سنجش از دور
و
GIS ایران

سنجش از دور و GIS ایران
Iranian Remote Sensing & GIS

سال دهم، شماره دوم، تابستان ۱۳۹۷
Vol.10, No. 2, Summer 2018

۷۵-۹۴

بهبود ترمیم خودکار چندضلعی‌های مسطح در سیستم‌های اطلاعات جغرافیایی

کیوان برنا*[۱] و فرهاد فتحی[۲]

۱. استادیار دانشگاه خوارزمی، دانشکدهٔ علوم ریاضی و کامپیوتر، گروه علوم کامپیوتر، تهران

۲. کارشناس ارشد مهندسی کامپیوتر (نرم‌افزار)، مؤسسهٔ آموزش عالی علمی- کاربردی جهاد کشاورزی، سازمان تحقیقات، آموزش و ترویج کشاورزی، تهران

تاریخ دریافت مقاله: ۱۳۹۶/۳/۱۶ تاریخ پذیرش مقاله: ۱۳۹۷/۳/۱۲

چکیده

ترمیم چندضلعی‌های نادرست، برای استفاده در نرم‌افزارهای سیستم اطلاعات جغرافیایی، کاری نیمه‌خودکار و وقت‌گیر است. ترمیم خودکار چندضلعی را تفسیر چندضلعی‌های مبهم و نادرست و ازبین‌بردن تمامی اشکالات، براساس تعریف‌ها و استانداردهای جهانی، می‌شمرند که کاربردهای بسیاری در نرم‌افزارهای مرتبط با سیستم اطلاعات جغرافیایی دارد. با توجه به پیچیدگی محاسبات و حجم داده‌ها در کار با مه‌داده‌ها، همیشه رقابتی بین سرعت و میزان حافظهٔ مصرفی دیده می‌شود. در این مقاله، ضمن معرفی استاندارد مشخصات عوارض ساده در چندضلعی‌ها، با استفاده از مثلث‌بندی دلونی و توابع جی.تی.اس. در جاوا و با کمک بانک اطلاعات H2، روشی پیشنهاد شده است که چندضلعی‌ها را در قالب فایل CSV از ورودی دریافت می‌کند و می‌توان با اعمال الگوریتم‌های کارآ، چندضلعی‌های موجود را بررسی و به‌شکل خودکار ترمیم کرد. چندضلعی‌های موجود در مجموعه داده‌های مکانی، به‌شکل خودکار و در زمانی بهینه و با مصرف حداقل حافظه، بررسی و در صورت نیاز، ترمیم می‌شوند. نتایج نشان می‌دهد این روش، در مقایسه با نمونه‌های پیشین، باعث بهبود نسبی در سرعت اجرا می‌شود و هنگام کار با داده‌های بزرگ، میانگین صرفه‌جویی بیش از ۵۰٪ در حافظهٔ اصلی را نشان می‌دهد.

کلیدواژه‌ها: مثلث‌بندی دلونی، ترمیم چندضلعی، بانک اطلاعات مکانی، سیستم اطلاعات جغرافیایی.

* نویسنده عهده‌دار مکاتبات: کرج، حصارک، دانشگاه خوارزمی، دانشکدهٔ علوم ریاضی و کامپیوتر، گروه علوم کامپیوتر. تلفن: ۰۲۶۳۴۵۷۹۶۰۰، فکس: ۰۲۶۳۴۵۶۹۵۵۸
Email: borna@khu.ac.ir

۱- مقدمه

ترمیم چندضلعی شامل حذف و دوباره‌سازی چندضلعی‌ها می‌شود که کاری نیمه‌خودکار و وقت‌گیر است. همچنین، رفتار نرم‌افزارهای گوناگون نیز، در مواجهه با این اشکالات، با یکدیگر متفاوت است. منابع خطا در داده‌های مکانی ممکن است ناشی از خطای فتوگرامتری[1]، نقشهٔ دیجیتالی، کارهای تحقیقاتی یا سایر منابع باشد. ترکیب خطاهای آغازین با هم می‌تواند خطاهای پیچیده‌تری را تشکیل دهد. برای نمونه، ترکیب مشکل هم‌پوشانی نقشه‌ها در جی.آی.اس. و خطاهای فردی باعث ناهم‌خوانی بین سطح واقعی و نقشه می‌شود (Bolstad and Smith, 1992). پرسش اینجاست چرا فروشندگان گوناگون نرم‌افزارهای جغرافیایی نمی‌توانند چندضلعی‌ها را به‌طور یکسان اجرا کنند؟ دو مشکل وجود دارد که اساساً باعث اختلاف اجرا در سیستم‌های متفاوت می‌شود (Oosterom et al., 2005).

۱. آیا مرز خارجی اجازهٔ تعامل با خود و امکان تعامل با مرز داخلی را دارد؟ اگر بله؛ در چه شرایطی؟

۲. به‌علت محدودیت‌های رایانه، مختصات واقعی ممکن است گاهی، به‌دلیل استفاده از نوع متغیرها، کمی از مقادیر ریاضی فاصله داشته باشد.

ازآنجاکه اغلب، چندضلعی‌ها با روش‌های متفاوتی جمع‌آوری می‌شوند؛ در زمان ایجاد و ویرایش و تبدیل داده‌ها به قالب‌های متنوع، اشکالات گوناگونی رخ می‌دهد. این وضعیت برای الگوریتم‌ها در نرم‌افزارهای متفاوت جی.آی.اس. مشکلات چندی ایجاد می‌کند که معمولاً، بدون اینکه اخطار خاصی به کاربر بدهند، نتایج نادرستی عرضه می‌کنند. همچنین، حل این مشکلات در نرم‌افزارهای گوناگون به‌صورت خودکار و یک‌باره انجام نمی‌شود و کاری مشکل و زمان‌بر است. برای نمونه، در اصلاح چندضلعی پاپیون، در نرم‌افزار PostGIS، نیاز به استفاده از سه دستور (ST_ExteriorRing+ST_Union+ST_BuildingArea) است و یا اسکریپت CleanGeometry.sql[2]، در این نرم‌افزار، بر روی چندضلعی‌هایی که دارای حلقه‌های داخلی هستند به‌درستی کار نمی‌کند (Ledoux et al., 2014).

در این مقاله، با استفاده از مثلث‌بندی دلونی و با کمک توابع جی.تی.اس.، روشی را در جاوا اعمال کردیم که به‌صورت تمام‌خودکار و براساس استانداردهای مشخصات عوارض سادهٔ کنسرسیوم مکانی باز (OGC)[3] و استانداردهای ایزو، با صرف زمان بهینه و کمینه‌سازی در مصرف حافظه، چندضلعی‌های مجموعهٔ دادهٔ ورودی را بررسی و ترمیم می‌کند. در آزمایش‌ها، از مجموعه داده‌های واقعی پایگاه کورین[4] استفاده کردیم. پایگاه دادهٔ کورین[5] دارای مجموعه داده‌های حجیم، در مقیاس تعداد نقاط و چندضلعی‌ها، است که آژانس محیط‌زیست اروپا آن را تهیه کرده است. اجرای الگوریتم‌های پیشنهادی منجر به تولید نرم‌افزار Geometry Repair شد. نتایج نشان می‌دهد این روش، در مقایسه با موارد مشابه پیشین، باعث بهبود نسبی در سرعت اجرا می‌شود و استفاده از پایگاه داده، در زمان کار با داده‌های بزرگ، نیز مصرف حافظهٔ اصلی را بیش از ۵۰٪ کاهش می‌دهد.

ساختار این مقاله از هفت بخش تشکیل شده است؛ بخش اول، مقدمه، شامل بیان مشکل و مروری بر کارهای پیشین و بیان کلی روش و نتایج روش پیشنهادی است. در بخش دوم، ویژگی‌های چندضلعی‌ها و چندضلعی‌های چندگانه[6] براساس استانداردهای معتبر، با استفاده از اشکال متنوع، مطرح شده است که اساس کار این نرم‌افزار را، برای تشخیص درستی چندضلعی و ترمیم آنها، تشکیل می‌دهد. بخش سوم مقاله مروری بر تحقیقات انجام‌شده در این زمینه دارد. در بخش چهارم، الگوریتم‌های گوناگون، به‌همراه محاسبهٔ پیچیدگی محاسباتی هریک، بیان شده است

1. photogrammetry
2. https://trac.osgeo.org/postgis/wiki/UsersWikiCleanPolygons
3. Open Geospatial Consortium
4. http://www.eea.europa.eu/data-and-maps/data/clc-2006-vector-data-version-3
5. Corine
6. multi-polygon

که عبارت‌اند از الگوریتم Repair_Multipolygon برای ترمیم چندضلعی‌های چندگانه؛ الگوریتم Repair Self-intersection برای ترمیم مشکل خودتقاطعی در چندضلعی‌ها؛ الگوریتم Repair-By-Triangulation برای رفع مشکل چندضلعی براساس روش مثلث‌بندی دلونی؛ الگوریتم DRUD[1] برای رفع مشکل چندضلعی‌ها، با کمک پایگاه داده. در بخش پنجم، پیچیدگی محاسباتی روش پیشنهادی آمده و در بخش ششم، ضمن معرفی نرم‌افزار اجرایی به نام Geometry Repair و روش کار آن، کارآیی روش پیشنهادی و مقایسهٔ آن با دیگر روش‌ها و تأثیر استفاده از پایگاه داده در مجموعه داده‌ها، به‌شکل نموداری، بررسی شده است. در پایان، در بخش هفتم، نتیجه‌گیری آمده و کارهای آتی پیشنهاد شده است.

۱-۱- ویژگی‌های چندضلعی و چندضلعی چندگانهٔ صحیح در جی.آی.اس.

