موضوعات پیشنهادی برای پروژه‌ی کارشناسی

در این مستند برخی از موضوعات پیشنهادی برای پروژه‌ی کارشناسی فهرست می‌شوند (انجام نشده □، در حال انجام ■ و انجام شده ■). بیشتر این موضوعات در دو دسته‌ی کلی برنامه‌های سیستمی و الگوریتم‌های هندسی قرار می‌گیرند. به عنوان نمونه، برخی از این موضوعات در صفحه‌های بعدی این مستند با جزئیات بیشتری توضیح داده می‌شوند. 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

- □ رابطی مشابه ShareLatex برای نیتراف
- □ مرورگر متنی برای صفحه‌های وب فارسی
- □ ارائه‌ی سرویسی مبتنی بر شبکه برای تولید فایل‌های خروجی (برای مثال ترجمه‌ی فایل‌های تیراف و برگشت خروجی)
- □ اجرای فایل‌های اجرایی خارجی در یک سیستم عامل
- □ مدیریت میلیون‌ها نامه در یک یا چند سرور با تأخیر کم
- □ استخراج و نمایش آمار سایت‌ها در بازه‌های مختلف با پیاده‌سازی یک سرور نام دامنه (DNS)
- ■ ذخیره‌سازی مسیر و نمایش تحلیل آن
- □ افزایش سرعت حل یک مسئله‌ی مهم به کمک گسترش‌های SIMD پردازنده‌ها؛ مسئله‌های مطرح شده در شماره‌های اخیر 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
- □ پیاده‌سازی یک محیط مجازی با بازنویسی تابع‌هایی که در زمان اجرا Link می‌شوند
- □ تغییر فایل‌های اجرایی ELF به منظور بهینه‌سازی یا یافتن خطا
- □ 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 سایت‌هایی مثل crossref.org و semanticscholar.org و تحلیل داده‌های آنها
- □ 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 (Image Tracing) به صورت موازی

- [■] پیاده‌سازی یک رابط گرافیکی برای نیتوی (Neatvi)

ابزارهای حروفچینی

بیشتر موارد زیر در مورد نیتراف هستند. نیتراف یک برنامه‌ی حروفچینی است که مانند TeX، توصیف یک متن را به فایل خروجی مثل PDF تبدیل می‌نماید. برای معرفی، مستند «تولید مستندهای فارسی با نیتراف» (پیوند) را مطالعه کنید.

- [□] تولید مستقیم فایل‌های PDF به عنوان یک پس پردازشگر نیتراف
- [□] 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
- [□] 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
- [□] پیش پردازشگر نیتراف برای تولید حرکت در شکل‌ها
- [□] پیاده‌سازی بسته‌ی ماکرو برای نمودارهای متنوع (مثل ستونی، منحنی، پای، پشته‌ای) در نیتراف
- [■] نوشتن یک بسته‌ی نیتراف برای پایان‌نامه‌های فارسی
- [■] مطالعه‌ی فرمت فایل‌های رایج ارجاع علمی (Citation) و مدیریت آنها
- [■] تولید فایل‌های PDF دارای لینک یا بخش با استفاده از pdfmark برای نیتپست

مترجم‌ها

بسیاری از موارد زیر باید برای یک مترجم موجود پیاده‌سازی شوند. یکی از مترجم‌هایی که می‌تواند برای این منظور انتخاب شود، مترجم Neatcc است که نسبتا کوچک است و برای سه معماری کد تولید می‌کند.

- [□] 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
- [□] تولید کد Position-Independent در یک مترجم
- [□] اضافه کردن مترجم Just-in-Time به یک مفسر
- [□] مطالعه، مقایسه و ارزیابی تجربی بهینه‌سازی‌های مختلف مترجم‌های امروزی

الگوریتم‌ها

اگر چه هدف موارد زیر پیاده‌سازی یک الگوریتم است، این الگوریتم‌ها باید در کنار رابط کاربری پیاده‌سازی شوند که با توجه کاربرد بتوان از آنها به راحتی استفاده نمود.

