موضوعات پیشنهادی برای پروژه‌ی کارشناسی

در این مستند برخی از موضوعات پیشنهادی برای پروژه‌ی کارشناسی فهرست شده‌اند. بیشتر این موضوعات در دو دسته‌ی کلی الگوریتم‌ها و ساختمان‌های داده و برنامه‌های سیستمی قرار می‌گیرند. دانشجویانی که علاقمند به انجام یکی از این موضوعات یا موضوعات مشابه در پروژه‌ی کارشناسی خود هستند، برای توضیحات بیشتر با gholamirudi@nit.ac.ir تماس بگیرند.

نرم‌افزارهای سیستمی

- تغییر فایل‌های اجرایی ELF به منظور بهینه‌سازی یا یافتن خطا
- افزایش سرعت حل یک مسئله‌ی مهم به کمک گسترش‌های SIMD پردازنده‌ها؛ مسئله‌های مطرح شده در فصل یکم پنج‌شنبه‌های سخت، نمونه‌های خوبی برای این مورد و دو مورد بعدی هستند.
- توازی یک الگوریتم مهم با استفاده از «Threading» یا «OpenMP»
- توازی یک الگوریتم با اهمیت با استفاده از «OpenCL» یا «CUDA»
- موتور جستجو برای فایل‌های PDF فارسی
- یک پایگاه داده‌ی مبتنی بر شبکه برای نگهداری کلید و مقدار (Key-value) با کارایی بالا
- استفاده از رابط برنامه‌نویسی سایت semanticscholar.org برای نمایش اطلاعات در مورد مقاله‌ها
- تولید گزارش توسط تیراف به صورت خودکار در یک کاربرد مهم
- یافتن کاربردهای یک کلمه با سرعت بالا با حجم زیاد داده
- تبدیل تصاویر «Raster» به تصاویر برداری (Vectorization)
- پیاده‌سازی یک رابط گرافیکی برای نیتوی (Neatvi)
- پیاده‌سازی لوله‌های یونیکس موازی
- ارائه‌ی سرویسی مبتنی بر شبکه برای تولید فایل‌های خروجی (برای مثال ترجمه‌ی فایل‌های تیراف و برگشت خروجی)

ابزارهای حروفچینی

- تولید فایل‌های PDF دارای لینک یا بخش با استفاده از pdfmark برای نیتپست
- تولید مستقیم فایل‌های PDF به عنوان یک پس پردازشگر نیتراف
- مطالعه‌ی فرمت فایل‌های رایج ارجاع علمی (Citation) و مدیریت آنها
- بهینه‌سازی توصیف فونت‌های OpenType در نیتراف
- مطالعه و مقایسه‌ی الگوریتم‌های شکستن خط‌ها و کشیدن حروف در پاراگراف‌های فارسی
- پیش پردازشگرهای تیراف برای کاربردی مثل ترسیم مدارها

. پیش پردازشگر ریاضی تیراف با ساختار فرمول‌ها در TeX
. پس پردازشگرهای نیتراف برای فرمت‌های فایل جدیدی مثل OpenXPS
. پیش پردازشگر نیتراف برای کشیدن فضای سه بعدی
. محیط مبتنی بر وب برای نوشتن مستندهای نیتراف و نمایش خروجی آنها
. پس پردازشگر نیتراف برای نمایش فضای سه‌بعدی با استفاده از الگوریتم Z-Buffering

مترجم‌ها

. تولید کد برای معماری MIPS یا ARM64 در یک مترجم
. اضافه کردن ویژگی‌هایی مثل «Array Bounds Checking» به یک مترجم
. پیاده‌سازی برخی از بهینه‌سازی‌های مهم برای یک مترجم
. مطالعه، مقایسه و ارزیابی تجربی بهینه‌سازی‌های مختلف مترجم‌های امروزی

الگوریتم‌ها

. مطالعه و پیاده‌سازی الگوریتم‌های موجود برای مسئله‌های مطرح شده در فصل یکم پنج‌شنبه‌های سخت
. تطابق (Matching) در گراف‌های پویا
. تخمین فاصله در گراف‌ها به کمک «Distance Oracles»
. داده ساختاری برای عملیات رده و انتخاب (Rank/select)
. برچسب‌گذاری فاصله (Distance Labeling) در گراف‌های کم‌پشت (Sparse)
. برای پیشنهادهای بیشتر، مکاتبه کنید.

