سیزدهمین کنفرانس ملی انجمن کامپیوتر ایران
جزیره کیش، خلیج فارس، ایران
۱۹ الی ۲۱ اسفند ۱۳۸۶
مقاله کامل
انجمن کامپیوتر ایران
Computer Society of Iran
دانشگاه صنعتی شریف
دانشکده مهندسی کامپیوتر

یک الگوریتم مرتب‌سازی موازی برای اتوماتای سلولی خطی

سید میثم حسینی سدهی
دانشکده مهندسی کامپیوتر و فناوری اطلاعات
دانشگاه صنعتی امیرکبیر
تهران، ایران
sehrestan@Hotmail.com

محمد رضا میبدی
دانشکده مهندسی کامپیوتر و فناوری اطلاعات
دانشگاه صنعتی امیرکبیر
تهران، ایران
mmeybodi@aut.ac.ir

چکیده: مرتب‌سازی یکی از مسائل کاربردی در بسیاری از علوم مهندسی می‌باشد. به همین جهت الگوریتم‌های متنوعی از مرتب‌سازی با دیدگاه‌های متفاوت و پیچیدگی‌های مختلف گزارش شده است. در این مقاله دو الگوریتم مرتب‌سازی جدید برای اتوماتای سلولی خطی پیشنهاد شده است. این دو الگوریتم در مقایسه با تنها الگوریتم گزارش شده برای اتوماتای سلولی خطی از نوع همسایگی، شعاع همسایگی و قوانین متفاوتی برخوردار است، همچنین ساختار اتوماتای سلولی استفاده شده در آن نیازی به حافظه اضافی برای مرتب‌سازی ندارد. الگوریتم‌های معرفی‌شده در مقایسه با تنها الگوریتم گزارش شده از همسایگی کوچکتری برخوردار می‌باشند و نیازمند محاسبات کمتری برای مرتب‌سازی می‌باشند.

واژه‌های کلیدی: الگوریتم‌های مرتب‌سازی، پردازش موازی، اتوماتای سلولی.

۱- مقدمه

اتوماتای سلولی، مدلی مبتنی بر ساختار سلولی بوده که رفتارش بر اساس ارتباط محلی استوار می‌باشد. هر سلول بر اساس اطلاعات محلی، اطلاعات موجود در خود را پردازش می‌کند. قوانین محلی حاکم بر اتوماتای سلولی در هر مرحله، وضعیت جدید هر سلول را با توجه به وضعیت همسایه‌های مجاور سلول بدست می آورد [1]. تنوع و سادگی کارکرد اتوماتای سلولی، امکان استفاده در علوم مختلفی مانند، فیزیک، ریاضی، کامپیوتر و حتی بیولوژیک را فراهم ساخته است. ارتباط محلی از طریق تعامل همسایگی، پردازش محلی اطلاعات از طریق قوانین محلی حاکم، پردازش موازی اطلاعات در تمامی سلول‌ها به صورت همزمان و توزیعی بودن پردازش اطلاعات موجب گسترش اتوماتاهای سلولی در زمینه‌های مختلف کاربردی، شبیه‌سازی و تحقیقاتی شده است. یکی از موارد مورد توجه محققین، حل مسائل پایه مانند مسائل مرتبط به گراف، مرتب‌سازی اعداد و تولید زنجیره اعداد تصادفی توسط اتوماتای سلولی می‌باشد.

مرتب‌سازی داده‌ها یکی از مسائل مورد توجه در کاربردهای متنوعی چون پردازش تصاویر، محاسبات گراف‌ها، مسائل پایگاه داده، کاربردهای اندازه‌گیری و از این دست می‌باشد. می‌توان الگوریتم‌های مرتب‌سازی را به دوسته الگوریتم‌های ترتیبی و الگوریتم‌های موازی تقسیم نمود. الگوریتم‌های موازی از دیدگاه نحوه حل مسئله و موارد مورد استفاده آن به دو دسته الگوریتم‌های بر مبنای مدل‌های ریاضی و الگوریتم‌های وابسته به بستر پیاده‌سازی تقسیم می‌شوند. الگوریتم‌های بر مبنای ماشین‌های SIMD از دسته الگوریتم‌های وابسته به بستر پیاده‌سازی می‌باشند. در این الگوریتم‌ها عموماً آرایه یک بعدی از پردازنده‌ها از طریق الگوریتم انتقال زوج و فرد[1]، n عنصر داده‌ای را با استفاده از n پردازنده مقایسه و مرتب می‌نمایند. دسته‌ای دیگر از الگوریتم‌های وابسته به بستر بر اساس ماشین‌های موازی MIMD و بر مبنای الگوریتم مرتب‌سازی سریع[2] می‌باشند. بسیاری از الگوریتم‌های ترتیبی مانند مرتب‌سازی انتخابی[3]، مرتب‌سازی ترکیبی[4] و یا مرتب‌سازی سریع با راهکارهای متناظر با پردازش موازی، در سیستم‌های موازی مورد استفاده قرار گرفته‌اند [3-9].

دسته‌ای دیگر از الگوریتم‌های مرتب‌سازی، الگوریتم‌های بر مبنای مدل‌های ریاضی مانند اتوماتاهای سلولی[5] و آرایه‌های تپنده[6] [2] بوده که منفک شده از مفهوم بستر پیاده‌سازی می‌باشند. اگرچه الگوریتم‌های بر مبنای مدل‌های ریاضی از دیدگاه‌های مختلف پردازش موازی حاصل شده‌اند، با این وجود بر اساس قوانین حاکم بر محیطشان، قابلیت تبدیل شدن به یکدیگر را دارا می‌باشند.

