

بهبود روش خوشه بندی مورچه ای به کمک اتوماتاهای یادگیر

برنا جعفرپور محمد رضا میبیدی

آزمایشگاه محاسبات نرم

دانشکده مهندسی کامپیوتر و فناوری اطلاعات

دانشگاه صنعتی امیر کبیر

تهران ایران

jafarpour@cic.aut.ac.ir, mmeybodi@aut.ac.ir

چکیده: خوشه بندی مورچه ای روشی الهام گرفته از طبیعت برای خوشه بندی می باشد. این روش دارای مزایایی می باشد که آن را در سالهای اخیر مورد توجه محققان قرار داده است. در این مقاله از یک پیشنهاد برای بهبود خوشه بندی مورچه ای ارائه میشود. در روش پیشنهادی، از اتوماتای یادگیر برای یادگیری مکان قرار دادن داده ها بر روی صفحه استفاده شده است. برای این منظور صفحه ای که مورچه ها بر روی آن خوشه بندی می کنند ناحیه بندی میشود و سپس هر مورچه مجهز به یک اتوماتای یادگیر است. وظیفه اتوماتای یادگیر هر مورچه یادگیری ناحیه ای میباشد که قرار است مورچه داده خود را در آن قرار دهد. این روش با تنها روش موجود برای تشخیص مکانهای مهم در صفحه برای خوشه بندی که "حافظه ی کوتاه مدت" نام دارد مقایسه شده است. نتایج آزمایشها بر روی ۵ مجموعه داده استاندارد نشان می دهد که روش پیشنهادی با محاسبات کمتر، خوشه بندی بهتری در مقایسه با روش "حافظه ی کوتاه مدت" بدست می آورد. همچنین مقایسه ها برتری روش ارائه شده را نسبت به K-means نشان می دهد.

کلمات کلیدی: خوشه بندی مورچه ای، اتوماتاهای یادگیر، حافظه ی کوتاه مدت

Improving Ant based Clustering Technique using Learning Automata

B. Jafarpour M. R. Meybodi

Soft Computing Laboratory

Computer Engineering and Information Technology Department

Amirkabir University of Technology

Tehran Iran

jafarpour@cic.aut.ac.ir, mmeybodi@aut.ac.ir

Abstract

Ant based clustering is a bio-inspired method for clustering. This method has some interesting characteristic that has prompted special attention from the researchers in the past few years. In this paper an improvement on ant based clustering using learning automata is proposed. In improved ant based clustering every ant is equipped with a learning automaton which learns the place that the ant should drop its data item on the grid. To show the performance of the proposed method, it is tested on five standard data sets and the results are compared with the results obtained for the only reported method in the literature for determining important places in grid which is called "Short Term Memory" and K-means method. The results of comparison have shown the superiority of the proposed method over "Short Term Memory" and K-means methods in term of computational effort and accuracy.

Keywords: Ant based Clustering, Learning Automata, Short Term Memory

۱- مقدمه

روش خوشه بندی مورچه ای که الهام گرفته از رفتار مورچه ها در طبیعت می باشد اولین بار در [1] برای مجموعه ای از ربات ها که اشیاء را دسته بندی می کردند ارائه شد. این الگوریتم در [2] برای خوشه بندی داده ها تصحیح و ارائه شد. در روش خوشه بندی مورچه ای از تعدادی

مورچه با عملکرد ساده و بدون کنترل مرکزی استفاده می شود. در این روش یک شبکه ۲ بعدی از سلولها وجود دارد که داده‌ها و مورچه‌ها بر روی سلولهای آن قرار دارند و مورچه‌ها می‌توانند بر روی آنها حرکت کنند. در این روش بر خلاف بسیاری از روش‌های خوشه بندی احتیاج به از پیش تعیین کردن تعداد خوشه‌ها نمی‌باشد و تعداد آنها بطور خودکار در حین فرایند خوشه بندی تعیین میگردد. یکی دیگر از ویژگی‌های مهم این روش اینست که خوشه بندی در یک فضای ۲ بعدی به جای فضای n بعدی انجام می شود که امکان ارزیابی خوشه بندی توسط انسان را فراهم می کند. یکی از بهبودهایی که در این روش ایجاد شده است، افزودن حافظه کوتاه مدت^۱ به مورچه‌ها برای یادآوری مکانهایی است که قبلاً در آنجا داده‌ای را قرار داده اند [3] و نشان داده شده است که این تغییر می تواند کارایی را در خوشه بندی مورچه‌ای افزایش دهد. این روش دارای کاربردهای زیادی از جمله خوشه بندی مستندات [4][5]، قسمت بندی^۲ گراف [6][7] و قسمت بندی در VLSI [8] و غیره می باشد.

یک اتوماتای یادگیر [13] ماشینی است که میتواند تعدادی متناهی عمل را انجام دهد. هر عمل انتخاب شده توسط یک محیط احتمالی ارزیابی میشود و نتیجه ارزیابی در قالب سیگنالی تقویتی^۳ به اتوماتای یادگیر داده میشود و اتوماتا از این پاسخ در انتخاب عمل بعدی استفاده میکند و بدین ترتیب به سمت انتخاب عملی که بیشترین پاداش را از محیط می گیرد، میل میکند. سادگی و کارایی اتوماتای یادگیر آن را مناسب برای بسیاری کاربردها میکند. برای مثال از این ابزار در آموزش شبکه ی عصبی [9]، محاسبات تکاملی [10][11]، حل مسئله ی کوتاهترین مسیر اتفاقی [12] و غیره استفاده شده است.

