دانشگاه صنعتی شریف
دانشکده‌ی مهندسی کامپیوتر

پروژه‌های درس سیستم‌های بی‌درنگ

مدرس: دکتر سپیده صفری

زمستان ۱۴۰۲ و بهار ۱۴۰۳

دانشجویان محترم لیست پروژه‌های درس سیستم‌های بی‌درنگ در ادامه آورده شده است. هر گروه (یک نفره یا دو نفره) باید ۵ اولویت اول خود از بین پروژه‌های زیر را انتخاب کند. شما می‌توانید با دستیار آموزشی هر پروژه در طول ترم در ارتباط باشید و پروژه را پیاده‌سازی کنید.

پروژه شماره ۱ **درس:** سیستم‌های بی‌درنگ

عنوان: زمان‌بندی وظایف بحرانی-مختلط در سامانه‌های چندهسته‌ای با کمک پروتکل SRP **دستیار آموزشی:** خانم ملکی

در یک کارخانه تولیدی، یک سامانه کنترل ممکن است دارای اجزائی با سطح بحرانی بالا برای خاموش شدن اضطراری و اجزائی با سطح بحرانی پایین برای نظارت و گزارش باشد. اطمینان از اینکه اجرای وظایف با سطح بحرانی بالا توسط وظایف با سطح بحرانی پایین به خطر نیفتد، الزامی است. سامانه بحرانی-مختلط دو سطحی، نوعی از سامانه‌های بحرانی-مختلط است که فقط دو سطح بحرانی دارد و شامل وظایف دوره‌ای با درجه‌ی بحرانی پایین LC و وظایف دوره‌ای با درجه‌ی بحرانی بالا HC می‌باشد که وظایف LC دارای یک بدترین زمان اجرا و وظایف HC دارای دو بدترین زمان اجرا هستند. سامانه در حالت عادی (Normal) شروع به کار می‌کند و در صورتی که یکی از وظایف HC زمان اجرای کوچک خود را رد کند ولی هنوز کامل اجرا نشده بود سامانه وارد حالت سرریز (Overrun) می‌شود. این کارخانه دارای چندین خط تولید است و تجهیزات تست و بازرسی، منابع مشترکی دراین کارخانه هستند که خطوط تولید متعدد ممکن است نیاز به استفاده از آن‌ها داشته باشند.

شما به‌عنوان مهندس خط تولید این کارخانه، باید وظایف موردنظر را تحت پروتکل Multi-processor Stack Resource Policy (MSRP) و توسط نسخه تغییر یافته‌ی الگوریتم زمان‌بندی EDF-VD زمان‌بندی کنید و برای نگاشت وظایف از الگوریتم Worst Fit Decreasing (WFD) استفاده کنید. همان‌طورکه از الگوریتم EDF-VD می‌دانید، این الگوریتم برای وظایف با سطح بحرانی بالا در حالت عملیاتی عادی، یک موعد زمانی مجازی در نظر می‌گیرد و اگر یک وظیفه این موعد زمانی را نقض کند، سامانه زمان‌بندپذیر نخواهد بود. حال شما باید این الگوریتم را به گونه‌ای تغییر دهید که موعد زمانی وظایف با سطح بحرانی بالا پس از نقض موعد زمانی مجازیشان، به حالت اولیه برگشته و سطوح قبضگی نیز به‌روزرسانی شوند. برای ارزیابی این روش، شما باید به کمک الگوریتم Uunifast، مجموعه‌ای از وظایف مصنوعی تولید کنید و پارامترهای خواسته شده زیر را به کمک هر دو الگوریتم **EDF-VD** و **EDF-VD تغییر یافته**، بررسی کنید و نتایج را ثبت و گزارش کنید.

خروجی‌های لازم: نمودار زمان‌بندپذیری در هر دو حالت عملیاتی سامانه.