در الگوریتم‌های مبتنی بر اتوماتای سلولی خطی، محیط حاکم بر مرتب‌سازی، سلول‌های اتوماتای سلولی، همسایگی‌های سلولی و قوانین محلی حاکم بر سلول‌های اتوماتای سلولی می‌باشند. بر روی مرتب‌سازی از دیدگاه اتوماتای سلولی خطی، کار زیادی صورت نگرفته و معروفترین مرتب‌سازی بر پایه اتوماتای سلولی توسط گوردیلو و لونا پیشنهاد شده است[10].

در الگوریتم اول گوردیلو_لونا تعداد همسایگان مورد بررسی هر سلول سه و در الگوریتم دوم گوردیلو_لونا تعداد همسایگان مورد بررسی هر سلول، پنج سلول می‌باشد. برای مرتب‌سازی آرایه‌ها در الگوریتم گوردیلو_لونا هر سلول نیازمند سه حافظه خواندنی و نوشتنی بوده که یکی از آنها در بردارنده مقدار عددی و دو حافظه دیگر شامل کنترل کننده‌های جابجایی مقدار عددی به سمت چپ و راست می‌باشند. این

کنترل‌کننده ها علاوه بر تعیین جابجایی برای عدم تصادم مقادیر سلول‌ها در نظر گرفته شده است. در این الگوریتم‌ها در هر مرحله، محتوای دو سلول اتوماتا با یکدیگر بر اساس قانون حاکم تعویض می‌شوند. همچنین بر اساس ساختار جابجایی این الگوریتم‌ها، در یک مرحله، جابجایی محتوای یک سلول به سمت چپ و راست بررسی شده و در مرحله دوم، محتوای دو سلول همسایه بر اساس بررسی انجام شده و مقدار دو حافظه کنترل کننده جابجایی، جابه‌جا می‌شوند.

در این مقاله دو الگوریتم مرتب‌سازی برای اتوماتای سلولی خطی پیشنهاد شده است. هر سلول در الگوریتم پیشنهادی اول دارای دو همسایه سمت راست بوده و در الگوریتم پیشنهادی دوم تعداد همسایگان هر سلول برابر چهار می‌باشد. الگوریتم دوم با دو نوع همسایگی متقارن و همسایگی نامتقارن پیاده‌سازی شده است. دو الگوریتم پیشنهادی، پیچیدگی زمانی مشابهی با الگوریتم‌های گوردیلو_لونا دارند، ولی از نوع همسایگی، شعاع همسایگی و قوانین متفاوتی برخوردار می‌باشند. همچنین در ساختار استفاده شده برای مرتب‌سازی نیازی به حافظه اضافی برای کنترل جابه‌جایی مقادیر سلول‌های همسایه وجود ندارد. الگوریتم‌های پیشنهادی در مقایسه با الگوریتم‌های گوردیلو_لونا از همسایگی کوچکتری برخوردار می‌باشند. در الگوریتم‌های پشنهادی بر خلاف الگوریتم‌های گوردیلو_لونا، بررسی جابجایی و جابجایی مقادیر سلول‌های همسایه بر اساس قانون حاکم بر اتوماتای سلولی تنها در یک مرحله صورت گرفته و نیازمند دو مرحله مجزا نمی‌باشد. حذف یک مرحله اجرایی موجب افزایش سرعت الگوریتم مرتب‌سازی می‌شود.

ادامه مقاله بدین صورت سازماندهی شده است. در بخش ۲ اتوماتای سلولی به اختصار شرح داده میشود. در بخش ۳ ابتدا الگوریتم مرتب‌سازی گوردیلو_لونا و سپس دو الگوریتم پیشنهادی معرفی می‌شوند. بخش ۴ نتیجه گیری می‌باشد.

۲- اتوماتای سلولی

اتوماتای سلولی مدلی ریاضی است که در آن چندین مؤلفه ساده براساس یک رابطه معین با یکدیگر همکاری نموده و توانایی ایجاد الگوهای پیچیده را دارند. اتوماتای سلولی از یک شبکه منظم سلولی تشکیل شده که در آن هر سلول می‌تواند $r > 1$ مقدار مختلف اختیار نماید. سلول‌های اتوماتای سلولی در زمان‌های گسسته و بطور همزمان و بر طبق یک قانون محلی بنام Φ بهنگام می‌شوند که در آن مقدار هر سلول براساس مقادیر سلول‌های همسایه تعیین می‌شود.