- [□] مطالعه و پیاده‌سازی الگوریتم‌های موجود برای مسئله‌های مطرح شده در شماره‌های اخیر پنج‌شنبه‌های سخت

- [ ] شکستن گسسته‌ی مسیر با در نظر گرفتن پارامترهایی مثل سرعت
- [ ] یافتن مربع‌های مشهور مسیرها با اندازه‌ی ثابت
- [ ] پیشنهاد مسیر با توجه به مسیرهای گذشته
- [ ] نمایش اجرای یک الگوریتم هندسی برای نمونه‌های ورودی به صورت گام به گام
- [ ] 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 دوره‌ی 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 می‌توانید با توجه به علاقه‌تان از بین موضوع‌هایی که پیشنهاد می‌شوند یکی را به عنوان پروژه‌ی کارشناسی انتخاب کنید. انتخاب خوب موضوع پروژه تصمیم بسیار دشواری است، چرا که یک بار در زندگی آن را می‌گیرید و قطعا انتخاب خوب می‌تواند گام اول از یک آینده‌ی درخشان کاری باشد. اما در انتخاب خیلی تردید نکنید؛ فرصت‌های زیادی برای انجام پروژه‌های دیگری که به آنها نیز علاقه دارید وجود دارند. می‌توانید قابلیت‌های بدون مرز خودتان را در سایر زمینه‌ها نیز در آینده نشان دهید.

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 بخش‌های گزارش، در بخش 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 بخش‌های پروژه اشاره نمایید. در بخش دوم می‌توانید به مفاهیم پایه بپردازید: مفاهیمی که برای درک مسئله، الگوریتم‌ها یا روش اصلی حل آن لازم هستند. در بخش 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 بخش چهارم، به پیاده‌سازی و ارزیابی روش‌های مطرح شده در بخش سوم بپردازید. اگر پیاده‌سازی نکرده‌اید، الگوریتم‌ها را ارزیابی و تحلیل کنید. در بخش پنجم، گزارش را با جمع‌بندی کارهای انجام شده و بیان پیشنهاد برای کارهای آتی خاتمه دهید.

شیوه‌ی ارزشیابی

بیشترین پاداش شما از انجام پروژه، تجربه و مهارتی است که بدست می‌آورید. اما این کل ماجرا نیست و نمره‌ای هم برای نشان دادن تلاش‌تان به یادگار به شما داده می‌شود. در ارزشیابی درس پروژه‌ی کارشناسی با توجه به نوع پروژه متغیرهای متفاوتی در نظر گرفته می‌شوند. نمره‌ی پروژه‌ی کارشناسی به دو قسمت تقسیم می‌شود. قسمت اول، نظم و کیفیت گزارش است. در

این قسمت متغیرهایی مثل گزارش‌های منظم (حداقل ده بار، هر هفته یا دو هفته یک بـار؛ بیست درصد) و کیفیـت گزارش پـروژه (کیفیت مقدمه، نگارش، ساختار، بررسی روش‌های موجود، توضیح روش پیشنهادی و نمایش نتایج؛ هشتاد درصد) در نظر گرفتـه می‌شود. قسمت دوم، به کیفیت کار فنی که انجام داده‌اید، اختصاص می‌یابد. در پروژه‌هایی که نرم‌افزاری را پیاده‌سازی می‌کنند، این قسمت شامل دستیابی به اهداف، محدودیت‌های پیاده‌سازی و درستی پیاده‌سازی (پنجاه درصد)، کیفیت آزمون‌های درستی برنامه (ده درصد)، خوانایی (بیست درصد)، حمل‌پذیری، موازی‌سازی و بهینه بودن (با توجه به پروژه؛ بیست درصد) می‌باشد.