توضیحات کلی در مورد پروژه‌ی کارشناسی

هدف درس پروژه‌ی کارشناسی، استفاده‌ی تجربی از مفاهیم و مهارت‌هایی است که در درس‌های مختلف کارشناسی فرا گرفته‌اید. با توجه به موضوعی که انتخاب می‌کنید، معمولا لازم است در مورد آن مطالعه کنید، راه‌های مختلف انجام آن را مقایسه نمایید و در نهایت آن را انجام دهید. علاوه بر انجام پروژه، گزارشی نیز آماده می‌کنید که در آن در مورد موضوع پروژه و کارهایی که انجام داده‌اید توضیح می‌دهید تا آیندگان را از مهارت‌های خود شگفت‌زده نمایید.

در مورد عنوان پروژه، هر استاد معمولا در زمینه‌های مشخصی تعدادی عنوان پروژه یا موضوع کلی را به شما پیشنهاد می‌دهد. شما با توجه به علاقه‌تان از بین موضوع‌هایی که پیشنهاد می‌شوند می‌توانید انتخاب کنید.

سازماندهی گزارش پروژه

برای شکل کلی گزارش پروژه، اگر با نهایت افتخار از نیتراف استفاده می‌کنید می‌توانید از بسته‌ی ths استفاده کنید و اگر (احتمالا با شرمساری و تأسف) از نرم افزار Word شرکت مایکروسافت استفاده می‌کنید بهتر است از روی ناچاری از الگوی پارسا استفاده کنید. تعداد صفحات گزارش پروژه محدودیت مشخصی ندارد. ولی مطالب پروژه باید پیوسته و کامل باشند. در حین نگارش پروژه ممکن است به این نتیجه برسید بخش‌هایی باید به آن اضافه شوند یا بخش‌هایی از آن حذف شوند.

در مورد سازماندهی فصل‌های پروژه، در فصل اول به مقدمات بپردازید: موضوع پروژه را بیان کنید، اهمیت و کاربردهای آن را ذکر نمایید و کمی در مورد تاریخچه‌ی آن صحبت کنید. سپس صورت مسئله‌ی پروژه را دقیق‌تر بیان کنید، اهداف پروژه را ذکر کنید و گام‌های انجام آن را بیان کنید. در پایان به سازماندهی فصل‌های پروژه اشاره نمایید. در فصل دوم می‌توانید به مفاهیم پایه بپردازید: مفاهیمی که برای درک مسئله، الگوریتم‌ها یا روش اصلی حل آن لازم هستند. در فصل سوم، الگوریتم‌های اصلی یا روش‌های حل مسئله پروژه را شرح دهید. الگوریتم‌ها و روش‌ها را کامل بیان کنید ولی از بیان اثبات‌های طولانی و جزئیات آنها خودداری کنید. در فصل چهارم، به پیاده‌سازی و ارزیابی روش‌های مطرح شده در فصل سوم بپردازید. اگر پیاده‌سازی نکرده‌اید، الگوریتم‌ها را ارزیابی و تحلیل کنید. در فصل پنجم، گزارش را با جمع‌بندی کارهای انجام شده و بیان پیشنهاد برای کارهای آتی خاتمه دهید.

نوشتن یک بسته‌ی نیتراف برای پایان‌نامه‌های فارسی

تیراف (Troff) یکی از قدیمی‌ترین و همین‌طور قدرتمندترین ابزارهای حروفچینی (Typesetting) است که در کنار سیستم عامل یونیکس طراحی و نوشته شده است. مشابه یونیکس که تأثیر چشم‌گیری بر سیستم عامل‌های بعد از خود گذاشته است، ایده‌های بسیار جالبی در تیراف و معماری آن معرفی شده‌اند که با وجود گذشت بیشتر از چهار دهه از تولد آن، ساختار، انعطاف و خروجی این ابزار در میان ابزارهای حرفچینی مشابه بسیار شاخص است. نیتراف (Neatroff) یک پیاده‌سازی جدید از تیراف است که امکانات لازم برای حروفچینی متن فارسی را پشتیبانی می‌کند.