اتوماتای سلولی براساس معیارهای مورد بررسی به دسته‌های مختلف تقسیم می‌گردد. براساس معیار بعد شبکه، اتوماتای سلولی به اتوماتای سلولی یک بعدی، دوبعدی و غیره تقسیم شده و براساس مقدار r به اتوماتای سلولی دودوئی $(r = 1)$ و اتوماتای سلولی چند مقداره تقسیم می‌شود [1]. اتوماتای سلولی d بعدی یک چندتایی $CA = (Z^d, \phi, N, \Phi)$ است به طوریکه:

- Z^d یک شبکه از d-تایی‌های مرتب از اعداد صحیح می‌باشد. این شبکه می‌تواند یک شبکه متناهی، نیمه متناهی یا نامتناهی باشد.
- $\phi = \{1, \cdots, m\}$ یک مجموعه متناهی از حالت‌ها است.
- $N = \{\bar{x}_1, \cdots, \bar{x}_m\}$ ، $\bar{x}_i \in Z^d$ ، یک زیر مجموعه متناهی از Z^d بوده که بردار همسایگی خوانده می‌شود. بردار همسایگی، موقعیت نسبی همسایگان برای هر سلول u در شبکه سلولی توسط رابطه (۱) مشخص می‌شود.

$$N(u) = \{u + \bar{x}_i \mid i = 1,...,\overline{m}\} \qquad (۱)$$

تابع $N(u)$ دو شرط زیر را ارضا می‌کند:

$$\forall u \in Z^d \Rightarrow u \in N(u) \qquad (۲)$$

$$\forall u, v \in Z^d \Rightarrow u \in N(v) \wedge v \in N(u)$$

- $\Phi : \underline{\phi}^{\overline{m}} \to \underline{\phi}$ قانون محلی اتوماتای سلولی بوده که به سه دسته قانون عام[۷]، کلی‌گرا[۸] و کلی‌گرا بیرونی[۹] تقسیم می‌شود.

در اتوماتای سلولی یک بعدی مقدار سلول i (برای $1 \le i \le n$) در زمان t که با $a_i(t)$ نشان داده می‌شود، طبق رابطه (۳) محاسبه می‌گردد:

$$a_i(t+1) = \Phi[a_{i-1}(t), a_i(t), a_{i+1}(t)] \qquad (۳)$$

در رابطه فوق، اگرقانون Φ فقط به مقدار همسایه‌ها بستگی داشته باشد آنرا قانون عام و اگر قانون Φ تابعی از مجموع مقادیر سلول‌های همسایه یک سلول مرکزی باشد آنرا قانون کلی‌گرا می‌نامند و طبق رابطه (۴) بیان می‌شود :

$$a_i(t+1) = \Phi[a_{i-1}(t) + a_i(t) + a_{i+1}(t)] \qquad (۴)$$

در صورتی که قانون تابعی از مجموع مقادیر سلول‌های همسایه و مقدار سلول مرکزی باشد آنرا قانون کلی‌گرای بیرونی می‌گویند و بصورت رابطه (۵) نشان داده می‌شود.

$$a_i(t+1) = \Phi[a_i(t), a_{i-1}(t) + a_{i+1}(t)] \qquad (۵)$$

بر اساس تعاریف فوق اتوماتای سلولی را می‌توان، سیستم محاسباتی توزیع شده‌ای دانست که از عناصر پردازشی همسان بسیار ساده‌ای شکل گرفته که از تعامل بین این عناصر ساده، بروز رفتارهای پیچیده امکان‌پذیر می‌باشد. در این سیستم محاسباتی، قوانین محاسباتی بر اساس همسایگی و به صورت محلی تعریف می‌شوند.

در این مقاله به اتوماتای سلولی از دید یک ماشین محاسباتی عمومی نگریسته شده که برای حل مسئله مرتب‌سازی مورد استفاده قرار گرفته است. برای حل مسائل محاسباتی نیاز به ساختمان داده‌ای شامل ورودی، خروجی و یک رویه برای تبدیل ورودی به خروجی می‌باشد. مراحل پردازش و تبدیل ورودی به خروجی در اتوماتای سلولی توسط انتقال حالت‌های اتوماتای سلولی پیاده سازی می‌شود. هرچند در تعریف اتوماتای سلولی استاندارد حافظه منظور نشده است، ولی اگر

اتوماتای سلولی بخواهد نقش یک ماشین محاسبه‌گر عمومی را بازی کند اولاً هر سلول نیاز به تعدادی حافظه داشته و همچنین بایستی قابلیت نوشتن مقادیر خروجی در حافظه را داشته باشد.

۳- الگوریتم‌های مرتب‌سازی اعداد با اتوماتای سلولی خطی

در این بخش ابتدا الگوریتم مرتب‌سازی گوردیلو_لونا شرح داده خواهد شد، سپس دو الگوریتم پیشنهادی معرفی می‌شوند.

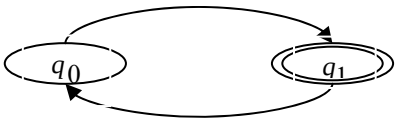
۳-۱- الگوریتم مرتب سازی گوردیلو_لونا

در الگوریتم‌های گوردیلو_لونا با ترکیب اتوماتای سلولی با اتوماتای میلی، ماشین محاسبه‌گری ایجاد شده که هر سلول آن مجهز به سه خانه حافظه بوده و همچنین قابلیت نوشتن مقادیر خروجی در حافظه امکان پذیر می‌باشد.

همسایگی استفاده شده در الگوریتم مرتب‌سازی اول، از نوع متقارن با شعاع یک همانند شکل (۱) می‌باشد. در این الگوریتم، مرتب‌سازی دارای دو مرحله q_0 و q_1 همانند شکل (۲) بوده که در ابتدا تمامی سلول‌های اتوماتای سلولی در مرحله q_0 قرار دارند.

شکل(۱): همسایگی در الگوریتم اول گوردیلو_لونا.