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 و راه‌اندازی نیتراف (فارسی)	http://litcave.rudi.ir/neatfbeg.pdf
معرفی تیراف	http://www.troff.org/54.pdf
کتاب پردازش متن در یونیکس	http://www.oreilly.com/openbook/utp/UnixTextProcessing.pdf
تفاوت‌های نیتراف نسبت به تیراف	http://litcave.rudi.ir/neatroff.pdf

برچسب‌گذاری فاصله در گراف‌ها (قدیمی)

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 (قدیمی)

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 (قدیمی)

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 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 و حالت‌های آدرس‌دهی مختلف آنها را بدست آورید. سپس باید کد میانی کامپایلر به کد ماشین تبدیل شود. جزئیات زیادی در تولید کد در یک کامپایلر قابل استفاده مطرح می‌شوند که در این پروژه با آنها روبرو می‌شوید.

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 تیراف کدی تولید می‌کند که توسط پس-پردازشگرهای (Post-processor) آن به فرمت‌های نمایش خروجی مثل PostScript تبدیل می‌گردد. وجود کد میانی تیراف سبب می‌شود که بدون تغییر برنامه‌ی اصلی تیراف 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 تیراف، 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 تیراف

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 یا 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 http://dx.doi.org/10.1145/2525314.2525359

در مورد گسستن مسیر http://dx.doi.org/10.5311/JOSIS.2011.3.66

حدس زدن مقصد با توجه به شروع مسیر

با توجه به شروع یک مسیر (مثل شروع حرکت یک تندباد) می‌توان مقصد آن را با توجه به مسیرهای گذشته حدس زد. یک مسیر، نگاشتی از زمان به مکان یک موجود متحرک است. در این پروژه با دریافت مجموعه‌ای از مسیرها، با توجه به شروع یک مسیر، مقصد ممکن آن حدس زده می‌شود. در این پروژه از ساختمان‌های داده‌ای مثل درخت پسوندی استفاده می‌شود.

گام‌های اصلی پروژه:

۱ خواندن مسیرها

۲ مطالعه‌ی روش‌های پیشین

۳ ارائه‌ی الگوریتم برای جستجوی مسیر

۴ ارزیابی الگوریتم

اطلاعات بیشتر:

یکی از روش‌ها http://dx.doi.org/10.1145/2505821.2505824

پیشنهاد مسیر با توجه به مسیرهای گذشته

گاهی لازم است مسیری برای حرکت از نقطه‌ای به یک مقصد تعیین شود. یک راه برای این کار استفاده از نقشه و اجرای الگوریتم‌های مسیریابی روی آن است. در این پروژه رویکرد دیگری انتخاب می‌شود. با داشتن تعدادی مسیر از گذشته، بررسی می‌شود که کدام یک از این مسیرها از دو نقطه‌ی مورد نظر گذشته‌اند و از بین آنها، مسیری که در کمترین زمان این فاصله را طی کرده‌است پیشنهاد می‌گردد. می‌توان پیشنهادها را با توجه به روز هفته (شنبه‌ها) یا زمان مسافرت در روز (برای مثال از بازه‌ی هفت تا هشت صبح) دسته‌بندی کرد. برای این پروژه لازم است الگوی مشخصی برای گرفتن مسیرهای پیشین و نشان دادن مسیر پیشنهادی تعیین شود.

در این پروژه، ابتدا باید مسیرها از مجموعه‌ی داده‌هایی خوانده شوند. در یک مسیر، مکان یک موجود متحرک در زمان‌های مشخصی بیان می‌شود تا بتوان با توجه به این اطلاعات مکان جسم را در هر لحظه‌ای حدس زد. سپس، با گرفتن مبدأ و مقصد و فرض وجود فقط یک مسیر، زمان عبور موجود از همسایگی مبدأ به همسایگی مقصد را می‌توان محاسبه کرد (در صورت عبور جسم از این ناحیه‌ها). پس از آن، خوب است که یک رابط گرافیکی آماده شود که در آن مسیر نمایش داده شود و مبدأ و مقصد انتخاب گردند. سپس، جستجو روی بیش از یک مسیر گذشته پیاده‌سازی می‌گردد و از بین آنها باید قسمتی از مسیری برگشت داده شود که در زمان کمتری از همسایگی مبدأ به همسایگی مقصد می‌رود. در پایان، می‌توان جستجو را به ساعت‌های مشخصی از روز و روزهای مشخصی از هفته محدود کرد.

