سومین همایش ملی آموزش
۲۸و۲۹ اردیبهشت ۱۳۹۰
The 3rd National Conference on Education May 18&19, 2011
دانشگاه تربیت دبیر شهید رجائی

# یک الگوریتم یادگیری برای حل مسئله نگهبان راسی کمینه

**علی نوراله [۱]، لعیا محمدی [۲]**

*دانشکده مهندسی برق و کامپیوتر، دانشگاه تربیت دبیر شهید رجایی تهران و دانشکده مهندسی برق و کامپیوتر، دانشگاه آزاد اسلامی قزوین*

*nourollah@aut.ac.ir*

## چکیده

یکی از مسائل بهینه سازی مطرح شده مهم در هندسه محاسباتی، مسئله موزه هنری می‌باشد که هدف آن پیدا کردن کمترین تعداد نگهبان‌های راسی مورد نیاز برای پوشش یک موزه چندضلعی شکل داده شده است و به این صورت مطرح می‌شود: با داشتن یک چندضلعی $P$ با $n$ راس، کمینه نگهبان‌های مورد نیاز برای پوشش آن چه تعداد است؟ این مسئله NP-hard است. در این مقاله یک الگوریتم یادگیری جدید برای حل نسخه راسی مسئله موزه هنری پیشنهاد می‌شود که در آن به نگهبان‌های راسی آموزش داده می‌شود که چگونه از یک موزه هنری حفاظت کنند. براساس این الگوریتم با تکرار آموزش‌ها نتیجه گرفته می‌شود که میانگین کمترین تعداد نگهبان‌های راسی برای پوشش یک چندضلعی دلخواه و یک چندضلعی مونوتون با $n$ راس به ترتیب $\frac{n}{6.49}$ و $\frac{n}{6.84}$ است که نسبت به الگوریتم‌های موجود بهترین مقدار است.

**واژه‌های کلیدی:** هندسه محاسباتی، مسئله موزه هنری، الگوریتم یادگیری ، نگهبان راسی، قابلیت دید.

## ۱- مقدمه

مسئله موزه هنری(Art Gallery Problem)، نوعی از مسئله معروف قابلیت دید در هندسه محاسباتی است که در ۱۹۷۳ با طرح سوالی توسط Victor Klee بیان شد: چند نگهبان ساکن برای پوشش یک اتاق موزه هنری با $n$ دیوار مورد نیاز است[1]؟ سطح موزه هنری به وسیله یک چندضلعی ساده $P$(چندضلعی بسته ساده با درونش) مدل می‌شود. یک نگهبان در $P$، یک راس ثابت با زاویه دید $2\pi$ در نظر گرفته می‌شود. راس $x$، راس $y$ را می‌بیند اگر $xy \subseteq P$ یعنی پاره‌خط $xy$ به طور کامل در درون $P$ قرار گیرد و هیچ قسمتی از آن خارج $P$ واقع نباشد. یک مجموعه نگهبان، $P$ را می‌پوشاند اگر هر راس $P$ حداقل به وسیله یک نگهبان قابل دید باشد. به این ترتیب مسئله موزه هنری، یافتن کمترین تعداد نگهبان‌ها در یک موزه هنری چندضلعی شکل است و مجموعه این نگهبان‌ها همه رئوس چندضلعی را می‌بینند. هدف کاربردی و اصلی، حفاظت از آثار موزه‌ها با قراردادن کمترین تعداد نگهبان(دوربین) می‌باشد. همچنین مسائل از نوع موزه هنری کاربردهایی در روباتیک، برنامه‌ریزی حرکت، بینایی کامپیوتر و تشخیص الگو ،گرافیک، CAM،CAD و شبکه‌های بیسیم نیز دارند[2].

---

۱- استادیار

۲- عضو باشگاه پژوهشگران جوان، دانشگاه آزاد اسلامی، واحد قزوین، قزوین، *l.mohammadi@qiau.ac.ir*

ادامه این مقاله به این شکل سازماندهی شده است: بخش۲ به کارهای انجام شده می‌پردازد. در بخش۳ تعاریف مورد نیاز معرفی می‌شوند. در بخش۴ الگوریتم یادگیری پیشنهادی مطرح می‌شود. بخش۵ آزمایش‌ها و نتایج الگوریتم مطرح شده را مورد بررسی قرار می‌دهد و نهایتا در بخش۶ نتیجه‌گیری خواهد شد.