هر چندضلعی از مجموعه‌ای از بخش‌های محدود تشکیل شده است که هر بخش آن، حداکثر در دو یال، اشتراک دارد و هیچ زیرمجموعه‌ای از یال‌ها ویژگی‌های یکسان ندارد (Preparata & Shamos, 1985). سازمان استاندارد جهانی (ایزو) ۱۹۱۰۷ چندضلعی ساده را این‌گونه تعریف می‌کند: «چندضلعی سطحی است که با مجموعه‌ای از منحنی‌های مرزی و یک سطح زیرین تعریف می‌شود که این منحنی‌ها را به هم می‌چسباند. به‌طور پیش‌فرض، منحنی‌ها هم‌سطح‌اند و چندضلعی از درون‌یابی مسطح در داخل خود استفاده می‌کند» (ISO, 2003). این تعریف چندضلعی چندان ساده به‌نظر نمی‌رسد. همچنین، مشخص نیست که مرز خارجی می‌تواند خود را قطع کند یا، اگر بتواند مرز داخلی را قطع کند، چه شرایطی باید برقرار باشد. در این تعریف، یک مطلب واضح است: فقط یک مرز خارجی و صفر وجود دارد یا بیشتر مرزها داخلی‌اند. این بدان معناست که چندضلعی دارای دو مرز خارجی، بالقوه، می‌تواند دو ناحیهٔ جدا از هم باشد و مطمئناً نادرست است. استاندارد ایزو دربارهٔ مرزهای داخلی و خارجی بسیار صریح است. چندضلعی‌های دوبعدی دارای مرزهای خارجی در جهت خلاف عقربه‌های ساعت‌اند[2] و مرزهای داخلی در جهت عقربه‌های ساعت[3] نشان داده می‌شوند.

OGC، براساس تعریف ایزو، چندضلعی را بدین‌صورت تعریف کرد: «چندضلعی سطح مسطحی است که با یک مرز خارجی و صفر، یا بیشتر مرز داخلی، تعریف می‌شود. هر مرز داخلی یک حفره را در چندضلعی تعریف می‌کند». بر این اساس، این قوانین برای چندضلعی‌های صحیح تعریف می‌شود (OGC,۱۹۹۹):

چندضلعی‌ها به‌شکل توپولوژیکی بسته‌اند؛ مرز آنها متشکل از مجموعه‌ای از حلقه‌های خطی است که مرزهای داخلی و خارجی را تشکیل می‌دهند؛ مرزهای هیچ دوحلقه‌ای متقاطع نیستند. حلقه‌ها، در مرزهای چندضلعی، ممکن است در یک نقطه مماس باشند؛ چندضلعی نباید شامل خط جدا، بیرون‌زدگی یا تورفتگی[4] باشد؛ فضای داخلی هر چندضلعی مجموعه‌ای از نقاط پیوسته است؛ فضای خارجی چندضلعی دارای یک حفرهٔ، یا بیشتر، متصل نیست. هر حفره به‌صورت اجزای خارجی تعریف می‌شود.

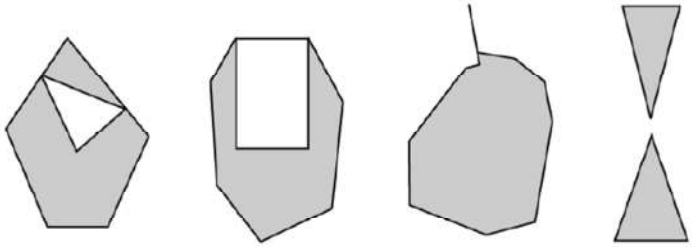
این قوانین مرز داخلی، خارجی و بستار[5] را در ساختار توپولوژیکی استاندارد تعریف می‌کنند و پایهٔ کار ما در این تحقیق به‌شمار می‌روند. بر این اساس، حلقه‌ها ممکن است، درنهایت، در یک نقطه با هم مماس باشند و درون چندضلعی باید یک مجموعهٔ متصل باشد؛ خطوط جدا و لبه‌ها نیز نباید وجود داشته باشند. شکل ۱ نمونه‌هایی نشان می‌دهد که قوانین بالا را نقض می‌کنند و نمی‌توان آنها را به‌شکل چندضلعی ساده نشان داد. چندضلعی‌ای که با خطوط مستقیم در قالب حلقه‌ها تعریف شده شامل، دست‌کم، یک مرز خارجی (خلاف جهت عقربه‌های ساعت) و مرز داخلی صفر یا بیشتر (در جهت عقربه‌های ساعت) است (OGC, 2011).

1. Data Base Repair by Using Database
2. counter clock wise (CCW)
3. clock wise (CW)
4. cut line, spikeor puncture
5. clusure

طبق استاندارد مشخصات عوارض سادهٔ OGC[1]، بخش داخلی چندضلعی باید بسته و همبند باشد و هر حفره می‌تواند حلقهٔ خارجی خود را، فقط در یک نقطه، لمس کند. استاندارد ایزو بیان می‌کند که چرخش حلقه‌های خارجی باید خلاف جهت عقربه‌های ساعت و چرخش مرزهای داخلی در جهت عقربه‌های ساعت باشد. اوستروم و همکاران[2] (۲۰۰۵) این مشکلات را، در مقالهٔ خود و در ۳۷ حالت متفاوت، نشان داده‌اند و تناقض آنها را با مشخصات عوارض سادهٔ OGC بیان کرده‌اند.

چندضلعی‌های چندگانه سطوح گوناگونی دارند که اجزای هر سطح آنها از چندضلعی‌هایی تشکیل می‌شود و براساس استانداردهای OGC باید دارای این ویژگی‌ها باشند (.Ibid): دو چندضلعی داخلی، که عضو یک چندضلعی چندگانه‌اند، نباید متقاطع باشند؛ مرزهای دو چندضلعی، که عضو یک چندضلعی چندگانه‌اند، نباید متقاطع باشند، فقط می‌توانند در چند نقطهٔ محدود یکدیگر را لمس کنند؛ چندضلعی چندگانه به‌صورت توپولوژیکی بسته است؛ درون چندضلعی چندگانه‌ای که در آن بیش از یک چندضلعی قرار دارد متصل نیست؛ همان‌گونه که در شکل ۲ نشان داده شده است، چندضلعی چندگانه نباید شامل خط جدا، بیرون‌زدگی یا تورفتگی باشد. این نوع چندضلعی مجموعه‌ای بسته است.

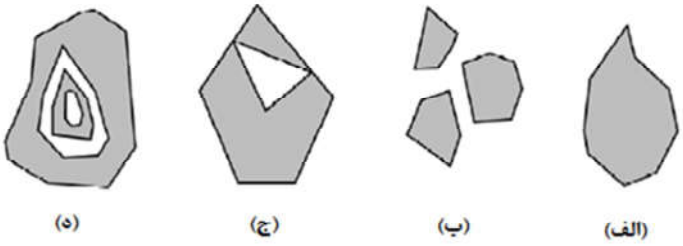
مرز چندضلعی چندگانه شامل مجموعه‌ای از منحنی‌های بسته است که مرزهای چندضلعی‌ها را تشکیل می‌دهند. هر منحنی در مرز چندضلعی دقیقاً مرز آن را تشکیل می‌دهد. در شکل ۳، چهار نمونه از چندضلعی‌های چندگانهٔ صحیح با (الف) یک چندضلعی، (ب) سه چندضلعی ، (ج) دو چندضلعی، (د) دو چندضلعی، نشان داده شده است، همچنین در شکل ۴، چند نمونهٔ نادرست از چندضلعی‌های چندگانه آمده است (.Ibid).



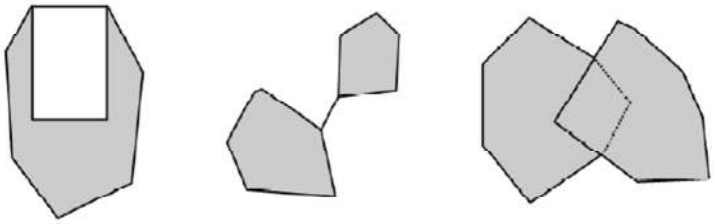
شکل۱. نمونه‌هایی نادرست از چندضلعی ساده
منبع: OGC, 2011

شکل ۲. حالت‌های گوناگون سطح صفر در چندضلعی‌ها

1. Simple Feature Specification (SFS)
2. Oosterom et al.



شکل۳ . نمونه‌هایی از چندضلعی‌های چندگانهٔ صحیح
منبع: OGC, 2011



شکل ۴. چند نمونه چندضلعی چندگانهٔ نادرست
منبع: Ibid.

۱-۲- مروری بر تحقیقات انجام‌شده

برخی محققان، برای رفع مشکلات موجود در چندضلعی‌ها، از شاخص‌گذاری استفاده کردند. فو و همکاران[1] (۲۰۱۲) شاخص‌گذاری داده‌های مکانی توزیع‌شده، براساس طرح درخت مستطیلی چندمرحله‌ای، را مطرح کردند. لی و همکاران[2] (۲۰۱۳) نیز شاخص‌گذاری موازی، در شبکهٔ داده‌های مکانی[3]، برمبنای اِفراز فضای هیلبرت در چندضلعی محدب را پیشنهاد دادند. آنها مدل جدیدی را برای ذخیره‌سازی داده‌های حجیم، براساس سیستم توزیع‌شدهٔ هادوب، مطرح کردند که در آن، برای ذخیرهٔ داده‌های مکانی در حالت برداری و نقطه‌ای و تشکیل شاخص‌گذاری داده‌ای مکانی توسعه‌یافته‌ای، از منحنی هیلبرت و درخت چهارچهاری[4] استفاده می‌شود. برای بازگرداندن پایداری توپولوژیکی به پایگاه داده، تلاش‌های گوناگونی صورت گرفته است (Servigne et al., 2000; Rodriguez et al., 2010; Xie et al., 2010) ولی این روش‌ها نظام‌مند نیستند و به‌شکل خودکار، اعمال نمی‌شوند.