۱. نمودار اول:
- ✓ میانگین زمان‌بندپذیری ۴۰۰ وظیفه با نسبت تعداد وظایف HC به LC برابر با یک در ۱۰ بار اجرا (وظایف متفاوت در هر بار اجرا) ، انتخاب تعداد منابع به صورت تصادفی از بازه [۶-۲]، تعداد هسته برابر با ۲ و بهره‌وری ۰.۲۵ به ازای هر هسته.
- ✓ میانگین زمان‌بندپذیری ۴۰۰ وظیفه با نسبت تعداد وظایف HC به LC برابر با یک در ۱۰ بار اجرا (وظایف متفاوت در هر بار اجرا) ، انتخاب تعداد منابع به صورت تصادفی از بازه [۶-۲]، تعداد هسته برابر با ۴ و بهره‌وری ۰.۲۵ به ازای هر هسته.
- ✓ میانگین زمان‌بندپذیری ۴۰۰ وظیفه با نسبت تعداد وظایف HC به LC برابر با یک در ۱۰ بار اجرا (وظایف متفاوت در هر بار اجرا) ، انتخاب تعداد منابع به صورت تصادفی از بازه [۶-۲]، تعداد هسته برابر با ۸ و بهره‌وری ۰.۲۵ به ازای هر هسته.

۲. نمودار دوم:
- ✓ میانگین زمان‌بندپذیری ۴۰۰ وظیفه با نسبت تعداد وظایف HC به LC برابر با یک در ۱۰ بار اجرا (وظایف متفاوت در هر بار اجرا) ، انتخاب تعداد منابع به صورت تصادفی از بازه [۶-۲]، تعداد هسته برابر با ۲ و بهره‌وری ۰.۷۵ به ازای هر هسته.
- ✓ میانگین زمان‌بندپذیری ۴۰۰ وظیفه با نسبت تعداد وظایف HC به LC برابر با یک در ۱۰ بار اجرا (وظایف متفاوت در هر بار اجرا) ، انتخاب تعداد منابع به صورت تصادفی از بازه [۶-۲]، تعداد هسته برابر با ۴ و بهره‌وری ۰.۷۵ به ازای هر هسته.
- ✓ میانگین زمان‌بندپذیری ۴۰۰ وظیفه با نسبت تعداد وظایف HC به LC برابر با یک در ۱۰ بار اجرا (وظایف متفاوت در هر بار اجرا) ، انتخاب تعداد منابع به صورت تصادفی از بازه [۶-۲]، تعداد هسته برابر با ۸ و بهره‌وری ۰.۷۵ به ازای هر هسته.

فازبندی پروژه:

فاز اول: در فاز اول **پیاده‌سازی الگوریتم تولید وظایف (به همراه گزارش وظایف تولید شده)**، پیاده‌سازی **نگاشت وظایف به هسته‌ها، تولید و نگاشت منابع، تخصیص بخش بحرانی به وظایف** و **تعیین سطوح قبضگی تمام وظایف (براساس پروتکل SRP)** به عنوان نتایج خروجی گزارش شوند.

فاز دوم: در فاز نهایی، **زمان‌بندی** به صورت کامل **پیاده‌سازی** شود **نمودارهای خواسته شده** گزارش شوند.

پروژه شماره ۲ **درس:** سیستم‌های بی‌درنگ

عنوان: زمان‌بندی وظایف بی‌درنگ در رایانش ابری با استفاده از بهینه‌سازی کلنی مورچه‌ها **دستیار آموزشی:** آقای اثنی عشری