## ۲- کارهای انجام شده

دو سال بعد از طرح سوال Victor Klee، Chvatal قضیه موزه هنری را مطرح کرد: $\lfloor \frac{n}{3} \rfloor$ نگهبان گهگاه لازم و همیشه کافیست تا یک چندضلعی با $n$ راس پوشش داده شود[3]. تا کنون انواع زیادی از مسئله موزه هنری مطرح شده و مورد مطالعه قرار گرفته‌اند، مانند: مکان نگهبان‌ها(هر جا یا در مکان خاصی مانند روی رئوس)، انواع مختلفی از نگهبان‌ها( مانند نگهبان‌های ساکن و نگهبان‌های سیار) و فرض‌های مختلفی روی چندضلعی ورودی (مانند چندضلعی‌های متعامد ساده، یعنی چندضلعی‌هایی که همه یالهایشان با زاویه قائم دیده می‌شوند)[1].

یک نوع جالب این مسئله، قضیه موزه هنری متعامد است. این قضیه که ابتدا توسط Kahn مطرح و اثبات شد، نشان می‌دهد $\lfloor \frac{n}{4} \rfloor$ نگهبان گهگاه لازم و همیشه کافیست تا یک چندضلعی متعامد ساده با $n$ راس پوشش داده شود[4]. زیر کلاس دیگری از چندضلعی‌ها، چند ضلعی‌های مونوتون(Monotone Polygons) هستند. چندضلعی$P$، نسبت به خط فرضی $l$ مونوتون است اگر مرز آن به دو زنجیره $a$ و $b$ تقسیم‌بندی شود و هر کدام از این زنجیره‌ها نسبت به خط $l$ مونوتون باشند. یک زنجیره را مونوتون گویند اگر بتوان خطی عمود بر آن رسم کرد به طوریکه اشتراک آن با مرز چندضلعی تهی، یک راس یا یک پاره‌خط باشد. قضیه موزه هنری مونوتون، در ۱۹۸۴ توسط Aggarwal مطرح و اثبات شد[5]. این قضیه نشان می‌دهد که $\lfloor \frac{n}{2} \rfloor + 1$ نگهبان، گهگاه لازم و همیشه کافیست تا یک چندضلعی مونوتون با $n$ راس که $r$ راس آن بازتابی(Reflex) هستند حفاظت شود. راس بازتابی، راسی است که زاویه آن بزرگتر از $\pi$ است.

الگوریتم‌هایی کارا، بر اساس اثبات قضایای بالا توسعه پیدا کردند تا چندضلعی‌های ساده، متعامد و مونوتون را به ترتیب با $\lfloor \frac{n}{3} \rfloor$، $\lfloor \frac{n}{4} \rfloor$ و $\lfloor \frac{r}{2} \rfloor + 1$ نگهبان حفاظت کنند. اگر چه این تعداد در برخی موارد ضروری هستند، ولی در اغلب موارد می‌توان یک چندضلعی را با تعداد نگهبان‌های کمتری حفاظت کرد.

یک نوع خاص مسئله موزه هنری، مسئله نگهبان راسی کمینه(Minimum Vertex Guard) (MVG) است که در آن پیدا کردن تعداد کمینه نگهبان‌های راسی مورد نیاز برای پوشش یک چندضلعی ساده مد نظر است. این مسئله، برای هر دو مورد چندضلعی‌های دلخواه و متعامد NP-hard است و حالت کلی آن به این صورت است: با داشتن یک چندضلعی$P$ با $n$ راس، کمینه نگهبان‌های مورد نیاز برای پوشش آن چه تعداد است(مسئله مجموعه نگهبان کمینه(Minimum Set Guard) (MSG))؟[1,2]