هدف اصلی این پروژه، نوشتن یک بسته‌ی نیتراف برای حروفچینی پایان‌نامه‌های کارشناسی به زبان فارسی است. در بسته‌های حروفچینی، ساختار و شکل مستندها با تعریف ماکروهایی مشخص می‌شوند؛ نوشتن این بسته‌ها نیاز به مهارت در استفاده از تیراف و دستورات آن دارد. از این رو در قسمت اول این پروژه، دستورات و شیوه‌ی استفاده از تیراف مستند می‌گردند و مهارت‌های لازم برای استفاده از تیراف کسب می‌شوند.

گام‌های اصلی پروژه:

۱ مستندسازی دستورات و شیوه‌ی استفاده از تیراف

۲ طراحی بسته‌ای برای پایان‌نامه‌های کارشناسی

۳ مستندسازی بسته‌ی معرفی شده

اطلاعات بیشتر:

معرفی تیراف	http://www.troff.org/54.pdf
تفاوت‌های نیتراف نسبت به تیراف	http://litcave.rudi.ir/neatroff.pdf
معرفی نیتراف به زبان فارسی	http://litcave.rudi.ir/neatfarsi.pdf

برچسب‌گذاری فاصله در گراف‌ها

در بسیاری از کاربردهای مبتنی بر گراف لازم است فاصله‌ی هر دو رأس از گراف محاسبه شود. در صورتی که وزن هر یال حداکثر w و n تعداد رأس‌های گراف باشد، فاصله‌ی دو رأس حداکثر $w(n-1)$ خواهد بود و نگهداری آن به $O(\log wn)$ بیت احتیاج خواهد داشت. بنابراین برای نگهداری فاصله‌ی هر رأس از هر رأس دیگر $O(n^2 \log wn)$ بیت لازم است (برای گراف‌های غیر وزن‌دار w برابر یک است و نگهداری همه‌ی فاصله‌ها به $O(n^2 \log n)$ بیت احتیاج دارد). در صورتی تعداد رأس‌های گراف بسیار زیاد باشد، گاهی اختصاص این مقدار حافظه برای نگهداری فاصله‌ی هر دو رأس امکان ندارد.

یک راه برای کاهش این مقدار حافظه، اختصاص برچسب‌هایی به رأس‌ها است (Graph distance labeling) که با داشتن فقط برچسب هر دو رأس بتوان فاصله‌ی آن دو رأس را محاسبه کرد (برچسب‌گذاری فاصله مزیت‌های دیگری نیز، مخصوصا هنگامی که گراف توزیع شده باشد، دارد). اگر برچسب اختصاص داده شده به رأس u با $l(u)$ نمایش داده شود، الگوریتم محاسبه‌ی فاصله با گرفتن برچسب‌های $l(u)$ و $l(v)$ می‌تواند فاصله‌ی دو رأس u و v را محاسبه کند. برای ارزیابی روش‌های مختلف برچسب‌گذاری فاصله گراف، دو مسئله اهمیت زیادی دارند: طول برچسب و پیچیدگی زمانی الگوریتمی که با گرفتن برچسب دو رأس، فاصله‌ی آنها را محاسبه می‌کند. در ساده‌ترین حالت، برچسب یک رأس می‌تواند فاصله‌ی آن رأس تا هر رأس دیگر باشد که در آن صورت طول هر برچسب $O(n \log nw)$ خواهد بود و فاصله‌ی دو رأس با توجه به برچسب آنها با پیچیدگی زمانی $O(1)$ قابل محاسبه خواهد بود. اما این برچسب‌ها را می‌توان با الگوریتم‌هایی بهبود داد. در این پروژه برخی از این الگوریتم‌ها مطالعه، پیاده‌سازی و عملکرد آنها روی گراف‌های بزرگ ارزیابی می‌شوند.