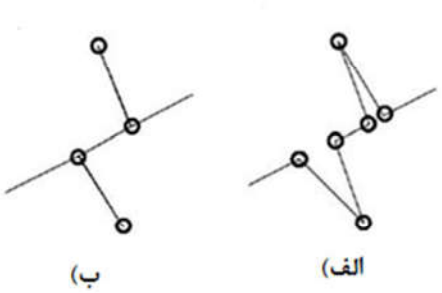
ثُبات بخشی از مفهوم کیفیت داده است که یکی از موضوعات بسیار مهم در استفاده از داده‌های مکانی به‌شمار می‌رود. سروینیه و همکاران[5] (۲۰۰۰) برای بهبود وضعیت ثبات داده‌های مکانی راهکارهایی پیشنهاد کردند، شامل تغییر هندسی چندضلعی‌ها، تغییر شکل، حذف و تقسیم آنها، که در شرایطی خاص می‌توان آنها را اعمال کرد. اگرچه این کار برای نشان‌دادن تناقض مفید است؛ برای حل مشکل، تمایل اندکی به‌سمت مسائل تئوری دارد. افزون‌بر آن، تغییر در ساختار هندسی برای رفع مشکل ممکن است باعث ایجاد مشکل در ساختار هندسی دیگر شود که، در این تحقیق، به آن توجه نشده است. همچنین، دِنگ و همکاران[6] (۲۰۰۳) از ایجاد دگرگونی در ساختار هندسی برای بازگرداندن ثبات در بانک‌های اطلاعاتی مکانی استفاده کردند. آنها در تحقیق خود الگوریتمی را مطرح کردند که، با ترکیب گروهی از نقاط دارای

1. Fu et al.
2. Li et al.
3. spatial-grid
4. quad tree
5. Servigne et al.
6. Deng et al.

اختلاف فازی معین، نقطهٔ جدیدی ایجاد می‌کرد. تعمیم این روش باعث ایجاد توابع ادغام در پکیج سامانه‌های تجاری اطلاعات جغرافیایی شد که می‌توانست، با دریافت میزان خطای دقت از کاربر، به ارزیابی داده‌ها اقدام کند. به همین صورت، سیه و همکاران[1] (۲۰۱۰) روشی را برای تصحیح ناپایداری توپولوژیکی پیشنهاد دادند که مبنای آن عملگرهای بافر و مثلث‌بندی دلونی بود. با هدف دادن پاسخ‌های سازگار به پرس‌وجوها از پایگاه داده‌ای ناسازگار، رودریگس و همکاران[2] (۲۰۱۳) مرمت معنایی را تعریف کردند که برمبنای کوچک‌کردن ساختار هندسی، با توجه به محدودیت‌های توپولوژیکی، عمل می‌کرد. در این تحقیق، اندازه‌گیری فاصله تعریف می‌شود و برای ایجاد نمونه‌ای از پایگاه دادهٔ جدید به‌کار می‌رود که با نمونهٔ پایگاه دادهٔ ناپایدار اولیه متفاوت است و محدودیت‌های یکپارچگی را برآورده می‌کند. این روش جدید «مرمت حداقل[3] برای بازگرداندن ثبات» نامیده می‌شود. در این کار، مجموعه‌ای از محدودیت‌های یکپارچگی در نظر گرفته شده و نیز، نشان داده شده است که یافتن مرمتی حداقل، از روی بانک اطلاعات اولیه، اغلب دشوار است. در چنین مواردی، می‌توانیم از الگوریتم‌های تقریبی استفاده کنیم و مسئله را، به تعداد مواردی که می‌توان انجام داد، کاهش دهیم.

چفت‌کردن[4] در اغلب روش‌های تصحیح ناسازگاری در نرم‌افزارهای تجاری استفاده شده است و براساس حد آستانهٔ بین دو نقطه، و یا یک خط و یک نقطه، انجام می‌شود. همان‌طور که در شکل ۵ نشان داده شده، این روش، ممکن است نتایج ناخواسته‌ای، به‌شکل تورفتگی یا بیرون‌زدگی، دربر داشته باشد.

روش چفت‌بندی در اغلب روش‌های تصحیح ناسازگاری و نرم‌افزارهای تجاری سیستم اطلاعات جغرافیایی استفاده شده است (Cho et al., 2014). لدو و همکاران[5] (۲۰۱۴) از مثلث‌بندی کراندار، برای اصلاح چندضلعی‌ها، استفاده کردند. در این روش، پس از



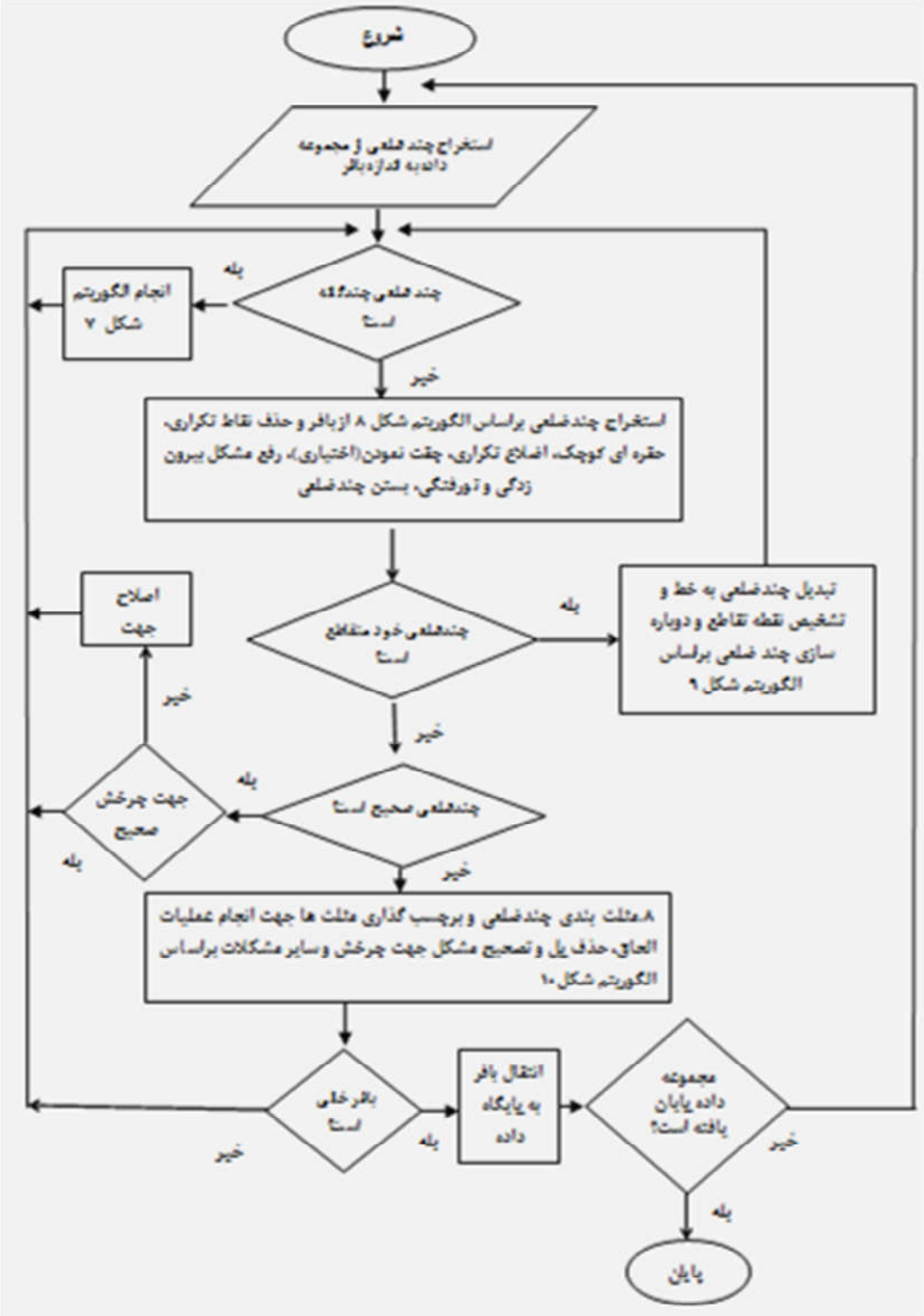
شکل ۵. چفت‌بندی رئوس
منبع: Cho et al., 2014

اطمینان از بسته‌بودن چندضلعی‌ها، مثلث‌بندی دلونی و برچسب‌گذاری مثلث‌ها برای تشخیص هم‌پوشانی، حفره و حذف بخش‌های اضافی به‌کار رفت. این روش، اگرچه چندضلعی‌ها را به‌درستی اصلاح می‌کرد، مدیریت استفاده از حافظه را در دست نداشت. اهوری و همکاران[6] (۲۰۱۲) تحقیقی درمورد اصلاح نازسازگاری در پارتیشن مسطح انجام دادند. این روش، به‌کمک مثلث‌بندی و برچسب‌گذاری مثلث‌ها، نواحی هم‌پوشان و حفره‌ها را تشخیص می‌داد و با استفاده از شش عملگر گوناگون، مثلث‌هایی را که در نواحی هم‌پوشانی و حفره قرار دارند، به چندضلعی‌های دیگر تخصیص می‌داد. لدو و اُهوری (۲۰۱۷) بعدها این الگوریتم را، با سه تغییر، بازنگری کردند و بهبود دادند. این تغییرات شامل این موارد بودند: ۱. اختصاص کد اولویت به چندضلعی‌ها در مجموعه داده؛ ۲. رفتار متفاوت با حفره به‌نسبت مناطق هم‌پوشان؛ ۳. افزودن کران اضافی در زیرمجموعهٔ حفره‌ها.