در این مقاله مسئله خاص نگهبان راسی کمینه(MVG) مورد مطالعه قرار می‌گیرد که مجموعه نگهبان‌های $G$، زیرمجموعه‌ای از رئوس$P$ است[3]. یک راه حل برای بررسی این پیچیدگی محاسباتی، توسعه الگوریتم‌های تقریبی است. به طورکلی این روش‌های تقریبی می‌توانند مانند راهکار حریصانه به طور خاص برای حل این مسئله طراحی شوند یا بر مبنای روش‌های ابتکاری(Heuristic) و یا فراابتکاری(Metaheuristic) مانند شبیه‌سازی گرم‌شده و الگوریتم ژنتیک طراحی شوند[6,7,8]. در چند مقاله، الگوریتم‌های تقریبی برای حل مسئله MSG توسعه داده شده‌اند[6,7]. در[9]یک الگوریتم ژنتیک به عنوان یک روش تقریبی برای حل مسئله MVG پیشنهاد شده است. بر اساس این الگوریتم، نتیجه گرفته می‌شود که میانگین تعداد نگهبان‌های راسی کمینه مورد نیاز برای پوشش یک چندضلعی ساده با $n$ راس $\frac{n}{6.38}$ و نرخ تقریبی مشاهده شده کوچکتر یا مساوی 2 و برای چندضلعی‌های متعامد $\frac{n}{6.40}$ با یک نرخ تقریب 1.9 است.

در این مقاله یک الگوریتم یادگیری برای حل مسئله MVG پیشنهاد شده و نتایج آزمایش به وسیله پیاده‌سازی توصیف می‌شود. این نتایج روی یک مجموعه بزرگ از چندضلعی‌های تصادفی تولید شده به دست می‌آیند.

## ۳- تعاریف اولیه

در این مقاله، یک الگوریتم یادگیری برای محاسبه مجموعه نگهبان راسی کمینه روی یک چندضلعی ساده $P$، پیشنهاد می‌شود. $P$ یک چندضلعی ساده با $n$ راس $p_0, p_1, \dots, p_{n-1}$ است. $r$ تعداد رئوس بازتابی را نشان می‌دهد. برای یک راس $p_i \in P$ ، چندضلعی قابل دید از $p_i$، $Vis(p_i)$ نامیده می‌شود که شامل مجموعه‌ای از رئوس $p_j \in P$ است که از راس $p_i$ قابل دید هستند، یعنی:

$$Vis(p_i) = \{p_j \in P; \quad p_i \; sees \; p_j\}$$

$G \subseteq P$ یک مجموعه نگهبان راسی از $P$ است، اگر $G$، $P$ را بپوشاند. میانگین تعداد نگهبان‌های راسی کمینه با $|G|$ نشان داده می‌شود. همچنین فرض می‌شود کل رئوس چندضلعی در لیست $P(i)$ قرار دارند ($0 \le i \le n\text{-}1$) و لیست نگهبان‌ها $G(g)$ می باشد که $g$ تعداد نگهبان‌ها خواهد بود. در بخش بعدی الگوریتم یادگیری پیشنهادی برای پیداکردن مجموعه نگهبان راسی کمینه توضیح داده می‌شود.

## ۴- الگوریتم یادگیری پیشنهادی

الگوریتم یادگیری موردنظر به تعداد itration تکرار می‌شود و در نهایت روی نتایج آموزش‌ها میانگین گرفته می‌شود و $|G|$ میانگین نگهبان‌های حاصل از تکرار آموزش‌ها را نشان می‌دهد. در این الگوریتم یادگیری، ابتدا برای تولید رئوسی با مختصات $x$ و $y$ به عنوان رئوس چندضلعی $P$، $n$ راس تصادفی با مختصات $x$ و $y$ تولید می‌شود. این کار با فراخوانی تابع *Random-Generate-Points*($n$) انجام می‌شود. سپس برای ساخت یک چندضلعی ساده با این رئوس داده شده، تابع *Random-Polygon-Generator(P)* صدا زده می‌شود. این تابع برای تولید چندضلعی‌های تصادفی دلخواه، الگوریتم TwoOptMoves [10] و برای تولید چندضلعی‌های مونوتون تصادفی، الگوریتم TwoPeasants [10] را شبیه‌سازی می‌کند.