گام‌های اصلی پروژه:

۱ مطالعه‌ی چند روش برچسب‌گذاری فاصله در گراف‌ها

۲ پیاده‌سازی برخی از روش‌های مطالعه شده

۳ ارزیابی روش‌های پیاده‌سازی شده

اطلاعات بیشتر:

یکی از روش‌های برچسب‌گذاری فاصله http://arxiv.org/pdf/1504.04498v1

پیش-پردازشگرهای نیتراف

معماری تیراف انعطاف زیادی برای گسترش این ابزار حروفچینی ارائه می‌دهد. اضافه کردن یک پیش-پردازشگر (Preprocessor) جدید یا نوشتن تعدادی ماکرو برای تیراف یا پیش‌پردازشگرهای آن، دو راه برای انطباق تیراف برای کاربردهای جدید می‌باشد. برای مثال، پیش-پردازشگر pic، با استفاده از دستورات سطح پایین تیراف، امکان کشیدن شکل را در تیراف فراهم می‌سازد.

یکی از کاربردهای ممکن برای ابزارهای تولید مستند، کشیدن مدارهای الکترونیکی می‌باشد. در این پروژه، یک پیش-پردازشگر برای کشیدن مدارهای الکترونیکی پیاده‌سازی می‌شود. در پروژه‌ی مشابهی می‌توان پیش-پردازشگری نوشت که انواع مختلف نمودارها را بکشد.

گام‌های اصلی پروژه:

۱ معرفی پیش-پردازشگرهای مرتبط

۲ پیاده‌سازی پیش‌پردازشگر برای کشیدن مدار

۳ مستندسازی پیش-پردازشگر

اطلاعات بیشتر:

معرفی پیش-پردازشگر pic https://en.wikipedia.org/wiki/Pic_language

برخی از پیش‌پردازشگرهای تیراف http://troff.org/macros.html

محاسبه‌ی فراگیرنده

پردازش گراف‌های بسیار بزرگ به منابع زیاد و گاهی غیر قابل دسترس احتیاج دارد. برای مثال اگر گرافی با n رأس و m یال چند میلیون رأس داشته باشد، تخصیص $O(n^2)$ کلمه‌ی حافظه یا اجرای یک الگوریتم با پیچیدگی زمانی $O(n^2)$ با کامپیوترهای رایج غیر ممکن یا بسیار کند است. اما در صورتی که تعداد یال‌های گراف ورودی کم باشد، الگوریتم‌هایی که پیچیدگی زمانی یا حافظه‌ی آنها $O(m)$ باشد، به راحتی قابل اجرا خواهند بود. از این رو، یکی از راه‌هایی که برای پردازش گراف‌های بسیار بزرگ به کار گرفته می‌شود، حذف تعدادی از یال‌های این گراف‌ها است تا پردازش آن سریع‌تر گردد و از سوی دیگر ویژگی‌های مورد نظر در گراف چندان تغییر نکنند. در صورتی که ویژگی مورد نظر فاصله‌ی رأس‌ها از یکدیگر باشد، گراف حاصل یک فراگیرنده (Spanner) نامیده می‌شود.

یک فراگیرنده H از گراف G دارای کشش (α, β) است اگر به ازای هر دو رأس مثل u و v شرط $d_G(u,v) \leq d_H(u,v) \leq d_G(u,v) \times \alpha + \beta$ برقرار باشد ($d_G(u,v)$ فاصله‌ی رأس‌های u و v در گراف G است). در این پروژه برخی الگوریتم‌های انتخاب فرگیرنده از یک گراف مطالعه، پیاده‌سازی و ارزیابی می‌شوند.

گام‌های اصلی پروژه:

۱ مطالعه‌ی چند الگوریتم انتخاب فراگیرنده

۲ پیاده‌سازی برخی از الگوریتم‌های مطالعه شده

۳ ارزیابی الگوریتم‌های پیاده‌سازی شده

اطلاعات بیشتر:

یکی از الگوریتم‌های انتخاب فراگیرنده http://arxiv.org/pdf/1403.0178

طراحی و پیاده‌سازی رابطی مبتنی بر فایل

یکی از ایده‌های بسیار موفق یونیکس، معرفی رابطی (Interface) مبتنی بر فایل برای بسیاری از منابع موجود در سیستم عامل بوده است. استفاده از چنین رابطی سبب سادگی بسیاری از جنبه‌های یونیکس، از جمله برنامه‌های سیستمی و رابط‌های سیستم عامل شده است. استفاده از فایل به عنوان رابط، مزیت‌های دیگری نیز دارد، از جلمه: عدم وابستگی به یک زبان برنامه‌نویسی، استفاده از مکانیزم‌های کنترل دسترسی به فایل‌ها برای کنترل دسترسی به منابع، و استفاده از برنامه‌هایی که برای کار با فایل نوشته شده‌اند بدون تغییر. در سیستم‌های عامل جدیدتر نیز برای بسیاری از منابع سیستم عامل که در زمان سیستم عامل یونیکس مرسوم نبوده‌اند رابطی مبتنی بر فایل در نظر گرفته شده است. در سیستم عامل Plan 9 حتی برای منابعی مثل اتصالات شبکه نیز رابط مبتنی بر فایل طراحی شده است.

در این پروژه، رابطی مبتنی بر فایل برای برخی از منابع گوشی‌های همراه، مشابه خدماتی که سیستم عامل اندروید (Android) به پردازه‌ها ارائه می‌دهد، طراحی و پیاده‌سازی می‌شود. ابتدا خدماتی که سیستم عامل به پردازه‌های کاربری ارائه می‌دهد دسته‌بندی می‌گردند و سپس برای برخی از این منابعی رابط جدیدی مبتنی بر فایل ارائه داده می‌شود و مستند می‌گردد. سپس برای ارزیابی این رابط، با استفاده از فایل سیستم‌های محیط کاربری (Userspace) آنها پیاده‌سازی می‌گردند.

گام‌های اصلی پروژه:

۱ دسته‌بندی خدمات ارائه شده به برنامه‌ها در اندروید

۲ طراحی رابط برای برخی از خدمات دسته‌بندی شده

۳ پیاده‌سازی خدمات طراحی شده با FUSE

اطلاعات بیشتر:

فایل سیستم‌های محیط کاربری در لینوکس https://github.com/libfuse/libfuse

رابط گرافیکی برای نیتوی

یکی از قدیمی‌ترین و قدرتمندترین ویرایشگرهای یونیکس، وی (VI) می‌باشد که کاربران بسیار زیادی دارد. این ویرایشگر، امکانات بسیار زیادی را برای ویرایش سریع فایل‌ها در اختیار کاربر قرار می‌دهد که در ویرایشگرهای گرافیکی جدید یافت نمی‌شود. این ویرایش‌گر در دو محیط اصلی دستورات را اجرا می‌کند: در محیط EX دستورات خط به خط خوانده می‌شوند و اجرا می‌گردند و در محیط VI دستورات ویرایشی به صورتی تعاملی (Interactive) وارد و اجرا می‌گردند. نیتوی (Neatvi) یک پیاده‌سازی جدید از وی می‌باشد که امکان ویرایش خط‌های راست-به-چپ و فارسی را پشتیبانی می‌کند.

در این پروژه، یک رابط گرافیکی با استفاده از کتابخانه‌ی GTK یا QT برای نیتوی طراحی می‌شود که با آن بتوان به صورت گرافیکی متن فارسی را ویرایش کرد.

گام‌های اصلی پروژه:

۱ مستندسازی VI و شیوه‌ی کار با آن

۲ تغییر نیتوی برای افزودن رابط گرافیکی

اطلاعات بیشتر:

کد نیتوی http://repo.or.cz/neatvi.git

انتقال یک مترجم به معماری‌های جدید

بسیاری از مترجم‌ها (Compilers) می‌توانند کد نهایی را برای محیط‌ها یا معماری‌های گوناگونی تولید کنند. از این رو در مترجم‌ها معمولا قسمت‌های مربوط به تولید کد نهایی به شکلی پیاده‌سازی می‌شود که افزودن پشتیبانی یک معماری جدید به راحتی قابل انجام باشد. یکی از مترجم‌هایی که با این دید نوشته شده است نیتسیسی (Neatcc) می‌باشد.