۲- مواد و روش‌ها

مراحل کار ترمیم چندضلعی با استفاده از پایگاه داده در فلوچارت شکل ۶ نشان داده شده است. مجموعه دادهٔ ورودی در قالب CSV به نرم‌افزار داده می‌شود و در خروجی، فایل اصلاح‌شده در این قالب و در پایگاه دادهٔ مکانی اصلاح‌شده‌ای، استخراج می‌شود.

1. Xie et al.
2. Rodriguez et al.
3. Minimal Repair
4. Snapping
5. Ledoux et al.
6. Ohori et al.



شکل ۶. مراحل ترمیم و اعتبارسنجی چندضلعی‌ها

الگوریتم بررسی چندضلعی چندگانه در شکل ۷ نشان داده شده است. چندضلعی چندگانه، نخست، به چندضلعی‌های تشکیل‌دهندهٔ خود تجزیه می‌شود. سپس، هریک از چندضلعی‌ها به‌شکل مجزا بررسی و در صورت نیاز، مراحل ترمیم براساس شکل ۶ انجام می‌شود (خط سوم الگوریتم شکل ۷) و چندضلعی چندگانه دوباره ساخته می‌شود. کار ساخت و دوباره‌سازی چندضلعی در زمان O(v) صورت می‌گیرد. v تعداد رئوس چندضلعی‌های چندگانه است؛ بنابراین، پیچیدگی این الگوریتم برابر با پیچیدگی ساخت مثلث و ترمیم چندضلعی است.

با هدف بهینه‌سازی استفاده از حافظهٔ اصلی در عملیات ترمیم، از پایگاه دادهٔ H2 استفاده شده است. برای این کار، نخست، مجموعه دادهٔ ورودی را در پایگاه داده ذخیره و به‌اندازهٔ بافر مشخص‌شده، داده‌های ورودی را وارد حافظهٔ اصلی می‌کنیم. سپس، عملیات بررسی و ترمیم را، روی هر چندضلعی، انجام می‌دهیم. مراحل کار در الگوریتم شکل ۸ آورده شده است. مراحل خواندن و نوشتن در پایگاه داده، هریک، در زمان O(g) انجام می‌شود که g تعداد چندضلعی‌های موجود در مجموعه داده است.

برای ترمیم چندضلعی، در گام نخست، باید مطمئن شویم چندضلعی نیازی به عملیات ترمیم دارد یا خیر. اگر چندضلعی صحیح باشد، نیازی به عملیات مثلث‌بندی و ترمیم ندارد. در وهلهٔ اول، نقاط تکراری باید حذف شوند. ازآنجاکه این‌گونه نقاط معمولاً سطوح دارای مساحت صفرند، این کار در زمان O(v) انجام می‌شود. v تعداد رئوس چندضلعی است که هزینهٔ آن بسیار کمتر از مثلث‌بندی است. اگر چندضلعی خودمتقاطع نباشد؛ نیازی به پردازش حلقه‌های آن نیست. اینک اگر چندضلعی نیاز به ترمیم داشت؛ از روش مثلث‌بندی کراندار محدودشده برای ترمیم استفاده می‌کنیم. چندضلعی صحیح باید از حلقه‌های صحیح تشکیل شده باشد. فرق چندضلعی و حلقه در وجود حفره در چندضلعی است. حلقه باید بسته و بدون خودتقاطعی باشد و جهت آن، برای مرزهای خارجی برخلاف عقربه‌های ساعت و برای مرزهای داخلی در جهت عقربه‌های ساعت باشد. نواحی دارای مساحت صفر مانند خط بریده، بیرون‌زدگی و تورفتگی نیز باید حذف شوند.

Repair_Multipolygon Algorithm
Input: an Unreliable Multipolygon
Output: a Confident Multipolygon
1. Decomposition Multipolygon to Polygons
2. For all Polygon Created
3. Check Validity and Repaired if needed
4. Reconstruct Multipolygon
5. Return Multipolygon
6.End

شکل ۷. بررسی و ترمیم چندضلعی چندگانه

DRUD Algorithm

Input: Csv file Contains Unreliable GIS data

Output: Csv file & Database Contains Confident GIS Polygon

```
1. Creat Data Base
2. Read CSV file and convert it to database table
3. Assign Buffer_Size
4. Set Page_Index=1
5. For all record in Table do
6.      Skip_page:=(Page_index-1)*Buffer_Size
7.       Read Geometry from Table and Load it in Buffer
8.       For all Geometry in Buffer do
9.            Check Geometry Validation and Repair it if needed
10.      Update data Base
11.      Page_index++
12.Convert Table to CSV File format
13.End
```

شکل ۸. مراحل ذخیره در بانک اطلاعاتی و بازیابی

Repair Self-Intersection Algorithm

Input: A Polygon

Output: A polygon or A Multipolygon Without Self-Intersection

```
1. if  polygon has self_intersection
    2. Convert Polygon To Lines
    3. Reconstruct polygon or Multipolygon by adding lines to each
      other
    4.  Return MultiPolygon
5. Else Return Polygon
6.End.
```

شکل ۹. الگوریتم بررسی خودتقاطعی

پس از اطمینان از محدود و بسته‌بودن حلقه‌ها، مثلث‌بندی کراندار انجام می‌شود اگر در نقاط داخلی یک یال تقاطعی وجود داشت؛ آن یال به دو یال تقسیم می‌شود. اگر یال دوبار تکرار شده باشد؛ حذف می‌شود. گاهی اوقات ممکن است چندضلعی، در یک یال یا بخشی از آن، خودمتقاطع باشد. این نکتهٔ مهمی است که باید بررسی شود چون، وقتی چنین موردی داشته باشیم، بخش داخلی و خارجی چندضلعی در دو سوی خط قرار می‌گیرند که باعث اختلال در برچسب‌گذاری، در مراحل بعدی، می‌شود. بنابراین، یالی که دوبار از یک زوج‌نقطه عبور کرده است باید حذف شود.

برای بررسی این وضعیت، باید نقاط انتهای یال‌ها را چک کنیم که دارای نقطهٔ ابتدایی یکسان نباشند. این کار با استفاده از سلسله‌مراتب دلونی، در زمان O(log p)، برای هر چندضلعی، و در مجموع O(p log p) انجام می‌شود. p تعداد کل نقاط، پس از مثلث‌بندی، است.

مراحل بررسی حالت خودتقاطعی در چندضلعی، در الگوریتم شکل ۹، نشان داده شده است.

اگر تا این مرحله مشکلات چندضلعی حل نشده باشد؛ براساس الگوریتم شکل ۱۰ و با استفاده از مثلث‌بندی دلونی، پل‌ها و نقاطی را که چندضلعی و حفره‌ها در آنها یکدیگر را لمس کرده‌اند از بین می‌بریم.

نتیجهٔ مثلث‌بندی، برای مجموعه‌ای از نقاط، یکتا نیست که در این پژوهش، نامطلوب است. برای ما اهمیت دارد بهترین مثلث‌بندی را، برای چندضلعی‌ها، به‌گونه‌ای انجام دهیم که مثلث‌بندی یکتا باشد و حداقل زاویهٔ مثلث‌ها را حداکثر کند. بدین‌منظور، از مثلث‌بندی دلونی استفاده می‌کنیم که دایرهٔ محدودکنندهٔ سه نقطه را به‌کار می‌برد؛ به‌گونه‌ای‌که نقطهٔ دیگری نباید داخل این دایره وجود داشته باشد. روش دیگری که در این مقاله از آن استفاده شده مثلث‌بندی کراندار[1] است. برای این کار، در مثلث‌بندی، باید یال‌های مرزی را نیز در نظر بگیریم. ازآنجاکه مثلث‌بندی کراندار اغلب یکتا نیست و بسیاری از آنها دلونی نیستند؛ با ترکیب آنها، می‌توان مثلث‌بندی دلونی کراندار را ایجاد کرد که معمولاً مثلث‌بندی یکتایی را نتیجه می‌دهد (Ohori, 2010). برای برچسب‌گذاری مثلث‌ها، از نقطه‌ای دوردست[2] به‌منزلهٔ نقطهٔ خارجی، کار را آغاز می‌کنیم و تا زمانی‌که به ضلع مرزی برخورد نکردیم؛ به‌صورت بازگشتی، نقاط برچسب یکسان (خارجی) می‌گیرند. (Ledoux et al., 2014). ازآنجاکه حلقه‌ها محدود و بسته هستند و بی‌نهایت مثلث در محدودهٔ خارجی داریم (Yvinec, 2010). مثلث‌ها به‌صورت یکسان برچسب خارجی می‌گیرند.

ازاین‌رو برچسب‌گذاری مثلث‌ها، تا زمانی‌که از ضلعی مرزی عبور نکرده باشند، به تمامی وجوه همسایه نیز ممکن است گسترش یابد. وقتی از روی ضلع مرزی عبور کنند، وجوه داخلی آغاز می‌شود. همچنین، وجوهی که از دو ضلع مرزی عبور کنند خارجی‌اند و به همین ترتیب، کار ادامه می‌یابد. الگوریتم شکل ۱۱ به‌کمک دو پشته این کار را انجام می‌دهد؛ از یک پشته برای نگهداری وجوه داخلی و از دیگری برای نگهداری وجوه خارجی استفاده می‌شود. پیچیدگی آن O(t) برای t مثلث است.

Repair_By_Triangulation Algorithm

Input: an Unreliable Polygon
Output: a Confident Polygon
1. Polygon Triangulation by Delaunay Method
2. Labeling Triangles by Far-Away-Point Algorithm
3.Convert Polygon To Polyline
4. Remove Bridge & Touch Section from Polyline
5. Extraction Holes from Polyline
5. Convert Polyline to Polygon
6. Return Polygon
7.End.