این پروژه از بهینه‌سازی کلونی مورچه‌ها[1] استفاده می‌کند، که یک تکنیک هوش ازدحامی است که از رفتار جستجوی مورچه‌ها الهام گرفته‌شده است، تا وظایف بی‌درنگ را در رایانش ابری زمان‌بندی کند. پروژه مسئله‌ی زمان‌بندی را به‌عنوان یک نمودار مدل‌سازی می‌کند، جایی که گره‌ها وظایف را نشان می‌دهند و یال‌ها وابستگی‌ها و هزینه‌های ارتباطی را نشان می‌دهند. سپس پروژه از الگوریتم ACO برای یافتن مسیر بهینه‌ای استفاده می‌کند که هزینه کل را به حداقل می‌رساند و موعد زمانی وظایف را برآورده می‌کند. این پروژه همچنین ماهیت پویای محیط ابری و در دسترس بودن منابع متفاوت را در نظر می‌گیرد. برای ارائه یک تجربه عملی، با استفاده از الگوریتم UUNIFAST، مجموعه‌ای از وظایف نرم مصنوعی را تولید می‌کنید. وظایف فوق دارای مشخصات زمان اجرا[2]، موعد زمانی[3]، اولویت[4]، میزان بهره‌وری[5] و هر مشخصه‌ی دیگری که در این مسیر نیاز دارید، خواهد بود. در ادامه باید مجموعه وظایف فوق را با استفاده از سیاست کمترین سستی نخست[6] نیز زمان‌بندی کرده و نتایج به‌دست‌آمده از آن را با نتایج به‌دست‌آمده از روش ACO مقایسه کنید.

خروجی‌های مورد نیاز: در انتها باید خروجی‌های زیر را بر روی یک سامانه‌ی ۱۶ هسته‌ای همگن زمانی که میزان بهره‌وری هر هسته ۰.۲۵ و ۰.۵ است و بر روی یک سامانه‌ی ۳۲ هسته‌ای همگن زمانی که میزان بهره‌وری هر هسته ۰.۳ و ۰.۷ است ، ارائه کنید:

- نمودار کیفیت خدمات[7] در روش ACO برای هر دو سامانه‌ی ۱۶ و ۳۲ هسته‌ای
- نمودار کیفیت خدمات در روش LLF برای هر دو سامانه‌ی ۱۶ و ۳۲ هسته‌ای
- نمودار زمان‌بندی‌پذیری وظایف در روش ACO برای هر دو سامانه‌ی ۱۶ و ۳۲ هسته‌ای
- نمودار زمان‌بندی‌پذیری وظایف در روش LLF برای هر دو سامانه‌ی ۱۶ و ۳۲ هسته‌ای
- نمودار makespan
- جدول وظایف و مشخصات آن‌ها

فاز اول پروژه (پیاده‌سازی LLF):

تعداد ۵۰ وظیفه با استفاده از الگوریتم UUNIFAST ایجاد و وظایف فوق را با استفاده از سیاست زمان‌بندی LLF زمان‌بندی کنید و نمودارهای کیفیت خدمات و زمانبندی‌پذیری را رسم نمایید.

فاز دوم پروژه:

پیاده‌سازی تمامی موارد گفته‌شده، تولید تمام خروجی‌های خواسته‌شده و تهیه گزارش پایانی.

[1] Ant Colony Optimization (ACO)
[2] Execution Time
[3] Deadline
[4] Priority
[5] Utilization
[6] Least Laxity First (LLF)
[7] Quality of Service (QoS)

پروژه شماره ۳ **درس:** سیستم‌های بی‌درنگ

عنوان: زمان‌بندی پویا وظایف بحرانی-مختلط برای سیستم‌های خودرو **دستیار آموزشی:** آقای طوقانی

در این پروژه به زمان‌بندی وظایف در یک خودروی امروزی خواهیم پرداخت. در این خودرو برخی از وظایف به صورت پریودیک تکرار می‌شوند (مانند سیستم پیش‌بینی برخورد، سیستم بررسی وضعیت خودرو و ...) و برخی دیگر از وظایف در طول زمان و به صورت آپریودیک منتشر می‌شود (مانند سیستم باز شدن کیسه هوا). همچنین هرکدام از وظایف دارای یک سطح بحرانی (کم یا زیاد) هستند. هدف از این پروژه طراحی یک زمان‌بند است که از اجرای وظایف با درجه‌ی بحرانی زیاد اطمینان حاصل شود و برای وظایف با درجه‌ی بحرانی کم میزانی از کیفیت خدمات ارائه شود.