در این مقاله از اتوماتاهای یادگیر برای یادگیری مکانهایی که مورچه‌ها برای انداختن داده‌ها بر روی صفحه باید در نظر بگیرند استفاده شده است. در روش پیشنهادی، صفحه ناحیه بندی می شود و هر مورچه دارای یک اتوماتای یادگیر است که تعداد اعمال آن برابر است با تعداد ناحیه‌ها. اتوماتای یادگیر هر مورچه به صورت احتمالاتی ناحیه ی قرار دادن داده‌ها را برای هر مورچه تعیین می کند. مورچه‌ها به صورت دست جمعی و بدون داشتن ارتباط مستقیم با یکدیگر، یاد می گیرند که چه ناحیه‌هایی برای انداختن داده‌ها مناسب تر می باشد. مقایسه ی روش پیشنهادی با تنها روش موجود برای تعیین مکان انداختن داده‌ها نشان می دهد که روش پیشنهادی احتیاج به محاسبات کمتری دارد و خوشه بندی بهتری تولید می کند. ادامه مقاله بدین صورت سازماندهی شده است. در بخشهای ۲ و ۳ به ترتیب خوشه بندی مورچه‌ای و اتوماتاهای یادگیر به اختصار شرح داده میشود. در بخش ۴ روش پیشنهادی و در بخش ۵ نتایج آزمایشها ارائه میگردد. بخش ۶ نتیجه گیری میباشد.

۲- خوشه بندی مورچه‌ای

روش خوشه بندی مورچه‌ای الهام گرفته از رفتار مورچه‌ها در طبیعت می باشد. مورچه‌ها در طبیعت می توانند اشیاء مختلف را بدون وجود کنترل مرکزی در دسته‌های مشابه در کنار یکدیگر جمع آوری کنند. با الهام گرفتن از این مدل، خوشه بندی مورچه‌ای توسعه داده شده است و دارای کاربردهای زیادی در علم می باشد. در این مدل، مورچه‌ها و داده‌هایی که باید خوشه بندی شوند بر روی صفحه‌ای دو بعدی قرار داده می شوند. مورچه‌ها می توانند داده‌هایی را که به داده‌های اطراف خود شبیه نیستند از جای خود در صفحه بر داشته و در کنار داده‌های مشابه قرار دهند. مورچه‌ها برای ارزیابی اینکه داده‌ای به داده‌ی اطراف خود شبیه است یا خیر دارای دید محدودی می باشند. این مورچه‌ها، همانند مورچه‌های واقعی از کنترل مرکزی بر خوردار نیستند و به صورت مستقل از یکدیگر عمل می کنند. در هر سلول شبکه همزمان دو داده یا دو مورچه نمی‌توانند قرار بگیرند و چنانچه مورچه‌ای از یک طرف شبکه بیرون رود از طرف دیگر وارد خواهد شد. مورچه‌ها با احتمالات زیر داده i را از جای خود برداشته یا در مکانی قرار می دهند [2].

$$P_{pick}(i) = \left(\frac{k_{pick}}{k_{pick} + f(i)} \right) \quad (1)$$

$$P_{drop}(i) = \left(\frac{f(i)}{k_{drop} + f(i)} \right) \quad (2)$$

¹ Short Term Memory

² Partitioning

³ Reinforcement Signal

چنانچه داده ی i شبیه داده هایی که در همسایگی اش هستند باشد، $f(i)$ مقادیر بالا خواهد داشت و در نتیجه داده i با احتمال کمی برداشته می شود (فرمول (۱)) و با احتمال بالایی در آن مکان انداخته می شود (فرمول (۲)). تفاوت داده ی i با داده های همسایه اش باعث می شود که $f(i)$ مقادیر پایین داشته باشد و در این حالت احتمال اینکه داده توسط مورچه از جای خود برداشته شود زیاد می شود و احتمال گذاشته شدن در آن مکان برای داده پایین می آید. تابع ارائه شده در [1] امکان مقایسه ی داده های عددی را فراهم نمی کند ولی این مشکل در [2] حل و تابع زیر ارائه شد.

$$f(i) = \max(0, \frac{1}{\sigma^2} \sum_{j \in Neigh(i)} (1 - \frac{d(i, j)}{\alpha})) \quad (3)$$