در این پروژه، قابلیت تولید کد نهایی برای یکی از معماری‌های رایج مثل MIPS یا ARM64 به مترجم نیتسیسی اضافه می‌شود. چون تولید کد نهایی، احتیاج به اطلاع از دستورات و جزئیات این معماری‌ها دارد، لازم است در گام اول این پروژه مهارت استفاده از دستورات این معماری‌ها ایجاد گردد.

گام‌های اصلی پروژه:

۱ مستندسازی ویژگی‌های اصلی و دستورات معماری

۲ انتقال مترجم به معماری جدید

۳ انجام آزمون‌های درستی کد نهایی

اطلاعات بیشتر:

معماری MIPS	https://en.wikipedia.org/wiki/MIPS_instruction_set
معماری ARM64	https://en.wikipedia.org/wiki/ARM64
کد نیتسیسی	http://repo.or.cz/neatcc.git

پس-پردازشگرهای نیتراف

مشابه کد میانی در مترجم‌ها، هسته‌ی نیتراف کدی تولید می‌کند که توسط پس-پردازشگرهای (Post-processor) آن به فرمت‌های نمایش خروجی مثل PostScript تبدیل می‌گردد. وجود کد میانی نیتراف سبب می‌شود که بدون تغییر برنامه‌ی اصلی نیتراف و پیش-پردازشگرهای آن، امکان تولید مستند به یک فرمت خروجی جدید فراهم شود. یکی از فرمت‌های نمایش جدید OpenXPS (Open XML Paper Specification) می‌باشد.

در این پروژه یک پس-پردازشگر نیتراف برای تولید خروجی به فرمت OpenXPS پیاده‌سازی می‌گردد. این برنامه با خواندن کد میانی نیتراف، آن را به فرمت OpenXPS تبدیل می‌کند. نوشتن چنین پس‌پردازشگری نیاز به آشنایی با معماری نیتراف و کد میانی آن و همین‌طور فرمت OpenXPS دارد.

گام‌های اصلی پروژه:

۱ معرفی فایل‌های توصیف فونت و کد میانی نیتراف

۲ معرفی فرمت خروجی OpenXPS

۳ پیاده‌سازی پس-پردازشگر نیتراف

اطلاعات بیشتر:

فرمت OpenXPS http://www.ecma-international.org/publications/standards/Ecma-388.htm

جمع‌آوری و نمایش مسیر حرکت

بیشتر تلفن‌های همراه به کمک فن‌آوری‌هایی مثل GPS مکان جغرافیایی را گزارش می‌دهند. دنباله‌ی مکان تلفن همراه در بازه‌های زمانی مشخص، مسیر یک گوشی را بیان می‌کند. با داشتن این مسیر، می‌توان اطلاعات جالبی را روی مسیر به کاربر نشان داد و آن را تحلیل کرد.

در این پروژه، برنامه‌ای نوشته می‌شود که مسیر حرکت تلفن همراه را ذخیره می‌کند و آن را نمایش می‌دهد. سپس، این مسیر یا قسمتی از آن را که کاربر مشخص می‌کند به برنامه‌ی کمکی می‌دهد که مکان‌های پر رفت و آمد را تشخیص می‌دهد و به برنامه‌ی کمکی دیگری می‌دهد که با توجه به متغیرهایی مثل سرعت، مسیر را تکه تکه می‌کند (این دو برنامه توسط افراد دیگری نوشته می‌شوند). برنامه‌ی این پروژه باید خروجی این برنامه‌ها را به کاربر نمایش دهد.

گام‌های اصلی پروژه:

۱ استفاده از رابط برنامه‌نویسی اندروید برای دریافت مکان جغرافیایی

۲ دریافت و نمایش یک مسیر

۳ دادن ورودی به برنامه‌های کمکی و خواندن خروجی آنها

۴ نمایش خروجی برنامه‌های کمکی

اطلاعات بیشتر:

در مورد مکان‌های پر رفت و آمد [۱]

در مورد گسستن مسیر [۲,۳]