محدودیت‌های پیاده‌سازی:

- سیستم دارای ۲، ۴، ۸ و یا ۱۶ واحد پردازشی همگن است و وظایف دارای سطح بحرانی زیاد یا کم هستند. برای وظایف با سطح بحرانی زیاد دو بدترین زمان اجرا در نظر گرفته می‌شود و در صورتی که یک وظیفه بحرانی زیاد به اندازه زمان اجرای کوچک خود اجرا شود ولی به پایان نرسد، سامانه وارد حالت overrun می‌شود. در پایان هر هایپرپریود حالت سیستم به حالت عادی بازمیگردد.
- وظایف پریودیک: ابتدا وظایف پریودیک را با استفاده از رویکرد BFD[8] بر روی هسته‌ها نگاشت کنید. برای هر هسته، یک وظیفه پریودیک با درجه بحرانی زیاد با عنوان server تعریف کنید و بهره‌وری آن را (U_s) به گونه بدست آورید که تغییری در زمان‌بندی‌پذیری با الگوریتم EDF-VD ایجاد نشود. سپس با استفاده از EDF-VD به زمان‌بندی این وظایف بر روی هر هسته بپردازید (U_s دقتی ۰.۰۵ دارد).
- وظایف آپریودیک: وظایف آپریودیک نیز دارای مهلت زمانی و درجه بحرانی می‌باشند. در صورت منتشر شدن درخواست آپریودیک، هسته‌ای را پیدا کنید که U_s آن توانایی پاسخگویی به این درخواست را داشته باشد. درصورت پاسخگویی به این درخواست، مقدار U_s آن هسته را موقتاً کاهش داده و بعد از پایان مهلت زمانی وظیفه آپریودیک، مقدار U_s را افزایش دهید.

محدودیت‌های شبیه‌سازی:

- هر وظیفه را با احتمال یکسانی یا بحرانی زیاد یا بحرانی کم در نظر بگیرید.
- مهلت زمانی تمام وظایف را بین $[10, 20, 30, 40, 50]$ به صورت تصادفی (با توزیع یکنواخت) انتخاب کنید.
- وظایف پریودیک: این وظایف را به وسیله الگوریتم UUnifast تولید کنید، برای وظایف بحرانی زیاد در نظر بگیرید: $U_\tau^H = 2 * U_\tau^L$ (مقدار خروجی الگوریتم UUnifast را برای U_τ^L آن لحاظ کنید). دوره تناوب وظایف پریودیک را برابر با مهلت زمانی آن‌ها در نظر بگیرید.
- وظایف آپریودیک: زمان اجرای این وظایف را به اندازه نصف مهلت زمانی آن‌ها در نظر بگیرید و زمان منتشر شدن آن‌ها را به صورت تصادفی در طول هایپرپریود وظایف پریودیک، قرار دهید.

خروجی‌ها:

۱. احتمال overrun یک وظیفه بحرانی زیاد را برابر ۰.۲، ۰.۱ و ۰.۰۱ در نظر بگیرید. برای این سه حالت خروجی‌های زیر را بدست آورید:
   - بهره‌وری وظایف پریودیک را ۰.۵ (به ازای هر هسته) و تعداد درخواست‌های آپریودیک در طول هر هایپرپریود را ۴۰ در نظر بگیرید. نمودار کیفیت خدمات را به ازای ۲، ۴، ۸ و ۱۶ هسته پردازشی نمایش دهید.
   - بهره‌وری وظایف پریودیک را ۰.۵ (به ازای هر هسته) در نظر بگیرید و تعداد هسته‌های پردازشی را ۸ لحاظ کنید. نمودار کیفیت خدمات را به ازای ۴۰، ۸۰، ۱۲۰ و ۱۶۰ درخواست آپریودیک نمایش دهید.
   - خروجی زمان‌بندی وظایف در یک هایپرپریود بر روی ۸ هسته پردازشی با بهره‌وری وظایف پریودیک ۰.۵. زمان شروع، پایان، هسته نگاشت شده و سطح بحرانی هر وظیفه باید مشخص باشد (می‌توانید در قالب json یا هر فرمت دیگری که خوانا باشد، بدست آورید).
۲. تعداد هسته‌های پردازشی را ۲، ۴، ۸ و ۱۶ و تعداد درخواست‌های آپریودیک را ۰ لحاظ کنید. برای این سه حالت خروجی زیر را بدست آورید:
   - نمودار زمان‌بندی‌پذیری سامانه به ازای بهره‌وری وظایف پریودیک (به ازای هر هسته) ۰.۲۵، ۰.۵، ۰.۶ و ۰.۷ نمایش دهید.