در انتقال از مرحله q_0 به مرحله q_1، مقدار هر سلول با مقدار دو سلول همسایه‌اش مقایسه شده و در صورتی که مقدار سلول مرکزی از مقدار سلول همسایه سمت راستش بزرگتر باشد، آن را به سمت راست خود و در صورتی که مقدارش از مقدار سلول همسایه سمت چپش کوچکتر باشد، آن را به سمت چپ خود انتقال می‌دهد. بنابر این یک سلول می‌تواند، در یک زمان مقدارش را هم با همسایه سمت راست و هم با همسایه سمت چپش جابجا نماید، که در این صورت تصادم رخ می‌دهد.



شکل(۲): مراحل اجرایی الگوریتم گوردیلو_لونا

با اعمال کنترل در جابجایی، از بروز تصادم می‌توان جلوگیری نمود. برای این منظور در هر سلول دو حافظه با نام‌های سد کننده راست و چپ در نظر گرفته می‌شود. با این روش، هر سلولی که مقدارش کوچکتر یا مساوی مقدار سلول همسایه سمت راستش باشد، مقدار سد کننده سمت راست خود را برابر یک و در صورتی که مقدارش کوچکتر از مقدار همسایه سمت چپش و کوچکتر از همسایه راستش باشد، مقدار سد کننده چپ خود را برابر یک قرار می‌دهد. مقادیر سد کننده راست و چپ بر پایه روابط (۶) و (۷) محاسبه می‌شوند.

$$S_i^r = \begin{cases} 1 & x_i > x_{i+1} \\ 0 & Otherswise \end{cases} \qquad (۶)$$

$$S_i^l = \begin{cases} 1 & x_i < x_{i-1} \wedge x_i \le x_{i+1} \\ 0 & Otherswise \end{cases} \qquad (۷)$$

در رابطه فوق S_i^r سد کننده راست و S_i^l سد کننده چپ و x_i بیانگر مقدار سلول مرکزی i-ام می‌باشند.

تصمیم‌گیری برای جابجایی بر اساس مقدار سد کننده چپ و راست انجام می‌شود. اگر سد کننده چپ سلول و سد کننده راست سلول همسایه سمت چپ سلول برابر یک باشد، مقدار همسایه چپ سلول جایگزین مقدار این سلول می‌شود و در صورتی که مقدار سد کننده راست سلول و مقدار سد کننده چپ سلول همسایه سمت راست سلول، برابر یک باشد، مقدار سلول سمت راست جایگزین این سلول شده و در غیر این صورت جابجایی اتفاق نمی‌افتد. عملیات فوق طبق رابطه (۸) صورت می‌پذیرد.

$$x_i = \begin{cases} x_{i+1} & S_i^r = 1 \wedge S_{i+1}^l = 1 \\ x_{i-1} & S_i^l = 1 \wedge S_{i-1}^r = 1 \\ x_i & Otherwise \end{cases} \qquad (۸)$$

اگر روابط (۶) و (۷) در رابطه (۸) جایگزین و کمی ساده‌سازی انجام گیرد، رابطه (۹) بدست می‌آید. بر اساس رابطه (۹) هر سلول برای جابجایی مقدار خود با مقدار سلول همسایه‌اش نیازمند بررسی محتوای خود با سه سلول همسایه خود (یک سلول سمت چپ و دو سلول سمت راست) می‌باشد.

$$x_i = \begin{cases} x_{i+1} & if\,(x_i > x_{i+1} \,\&\, x_{i+1} \le x_{i+2}) \\ x_{i-1} & if\,(x_i \le x_{i-1} \,\&\, x_i \le x_{i-1}) \\ x_i & Otherwise \end{cases} \qquad (۹)$$

با اینکه همسایگی تعریف شده در این الگوریتم شعاع یک متقارن و شامل دو همسایه می‌باشد ولی به دلیل دو مرحله اجرایی q_1 و q_0 برمبنای رابطه (۹)، همسایگی مورد استفاده هر سلول برای مرتب‌سازی، سه سلول همانند شکل (۳) می‌باشد.

شکل(۳): همسایگی واقعی در الگوریتم اول گوردیلو_لونا.

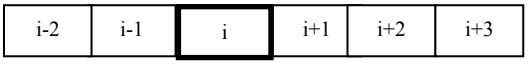
این الگوریتم هنگامی خاتمه می‌یابد که چرخه انتقال q_0 به q_1 و q_1 به q_0، $2n-3$ مرتبه تکرار شود. اگر آرایه‌ای در کمتر از $2n-3$ مرحله مرتب شود، نیاز به یک کنترل بیرونی برای توقف اجرای الگوریتم

می‌باشد. در این صورت می‌توان از حافظه‌های سد کننده برای کنترل توقف استفاده نمود. به عبارتی اگر هیچ حافظه سد کننده‌ای یک نشده باشد، عملیات مرتب‌سازی می‌تواند پایان یابد. با راهکار نظارت بیرونی، مرتب‌سازی در بدترین شرایط پیچیدگی زمانی $2n-3$ خواهد داشت. در جدول (۱)، مرتب‌سازی صعودی اعداد [۱-۱۰] با ترتیب نزولی با الگوریتم اول گوردیلو_لونا نشان داده شده است. پیچیدگی زمانی این الگوریتم در بدترین شرایط متناسب با $\lceil 3n/2-2 \rceil$ می‌باشد.