برای هر $p_i$، $Vis(p_i)$ آن، با پیدا کردن قطرهای داخلی $p_ip_j$ که همه رئوس قابل‌دید $p_j$ از $p_i$ را نشان می‌دهند، مشخص می‌شود. با فراخوانی تابع *Find-Guards*()، تعداد نگهبان‌ها در هر بار تکرار الگوریتم تعیین می‌شوند. الگوریتم۱، روند الگوریتم تعیین نگهبان راسی کمینه را نشان می‌دهد که در آن $|G|$ میانگین تعداد نگهبان‌های برگردانده شده توسط این الگوریتم است. در ادامه، کار هر یک از این توابع به اختصار شرح داده می‌شود.

```
Minimum-Vertex-Guard(n)    //A learning Algorithm
Random-Generate-Points(n) ;
Random-Polygon-Generator(P) ;
Int Sum=0;
For i ← 1 to itration do
        For i ← 0 to n − 1 do
                Determine Vis(p_i) ;
                g ← Find-Guards();
        End for
        Sum=Sum+g;
End for
|G| = Sum/itration;
Return |G|
```

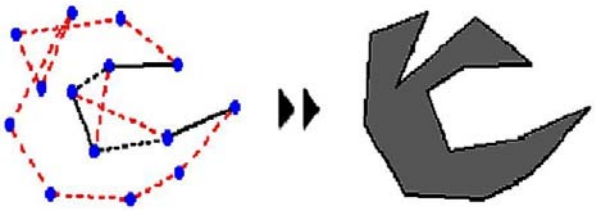
الگوریتم (۱)

## ۴-۱- تولید چندضلعی تصادفی

پس از تولید رئوس تصادفی، تابع تولید چندضلعی تصادفی صدا زده می‌شود. این تابع برای تولید چندضلعی‌های تصادفی دلخواه، الگوریتم TwoOptMoves و برای تولید چندضلعی‌های تصادفی مونوتون، الگوریتم TwoPeasants را شبیه‌سازی می‌کند.

روند الگوریتم TwoOptMoves به این صورت است که در یک چندضلعی کاملا تصادفی تولید شده، به طور مکرر یک جفت یال متقاطع $(s,t),(u,v)$ جستجو می‌شود و با جفت $(s,v),(u,t)$ جایگزین می‌شود تا تقاطع حذف شود. پس از حذف تمام تقاطع‌ها یک چندضلعی تصادفی حاصل می‌شود(شکل۱).

الگوریتم TwoPeasants برای تولید چندضلعی تصادفی مونوتون، به این ترتیب عمل می‌کند: ابتدا رئوس با کمترین و بیشترین مختصات $x$ در چندضلعی تعیین شده و به وسیله یک خط فرضی به هم متصل می‌شوند. سپس رئوس بالای خط با شروع از چپ‌ترین راس تا راست‌ترین راس به ترتیب به نزدیک‌ترین راس مجاور خود متصل می‌شوند (مرتب‌سازی رئوس بر اساس مختصات $x$) و بعد مجموعه رئوس پایین خط فرضی با شروع از راست‌ترین تا چپ‌ترین راس بر اساس مختصات $x$ مرتب می‌شوند. در مرحله بعد همه رئوس مرتب شده بالایی و مجموعه رئوس پایینی به ترتیب مرتب شده به هم متصل می‌شوند. چندضلعی حاصل یک چندضلعی مونوتون است.



شکل۱- جابه‌جایی دو یال متقاطع با TwoOptMoves [10]

## ۴-۲- تعیین چندضلعی قابل دید

برای تعیین نگهبان‌ها، ابتدا باید برای هر راس $p_i$ ، چندضلعی قابل دید آن یعنی $Vis(p_i)$ مشخص شود و برای تعیین $Vis(p_i)$ باید قطرهای داخلی $p_ip_j$ که همه رئوس قابل‌دید $p_j$ از $p_i$ را نشان می‌دهند، مشخص شوند. با فرض برقراری همه قطرهای داخلی و خارجی بین رئوس، برای مشخص کردن قطرهای داخلی، ابتدا قطرهای خارجی حذف شده و سپس قطرهایی که چندضلعی را قطع کرده اند حذف می‌شوند. اگر $p_{i-1}$، $p_i$ و $p_{i+1}$ سه راس متوالی از چندضلعی باشند، برای حذف قطر خارجی حالات زیر به ترتیب بررسی می‌شوند و در صورت صحیح بودن آن قطر خارجی $p_ip_j$ حذف می‌شوند:

- اگر $p_i$ راس غیربازتابی باشد و $p_{i-1}$ سمت چپ $p_ip_j$ و $p_{i+1}$ سمت راست $p_jp_i$ واقع نباشند.
- اگر $p_i$ راس بازتابی باشد و $p_{i+1}$ سمت چپ یا روی $p_ip_j$ و $p_{i-1}$ سمت چپ یا روی $p_jp_i$ واقع باشند.