$d(i, j)$ در محدوده ی $[0, 1]$ قرار دارد و تفاوت داده های i و j را تعیین می کند. α پارامتر مقیاس^۴ است که در محدوده ی $[0, 1]$ قرار دارد و باید با توجه به داده ها تعیین شود. σ محدوده دید را تعیین می کند به این معنی که محدوده ی دید یک مورچه با پارامتر σ در هر ۴ جهت برابر است با $\frac{(\sigma-1)}{2}$. $Neigh(i)$ مجموعه ی داده هایی است که در همسایگی σ داده ی i بر روی صفحه قرار دارند. برای افزایش کارایی الگوریتم "حافظه ی کوتاه مدت" در [2] ارائه شد. در این مدل مورچه ها حافظه ی محدودی دارند که تعدادی از آخرین مکانهایی را که داده ها در آنجا انداخته شده اند در آن ثبت می کنند و هر گاه که قصد انداختن داده ای را داشته باشند به حافظه ی خود رجوع می کنند و به سمت مکانی که شبیه ترین داده به داده ای که حمل می شود در آنجا انداخته شده حرکت می کنند. این مدل دارای مشکلاتی نیز می باشد، از جمله اینکه مورچه باید تمام مسیر را طی کند و زمانی که به مکان مورد نظر رسید ممکن است داده ای که قبلا در آنجا قرار داده است توسط مورچه ی دیگری جابجا شده باشد و طی این مسیر بی فایده باشد. برای حل این مشکل در [3] نسخه ی جدیدی از حافظه ی کوتاه مدت ارائه شد. در این مدل هر مورچه برای انداختن داده به تمام نقاطی در که در حافظه ی خود ثبت کرده می رود و با احتمال P_{drop} داده ی حمل شده را در مکانی که بیشترین f را داشته باشد قرار می دهد. برای افزایش سرعت، مورچه ها به جای طی مسیر تا نقاط مورد نظر به این نقاط پرش می کنند. چنانچه مورچه نتواند به کمک حافظه ی خود، داده ای را در صفحه قرار دهد استفاده از حافظه ی خود را تا انداختن موفق بعدی به حالت تعلیق در آورده و به صورت تصادفی عمل می کند. از جمله دیگر تغییراتی که برای بهبود در [3] ارائه شد پارامتر $step_size$ می باشد. در این مدل مورچه ها به جای حرکت از یک سلول به سلول مجاور می توانند به سلولهایی که مجموع فاصله ی عمودی و افقی آنها حد اکثر $step_size$ باشد پرش کنند. شبه کد الگوریتمی را که در این مقاله مورد بررسی قرار گرفته و برگرفته از [3] می باشد که در شکل ۱ آمده است. در این مدل هر مورچه برای برداشتن داده ها به لیستی که از داده های آزاد (داده هایی که توسط دیگر مورچه ها حمل نمی شوند) وجود دارد مراجعه می کند و یکی از داده ها را به صورت اتفاقی انتخاب می کند و آن را با احتمالی که فرمول (۱) تعیین می کند بر می دارد. این کار تا زمانی ادامه پیدا می کند که مورچه موفق به برداشتن یک داده بشود. پس از آن مورچه در تکرار های بعدی سعی خواهد کرد که داده اش را در مکان مناسبی قرار دهد. در هنگام انداختن داده چنانچه حافظه ی کوتاه مدت مورچه فعال باشد، مورچه به تمام نقاطی که در حافظه اش ثبت شده مراجعه می کند و با احتمالی که فرمول (۲) تعیین می کند داده اش را در مکانی که بیشترین f را تولید می کند قرار می دهد. چنانچه حافظه ی کوتاه مدت مورچه به علت تلاش ناموفق برای انداختن داده غیر فعال باشد، مورچه اقدام به جهش های اتفاقی با حداکثر اندازه ی $step_size$ می کند و از فرمول (۲) برای تصمیم گیری در مورد انداختن داده تصمیم می گیرد. چنانچه مورچه تصمیم به انداختن داده ای کرد و سلول مورد نظر توسط داده ی دیگری اشغال شده بود، مورچه داده را به صورت اتفاقی در یکی از سلول های همسایه می اندازد و چنانچه تمام سلولهای همسایه داده در خود داشتند مورچه از انداختن داده صرف نظر می کند.

Algorithm Ant Clustering
 1: INITIALIZATION PHASE
 2: Randomly scatter data items on the toroidal grid
 3: **for** each j in 1 to #agents **do**
 4: $i :=$ random select(remaining items)
 5: pick up(agent(j), i)
 6: $g :=$ random select(remaining empty grid locations)
 7: place agent(agent(j), g)

⁴ Scaling Parameter

```

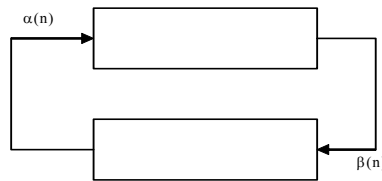
8: end for
9: for each it_ctr in 1 to #iterations do
10: j := random select(all agents)
11: step(agent(j), stepsize)
12: i := carried item(agent(j))
13: drop := drop item?(f(i)) // see equation 2
14: if drop = TRUE then
15:   while pick = FALSE do
16:     i := random select(free data items)
17:     pick := pick item? // see equation 1
18:   end while
19: end if
20: end for

```

الگوریتم خوشه بندی مورچه ای

۳- اتوماتاهای یادگیر

یک اتوماتای یادگیر یک مدل انتزاعی است که تعداد معدودی عمل را می تواند انجام دهد. هر عمل انتخاب شده توسط محیطی احتمالی ارزیابی شده و پاسخی به اتوماتای یادگیر داده می شود. اتوماتای یادگیر از این پاسخ استفاده نموده و حالت درونی خود را به روز می کند و دوباره عمل خود را برای مرحله بعد انتخاب می کند [13]. اتوماتا با تعامل با محیط عمل بهینه را فراگیری می کند و به این شکل پاداش دریافتی خود را از محیط حد اکثر می کند. شکل ارتباط بین اتوماتای یادگیر و محیط را نشان می دهد.