جدول(۱): مرتب‌سازی اعداد [۱-۱۰]، الگوریتم گوردیلو_لونا.

N.	۱۰	۹	۸	۷	۶	۵	۴	۳	۲	۱
۱	۱۰	۹	۸	۷	۶	۵	۴	۳	۱	۲
۲	۱۰	۹	۸	۷	۶	۵	۴	۱	۳	۲
۳	۱۰	۹	۸	۷	۶	۵	۱	۴	۲	۳
۳	۱۰	۹	۸	۷	۶	۱	۵	۲	۴	۳
۴	۱۰	۹	۸	۷	۱	۶	۲	۵	۳	۴
۴	۱۰	۹	۸	۱	۷	۲	۶	۳	۵	۴
۵	۱۰	۹	۱	۸	۲	۷	۳	۶	۴	۵
5	۱۰	۱	۹	۲	۸	۳	۷	۴	۶	۵
۶	۱	۱۰	۲	۹	۳	۸	۴	۷	۵	۶
۶	۱	۲	۱۰	۳	۹	۴	۸	۵	۷	۶
۷	۱	۲	۳	۱۰	۴	۹	۵	۸	۹	۷
۷	۱	۲	۳	۴	۱۰	۵	۹	۶	۸	۷
۸	۱	۲	۳	۴	۵	۱۰	۶	۹	۷	۸
۸	۱	۲	۳	۴	۵	۶	۱۰	۷	۹	۸
۹	۱	۲	۳	۴	۵	۶	۷	۱۰	۸	۹
۹	۱	۲	۳	۴	۵	۶	۷	۸	۱۰	۹
۱۰	۱	۲	۳	۴	۵	۶	۷	۸	۹	۱۰

اولین جابجایی یک آرایه نزولی در الگوریتم اول گوردیلو_لونا، جابجایی کوچکترین عدد با همسایه خود می‌باشد. در الگوریتم دوم گوردیلو_لونا با بهبود سد کننده‌های چپ و راست، جابجایی بزرگترین عدد با همسایه خود در اولین جابجایی امکان‌پذیر شده است. همسایگی تعریف شده در این الگوریتم با شعاع متقارن یعنی دو سلول سمت راست و دو سلول سمت چپ می‌باشد. با این وجود با توجه به دو وضعیت q_1 و q_0 تعداد سلول همسایه مورد استفاده هر سلول برای مرتب‌سازی، پنج سلول (سه سلول سمت راست و دو سلول سمت چپ) همانند شکل (۴) می‌باشد.



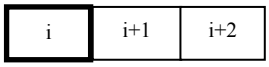
شکل(۴): همسایگی واقعی در الگوریتم دوم گوردیلو_لونا.

۳-۲- الگوریتم‌های مرتب‌سازی حطی پیشنهادی

در این قسمت دو الگوریتم مرتب‌سازی پیشنهادی جدید با استفاده از اتوماتای سلولی خطی معرفی می‌شوند. در این الگوریتم‌ها نحوه جابجایی از راست می‌باشد، به عبارتی جابجایی بر اساس مقایسه هر سلول و سلول‌های سمت راست آن صورت می‌گیرد.

۳-۲-۱- الگوریتم اول مرتب‌سازی با اتوماتای سلولی خطی

در این الگوریتم همسایگی برابر دو و به صورت نامتقارن تعریف شده است. همسایه هر سلول i شامل، اولین همسایه سمت راست $i+1$ (همسایه چپ $i-1$) و دومین همسایه سمت راست $i+2$ (دومین همسایه چپ $i-2$) آن می‌باشد. می‌توان همسایگی را از چپ و یا از راست تعریف نمود که در ادامه همسایگی از سمت راست همانند شکل (۵) در نظر گرفته شده است.



شکل(۵): مرتب‌سازی با همسایگی نامتقارن دو.

بر اساس شکل (۵) هر سلول با دو همسایه سمت راست (چپ) خود در تعامل بوده و جابجایی بین سلول i-ام و سلول i+1-ام، بر اساس محتوای همسایگی اتفاق می‌افتد. در ابتدا هر سلول محتوای خود را با سلول سمت راست خود مقایسه نموده و در نتیجه محتوای این سلول بزرگتر یا کوچکتر از سلول همسایه سمت راست خود خواهد بود. به عبارتی در یک همسایگی، یک مقایسه بین محتوای سلول i و $i+1$ و یک مقایسه بین محتوای سلول $i+1$ و $i+2$ اتفاق می‌افتد. از نتایج این دو مقایسه انتقال و یا عدم انتقال بین محتوای دو سلول i و $i+1$ مشخص می‌گردد.

برای بیان قانون حاکم بر همسایگی، از روش نمایش خاصی شامل دو بیت استفاده می‌نماییم. در این نحوه نمایش، نتایج مقایسات صورت گرفته در هر همسایگی با دو بیت که هر بیت بیانگر یک مقایسه می‌باشد، نشان داده می‌شود. این نحوه نمایش مقایسات به صورت بیت‌های مقایسه‌ای تنها برای نشان دادن قانون حاکم بر همسایگی سلول‌های اتوماتای سلولی در نظر گرفته شده‌اند. برای نمونه اگر محتوای سلول i بزرگتر از محتوای سلول $i+1$ باشد، مقدار یک در غیر این صورت مقدار صفر برای بیت سمت چپ منظور می‌گردد. محتوای بیت سمت راست نیز از مقایسه بین محتوای سلول‌های $i+1$ و $i+2$ بدست می‌آید. مرتب‌سازی در این روش برای سلول i از قوانین ساده محلی زیر طبعیت می‌نماید:

- در بیت‌های مقایسه‌ای به صورت ۰ ۰ و ۰ ۱ یا ۱ ۱ هیچ جابجایی صورت نمی‌گیرد.
- بیت‌های مقایسه‌ای به صورت ۱ ۰ باعث جابجایی محتوای سلول i و سلول $i+1$ می‌شوند.