## ۴-۳- تعیین نگهبان‌های راسی

برای تعیین نگهبان‌ها به این ترتیب عمل می‌شود که ابتدا تعداد رئوس قابل دید هر راس $p_i$ در $Vis(p_i)$ شمارش شده و در لیست شمارنده $C(Vis(p_i))$ قرار می‌گیرند. همینطور برای هر $p_i$ ، به ازای هر عضو $Vis(p_i)$ آن، تعداد تکرار هر عضو در همه $p_j \in P$ در $Vis(p_j)$ به دست آمده و در لیست شمارنده $C1(Vis(p_i))$ قرار می‌گیرد. سپس رئوس بر اساس راسی که بیشترین رئوس را می‌بیند، از بیشترین مقدار $C(Vis(p_i))$ تا کمترین مقدار آن مرتب می‌شوند.

اگر برای دو راس $p_i$ و $p_j$ ، $C1(Vis(p_i))$ و $C1(Vis(p_j))$ برابر باشند، اولویت برای مرتب‌سازی با راسی خواهد بود که رئوس قابل دید آن تعداد تکرار کمتری در بقیه لیست‌های قابل دید دارند، زیرا پوشش بهتری می‌دهد. حال بر اساس این مرتب‌سازی، $P_0$ راسی خواهد بود که بیشترین رئوس را می‌بیند. یک نگهبان روی راس $P_0$ قرار گرفته، $P_0$ به لیست نگهبان‌ها یعنی $G(g)$ اضافه می‌شود و به تعداد نگهبان‌های $g$ افزوده می‌شود. تمام رئوسی که نگهبان راس $P_0$ می‌بینند در لیست چندضلعی قابل دید خودش، لیست‌های چندضلعی قابل دید رئوس دیگر و لیست کل رئوس یعنی $P(i)$ حذف می‌شود. این روند تا وقتی که هیچ راسی بدون نگهبان باقی نماند ادامه می‌یابد و در نهایت هر راسی توسط یک نگهبان حفاظت می‌شود و هیچ راسی بدون نگهبان نخواهد بود. الگوریتم ۲ روند توضیح داده شده را نشان می‌دهد. در بخش بعدی نتایج آزمایش برای چندضلعی‌های تصادفی دلخواه و چند ضلعی‌های مونوتون مورد بررسی قرار می‌گیرد.

```
Find-Guards()
n1 ← n − 1 ;
While (n1 ≠ −1) do
    For each pi ∈ P do
        Compute C(Vis(pi)) ;
        Compute C1(Vis(pi)) ;
    End For
    Sort vertices P from Max(C(Vis(pi))) to Min(C(Vis(pi)))
    For each i ≠ j do
        If C(Vis(pi))=C(Vis(pj)) then
            determine Min{C1(Vis(pi)),C1(Vis(pj))}
            and then Sort vertices P;
          Add  p0 to G(g) ;
        g ← g + 1;
        For each pk ∈ Vis(p0) do
            For i ← 0 to n − 1 do
                Delete pk ∈ P;
            For j ← 1 to n − 1 do
                Delete  pk ∈ Vis(pj) ;
            Delete  pk ∈ Vis(p0) ;
        End For
    End For
    n1 ← n1 − 1;
End While
```

الگوریتم (۲)