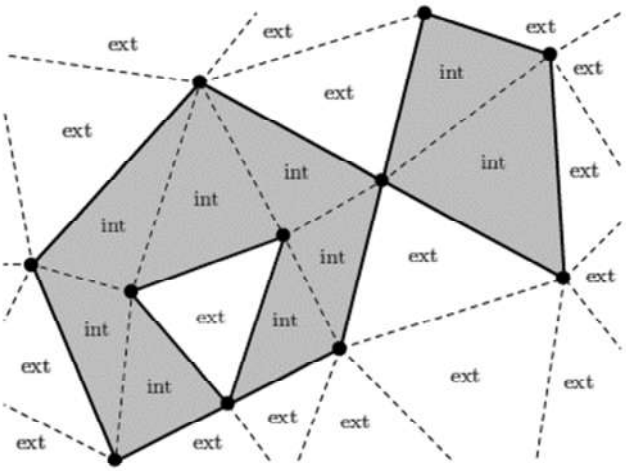
شکل ۱۰. الگوریتم رفع مشکل چندضلعی با استفاده از روش مثلث‌بندی

1. constrained triangulation
2. far-away-point

```
Input: A face IF of the triangulation of a ring in its exterior.
Output: A tagged triangulation, where each triangle is marked as being part of
        either the interior or the exterior of the ring.
Push IF into a stack ES, with another stack IS still empty
while IS and ES are not empty do
    if IS is not empty then
        Pop a face F from IS.
        foreach neighbour N of F not processed do
            Mark N as processed.
            if there is a constrained edge between F and N then push N into ES.
            else push N into IS.
        end
    else
        Pop a face F from ES.
        foreach neighbour N of F not processed do
            Mark N as processed.
            if there is a constrained edge between F and N then push N into IS.
            else push N into ES.
        end
    end
end
```

شکل ۱۱. الگوریتمی برای برچسب‌گذاری بازگشتی حلقه‌های داخلی و خارجی

نتیجهٔ کار ایـن الگـوریتم در شـکل ۱۲ نشـان داده شده است.



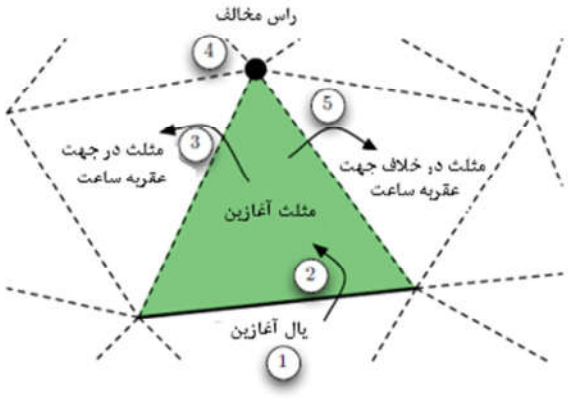
شکل ۱۲. عملیات برچسب‌گذاری وجوه

وقتی همهٔ مثلث‌ها برچسب‌گذاری شدند، حلقه‌ها باید بازسازی شوند. خروجی این الگوریتم شامل حلقه‌های جداست که برخی، در داخل خود، دارای حفره‌اند یا ممکن است، به‌علت ازدیاد فضای دارای سطح صفر، هیچ حلقه‌ای در خروجی موجود نباشد. به همین علت، چندخطی[1] مربوط به چندضلعی تولید می‌شود. این چندخطی از روی تمامی مرزهای چندضلعی عبور می‌کند. الگوریتم مورد نظر از مثلث آغازینی[2] که درونی است شروع می‌شود و دنباله‌ای طولانی از یال‌ها را، در جهت درست، ایجاد می‌کند و برای اتصال حفره‌ها و حلقه‌ها به‌منظور دوباره‌سازی چندضلعی، از پل[3] استفاده می‌شود (Ledoux et al., 2014). وقتی همهٔ مثلث‌ها برچسب‌گذاری شدند، حلقه‌ها باید بازسازی شوند. الگوریتم شکل ۱۳ چندخطی مربوط به چندضلعی را تولید می‌کند. الگوریتم از مثلث اولیه‌ای[4] که داخلی است آغاز می‌شود و دنباله‌ای طولانی از یال‌ها را، با جهت چرخش درست، ایجاد می‌کند. این چندخطی از روی تمامی مرزهای چندضلعی می‌گذرد.

پیچیدگی پیمایش مثلث‌ها خطی است؛ بنابراین، پیچیدگی این الگوریتم همان پیچیدگی مثلث‌بندی O(vlogv) خواهد بود. v تعداد رئوس چندضلعی است. الگوریتم از مثلثی اولیه آغاز می‌شود و در جهت عقربه‌های ساعت، به سه یال اعمال می‌شود و رأس مقابل را به فهرست چندخطی می‌افزاید.

```
Input: A face F of the triangulation in the interior of a polygon.
Output: An ordered list of vertices L representing a polyline, which contains all
        the boundaries of the polygon and has its interior always to the right,
        possibly including bridges joining inner and outer boundaries.
1 foreach edge E of F in clockwise order do
2   if E is not constrained and the neighbour N of F along E has not been processed
    then
3     Append to L the results of applying the algorithm recursively with N.
4     Append to L the vertex clockwise from E.
5   end
6 end
```

شکل ۱۳. الگوریتم تولید چندخطی



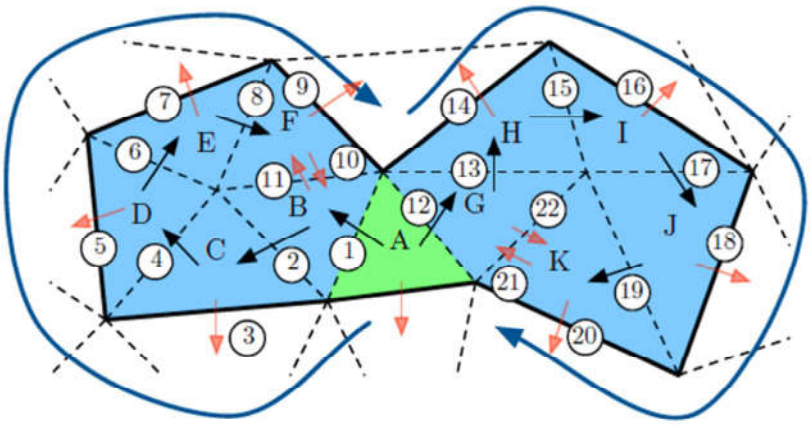
شکل ۱۴. مراحل اجرای الگوریتم تولید چندخطی

1. polyline
2. seeding triangle
3. bridge
4. seeding triangle

مطابق با شکل ۱۴، الگوریتم به‌شکل بازگشتی و با جست‌وجوی عمق اول[1]، از یک مثلث آغازین (۲) و یال آغازین وارد می‌شود (۱)؛ الگوریتم به‌شکل بازگشتی، در جهت عقربه‌های ساعت، بر مثلث‌ها اعمال می‌شود (۳)؛ سپس، رأس مخالف اضافه می‌شود (۴)؛ الگوریتم به‌شکل بازگشتی، برخلاف عقربه‌های ساعت، به مثلث اعمال می‌شود (۵)؛ مراحل تکرار می‌شود. با انجام‌دادن مراحل این الگوریتم، چرخشی در جهت عقربه‌های ساعت، روی مرز چندضلعی، در خروجی الگوریتم بازگردانده می‌شود. مثالی برای حرکت در جهت عقربه‌های ساعت، روی مثلث‌های چندضلعی، در شکل ۱۵ نشان داده شده است.

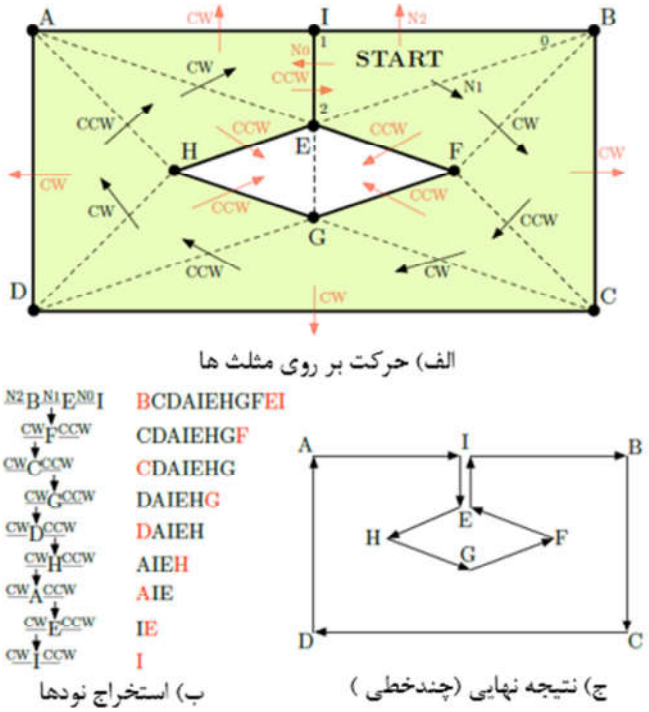
نخست از △A و ضلع بین △A و △B، بدین‌ترتیب، حرکت می‌کنیم: (۱) به △B می‌رویم؛ (۲) وجه سمت عقربه‌های ساعت را بررسی می‌کنیم: △C؛ به △C وارد می‌شویم و ...؛ (۸) به △F وارد می‌شویم؛ (۹) جهت عقربه‌های ساعت را بررسی می‌کنیم: وجه خارجی است؛ (۱۰) جهت خلاف عقربه‌های ساعت را بررسی می‌کنیم که پیش‌تر دیده شده است؛ به‌عقب و به △E برمی‌گردیم؛ ...؛ به △A بازمی‌گردیم. به همین شکل، از △A و یال بین △A و △G آغاز می‌کنیم: (۱۲) به △G وارد می‌شویم؛ (۱۳) جهت عقربه‌های ساعت را بررسی می‌کنیم: △H؛ به‌سمت △Hحرکت می‌کنیم؛ ...؛ (۱۹) به △K وارد می‌شویم؛ (۲۰) جهت عقربه‌های ساعت را بررسی می‌کنیم: وجه خارجی است؛ (۲۱) جهت خلاف عقربه‌های ساعت را بررسی می‌کنیم که پیش از این دیده شده است؛ به △J برمی‌گردیم؛ به △I برمی‌گردیم؛ ...؛ به △A بازمی‌گردیم. خطوط جهت‌دار بلند دور چندضلعی نوع حرکت در جهت عقربه‌های ساعت را نشان می‌دهند.