هرگاه طبق قوانین حاکم بر اتوماتای سلولی، شرط جابجایی سلول مورد بررسی که از قانون بیت‌های مقایسه‌ای طبعیت می‌نماید، ارضاء شود، محتوای سلول با محتوای سلول سمت راست آن تعویض می‌شود.

۳-۲-۲- الگوریتم پیشنهادی اول

مرتب‌سازی درالگوریتم پیشنهادی اول به صورت زیر انجام می‌شود. در ابتدا در هر همسایگی محتوای دو سلول i و $i+1$ و دو سلول $i+1$ و $i+2$ مقایسه می‌شوند (محاسبه بیت‌های مقایسه‌ای). جابجایی بین سلول i و $i+1$ تنها هنگامی رخ می‌دهد که محتوای سلول i بزرگتر از محتوای سلول $i+1$ بوده و محتوای سلول $i+1$ کوچکتر از محتوای سلول $i+2$ باشد؛ در غیر این صورت جابجایی صورت نمی‌پذیرد. جابجایی محتوای دو سلول i و $i+1$ در الگوریتم اول توسط رابطه (۱۰) نشان داده شده‌است.

$$if\ (x_i > x_{i+1}) \wedge (x_{i+1} < x_{i+2}) \quad \{swap(x_i, x_{i+1})\} \qquad (۱۰)$$
$$else\quad \{nothing\}$$

در را بطه فوق محتوای سلول x_i با x_{i+1} و سلول $i+1$ با نمایش داده شده است. تابع swap، برای جابجایی محتوای دو سلول می‌باشد.

در این الگوریتم یک آرایه n عنصری در بدترین شرایط (ترتیب نزولی اعداد) در $2n-3$ مرحله به صورت صعودی مرتب می‌شود. در جدول (۲)، مرتب‌سازی اعداد [۱-۱۰] در بدترین شرایط با الگوریتم پیشنهادی اول نشان داده شده است.

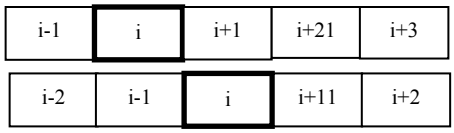
جدول(۲): مرتب‌سازی اعداد با الگوریتم پیشنهادی اول.

N.	۱۰	۹	۸	۷	۶	۵	۴	۳	۲	۱
۱	۱۰	۹	۸	۷	۶	۵	۴	۳	۱	۲
۲	۱۰	۹	۸	۷	۶	۵	۴	۱	۳	۲
۳	۱۰	۹	۸	۷	۶	۵	۱	۴	۲	۳
۴	۱۰	۹	۸	۷	۶	۱	۵	۲	۴	۳
۵	۱۰	۹	۸	۷	۱	۶	۲	۵	۳	۴
۶	۱۰	۹	۸	۱	۷	۲	۶	۳	۵	۴
۷	۱۰	۹	۱	۸	۲	۷	۳	۶	۴	۵
۸	۱۰	۱	۹	۲	۸	۳	۷	۴	۶	5
۹	۱	۱۰	۲	۹	۳	۸	۴	۷	۵	۶
۱۰	۱	۲	۱۰	۳	۹	۴	۸	۵	۷	۶
۱۱	۱	۲	۳	۱۰	۴	۹	۵	۸	۹	۷
۱۲	۱	۲	۳	۴	۱۰	۵	۹	۶	۸	۷
۱۳	۱	۲	۳	۴	۵	۱۰	۶	۹	۷	۸
۱۴	۱	۲	۳	۴	۵	۶	۱۰	۷	۹	۸
۱۵	۱	۲	۳	۴	۵	۶	۷	۱۰	۸	۹
۱۶	۱	۲	۳	۴	۵	۶	۷	۸	۱۰	۹
۱۷	۱	۲	۳	۴	۵	۶	۷	۸	۹	۱۰

اگر بخواهیم تعداد مراحل مرتب‌سازی الگوریتم در بدترین و بهترین شرایط یکسان نباشد و بر اساس عناصر موجود در آرایه تغییر نماید، نیازمند یک ناظر بیرونی برای خاتمه دادن به اجرای الگوریتم می‌باشیم. هرگاه جابجایی بین هیچ سلولی رخ ندهد، آرایه مرتب شده و الگوریتم می‌تواند خاتمه یابد.