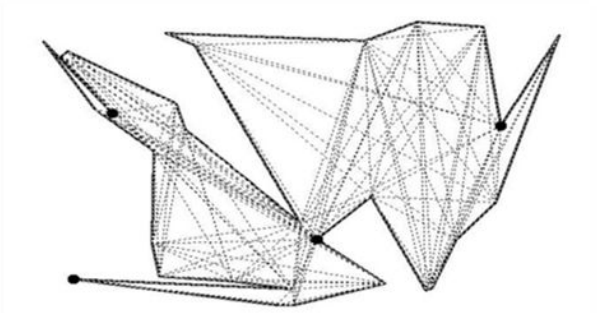
## ۵- آزمایش‌ها و نتایج

پیاده‌سازی این الگوریتم ابتکاری در C++ (Visual Studio 2003.Net) انجام شده است. در این بخش روی نتایج آزمایش‌ها بحث می‌شود. آزمایش‌ها روی یک مجموعه بزرگ از چندضلعی‌های تولید شده تصادفی انجام شده است(بخش ۴-۲). برای تولید چندضلعی‌های تصادفی دلخواه، الگوریتم TwoOptMoves و برای تولید چندضلعی‌های مونوتون الگوریتم TwoPeasants به کار گرفته می‌شود. هدف از آزمایش‌ها این است تا با تکرار الگوریتم به نگهبان‌ها آموزش داده شود که چگونه از موزه هنری حفاظت کنند، بنابراین هدف به دست آوردن میانگین تعداد نگهبان راسی کمینه یعنی $|G|$ روی این چندضلعی های تولید شده است. آزمایش‌ها به ازای ۸ مجموعه چندضلعی، هر یک با ۴۰ چندضلعی تولید شده به ترتیب با ۳۰، ۵۰، ۷۰،

۱۰۰،۱۱۰، ۱۳۰، ۱۵۰و ۲۰۰ راس انجام شده است. در بخش‌های ۵-۱ و ۵-۲ به ترتیب جزئیات نتایج روی چندضلعی‌های تصادفی دلخواه و چندضلعی‌های مونوتون مورد بررسی قرار می‌گیرد.

## ۵-۱- چندضلعی‌های دلخواه

در این بخش ابتدا آزمایش‌ها برای چندضلعی‌های دلخواه تولید شده به وسیله الگوریتم TwoOptMoves (بخش۴-۲) مورد بررسی قرار می‌گیرند. شکل ۲، تعداد نگهبان‌های به دست آمده از این الگوریتم ابتکاری را روی یک چندضلعی تولید شده با ۳۰ راس نشان می‌دهد و جدول۱، میانگین تعداد نگهبان‌های راسی را نسبت به کران بالای $\lfloor \frac{n}{3} \rfloor$ معروف[3] (قضیه موزه هنری معروف مطرح شده در بخش ۲) نشان می‌دهد.



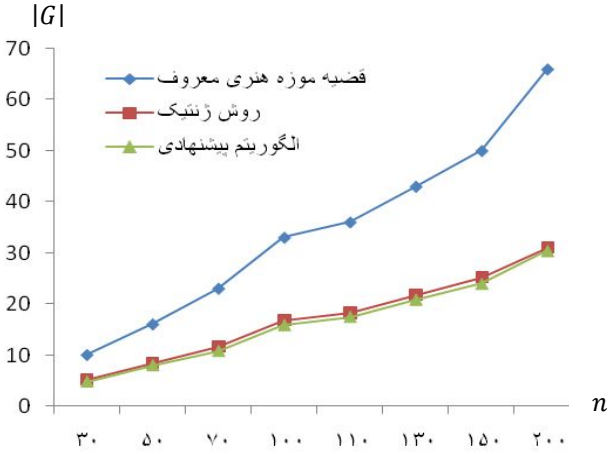
شکل۲ - مجموعه نگهبان‌های $|G|$ (مجموعه نقاط) به دست آمده با الگوریتم پیشنهادی برای یک چندضلعی دلخواه با $n = 30$

جدول ۱- نتایج به دست آمده برای چندضلعی‌های دلخواه

| تعداد رئوس | $\lfloor n/3 \rfloor$ | روش ژنتیک | \|G\| |
|---|---|---|---|
| 30 | 10 | 5.2 | 4.825 |
| 50 | 16 | 8.375 | 8.05 |
| 70 | 23 | 11.625 | 10.85 |
| 100 | 33 | 16.9 | 15.95 |
| 110 | 36 | 18.25 | 17.55 |
| 130 | 43 | 21.8 | 20.925 |
| 150 | 50 | 25.225 | 24.05 |
| 200 | 66 | 31.075 | 30.5 |