گام‌های اصلی پروژه:

۱ بررسی داده‌های مسیر و خواندن ورودی‌ها
۲ پیشنهاد مسیر بین دو نقطه
۳ نمایش مسیر به صورت گرافیکی و انتخاب مبدأ و مقصد
۴ محدود کردن زمان در مسیرها

اطلاعات بیشتر:

یافتن منابع مربوط

نوشتن برنامه‌ای برای GIS Cup

در کنار کنفرانس معتبر ACM SIGSPATIAL، رقابتی به نام GIS CUP انجام می‌شود که در مورد تحلیل داده‌های مکانی است. در رقابت امسال، با گرفتن تعدادی مبدأ و تعدادی مقصد، قرار است همه‌ی رأس‌هایی گزارش شوند که در روی یک مسیر از یک مبدأ به مقصد وجود دارند (این مسیر ممکن است کوتاه‌ترین مسیر نباشد ولی نباید رأس تکراری در این مسیر ظاهر شود). برنده در این رقابت، علاوه بر دریافت جایزه، امکان ارائه‌ی مقاله در این کنفرانس را خواهند داشت.

گام‌های اصلی پروژه:

۱ خواندن فایل‌های JSON

۲ خواندن ورودی مسئله

۳ ارائه و تحلیل الگوریتم برای این مسئله

۴ پیاده‌سازی و ارزیابی الگوریتم

اطلاعات بیشتر:

یافتن منابع مربوط

استخراج نقشه از مسیرها

با استفاده از مجموعه‌ای از مسیرها، می‌توان نقشه‌ی عبور افراد، ماشین‌ها یا سایر موجودات متحرک را بدست آورد. در این پروژه، با دریافت تعداد زیادی مسیر، چنین نقشه‌ای استخراج می‌گردد. با توجه به نرخ عبور موجودات از هر قسمت از نقشه، می‌توان ترافیک هر قسمت از نقشه را نیز استخراج نمود. سپس این نقشه‌ها به کاربر نمایش داده می‌شوند و برای نوشتن در یک فایل آماده می‌گردند.

گام‌های اصلی پروژه:

۱ خواندن یکی از مجموعه‌های مسیر

۲ ارائه و تحلیل الگوریتم برای این مسئله

۳ پیاده‌سازی و ارزیابی الگوریتم

۴ نمایش نقشه‌ی حاصل به صورت گرافیکی

اطلاعات بیشتر:

مجموعه‌ی تاکسی‌ها T-Drive

مجموعه‌ی داده‌ی نمونه GeoLife

سرویسی برای مدیریت مسیرها

در این پروژه سروری برای مدیریت مسیرها و پاسخ به پرسش‌ها پیاده‌سازی می‌گردد. این سرور می‌تواند از فرمتی مثل JSON استفاده کند. عملیاتی که توسط این سرور پشتیبانی می‌شوند شامل اضافه کردن مسیر، اعلان مکان، پرسش‌های کلی در مورد مسیرها، پرسش افراد نزدیک به یک محل، و پرسش مسیر بین دو نقطه هستند. سرور می‌تواند پاسخ‌های قبلی را برای افزایش سرعت تا مدتی نگه داشته باشد. در کنار این سرور، برنامه‌ای برای کاربر (مثلا برای اندروید) برای ارتباط با سرور نیز پیاده‌سازی می‌گردد.