مراحل ایجاد چندخطی روی یک شکل هندسی در شکل ۱۶ نشان داده شده است. الف) حرکت از مثلث N1 آغاز شده و مسیر حرکت با خط پررنگ (مشکی) مشخص است؛ ب) نودهایی که، در هر مرحلهٔ بازگشتی، به دنباله اضافه می‌شوند با رنگ قرمز مشخص شده است؛ ج) چندخطی نهایی پدیدآمده مشخص شده است. سطوح داخلی چندضلعی در سمت راست چندخطی قرار دارند. براساس حرکت در جهت عقربه‌های ساعت و با استفاده از مثلث‌های ایجادشده داخل چندضلعی، می‌توان چندخطی‌ای ساخت که در آن تمامی مرزهای داخلی چندضلعی دردسترس باشد. به‌دلیل حرکت در جهت عقربه‌های ساعت، جهت چرخش چندخطی، برای همهٔ مرزبندی‌های موجود، درست است ولی دارای پل‌هایی است که مرزهای داخلی و خارجی و یا برخی مثلث‌ها را به هم متصل می‌کنند. این چندخطی پیوسته است و تمامی مرزها را طی می‌کند؛ بنابراین، تمامی حفره‌ها و چندضلعی‌ها در این چندخطی وجود دارند که باید، با حذف پل‌ها، استخراج شوند.



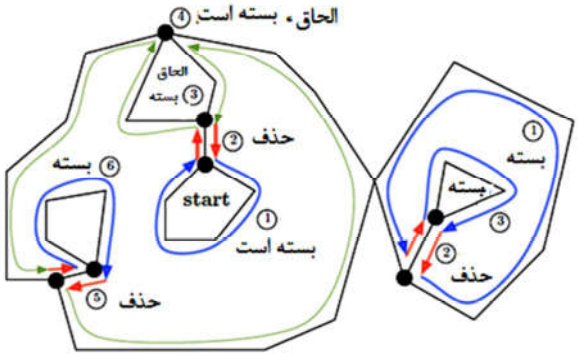
شکل ۱۵. مثالی برای حرکت روی مثلث‌ها، در جهت عقربه‌های ساعت

1. DFS



شکل ۱۶. نمایش گام‌به‌گام ایجاد چندخطی

برای تغییر جهتِ چرخش چندخطی ایجادشده، در جهت خلاف عقربه‌های ساعت، می‌توان فهرست پدیدآمده را وارونه کرد. پس از ایجاد چندخطی، برای دوباره‌سازی چندضلعی از روی چندخطی، مراحل کار آغازمی‌شود. برای این کار، همان‌گونه که گفته شد، چندخطی از یک مثلث آغازین داخلی شروع می‌شود و همهٔ حفره‌های دارای مرز خارجی را متصل می‌کند. مرز داخلی هم‌بند است و در یک سمت خط قرار گرفته است (در این مثال، سمت چپ). اغلب برای هریک از اجزای داخلی، چندخطی جداگانه‌ای ساخته می‌شود. پل‌ها زمانی ایجاد می‌شوند که از یک خط چندخطی دوبار عبور کنیم. در این چندخطی، درجهٔ هر رأس براساس یال‌های چندخطی محاسبه می‌شود. وقتی درجهٔ یک رأس بیشتر از دو باشد، برچسب «نقطهٔ بُرش»[1] را به‌خود می‌گیرد. این نقاط، با توجه به محل قرارگیری، به بخش‌های کوچک‌تر بریده می‌شوند. برخی از این بخش‌ها، که نقاط پایانی مشترک و جهت چرخش مخالف دارند، به بخش‌های دیگر متصل و خود حذف می‌شوند. باقی‌مانده نیز، در جهت چرخش درست، تشکیل حلقه می‌دهند.



شکل ۱۷. نمایش مراحل استخراج چندضلعی از چندخطی

1. cutting point

در شکل ۱۷، رئوس دارای درجهٔ بیشتر از دو، به‌شکل نقطهٔ پررنگ مشکی به‌منزلهٔ نقطهٔ برش، مشخص شده‌اند. از این نقاط، چندخطی‌ها از هم جدا می‌شوند. برای هریک از این نقاط، نخست، یک حلقهٔ بسته پیدا می‌شود که نقطهٔ آغازین است. از این نقطه آغازین، حلقه‌های بسته پیدا و از لیست اولیه حذف می‌شوند و در فهرستِ حلقه‌های کامل قرار می‌گیرند (خطوط آبی). چندخطی‌های تکراری حذف می‌شوند (خطوط قرمز) و این کار، تا بسته‌شدن تمامی حلقه‌ها و حذف خطوط تکراری، ادامه می‌یابد. این شیوه، برای همهٔ حلقه‌های ورودی نیازمند ترمیم، تکرار می‌شود. درنهایت، مجموعه‌ای از حلقه‌های بسته داریم که جهت چرخش هریک نشان‌دهندهٔ داخلی یا خارجی‌بودن مرز آن است. این به ما امکان می‌دهد چندضلعی ورودی را، به‌شکل استاندارد عوارض ساده، تعریف کنیم.

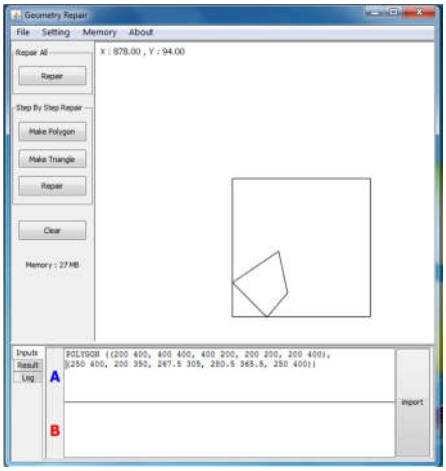
۲-۱- محاسبهٔ پیچیدگی زمانی الگوریتم ترمیم چندضلعی

پیچیدگی عملیات ترمیم چندضلعی براساس پیچیدگی مثلث‌بندی تعیین می‌شود که برابر O(V.logV) است و V تعداد رئوس چندضلعی‌ها در مجموعه داده محسوب می‌شود. در بدترین حالت، ممکن است مثلث‌بندی در زمان $O(v^2)$ انجام شود؛ یعنی در حالتی که چهار یال یکدیگر را قطع کنند (چندضلعی ستاره). در مجموعه داده‌های جی.آی.اس. در آزمایش واقعی، معمولاً تعداد تقاطع‌ها بسیار کمتر از تعداد رئوس است. همچنین، چندضلعی در زمان O(v.r.logr) دوباره‌سازی می‌شود که v تعداد رئوس در حلقه‌ها و r تعداد حلقه‌هاست. عمل بستن حلقه‌ها، ترمیم جهت چرخش و برچسب‌گذاری در زمان خطی و کمتر انجام می‌شود؛ بنابراین، زمان کلی الگوریتم O(v.log v + v.r.logr) است که، با توجه به مجموعه داده‌های آزمایش‌شده، دیده می‌شود تعداد v بسیار بزرگ‌تر از تعداد حلقه‌هاست. درنتیجه، پیچیدگی کلی همان O(v.logv) خواهد بود. درمورد ترمیم چندضلعی‌های چندگانه نیز، برای هر چندضلعی، عملیات ترمیم در زمان O(p.log p) محاسبه انجام می‌شود. p تعداد نقاط موجود در چندضلعی چندگانه را نشان می‌دهد و این زمان بسیار کمتر از O(V.logV) است. بنابراین، زمان ترمیم چندضلعی به‌صورت O(V.log V) محاسبه می‌شود.

۳- نتایج

در روش مثلث‌بندی، چون به مثلث‌های بسیار و برچسب‌گذاری آنها نیاز داریم، حافظهٔ گسترده‌ای به‌کار می‌آید و با اینکه اجرای مثلث‌بندی در جی.تی.اس. کاملاً سریع است، کاربرد حافظه بهینه نیست. بنابراین، هنگام استفاده از این روش، با محدودیت در حجم مجموعه داده روبه‌رو می‌شویم. اصلاح و بهینه‌سازی را می‌توان به دو روش انجام داد: کاهش مصرف حافظه در روش مثلث‌بندی (Blandford et al., 2005)؛ اجرای روش‌هایی برای حفظ بخشی از مثلث‌بندی در حافظهٔ اصلی در یک زمان (Isenburg et al., 2006). به‌دلیل حجیم‌بودن بیشتر مجموعه داده‌های به‌کاررفته، برای بهینه‌سازی در مصرف حافظه، می‌توان از پایگاه داده استفاده کرد؛ بنابراین، محدودیت عملیات‌های توپولوژی در پایگاه‌های داده‌ای مکانی باید محاسبه شوند (Zlatanova and Stoter, 2006). بخشی از مجموعه داده را می‌توان در حافظه بارگذاری و پردازش کرد و سپس، در پایگاه داده قرار داد. بدین‌ترتیب، امکانی فراهم می‌شود که مجموعه داده‌های بزرگ‌تری، در حافظهٔ اصلی، پردازش شوند.