برای به دست آوردن میانگین تعداد کمینه نگهبان راسی مورد نیاز، روش حداقل مربعات روی این ۸ مجموعه به کار می‌رود. شکل ۳، میانگین تعداد نگهبان‌ها را برای چندضلعی‌های دلخواه نشان می‌دهد. با استفاده از روش حداقل مربعات معادله خطی ۱ به دست می‌آید(شکل۳).

$$f(x) = 0.1540x + 0.48 \approx \frac{x}{6.49} + 0.48 \qquad (۱)$$

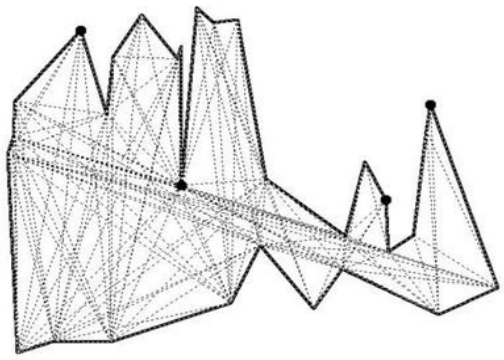



شکل۳ - میانگین تعداد نگهبان‌ها برای چندضلعی‌های دلخواه

به این ترتیب، نتیجه گرفته می‌شود که با تکرار آموزش‌ها میانگین تعداد نگهبـــان راسی مورد نظر برای پوشش یک چندضلعی دلخواه $\frac{n}{6.49}$ است. این نتیجه یک نتیجه خیلی بهتر نسبت به کران بالای $\lfloor \frac{n}{3} \rfloor$ (قضیه مطرح شده در بخش۲) است.

## ۵-۲- چندضلعی‌های مونوتون

یک مطالعه مشابه روی چندضلعی‌های مونوتون تصادفی تولید شده انجام شده است. شکل ۴، تعداد نگهبان‌های به دسـت آمده از این الگوریتم پیشنهادی را روی یک چندضلعی مونوتون تولید شده با ۳۰ راس نشان می‌دهد جـدول۲، میـانگین تعـداد نگهبان راسی کمینه را در مقایسه با کران بالای $\lfloor \frac{n}{3} \rfloor$ معروف و همچنین میانگین $1 + \lfloor \frac{r}{2} \rfloor$ برای چندضلعی‌های مونوتون (قضایای مطرح شده در بخش ۲) نشان می‌دهد[۲و۳].



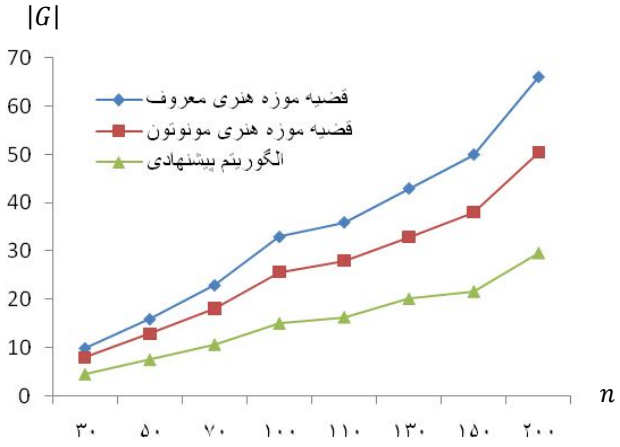
شکل۴ - نگهبان‌های $|G|$ (مجموعه نقاط) به دست آمده با الگوریتم پیشنهادی برای یک چندضلعی مونوتون با $n = 30$

جدول ۲- نتایج به دست آمده برای چندضلعی‌های مونوتون

| $\lvert G\rvert$ | $\lfloor r/2 \rfloor + 1$ | $\lfloor n/3 \rfloor$ | تعداد رئوس |
|---|---|---|---|
| 4.525 | 8.1 | 10 | 30 |
| 7.55 | 12.95 | 16 | 50 |
| 10.7 | 18.15 | 23 | 70 |
| 15.07 | 25.675 | 33 | 100 |
| 16.35 | 28.025 | 36 | 110 |
| 20.175 | 32.925 | 43 | 130 |
| 21.675 | 38 | 50 | 150 |
| 29.65 | 50.325 | 66 | 200 |