گام‌های اصلی پروژه:

۱ کار با ساکت‌های شبکه

۲ آشنایی با فرمت JSON

۳ تعیین الگوی درخواست‌ها و پاسخ‌ها

۴ پاسخ به درخواست‌ها

اطلاعات بیشتر:

یافتن منابع مربوط

بهینه‌سازی توصیف فونت‌ها در نیتراف

فونت‌های OpenType از ویژگی‌های زیادی پشتیبانی می‌کنند. برای مثال در این فونت‌های مشخص می‌شود که ارتفاع حرکت تشدید برای هر حرف چقدر باشد، اگر حرف الف پس از حرف لام قرار بگرید باید به چه شکل تغییر کند یا اگر حرف الف پس از حرف ر قرار بگیرید باید فاصله‌ی این حروف کم شود. برای اینکه نتیراف از فونت‌های OpenType استفاده کند، برنامه‌ی Neatmkfn این قواعد را از فونت‌ها استخراج می‌کند و در فایلی می‌نویسد. با استفاده از گروه‌ها برای قواعد مشابه، می‌توان اندازه‌ی این فایل‌ها را کاهش داد.

گام‌های اصلی پروژه:

۱ مطالعه‌ی ویژگی‌های فونت‌های OpenType

۲ آشنایی با توصیف فونت‌ها در نیتراف

۳ تغییرات لازم برای بهینه سازی فونت‌ها

۴ ارزیابی و مقایسه‌ی روش‌های بهینه‌سازی

اطلاعات بیشتر:

معرفی نیتراف http://litcave.rudi.ir/neatfbeg.pdf

فهرست ویژگی‌های فونت‌های OpenType ویکی‌پدیا یا مایکروسافت

نیتراف و برنامه‌های کمکی آن http://litcave.rudi.ir/

اجرای فایل‌های اجرایی خارجی

می‌توان با شبیه‌سازی فراخوانی‌های سیستمی یک سیستم عامل، فایل‌های اجرایی آن سیستم عامل را در یک سیستم عامل دیگر اجرا کرد، بدون اینکه یک سیستم عامل میهمان در یک ماشین مجازی نصب گردد. برنامه‌ی QEMU این امکان را با عنوان «User-Mode Emulation» برای سیستم عامل‌های Linux و BSD پیاده‌سازی می‌کند. می‌توان قابلیت شبیه‌سازی سیستم‌های عاملی مثل Solaris را نیز به این برنامه اضافه کرد.

گام‌های اصلی پروژه:

۱ دریافت کد یک سیستم عامل مثل Illumos و بررسی فراخوانی‌های سیستمی آن

۲ دریافت کد QEMU و ارزیابی تغییرات لازم

۳ پیاده‌سازی و آزمایش شبیه‌سازی فراخوانی‌های سیستمی

اطلاعات بیشتر:

برنامه‌ی ماشین مجازی QEMU https://www.qemu.org/

کد Illumos https://www.illumos.org/

نوشتن برنامه‌ای برای PACE

در کنار کنفرانس معتبر IPEC، رقابت برنامه‌نویسی به نام PACE برگزار می‌شود. رقابت امسال در مورد یافتن درخت Steiner در یک گراف بدون جهت وزن‌دار است. گرافی به عنوان ورودی داده می‌شود. سپس تعدادی رأس مشخص می‌گردد و هدف یافتن درختی است که همه‌ی این رأس‌ها را در بر داشته باشد و وزن آن کمینه باشد. این مسئله در دسته‌ی NP-سخت قرار دارد و بنابراین جواب چند جمله‌ای دقیق برای آن موجود نیست. جواب‌هایی که ارسال می‌شوند می‌توانند با هدف یافتن جواب دقیق برای نمونه‌های کوچک یا جواب غیر بهینه با تقریب یا جستجوی فضا باشند.

گام‌های اصلی پروژه:

۱ مطالعه در مورد درخت Steiner و الگوریتم‌های ساختن آن

۲ طراحی الگوریتم برای حالت دقیق و جستجو

۳ پیاده‌سازی و ارزیابی الگوریتم

اطلاعات بیشتر:

سایت کنفرانس http://algo2018.hiit.fi/ipec/

سایت رقابت https://pacechallenge.wordpress.com/pace-2018/