در شکل ۱۸، نمایی از نرم‌افزار حاصل از این تحقیق، به نام Geometry_Repair، برای ترمیم چندضلعی آورده شده است که، با استفاده از آن، می‌توان هر چندضلعی را جداگانه ترمیم کرد و یا مجموعه‌ای از داده‌ها را وارد و تمامی چندضلعی‌های موجود را، یک‌باره، ترمیم کرد.

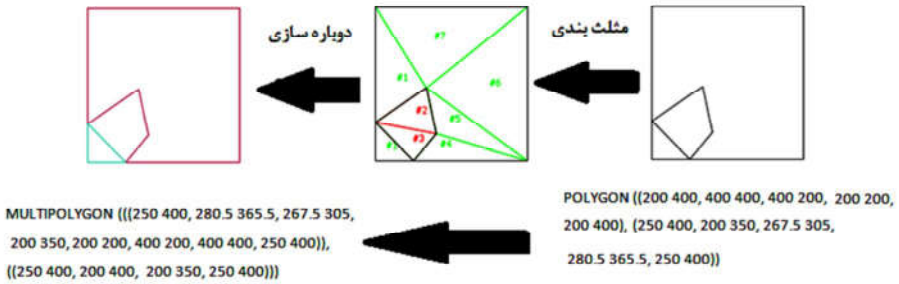


شکل ۱۸. رابط گرافیکی نرم‌افزار پیشنهادی

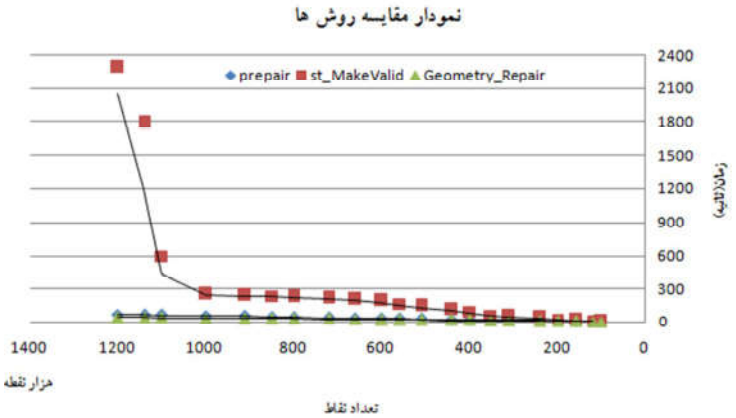
در شکل ۱۹، مراحل کار در چندضلعی، که یک حفرهٔ داخلی و دو مشکل تقاطع با یال خارجی و جهت چرخش نادرست دارد، آورده شده که چندضلعی چندگانه‌ای، شامل دو حلقه، از آن استخراج شده است.

برای آزمایش این نرم‌افزار، از مجموعه داده‌های واقعی پایگاه کورین بهره بردیم. پایگاه دادهٔ کورین دارای مجموعه داده‌های حجیم در مقیاس تعداد نقاط و چندضلعی‌هاست که آژانس محیط‌زیست اروپا آن را تهیه کرده است.

در این تحقیق، نتایج با کار تحقیقاتی لدو و همکاران (۲۰۱۴)، (دستور prepair) و نرم‌افزار PostGIS، (تابع ST_MakeValid)، روی رایانه‌ای همراه با پردازشگر 2.6 GHZ، دارای ۱۲ گیگابایت حافظه و سیستم‌عامل ویندوز ۱۰.۰ بررسی شد. نتایج نشان می‌دهد بیشتر خطاها مربوط به چرخش نادرست چندضلعی، خودتقاطعی و تعداد نقاط تکراری است که با نتایج پژوهش‌های پیشین مطابقت دارد. همچنین، مشاهده شد که با افزایش زیاد تعداد نقاط به ۱,۲۰۰,۰۰۰ نقطه، زمان اجرا در نرم‌افزار PostGis به‌شدت افزایش می‌یابد ولی این زمان، در نرم‌افزار Geometry_Repair و Prepair (شکل ۲۰)، به‌صورت خطی، تفاوت چشمگیری با تابع ST_MakeValid در نرم‌افزار PostGis دارد.



شکل ۱۹. مراحل ترمیم چندضلعی در برنامهٔ پیشنهادی

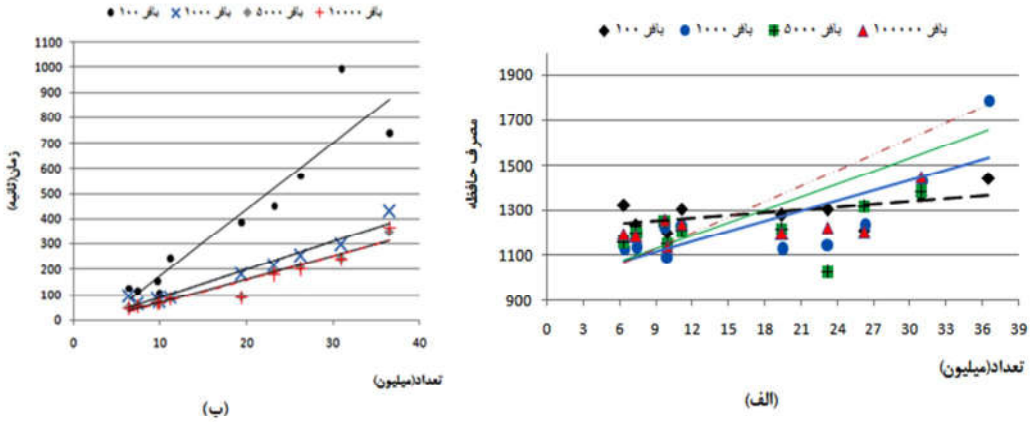


شکل۲۰. نتیجهٔ اجرای الگوریتم Prepair، ST_MakeValid و Geometry_Repair

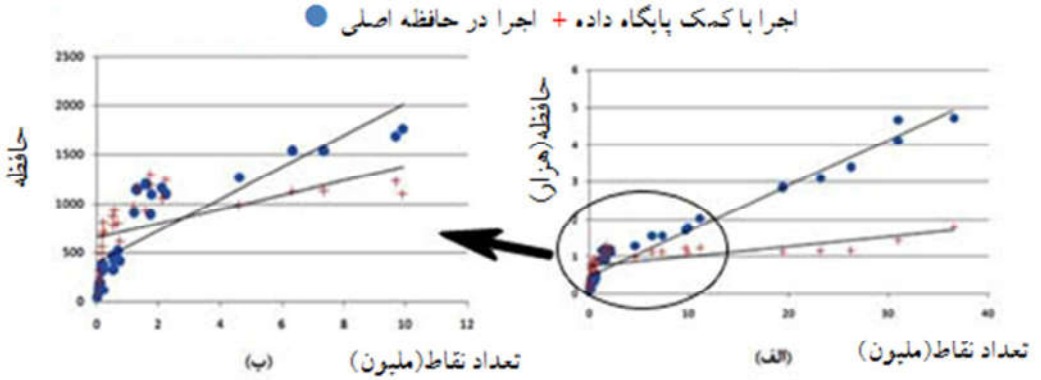
برای بررسی تأثیر استفاده از بانک اطلاعاتی مجموعه داده‌های گوناگون، که کوچک‌ترین آن دارای ۱۸۶۶۹ نقطه و تشکیل‌شده از ۲۰۰ چندضلعی و بزرگ‌ترین آن دارای ۳۶۵۲۶۹۰۶ نقطه و تشکیل‌شده از ۱۸۰۲۴۰ چندضلعی بود، آزمایش شد.

برای بهینه‌سازی مصرف حافظه، از پایگاه داده و حافظهٔ میانگیر بهره گرفته شد. برای یافتن حجم بافر اولیه، این مقدار را با اعداد ۱۰۰، ۱۰۰۰، ۵۰۰۰ و ۱۰۰۰۰ چندضلعی آزمایش کردیم. همان‌گونه که در نمودار شکل ۱۶- الف دیده می‌شود، کمترین میزان استفاده از حافظه با بافر کوچک‌تر (۱۰۰) به‌دست می‌آید. به همین ترتیب، با افزایش میزان بافر، مصرف حافظهٔ اصلی بیشتر خواهد شد. نمودار شکل ۲۱- ب نشان می‌دهد، با افزایش میزان بافر، زمان اصلاح چندضلعی‌ها کاهش می‌یابد؛ به‌طوری‌که با افزایش حجم بافر از ۱۰۰ به ۱۰۰۰، زمان کاهش چشمگیری یافت. در افزایش‌های بعدی حجمِ بافر، میزان کاهش زمان زیاد نبود؛ به‌گونه‌ای‌که با افزایش حجم بافر از ۵۰۰۰ به ۱۰۰۰۰، تأثیری در کاهش زمان در نمودار دیده نمی‌شود.

طبق شکل ۲۲، برای نقاط کمتر از سه میلیون با اجرا در حافظهٔ اصلی، حافظهٔ کمتری به‌کار می‌رود که دلیل آن انجام‌شدن روال اضافی کار با ابزار پایگاه داده در نرم‌افزار است. کم‌کم با افزایش تعداد نقاط به چهل میلیون نقطه، اختلاف روش استفاده از پایگاه داده اختلافی معناداری با روش به‌کارگیری حافظهٔ اصلی پیدا می‌کند.



