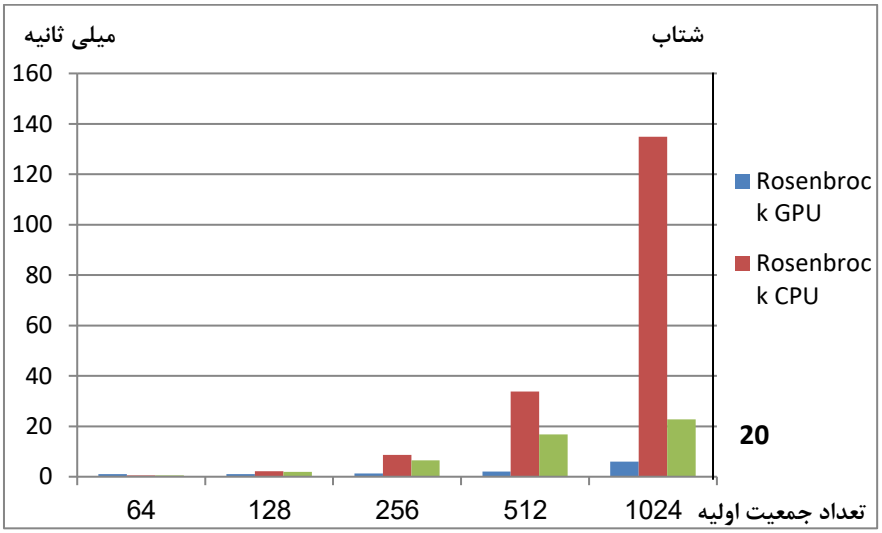

**شکل ۱۰. مقایسه زمان و شتاب اجرای الگوریتم پیشنهادی، الگوریتم کرم شب‌تاب استاندارد با تابع ارزیابی اسپیر**



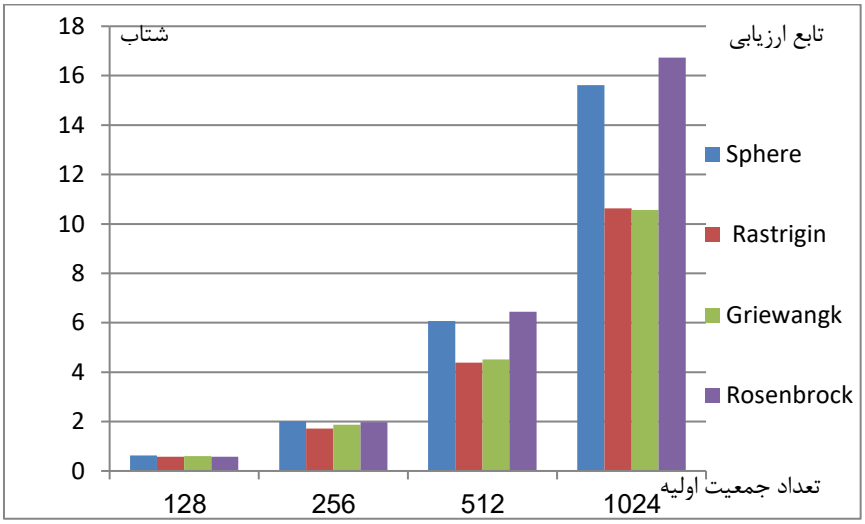

**شکل ۱۱. مقایسه زمان و شتاب اجرای الگوریتم پیشنهادی، الگوریتم کرم شب‌تاب استاندارد با تابع ارزیابی راستریجن**




**شکل ۱۲. مقایسه زمان و شتاب اجرای الگوریتم پیشنهادی، الگوریتم کرم شب‌تاب استاندارد با تابع ارزیابی گریوانک**

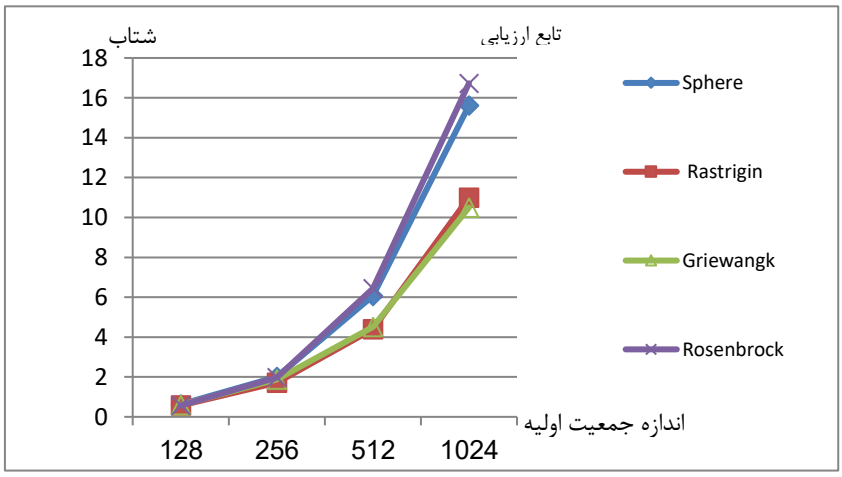



**شکل ۱۳. مقایسه زمان و شتاب اجرای الگوریتم پیشنهادی، الگوریتم کرم شب‌تاب استاندارد با تابع ارزیابی رزنبرک**



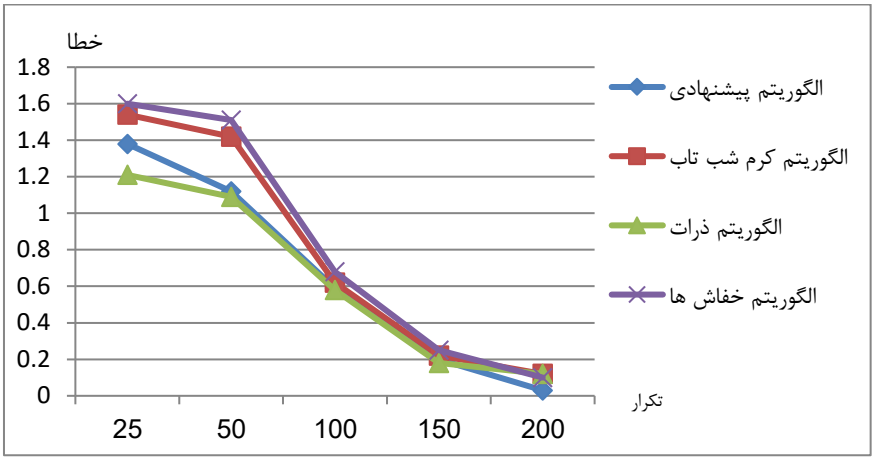

**شکل ١٤. مقایسه (میله‌ای) شتاب اجرای الگوریتم با استفاده از تابع ارزیابی اسپیر، راستریجن، گریوانک، رزنبرک**

در نمودار (۱۳) روندی متفاوت از نمودارهای میله‌ای برای بررسی شتاب کارت گرافیک نشان داده شده است. در این نمودار مشخص است شتاب کارت گرافیک