۳-۲-۳- الگوریتم مرتب‌سازی خطی پیشنهادی دوم

الگوریتم پیشنهادی دوم با گسترش شعاع همسایگی الگوریتم پیشنهادی اول ایجاد شده است. این الگوریتم با دو نوع همسایگی مختلف پیاده‌سازی شده است. در پیاده‌سازی اول همسایگی با شعاع دو و به صورت متقارن می‌باشد. به عبارتی برای هر سلول i، همسایگی شامل دو همسایه راست $i+1$ و $i+2$ و دو همسایه چپ $i-1$ و $i-2$ می‌باشد. در پیاده‌سازی دوم همسایگی با شعاع دو و به صورت نامتقارن می‌باشد. به عبارتی برای هر سلول i، همسایگی شامل سه همسایه راست $i+1$، $i+2$ و $i+3$ و یک همسایه چپ $i-1$ تعریف شده است. همسایگی‌های این روش در شکل (۶) نشان داده شده‌اند.



شکل(۶): مرتب‌سازی با شعاع همسایگی دو متقارن و نامتقارن.

در پیاده‌سازی با همسایگی متقارن، هر سلول با دو همسایه سمت راست و دو همسایه سمت چپ خود در تعامل بوده و بر اساس محتوای آنها جابجایی بین سلول i-ام و سلول i+1-ام اتفاق می‌افتد. همچنین در پیاده‌سازی با همسایگی نامتقارن هر سلول با سه همسایه سمت راست و یک همسایه سمت چپ خود در تعامل بوده و بر اساس محتوای آنها جابجایی بین سلول i-ام و سلول i+1-ام اتفاق می‌افتد. هر سلول محتوای خود را تنها با سلول سمت راست خود مقایسه نموده و در نتیجه محتوای سلول مذکور یا بزرگتر از سلول همسایه سمت راست خود خواهد بود و یا کوچکتر از آن. بر اساس این مقایسات امکان جابجایی محتوای دو سلول i-ام و سلول i+1-ام بررسی می‌شود. مرتب‌سازی در این روش برای سلول i در همسایگی متقارن از قوانین ساده محلی زیر طبعیت می‌نماید:

- در بیت‌های مقایسه‌ای به صورت | x | x | . | x |، | x | ۱ | ۱ | ۱ | و | ۰ | ۱ | ۱ | ۰ | هیچ جابجایی صورت نمی‌گیرد.

- بیت‌های مقایسه‌ای به صورت [ ][ ][ ][ ][0] و [ ][ ][1][1][0] باعث جابجایی محتوای سلول i و سلول $i+1$ می‌شوند.

در قوانین فوق x به معنای مهم نبودن محتوای مقایسه می‌باشد. هرگاه طبق قوانین فوق، شرط جابجایی سلول مورد بررسی که از قانون بیت‌های مقایسه‌ای طبعیت می‌نماید، ارضاء شود، محتوای سلول مرکزی با محتوای سلول سمت راست آن تعویض می‌شود.

۳-۲-۴- مراحل اجرای الگوریتم پیشنهادی دوم

مرتب‌سازی درالگوریتم پیشنهادی دوم به صورت زیر انجام می‌شود. در ابتدا در هر همسایگی محتوای هر دو سلول مجاور با یکدیگر مقایسه می‌شود (محاسبه بیت‌های مقایسه‌ای). جابجایی بین سلول i و $i+1$ طبق رابطه (۱۱) برای همسایگی متقارن، و رابطه (۱۲) برای همسایگی نامتقارن تغییر می‌نماید.

$$\begin{aligned} &if\ ((x_i > x_{i+1}) \wedge (x_{i-1} < x_i)) \vee \\ &\quad ((x_{i-2} > x_{i-1}) \wedge (x_{i-1} > x_i) \wedge \\ &\quad (x_i > x_{i+1}) \wedge (x_{i+1} < x_{i+2})) \\ &\qquad \{swap(x_i, x_{i+1})\} \\ &else\ \ \{nothing\} \end{aligned} \qquad (۱۱)$$

$$\begin{aligned} &if\ ((x_i > x_{i+1}) \wedge (x_{i+1} < x_{i+2})) \vee \\ &\quad ((x_i > x_{i-1}) \wedge (x_i > x_{i+1}) \wedge \\ &\quad (x_{i+1} > x_{i+2}) \wedge (x_{i+2} < x_{i+3})) \\ &\qquad \{swap(x_i, x_{i+1})\} \\ &else\ \ \{nothing\} \end{aligned} \qquad (۱۲)$$

در روابط فوق محتوای سلول i با x_i و سلول $i+1$ با x_{i+1} نمایش داده شده است. تابع swap، برای جابجایی محتوای دو سلول i و $i+1$ می‌باشد.

اگر بخواهیم تعداد مراحل مرتب‌سازی الگوریتم در بدترین و بهترین شرایط یکسان نباشد و بر اساس عناصر موجود در آرایه تغییر نماید، همانند الگوریتم پیشنهادی اول نیازمند افزودن نظارت بیرونی می‌باشیم. این نظارت بیرونی می‌تواند شامل یک حافظه مشترک برای تمام سلول‌ها باشد؛ به این نحو که هر سلول اگر محتوایش با سلول کناریش جابجا شود، در این حافظه مقدار یک قرار دهد.

در این الگوریتم یک آرایه n عنصری در بدترین شرایط (ترتیب نزولی اعداد) در $\lceil 3n/2-2 \rceil$ مرحله به ترتیب صعودی مرتب می‌شود. در جدول (۳)، مرتب‌سازی اعداد [۱-۱۰] در بدترین شرایط با الگوریتم پیشنهادی دوم نشان داده شده است.