شکل ۱: ارتباط بین اتوماتای یادگیر و محیط

محیط را می توان توسط یک سه تایی $E \equiv \{\alpha, \beta, c\}$ نشان داد که در آن $\alpha \equiv \{\alpha_1, \alpha_2, \dots, \alpha_r\}$ مجموعه ورودیها، $\beta \equiv \{\beta_1, \beta_2, \dots, \beta_m\}$ مجموعه خروجیها و $c \equiv \{c_1, c_2, \dots, c_r\}$ مجموعه احتمالهای جریمه α ها می باشد. هر گاه β مجموعه دو عضوی باشد، محیط از نوع P می باشد. در چنین محیطی $\beta_1 = 1$ به عنوان جریمه و $\beta_2 = 0$ به عنوان پاداش در نظر گرفته می شود. در محیط از نوع Q، $\beta(n)$ می تواند به طور گسسته یک مقدار از مقادیر محدود در فاصله $[0,1]$ و در محیط از نوع S، β متغیر تصادفی در فاصله $[0,1]$ است. c_i احتمال اینکه عمل α_i نتیجه نامطلوب^۵ داشته باشد، می باشد. در محیط ایستا^۶ مقادیر c_i بدون تغییر می مانند، حال آن که در محیط غیر ایستا^۷ این مقادیر در طی زمان تغییر می کنند. اتوماتای یادگیر به دو گروه با ساختار ثابت و با ساختار متغیر تقسیم می گردد [3], [13]. اتوماتای یادگیر با ساختار متغیر^۸ توسط ۴ تایی $\{\alpha, \beta, p, T\}$ نشان داده می شود که در آن $\alpha \equiv \{\alpha_1, \alpha_2, \dots, \alpha_r\}$ مجموعه عملهای اتوماتا، $\beta \equiv \{\beta_1, \beta_2, \dots, \beta_m\}$ مجموعه ورودیهای اتوماتا و $p = \{p_1, p_2, \dots, p_r\}$ بردار احتمال انتخاب هر یک از اعمال و $p(n+1) = T[\alpha(n), \beta(n), p(n)]$ الگوریتم یادگیری می باشد. در این نوع از اتوماتاها، اگر عمل α_i در مرحله n انتخاب شود و این عمل، پاسخ مطلوب از محیط دریافت نماید، احتمال $p_i(n)$ افزایش یافته و سایر احتمالها کاهش می یابند. برای پاسخ نامطلوب احتمال $p_i(n)$ کاهش یافته و سایر احتمالها افزایش می یابند. این تغییرات به گونه ای صورت می پذیرد که جمع احتمالات برابر با یک باقی بماند. فرمولهای (۴) و (۵) یکی از الگوریتمهای یادگیری خطی در اتوماتای با ساختار متغیر را نشان می دهد.

الف- پاسخ مطلوب برای عمل α_i :

$$p_i(n+1) = p_i(n) + a[1 - p_i(n)] \quad (۴)$$

^۵Unfavorable

^۶Stationary

^۷Non-Stationary

^۸ Variable Structure Learning Automata

$$p_j(n+1) = (1-a)p_j(n) \quad \forall j \neq i$$

ب- پاسخ نامطلوب برای عمل i :

$$p_i(n+1) = p_i(n) - (1-b)p_i(n)$$

$$p_j(n+1) = \frac{b}{r-1} + (1-b)p_j(n) \quad \forall j \neq i \quad (\Delta)$$

a و b به ترتیب پارامتر پاداش و پارامتر جریمه می باشد. زمانی که a و b با هم برابر باشند، الگوریتم L_{R-P} ^۹، زمانیکه b از a خیلی کوچکتر باشد، الگوریتم L_{R-EP} ^{۱۰} و زمانی که b مساوی صفر باشد، الگوریتم L_{R-I} ^{۱۱} نامیده می شود.