برای توابع بنچمارک اسپیر و رزنبرک بیشتر از دو توابع بنچمارک راستریجن و گریوانک است. این نتیجه‌گیری قابل انتظار بود زیرا شکل توابع بنچمارک اسپیر و رزنبرک ساده‌تر از توابع بنچمارک راستریجن و گریوانک است به طور کلی شکل تابع بنچمارک معرف پیچیدگی بالای مسئله بهینه‌سازی مرتبط است.




**شکل ۱۵. مقایسه (نموداری) شتاب اجرای الگوریتم با استفاده از تابع ارزیابی اسپیر، راستریجن، گریوانک و رزنبرک**

یکی از ارزیابی‌های موجود در این مقاله، بررسی دقت و سرعت همگرایی به جواب‌های بهینه است. در آخرین شبیه‌سازی، میزان میانگین خطا از جواب واقعی را در توابع بنجمارک اسپیر، راستریجن، گریوانک و رزنبرک در مورد الگوریتم پیشنهادی، الگوریتم کرم شب‌تاب استاندارد، الگوریتم ذرات و الگوریتم خفاش‌ها با ۱۰ بار آزمایش محاسبه و نتایج حاصل از آن در شکل (۱۶) نشان داده شده است. در نهایت، دقت الگوریتم پیشنهادی از الگوریتم استاندارد کرم شب‌تاب، الگوریتم ذرات و الگوریتم خفاش بیشتر شده است.