شکل ۲۱. میزان مصرف حافظه و زمان بافرها با ظرفیت‌های متفاوت



شکل ۲۲. میزان مصرف حافظهٔ برنامهٔ پیشنهادی

۴- بحث و نتیجه‌گیری

در حجم زیاد داده‌ها که گاه بیش از چند ده‌هزار چندضلعی را در یک مجموعه داده شامل می‌شود، روش مثلث‌بندی، با وجود استفادهٔ کتابخانه‌ها از روش‌های سریع و بهینه‌شده، اغلب حافظهٔ زیادی را مصرف می‌کند. روشی پیشنهادی، با کمک کتابخانه‌های GTS در محیط جاوا و با بهینه‌سازی استفاده از تکنیک مثلث‌بندی دلونی، عملیات ترمیم مجموعه داده‌های برداری مکانی را در زمان و با صرف حافظهٔ بهینه، انجام می‌دهد. از مزایای این روش می‌توان به استفاده از الگوریتم‌های ساده و مستحکم و استفاده از پایگاه داده برای مدیریت مصرف حافظه اشاره کرد که، به‌علت رابط گرافیکی سادهٔ خود، اشکالات چندضلعی را به‌شکل خودکار رفع می‌کند. استفاده از پایگاه داده برای داده‌های حجیم بیش از ۵۰٪ مصرف حافظه را بهینه می‌کند. استفاده از حافظهٔ اصلی و بانک اطلاعاتی، در کنار هم، به کاربر این امکان را می‌دهد که، براساس منابع موجود، برای انتخاب روش اصلاح خود تصمیم مناسبی بگیرد. در روش استفاده از حافظهٔ اصلی، برنامه سریع‌تر است و منابع بیشتری مصرف می‌کند. در روش استفاده از بانک اطلاعاتی، برنامه تا حد قابل قبولی کندتر می‌شود ولی مصرف حافظهٔ اصلی، به‌ویژه هنگام کار با داده‌های حجیم که دارای بیش از سی میلیون نقطه‌اند، بهینه می‌شود. در مقایسه با نرم‌افزار postGIS، در تعداد نقاط بالا، روش مطرح‌شده بسیار سریع‌تر است. در مقایسه با پژوهش لدو و همکاران (۲۰۱۴) با توجه به استفاده از پایگاه داده نیز، هنگام به‌کاربردن حافظهٔ اصلی، بهبود چشمگیری در کار با مه‌داده‌ها دیده می‌شود. این روش را می‌توان برای ترمیم چندضلعی‌ها در پارتیشن، ترمیم مشکل هم‌پوشانی و حفره در پارتیشن و کاداستر تعمیم داد. کارهای آینده روی بهینه‌سازی چندضلعی‌های ساده مانند املاک مزروعی، توسعهٔ الگوریتم برای حالت‌های سه‌بعدی و رفع مشکل چندضلعی‌های ناسازگار در کاداستر به‌کمک پایگاه داده را می‌توان با فرمت دیگری مانند shp، به‌منزلهٔ قالب فایل ورودی، ادامه داد.

۵- سپاسگزاری

نویسندگان مقاله از داوران محترم، که زحمت داوری این مقاله را پذیرفتند و باعث بهبود کیفیت مقاله شدند، صمیمانه تقدیر و تشکر می‌کنند.

۶- منابع

Blandford, D.K., Blelloch, G.E., Cardoze, D.E. & Kadow, C., 2005, **Compact Representationsof Simplicial Meshes in Two and Three Dimensions**, International Journalof Computational Geometry and Applications, 15(1), PP. 3–24.

Bolstad, P.V. & Smith, J.L., 1992, **Errors in GIS: Assessing Spatial Data Accuracy**, Journal of Forestry, 90, 11 (1992), PP. 21–29.

Cho, S., Xavier Punithan, M., Gim, J. & Huhd, Y., 2014, **Tagging-the-Triangle Algorithm for Partitioning Features with Inconsistent Boundaries**, International Journal of Geographical Information Science, Vol. 28, No. 12, PP. 2533–2550, http://dx.doi.org/10.1080/13658816.2014.937716.

Deng, M., Chen, Xi., Lia, W., Kusanagi, M., N. Phien, H., 2003, **Modelling Error Propagation for Spatial Consistency**, Journal of Geospatial Engineering, 5 (2), PP. 51–60.

Fu, Z., Liu, S., Tian, Z. & Xu, H., 2012, **Distributed Spatial Index Based on Multilevel R-Tree**, Bull. Surv. Mapp., 11, PP. 42–46.

Isenburg, M., Liu, Y., Shewchuk, J.R. & Snoeyink, J., 2006, **Streaming Computation of Delaunay Triangulations**, ACM Transactions on Graphics, 25(3), PP. 1049–1056.

ISO, 2003, ISO/TC 211/WG 2, ISO/CD 19107, Geographic information — Spatial schema.

JTS, Java Topology Suite, http://tsusiatsoftware.net/jts/main.html.

Ledoux, A. & Ohori, K.A., 2017, **Solving the Horizontal Conflation Problem with a Constrained Delaunay Triangulation**, Journal of Geographical Systems, Vol. 19, Issue 1, PP. 21–42

Ledoux, H.,Ohori, K.A. & Meijers, M., 2014, **A Triangulation-Based Approach to Automatically Repair GIS Polygon**, Computer & Geosciences, Vol. 66, Issue C, PP. 121–131

Li, X. & Zheng, W., 2013, **Parallel Spatial Index Algorithm Based on Hilbert Partition**, Proceedings of the 5th International Conference on Computational and Information Sciences (ICCIS), PP. 876–879.

OGC, 1999, **Open GIS Consortium**, Inc., Open GIS Simple Features Specification For SQL, Revision 1.1, Open GIS Project Document, 99-049, 5 May 1999.

OGC, 2011, OpenGIS® Implementation Standard for Geographic information - Simple feature access - Part 1: Common architecture.

Ohori, K.A., Ledoux, H. & Meijers, M., 2012, **Validation and Automatic Repair of Planar Partitions Using Aconstrained Triangulation**, Photogramm. Fernerkund. Geoinf., 1(October(5)), PP. 613–630.

Ohori, A., 2010, **Validation and automatic Repair of Palanar Partitions Using a Constrained Triangulation**, M, Sc in Geomatics, Pelft University of Technology in Helsinki.

Oosterom, p., Quak, W. & Tijssen, T., 2005, **About Invalid, Valid and Clean Polygons**, Developments in Spatial Data Handling, PP. 1–16.

Preparata, F.P. & Shamos, M.I., 1985, **Computational Geometry, an Introduction**, Springer-Verlag, New York Berlin Heidelberg Tokyo.

Rodríguez, M.A., Bertossi, L.E. & Caniupán, M., 2013, **Consistent Query Answering under Spatial Semantic Constraints**, Information Systems, 38 (2), PP. 244–263.

Rodríguez, M.A., Brisaboa, N., Meza, J. & Luaces, M.R., 2010, **Measuring Consistency with Respect to Topological Dependency Constraints**, In: Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems (GIS'10), San Jose, CA. New York: ACM, PP. 182–191.

Servigne, S., Ubeda, T., Puricelli, A. & Laurini, R., 2000, **A Methodology for Spatial Consistency Improvement of Geographic Databases**, GeoInformatica, 4 (1), PP. 7–34.

Wang, Y., Liu, Z., Liao, H. & Li, C., 2015, **Improving the Performance of GIS Polygon Overlay Computation with MapReduce for Spatial Big Dada Processing**, Cluster Computing, Vol. 18, Issue 2, PP. 507–516.

Xie, Z., Tian, G., Wu, L. & Xia, L., 2010, **A Framework for Correcting Geographical Boundary Inconsistency**, In: The 18th international conference on geoinformatics: GIScience in change, geoinformatics 2010, 18–20 June, Peking University, Beijing. IEEE, 1–5.

Yvinec, M., 2010, **2D Triangulations, in 'CGAL User and Reference Manual**, 3.6 edn, CGAL Editorial Board.

Zlatanova, S. & Stoter, J., 2006, **The Role of DBMS in the New Generation GIS Architecture**, Frontiers of Geographic Information Technology, Springer, Chapter 8, PP. 155–180.

سنجش از دور
و
GIS ایران

سنجش از دور و GIS ایران سال دهم، شماره دوم، تابستان ۱۳۹۷
Iranian Remote Sensing & GIS Vol.10, No. 2, Summer 2018

75-94

Improved Automatic Polygonal Restoration in Geographic Information Systems

Borna, K.[*1] and Fathi, F.[2]

1. Assistant Prof. of Kharazmi University, Faculty of Mathematics and Computer Science, Dep. of Computer Science, Tehran
2. M.Sc. of Computer Engineering (Software), Institute of Technical and Vocational Higher Education, Agriculture Jihad, Agricultural Research, Education and Extension Organization (AREEO), Tehran

Abstract

Repairing incorrect polygons for use in GIS software is semi-automated and time-consuming. Automatic polygon repair, interpretation of obscure polygons, and elimination of all existing bugs based on definitions and global standards that have many uses in software related to GIS. Due to the complexity of the computation and data volumes in working with big data, there is always a competition between the speed and the amount of memory used. In this paper, while introducing the standard of the characteristics of simple complications in polygons, using Delaunay Triangulation and GTS functions in Java and with the help of the H2 database, a method is presented that receives polygons in the form of a file in the CSV format and applies several effective algorithms to automatically repair them. The polygons in the spatial data set are automated at optimal time and with minimal memory consumption and are repaired if necessary. The results show that this method, compared with the previous ones, our method leads to relative improvement in execution speed and provides more than 50 percent saving (in average) in the main memory while working with big data.

Keyword: Delaunay triangulation, Repair polygon,Spatial database, GIS.

Correspondence Address: Faculty of Mathematics and Computer Science, Dep. of Computer Science, Tehran. Tell: 026-34579600, Fax: 026-34569558
Email: borna@khu.ac.ir