برای به دست آوردن میانگین تعداد نگهبان راسی مورد نیاز، روش حداقل مربعات روی این ۸ مجموعه چندضلعی مونوتون، به کار می رود. شکل ۵ میانگین تعداد نگهبان‌های راسی بدست آمده برای چندضلعی‌های مونوتون را نشان می‌دهد. با استفاده از روش حداقل مربعات معادله خطی ۲ به دست می‌آید(شکل ۴).

$$f(x) = 0.1465x + 0.38 \approx \frac{x}{6.82} + 0.38 \qquad (۲)$$




شکل۵ - میانگین تعداد نگهبان‌ها برای چندضلعی‌های مونوتون

به این ترتیب نتیجه گرفته می‌شود که بر اساس تکرار آموزش‌ها میانگین تعداد نگهبان‌های راسی کمینه مورد نیاز برای پوشش یک چندضلعی مونوتون $\frac{n}{6.82}$ است و این نتیجه نسبت به کران بالای $\left\lfloor \frac{n}{3} \right\rfloor$ و نیز نسبت به $\left\lfloor \frac{r}{2} \right\rfloor + 1$ برای چندضلعی‌های مونوتون بسیار بهتر می‌باشد.

## ۶- نتیجه‌گیری

در این مقاله، یک الگوریتم یادگیری جدید برای حفاظت موزه هنری به وسیله کمترین مجموعه نگهبان راسی مورد پیشنهاد شد. آزمایش‌های محاسباتی نشان داد که به طور میانگین بر اساس تکرار آموزش‌ها تعداد کمینه نگهبان‌های راسی موردنیاز برای پوشش یک چندضلعی ساده دلخواه $\frac{n}{6.49}$ و برای یک چندضلعی مونوتون $\frac{n}{6.82}$ است. تلاش بر این است تا در کارهای آینده الگوریتم‌های یادگیری طوری بر مبنای مسئله چندضلعی قابل دید و دیگر مسائل هندسی بهبود یابد تا بتوان به میانگین نگهبان‌های راسی کمتری دست یافت. همچنین تلاش بر این است تا نتایج روی چندضلعی‌های خاص دیگری بررسی شود.

## مراجع

[1] J. Urrutia, J.-R. Sack, *"Art Gallery and Illumination Problems"*, Handbook of Computational Geometry, Elsevier, 2000.
[2] M.L. Karavalas, C.D. Toth, and E.P. Tsigaridas, *"Guarding curvilinear art galleries with vertex or point Guards"*, Computational Geometry, Vol 42, issues 6-7, pp. 522-535, 2009.
[3] V. Chvátal, *"A combinatorial theorem in plane geometry"*, J. of Combinatorial Theory (Series B), pp. 39-41, 1975.
[4] J. Kahn, M. Klawe, D. Kleitman, *"Traditional galleries require fewer watchmen"*, SIAM J. Algebraic and Discrete Methods, pp. 194-206, 1983.
[5] J. Orourke, "*Art Gallery Theorems and Algorithms"*. Oxford University Press, 1987.
[6] Y. Amit, J. S. B Mitchell and E. Packer, *"Locating Guards for Visibility Coverage of Polyons"*, Proceedings of the Workshop on Algorithm Engineering and Experiments, pp. 1–15, 2007.
[7] C. Blum and R. Andreia, "Metaheuristics in combinatorial optimization: Overview and conceptual comparison", ACM Comput. Surv., Vol 35, issues 3, pp. 268–308, 2007.
[8] M. C. Couto, C.C. de Souza, P. J. de Rezende, P.J., *"An exact and efficient algorithm for the orthogonal art gallery problem"*, XX Brazilian Symp. on Comp. Graphics and Image Processing, pp. 87–94, 2007.
[9] A. L. Bajuelos, S. Canales, G. Hern´andez, A.M. Martins, *"Optimizing the Minimum Vertex Guard Set on Simple Polygons via a Genetic Algorithm"*, in WSEAS Transactions in Information Science and Applications, Vol 5, issues 11, pp. 1584–1596, 2008.
[10] http://www.geometrylab.de/polygon/Random Polygon.html.en