۴- روش پیشنهادی

همانطور که قبلا در بخش ۳ اشاره شد، "حافظه ی کوتاه مدت" یکی از روشهایی است که برای سرعت بخشیدن به همگرایی و افزایش کیفیت خوشه بندی در خوشه بندی مورچه ای ارایه شده است [2] [3]. هر چند این روش کارایی خوشه بندی مورچه ای را افزایش می دهد ولی دارای معایبی می باشد. اولین عیب این روش اینست که زمانی که مورچه ای از حافظه ی کوتاه مدتش استفاده می کند باید به تعداد خانه های حافظه اش تابع f را محاسبه کند. محاسبه ی f یکی از زمان بر ترین محاسبات این الگوریتم می باشد و فراخوانی مکرر این تابع می تواند محاسبات الگوریتم را افزایش دهد. عیب دیگر روش حافظه ی کوتاه مدت این است که توانایی ترکیب خوشه های کوچک تشکیل شده را ندارد. در ادامه این بخش روشی که بر پایه ی اتوماتای یادگیر میباید برای حل مشکلات ذکر شده پیشنهاد میگردد. در این روش صفحه را به n^2 ناحیه ی مربعی شکل مساوی تقسیم می کنیم و به هر مورچه یک اتوماتای یادگیر اختصاص می دهیم. این اتوماتاهای یادگیر دارای n^2 عمل می باشند که هر کدام از آنها متناظر با یکی از نواحی اختصاص داده شده می باشد. به این شکل هر مورچه برای انتخاب مکانی برای انداختن داده ی خود یک ناحیه را توسط اتوماتای یادگیرش تعیین می کند و نقطه ای را به صورت اتفاقی در مربع کوچکتری که در آن ناحیه در نظر گرفته می شود انتخاب می کند. اضلاع مربع داخلی برابر با $0/8$ ضلع ناحیه می باشد و در وسط آن قرار دارد. شکل یک ناحیه بندی ۱۶ تایی را نشان می دهد. خطوط سیاه ناحیه ها را از یکدیگر جدا می کنند و مربع های خاکستری نشان دهنده ی قسمتی از ناحیه است که برای انداختن داده در نظر گرفته می شود. چنانچه مورچه موفق به انداختن داده در ناحیه ی مورد نظر بشود اتوماتای یادگیر پاداش می گیرد ($\beta = 0$) و در غیر اینصورت جریمه می شود ($\beta = 1$). برای جلوگیری از همگرایی زودرس اتوماتای یادگیر، عمل یادگیری را بعد از گذشت یک پنجم از تعداد تکرار ها شروع می کنیم. برای یادگیری از قانون L_{R-EP} استفاده کرده ایم. دلیل این انتخاب اینست که تعداد تلاشهای موفق برای انداختن داده ها معمولا بسیار کمتر از تعداد تلاش های ناموفق است و اتوماتای یادگیر بیشتر جریمه می شود تا پاداش بگیرد.

4	8	12	16
3	7	11	15
2	6	10	14
1	5	9	13

شکل ۲: تقسیم بندی صفحه به ۱۶ قسمت

بدین ترتیب اتوماتای یادگیر هر مورچه ناحیه هایی را که در آنها داده قرار داده است با احتمال بیشتری مورد بررسی قرار می دهد. به این شکل یک پرتاب ناموفق باعث کنار گذاشتن اطلاعات موجود نخواهد شد (بر خلاف روش حافظه ی کوتاه مدت که یک پرتاب ناموفق باعث غیر فعال شدن حافظه خواهد شد) و تنها احتمال بررسی آن ناحیه کمتر خواهد شد. مزیت دیگر این اینست که ناحیه هایی که در آنها داده های بیشتری قرار دارند، با احتمال بیشتری بررسی می شوند و دیگر نیازی به ارزیابی تعداد زیادی مکان (که در روش حافظه ی کوتاه مدت نگه داری می شوند) برای انداختن داده در صفحه نیست، در نتیجه تعداد محاسبه f کاهش خواهد یافت. تفاوت دیگر اینست که به جای بررسی

^۹Linear Reward Penalty

^{۱۰}Linear Reward Epsilon Penalty

^{۱۱}Linear Reward Inaction

امکان انداختن در یک نقطه ی خاص، امکان انداختن در یک ناحیه بررسی می شود. این ویژگی باعث می شود که داده های یک کلاس که به خوشه های کوچک تقسیم شده با یکدیگر ترکیب شده و خوشه های بهتری تشکیل دهند. به این شکل الگوریتم ارائه شده تعداد محاسبه ی f را کاهش خواهد داد و خوشه بندی بهتری تولید می کند.

۵- نتایج شبیه سازیها

برای ارزیابی خوشه بندی انجام شده توسط روش پیشنهادی (Ant+LA) و روش موجود (Ant+Memory) که از حافظه ی کوتاه مدت استفاده می کند از ۴ معیار که در ادامه به آنها اشاره میشود استفاده خواهد شد.
۱. F-Measure: این معیار دو مفهوم دقت^{۱۲} و به یاد آوری^{۱۳} که از مفاهیم بازبایی اطلاعات هستند، استفاده می کند. این معیار به این شکل تعریف می شود و مقداری بین ۰ و ۱ دارد و مقادیر نزدیک به ۱ این معیار مطلوب هستند.

$$p(i, j) = \frac{n_{ij}}{n_j} \quad r(i, j) = \frac{n_{ij}}{n_i}$$

$$F(i, j) = \frac{(b^2 + 1) \cdot p(i, j) \cdot r(i, j)}{b^2 \cdot p(i, j) + r(i, j)} \quad (6)$$

$$F = \sum_i \frac{n_i}{n} \max_j \{F(i, j)\}$$

در فرمولهای فوق، n_i و n_j و n_{ij} به ترتیب برابراند با تعداد داده های کلاس i و تعداد داده های خوشه ی j (بدست آمده توسط الگوریتم خوشه بندی) و تعداد داده های کلاس i در خوشه ی j . p و r نیز به ترتیب نشان دهنده ی مفاهیم دقت و یادآوری می باشند. در این مقاله از $b=1$ استفاده شده است.