**شکل ۱۶. مقایسه دقت الگوریتم پیشنهادی، کرم شب‌تاب، ذرات و الگوریتم خفاش**

## ۱۰) نتیجه‌گیری و پیشنهادها

در این مقاله از آزمایش‌های مختلفی برای ارزیابی الگوریتم پیشنهادی استفاده شد. یکی از معیارهای ارزیابی الگوریتم پیشنهادی مقایسه زمانی آن با نسخه استاندارد الگوریتم کرم شب‌تاب بود. برای مقایسه از توابع بنچمارک اسپیر، راستریجن، گریوانک و رزنبرک و جمعیت‌های اولیه متغیر استفاده شد. در این آزمایش‌ها از ۵۰ بار تکرار الگوریتم برای میانگین زمان اجرا استفاده شد و تعداد تکرار الگوریتم پیشنهادی و کرم شب‌تاب استاندارد ۱۰۰، در نظر گرفته شد. در تمام پیاده‌سازی‌ها از کارت گرافیکی به مشخصه GeForce 710M استفاده شد. نتایج نشان می‌دهد هرچه اندازه جمعیت اولیه کرم‌های شب‌تاب بیشتر باشد، اختلاف زمان اجرا بین سی‌پی‌یو و جی‌پی‌یو بیشتر و زمان اجرا در جی‌پی‌یو به مراتب کمتر از سی‌پی یو می‌شود و به همان نسبت شتاب کارت گرافیک برای الگوریتم پیشنهادی نیز بیشتر می‌شود از طرفی اگر اندازه داده‌ها یا جمعیت اولیه کم باشد، زمان اجرا در جی‌پی‌یو بیشتر از سی‌پی‌یو می‌شود و به همان نسبت شتاب کارت گرافیک برای الگوریتم پیشنهادی کمتر می‌گردد. نتایج نشان می‌دهد موازی‌سازی با استفاده واحد پردازش کارت گرافیک، زمانی توجیه‌پذیر است که داده‌ها به‌اندازه‌ی کافی بزرگ باشند. داده‌ها در شبیه‌سازی به عنوان جمعیت اولیه بودند. علت این امر این‌گونه توجیه شد که سربار زمانی از داده‌های کم در انتقال از سی‌پی‌یو به جی‌پی‌یو در مقابل زمان اجرا الگوریتم پیشنهادی قابل توجه بود و هر چه داده‌ها بزرگ‌تر باشند این سربار در مقابل زمان اجرا در جی‌پی‌یو نیز ناچیز خواهد بود. از دیگر نتایج این پژوهش، رابطه پیچیدگی مسئله بهینه‌سازی و شتاب الگوریتم پیشنهادی بود. نتایج نشان داد که شتاب کارت گرافیک برای توابع بنچمارک اسپیر و رزنبرک بیشتر از دو تابع بنچمارک راستریجن و گریوانک است. این نتیجه‌گیری قابل‌انتظار بود، زیرا شکل توابع بنچمارک اسپیر و رزنبرک ساده‌تر از توابع بنچمارک راستریجن و گریوانک بود و شتاب کارت گرافیک برای توابع بنچمارک راستریجن و گریوانک تقریباً یکسان بود. به طور کلی شکل تابع بنچمارک، معرف پیچیدگی بالای مسئله بهینه‌سازی مرتبط بود و هر چه مسئله ساده‌تر باشد شتاب کارت گرافیک برای الگوریتم پیشنهادی بیشتر خواهد شد. به طور کلی اگر اندازه داده‌ها به قدر کافی بزرگ باشد شتاب الگوریتم پیشنهادی همیشه از روش استاندار بیشتر است. به عبارتی، زمان اجرای الگوریتم پیشنهادی از روش الگوریتم استاندارد کرم شب‌تاب کم‌تر شد. برای مقایسه دقت درروش پیشنهادی، این روش را با سه الگوریتم کرم شب‌تاب استاندارد، الگوریتم ذرات و الگوریتم خفاش‌ها مقایسه گردید. برای مقایسه دقت و سرعت همگرایی از میانگین نتایج حاصل از ۱۰ آزمایش

با چهار تابع بنچمارک اسپیر، راستریجن، گریوانک و رزنبرک استفاده شده است. در این آزمایش بعد از حدود ۲۰۰ تکرار به بعد، دقت الگوریتم پیشنهادی از الگوریتم کرم شب‌تاب استاندارد، الگوریتم ذرات و الگوریتم خفاش بیشتر شد.