جدول(۳): مرتب‌سازی اعداد با الگوریتم پیشنهادی دوم، اتوماتای سلولی با شعاع همسایگی دو متقارن/ نامتقارن (چهار همسایه)

N.	۱۰	۹	۸	۷	۶	۵	۴	۳	۲	۱
۱	۹	۱۰	۸	۷	۶	۵	۴	۳	۱	۲
۲	۹	۸	۱۰	۷	۶	۵	۴	۱	۳	۲
۳	۸	۹	۷	۱۰	۶	۵	۱	۴	۲	۳
۳	۸	۷	۹	۶	۱۰	۱	۵	۲	۴	۳
۴	۷	۸	۶	۹	۱	۱۰	۲	۵	۳	۴
۴	۷	۶	۸	۱	۹	۲	۱۰	۳	۵	۴
۵	۶	۷	۱	۸	۲	۹	۳	۱۰	۴	۵
۵	۶	۱	۷	۲	۸	۳	۹	۴	۱۰	۵
۱۰	۱	۶	۲	۷	۳	۸	۴	۹	۵	۶
۱۰	۱	۲	۶	۳	۷	۴	۸	۵	۹	۷
۱۰	۱	۲	۳	۶	۴	۷	۵	۸	۹	۸
۱۰	۱	۲	۳	۴	۶	۵	۷	۸	۹	۱۲
۱۰	۱	۲	۳	۴	۵	۶	۷	۸	۹	۱۳

تعداد مراحل مرتب‌سازی الگوریتم پیشنهادی دوم با هر دو پیاده‌سازی (همسایگی متقارن و نامتقارن) در بدترین شرایط یکسان می‌باشد، ولی در حالت کلی تعداد مراحل مرتب‌سازی اعداد آرایه‌هایی که در بدترین شرایط نیستند، با این دو پیاده‌سازی یکسان نمی‌باشد. به عبارتی تعداد مراحل اجرایی این دو پیاده‌سازی در آرایه‌های مختلف متفاوت می‌باشد.

۴- نتیجه

در این مقاله دو الگوریتم مرتب‌سازی برای اتوماتای سلولی خطی پیشنهاد گردید. تعداد مراحل مورد نیاز برای مرتب‌سازی در الگوریتم‌های پیشنهادی حدوداً به نصف تعداد مراحل مورد نیاز الگوریتم گوردیلو_اونا که تنها الگوریتم مرتب سازی گزارش شده برای اتوماتای سلولی خطی می‌باشد می‌رسد؛ چراکه الگوریتم گوردیلو_اونا در هر تکرار نیازمند دو مرحله اجرایی بوده ولی الگوریتم‌های پیشنهادی نیازمند یک مرحله اجرا می‌باشند. شعاع همسایگی هر دو الگوریتم پیشنهادی در مقایسه با شعاع همسایگی الگوریتم گوردیلو-لونا کمتر و بهمین دلیل محاسبات مورد نیاز برای جابجایی مقدار دو سلول کمتر می‌باشد. همچنین قوانین استفاده شده در الگوریتم‌های پیشنهادی بمراتب ساده‌تر از قوانین معرفی شده توسط الگوریتم‌های گوردیلو-لونیا می‌باشد.

سپاسگزاری

این کار تحقیقاتی توسط مرکز تحقیقات مخابرات ایران حمایت مالی شده است که بدین وسیله سپاسگزاری می‌شود.

مراجع

[1] Wolfram, S., *"Cellular Automata and Complexity"*, Perseus Books Group, 1994.
[2] Megson, G.M., *"An Introduction to Systolic Algorithm Design"*, Clara don press Oxford, 1992.
[3] Knuth, D.E., *"The Art of Computer Programming: Sorting and Searching"*, Addison Wesley, 1973.
[4] Akl, S. G., *"Parallel Sorting Algorithms*, Orlando", FL: Academic, 1985.
[5] Toffoli, T., Margolus, N., *"Cellular Automata Machines: A New Environment for Modeling"*, MIT Press Series in Scientific Computation, 1987.
[6] Chen, C.Y.R., Hou, C. Y. and Singh, U., *"Optimal Algorithms for Bubble Sort Based Non-Manhattan Channel Routing"*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 13, No. 5, pp. 603 – 609, May 1994.
[7] Thompson, D., Kung, H.T., *"Sorting on a Mesh Connected Parallel Computer",* Communication of ACM, Vol. 20, pp. 263-271, 1977.
[8] Batcher, K.E., *"Sorting Network and Their Applications"*, AFIP Proc, Vol. 32, pp. 307-314, 1968.
[9] Kummar, M., Hirschberg, D.S., *"An Efficient Implementation of Batcher's Odd-Even Merge Algorithms and its Application to Parallel Sorting Schemes",* IEEE Transaction on Comput C-32, pp. 254-264, 1983.
[10] Gordillo, L. and Luna, V., *"Parallel Sort on a Linear Array of Cellular Automata",* IEEE Trans. Computer Vol. 2, pp. 1904-1910, 1994.
[11] Sarkar, P., *"Brief History of Cellular Automata"*, ACM Computing Survey, Vol. 32, No. 1, 2000.

زیرنویس‌ها

[1] Odd-Even Transposition Sort
[2] Quick Sort
[3] Selection Sort
[4] Merge Sort
[5] Cellular Automata
[6] Systolic Array
[7] General
[8] Totalistic
[9] Outer Totalistic