۲. Rand Index: فرض کنید که V آرایه ای باشد که خوشه ی داده ها (بدست آمده توسط الگوریتم خوشه بندی) و U آرایه باشد که کلاس داده ها را نشان می دهد. این معیار به این شکل تعریف می شود و مقداری بین ۰ و ۱ دارد. مقادیر نزدیک به ۱ این معیار مطلوب هستند.

$$a = \{i, j | U(i) = U(j) \wedge V(i) = V(j)\}$$

$$b = \{i, j | U(i) = U(j) \wedge V(i) \neq V(j)\}$$

$$c = \{i, j | U(i) \neq U(j) \wedge V(i) = V(j)\}$$

$$d = \{i, j | U(i) \neq U(j) \wedge V(i) \neq V(j)\} \quad (7)$$

$$R = \frac{a+d}{a+b+c+d}$$

۳. Entropy: این پارامتر اعدادی بین ۰ و ۱ تولید می کند و اعداد نزدیک به صفر مطلوب می باشند.

$$p_{ij} = I(i, j) / n_i$$

$$e_i = - \sum_j p_{ij} \log p_{ij} \quad (8)$$

$$E = \sum_i e_i n_i / n$$

که در آن $I(i, j)$ تعداد داده های مشترک بین خوشه ی i و کلاس j می باشد و n_i و n به ترتیب تعداد داده های خوشه ی i و تعداد کل داده ها می باشد.

۴. تعداد فراخوانی های تابع f (F Evaluation): محاسبه ی تابع f یکی از زمان بر ترین محاسبات این الگوریتم می باشد و هرچه کمتر f محاسبه شود سرعت الگوریتم بیشتر خواهد بود.

۵ داده ی مورد استفاده عبارتند از: **Iris** که دارای ۱۵۰ داده ی ۴ بعدی می باشد و در ۳ کلاس به شکل مساوی تقسیم شده اند، **Wisconsin** که از ۶۸۳ داده ی ۱۰ بعدی تشکیل شده و دارای ۲ کلاس با تعداد داده های ۲۳۹ و ۴۴۴ می باشد، **Ecoli** که از ۳۳۶ داده ی ۷ بعدی تشکیل شده و دارای ۷ کلاس با تعداد داده های ۱۴۳، ۷۹، ۵۲، ۳۵، ۲۰، ۵ و ۲ می باشد، **Data1** که از ۱۰۰۰ داده ی ۲ بعدی تشکیل شده است که به شکل مساوی در ۴ کلاس با توزیع نرمال با میانگین های (۵,۲)، (-۵,۲)، (۵,-۲) و (-۵,-۲) و انحراف معیار ۲ تقسیم شده اند و

¹² Precision

¹³ Recall

Galss که شامل ۲۱۴ داده ی ۹ بعدی است و به ۶ کلاس که دارای ۰.۷۶، ۰.۷۰، ۰.۲۹، ۰.۱۷، ۰.۱۳ و ۰.۰۹ داده می باشند تقسیم شده است. تمام داده ها در تمام ابعاد قبل از استفاده نرمال شده اند. برای اینکه ماتریس d (ر.ک. فرمول (۳)) اعدادی بین ۰ و ۱ داشته باشد، اعداد این ماتریس نرمال شده اند. برای محاسبه ی ماتریس d برای تمام داده ها به غیر از Iris از فاصله ی اقلیدسی استفاده شده است و برای داده ی Iris از فاصله ی کسینوسی استفاده شده است. پارامترهای وابسته به داده ها که در هر دو روش به صورت یکسان مورد استفاده قرار گرفته اند در جدول ذکر شده اند.

جدول ۱: پارامترهای مشترک Ant+(Memory,LA)

	σ	α	تعداد تکرار ها
IRIS	5	0.03	50,000
Wisconsin	7	0.5	350,000
Ecoli	7	0.35	350,000
Data1	7	0.2	400,000
Glass	5	0.4	150,000