در این مقاله، نشان داده شد که الگوریتم پیشنهادی نسبت به الگوریتم استاندارد کرم شب‌تاب دارای سرعت اجرای بالاتری است. همچنین سرعت و شتاب با افزایش داده‌ها و جمعیت اولیه بیشتر می‌شود. در پژوهش آینده قصد داریم به جای استفاده از یک کارت گرافیک از چند کارت گرافیک که به‌نام سخت‌افزار مالتی جی‌پی‌یو شناخته می‌شوند، استفاده نماییم زیرا سخت‌افزارها به‌جای یک واحد پردازش کارت گرافیک، از چند کارت گرافیک و چند واحد پردازش کارت گرافیک استفاده می‌کند. این مجموعه از واحدهای پردازش کارت گرافیک، توان محاسباتی و شتاب بسیار بالایی را به الگوریتم‌های مختلف می‌دهند.

## منابع


A.Shukla, R. a. (2010). Swarm Intelligence. *in Towards Hybrid and Adaptive Computing*, 187-207. DOI:10.1007/978-3-642-14344-1

A.V Husselmann, a. K. (2012). Parallel Parametric Optimisation with Firefly Algorithms on . *Graphical Processing Units*, 1-7. Doi: 10.3390/app9010007

j.d.Owens, M. H. (2008). GPU Computing. *Proceedings of the IEEE*, 879-899. DOI:10.1109/GreenCom-CPSCom.2010.143

j.Kennedy, R. E. (1995). Particle swarm optimization. *in Neural Networks, Perth*, 1942-1948. DOI:10.1007/s12046-014-0244-7

G.Agarwal, S. (2013). Evaluation performance study of Firefly algorithm, particle swarm optimization and artificial bee colony algorithm for non-linear mathematical optimization functions. 1-8. DOI:10.1109/ICCCNT.2013.6726474

G.Zhua, S. (2010). Gbest-guided artificial bee colony algorithm for numerical function optimization. pp. 3166-3173. doi:10.1016/j.amc.2010.08.049

M.Imran, R. H. (2013). An Overview of Particle Swarm Optimization Variants. *Procedia Engineering*, 491-496. DOI:10.1016/j.proeng.2013.02.063

p.Pospichal, j. J. (2010). Parallel Genetic Algorithm on the CUDA Architecture. *Applications of Evolutionary Computation*, 442-451. DOI:10.1007/978-3-642-12239-2_46

R. Poli, J. K. (2007). Particle swarm optimization. *Swarm Intelligence*, 33-57. DOI:10.1007/s11721-007-0002-0

S.X.Wu, a. W. (2010). The use of computational intelligence in intrusion etection systems A review. 1-35. DOI:10.1016/j.jnca.2012.08.007

S-C. Chu, H.-C. H.-S. (2011). Overview of Algorithms for Swarm Intelligence,". *in Computational Collective Intelligence*, 28-41. DOI:10.1007/978-3-642-23935-9_3

x.s.yang, x. H. (2013). Firefly algorithm: recent advances and applications. International Journal of Swarm Intelligence. pp. 35-50. DOI:10.1007/978-981-15-0306-1_1

x.Yang. (2010). Engineering optimization: an introduction with metaheuristic applications. *John Wiley & Sons*. DOI:10.1002/9780470640425

X.S.Yang. (2010). A new metaheuristic bat-inspired algorithm. In Nature inspired cooperative strategies for optimization. 65-74. DOI:10.1007/978-3-642-12538-6_6

X.Yang. (2013). Bat Algorithm: Literature Review and Applications. 141–149. DOI:10.1504/IJBIC.2013.055093

Z.Yao, a. J. (2011). A quantitative performance analysis model for GPU architectures. *In High Performance Computer Architecture (HPCA)*, 382-393. DOI:10.1109/HPCA.2011.5749745