مطالعات تجربی در هندسه‌ی محاسباتی

اولین همایش هندسه‌ی محاسباتی ایران (The Iranian Conference on Computational Geometry) در زمستان سال جاری در تهران برگزار می‌شود. این همایش مقاله‌های کوتاه چهار صفحه‌ای را می‌پذیرد و مهلت ارسال مقاله به این همایش نهم آذر است. با برنامه‌ریزی مناسب می‌توانید مقاله‌ای را برای این همایش آماده کنید. از بین رویکردهای مختلفی که در هندسه‌ی محاسباتی وجود دارند، مطالعه‌ی تجربی الگوریتم‌ها برای دانشجویان دوره‌ی کارشناسی مناسب‌تر است. برای این کار، الگوریتم‌هایی را مطالعه می‌کنید، آنها را پیاده‌سازی می‌کنید و با الگوریتم‌های دیگر مقایسه می‌کنید، نتایج آزمایش را تحلیل می‌کنید و آنها را بهبود می‌دهید. در پیاده‌سازی الگوریتم‌ها می‌توانید از کتابخانه‌های هندسه‌ی محاسباتی مثل CGAL و LEDA نیز کمک بگیرید. برای ارزیابی و مقایسه‌ی الگوریتم‌ها به داده‌های آزمایشی خوب احتیاج دارید و لازم است آنها را نیز بیابید.

چند مورد از موضوعاتی از همایش The European Workshop on Computational Geometry را برای این رویکرد پیشنهاد می‌کنم: [۴]، [۵]، [۶]، [۷]، [۸] یا [۹].

گام‌های اصلی پروژه:

۱ انتخاب یکی از مقاله‌ها و مطالعه‌ی الگوریتم مطرح شده در آن

۲ جستجو برای داده‌های آزمایش و پیاده‌سازی‌های گذشته

۳ پیاده‌سازی و بهبود الگوریتم

۴ مقایسه و تحلیل نتایج

اطلاعات بیشتر:

سایت همایش	http://cg.aut.ac.ir/iccg2018
فایل چکیده‌ی مقاله‌های EuroCG	http://csconferences.mah.se/eurocg2017/proceedings.pdf
سایت کتابخانه‌ی CGAL	https://www.cgal.org/

مراجع

1. J. Gudmundsson, M. J. van Kreveld, F. Staals, “Algorithms for hotspot computation on trajectory data,” pp. 134–143 in *SIGSPATIAL/GIS* (2013).
2. M. Buchin, A. Driemel, M. J. van Kreveld, V. Sacristán, “Segmenting trajectories - A framework and algorithms using spatiotemporal criteria,” *Journal of Spatial Information Science* **3**(1), pp. 33–63 (2011).
3. B. Aronov, A. Driemel, M. J. van Kreveld, M. Löffler, F. Staals, “Segmentation of Trajectories on Nonmonotone Criteria,” *ACM Transactions on Algorithms* **12**(2), pp. 26:1–26:28 (2016).
4. M. Kleinhans, M. van Kreveld, T. Openhelders, W. Sonke, B. Speckmann, K. Verbeek, “Computing Representative Networks for Braided Rivers,” pp. 45–48 in *The European Workshop on Computational Geometry* (2017).
5. S. P. Fekete, D. Krupke, “Covering Tours with Turn Cost: Variants, Approximation and Practical Solution,” pp. 57–60 in *The European Workshop on Computational Geometry* (2017).
6. V. Alvarez, S. P. Fekete, A. Schmidt, “Computing Triangulations with Minimum Stabbing Number,” pp. 109–112 in *The European Workshop on Computational Geometry* (2017).
7. G. Avarikioti, I. Z. Emiris, I. Psarros, G. Samaras, “Practical Linear-space Approximate Near Neighbors in High Dimension,” pp. 161–164 in *The European Workshop on Computational Geometry* (2017).
8. J. Cleve, W. Mulzer, “An experimental Study of Algorithms for Geodesic Shortest Paths in the Constant Workspace Model,” pp. 165–168 in *The European Workshop on Computational Geometry* (2017).
9. T. A. Granberg, T. Polishchuk, C. Schmidt, “A Novel MIP-based Airspace Sectorization for TMAs,” pp. 173–176 in *The European Workshop on Computational Geometry* (2017).