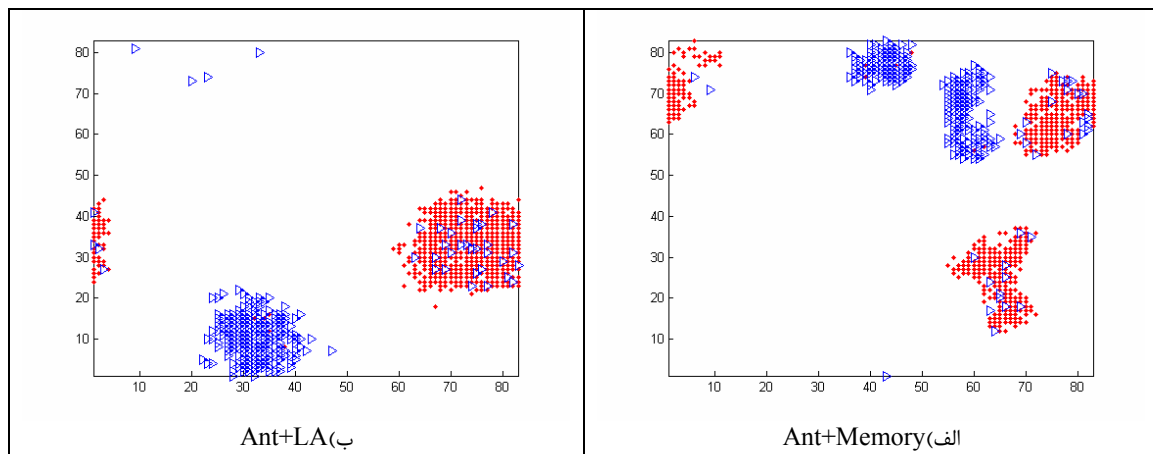
تعداد مورچه ها و k_{pick} و k_{drop} در تمام آزمایش ها به ترتیب برابر با 10، 0.1 و 0.15 قرار داده شده است. صفحه در تمام آزمایش ها مربعی با ضلع $\sqrt{10 \times N_{data}}$ می باشد. $step_size$ نیز برابر با $\sqrt{10 \times N_{data}}$ در نظر گرفته شده است. به این شکل یک مورچه با یک پرش به تمام نقاط صفحه دسترسی دارد. حافظه ی کوتاه مدت Ant+Memory ظرفیت نگه داری ۱۰ مکان انداختن داده را دارد. در روش Ant+LA صفحه را به ۱۶ ناحیه تبدیل کرده ایم، بنابراین اتوماتای یادگیر هر مورچه دارای ۱۶ عمل خواهد بود. Ant+LA از قانون یادگیری $L_{R-EP}(a=0.01, b=0.0001)$ استفاده می کند و جدول میانگین ۴ معیار را برای ۵ داده در ۲۰ تکرار گزارش می کند.

جدول ۲: مقایسه ی خوشه بندی روشهای Ant+LA و Ant+Memory و Kmeans برای ۵ داده و ۴ معیار

	<i>k-means</i>	<i>Ant + Memory</i>	<i>Ant + LA</i>
Wisconsin			
F-Measure	0.9699	0.8714	0.9653
Rand Index	0.9240	0.7274	0.8970
Entropy	0.1054	0.2994	0.1039
F Evaluations	-	440,631.20	375,152.05
IRIS			
F-Measure	0.8478	0.9533	0.9733
Rand Index	0.8627	0.9404	0.9656
Entropy	0.2519	0.1864	0.1051
F Evaluations	-	141,276.30	67,234.60
Data1			
F-Measure	0.9488	0.7567	0.9720
Rand Index	0.9526	0.8310	0.9727
Entropy	0.1787	0.4434	0.1324
F Evaluations	-	538,864.80	434,432.15
Ecoli			
F-Measure	0.6512	0.6230	0.7743
Rand Index	0.7991	0.7746	0.8325
Entropy	0.5156	0.7180	0.6226
F Evaluations	-	405,223.95	357,974.55
Glass			
F-Measure	0.4333	0.3850	0.4972
Rand Index	0.6692	0.6678	0.5743
Entropy	1.0769	1.2043	1.1190
F Evaluations	-	181,114.60	155,982.95

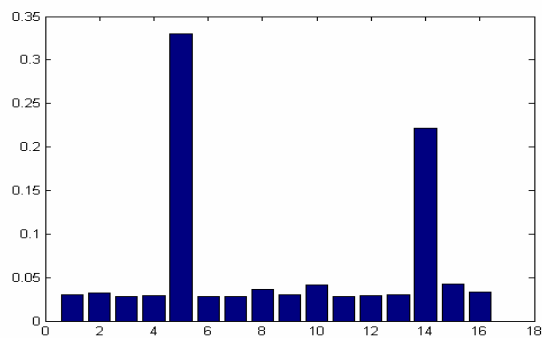
نتایج ارایه شده در جدول فوق نشان می دهد که استفاده از اتوماتای یادگیر برای تعیین کردن مکان انداختن داده بر روی صفحه می تواند کیفیت خوشه بندی را افزایش دهد (با توجه به معیار های F-Measure، Rand Index و Entropy). همچنین معیار تعداد محاسبه ی f نشان می دهد که الگوریتم پیشنهادی احتیاج به محاسبات کمتری دارد و این معیار بین ۱۲ تا ۵۲ درصد نسبت به حافظه ی کوتاه مدت

کاهش می یابد. شکل نتایج خوشه بندی داده ی Wisconsin را برای دو روش Ant+(Memory,LA) نشان می دهد. جدول بالا همچنین کارایی روش ارائه شده را در مقایسه با K-Means نشان می دهد.



شکل ۳: مقایسه ی خوشه بندی تولید شده برای داده ی Wisconsin توسط روش الف) Ant+Memory و ب) Ant+LA

همانطور که در شکل مشاهده میشود روش پیشنهادی توانسته است به خوبی خوشه ها را تشخیص داده و خوشه های کوچکی را که در طول اجرا تولید می شوند با هم ترکیب کند. این شکل همچنین نشان می دهد که Ant+Memory خوشه ها را به شکل صحیح تشخیص نداده است و به جای ۲ خوشه ۴ خوشه تشکیل داده است. شکل نشان می دهد که روش پیشنهادی چگونه توانسته است خوشه بندی صحیح را بدست آورد. در این شکل میانگین بردار احتمالات اتوماتای یادگیر مورچه ها را در روش Ant+LA در انتهای یک اجرا نشان می دهد.



شکل ۴: میانگین بردار احتمالات اتوماتای یادگیر در مورچه ها

همانطور که در شکل مشاهده میشود میانگین بردار احتمالات اتوماتای یادگیر مورچه ها نشان می دهد که آنها با احتمال زیاد دو ناحیه را برای انداختن داده ها استفاده می کنند. چنانچه شکل را با شکل ب- مقایسه کنیم می بینیم که داده ها در ناحیه های ۵ و ۱۴ تجمع کرده اند که نمودار شکل حاکی از این انتخاب توسط مورچه ها می باشد.

۶- نتیجه گیری

در این مقاله صفحه ای که مورچه ها در روش خوشه بندی مورچه ای بر روی آن خوشه بندی می کنند ناحیه بندی کردیم و به هر مورچه یک اتوماتای یادگیر که تعداد اعمال آن برابر با تعداد ناحیه هاست تخصیص دادیم. هدف از انجام این کار، فراهم آوردن مکانیزمی برای مورچه ها بود تا بتوانند ناحیه هایی را در صفحه که داده ها بیشتر در آنجا تجمع دارند، شناسایی کرده و با احتمال بیشتری توسط مورچه ها مورد بررسی بگیرند. آزمایشها نشان دادند که روش پیشنهادی، از تنها روش موجود برای تشخیص مکانهای مهم در صفحه برای خوشه بندی احتیاج به محاسبات کمتری دارد و خوشه بندی بهتری تولید می کند. همچنین مقایسه با روش K-Means برتری روش پیشنهادی را نشان داد.

مراجع

- [1] J. L. Deneubourg, S. Goss, N. Franks, A. Sendova-Franks, C. Detrain, and L. Chrétien, "The dynamics of collective sorting: Robot-like ants and ant-like robots", J.-A. Meyer and S. Wilson, editors, Proceedings of the First International Conference on Simulation of Adaptive Behaviour: From Animals to Animats 1, pp. 356–365. MIT Press, Cambridge, MA, 1991.
- [2] E.D. Lumer, B. Faieta, "Diversity and adaptation in populations of clustering ants" Proc. of the Third International Conference on The Simulation of Adaptive Behavior: From Animals to Animats 3, MIT Press, pp. 449–508, 1994.
- [3] J. Handl, J. Knowles, and M. Dorigo. "Strategies for the increased robustness of ant-based clustering", Lecture Notes in Computer Science, Vol. 2977, pp. 90–104, 2004.
- [4] J. Handl and B. Meyer. "Improved ant-based clustering and sorting in a document retrieval interface", Proceedings of the Seventh International Conference on Parallel Problem Solving from Nature, volume 2439 of LNCS, pp. 913–923. Springer-Verlag, Berlin, Germany, 2002.
- [5] K. Hoe, W. Lai, and T. Tai. "Homogenous ants for web document similarity modeling and categorization", Proceedings of the Third International Workshop on Ant Algorithms (ANTS 2002), volume 2463 of LNCS, pp. 256–261. Springer-Verlag, Berlin, Germany, 2002.
- [6] P. Kuntz and D. Snyers. "New results on an ant-based heuristic for highlighting the organization of large graphs", Proceedings of the 1999 Congress on Evolutionary Computation, pp. 1451–1458. IEEE Press, Piscataway, NJ, 1999.
- [7] P. Kuntz, D. Snyers, and P. Layzell. "A stochastic heuristic for visualizing graph clusters in a bi-dimensional space prior to partitioning", Journal of Heuristics, Vol. 5, No. 3, pp. 327–351, 1998.
- [8] P. Kuntz, P. Layzell and D. Snyers, "A colony of ant-like agents for partitioning in VLSI technology", 4th European Conference on Artificial Life. MIT Press, July 1997.
- [9] M. R. Meybodi, and H. Beigy, "A note on learning automata based schemes for adaptation of BP parameters", Journal of Neurocomputing, Vol. 48, No. 4, pp. 957-974, October 2002.
- [10] R. Rastegar, M. R. Meybodi, and K. Badie, "A New Discrete Binary Particle Swarm Optimization based on Learning Automata", Proceedings of the International Conference on Machine Learning and Applications, ICMLA '04, pp. 456-462, 2004.
- [11] M. Sheybani and M. R. Meybodi, "PSO-LA: A New Model for Optimization", Proceedings of 12th Annual International Computer Society of Iran Computer Conference CSICC2007, Iran, pp. 1162-1169, 2007.
- [12] H. Beigy, and M. R. Meybodi, "Utilizing Distributed Learning Automata to Solve Stochastic Shortest Path Problem", International Journal of Uncertainty, Fuzziness and Knowledge-based Systems, World Scientific Publishing Company, Vol. 14, No. 5, pp. 591-617, 2006
- [13] M. A. L. Thathachar and P. S. Sastry, "Varieties of Learning Automata: An Overview", IEEE Transaction on Systems, Man, and Cybernetics-Part B: Cybernetics, Vol. 32, No. 6, pp. 711-722, 2002.