فازبندی:

۱. فاز اول: تولید وظایف پریودیک با الگوریتم UUnifast، نگاشت آن‌ها بر روی ۴ هسته پردازشی با استفاده از BFD و زمان‌بندی آن‌ها با استفاده از الگوریتم EDF بر روی هر هسته. باید زمان‌بندی صورت گرفته برای هر هسته مشخص باشد.
۲. فاز دوم: پیاده‌سازی تمام موارد خواسته شده، بدست آوردن تمام خروجی‌های گفته شده و آماده کردن گزارش نهایی

[8] Best-fit decreasing

پروژه شماره ۴ **درس:** سیستم‌های بی‌درنگ

عنوان: پیاده‌سازی الگوریتم زمان‌بندی وظایف وابسته بهم در سیستم چندهسته‌ای ناهمگن **دستیار آموزشی:** آقای نامجو

در یک کارخانه، از سیستم‌های چندهسته‌ای ناهمگن برای کنترل فرآیندهای مختلف کارخانه استفاده می‌کنیم. از آن‌جایی که زمان‌بندی انجام کارها به منظور انجام درست فرآیندها در این کارخانه ضروری هستند، نیازمند یافتن الگوریتم‌های مناسبی برای زمان‌بندی وظایف در این کارخانه هستیم. نکته مهمی که وجود دارد این است که انجام مراحل مختلف کار به صورت ترتیبی بوده و هر وظیفه، وابستگی‌هایی به صورت یک گراف جهت‌دار بدون دور (DAG) به وظایف دیگر دارد.

در این پروژه می‌بایست با پیاده‌سازی الگوریتم PEFT که به کمک Optimistic Cost Table انجام می‌شود، زمان‌بندی دسته وظایفی که به صورت یک DAG ارائه می‌شوند را روی یک سیستم چندهسته‌ای ناهمگن انجام بدهید. الگوریتم‌های مختلفی برای تولید DAG وجود دارد، اما توصیه می‌شود با جست‌وجوی DAG Generation Program، الگوریتم نوشته شده توسط Suter را مطالعه کنید و سعی کنید با پیاده‌سازی آن به زبان پایتون، فرآیند ساخت DAG های خود را انجام بدهید.

بدین منظور، براساس مطالعه این الگوریتم، گراف‌هایی با $n = [10, 50,100,200,400,800]$ و $fat = [0.1,0.4,0.8]$ و $regularity = [0.2,0.5,0.8]$ و $density = [0.2,0.5,0.8]$ تولید کنید.

تعداد پردازنده‌ها را $[2,4,8,16]$ در نظر بگیرید. بازه زمانی اجرای هر تسک را به صورت یکنواخت در بازه $[10,100]$ و هزینه ارتباطی بین هسته‌ها را $[5,25]$ در نظر بگیرید. توجه کنید که مسئله ناهمگن بودن هسته‌ها را حتما مد نظر قرار بدهید. زمان اجرای وظایف روی هسته‌های مختلف باید متفاوت باشد.

در نهایت باید با پیاده‌سازی الگوریتم PEFT، زمان‌بندی وظایف را در همه این حالات مختلف انجام بدهید.

خروجی‌های مورد انتظار:

- یک کد به زبان Python که توانایی تولید DAG های مختلف را داشته باشد.
- یک کد به هر زبان دلخواه، که با گرفتن DAG ها به عنوان ورودی و تعداد هسته‌ها، عملیات زمان‌بندی را انجام بدهید.
- نمودار زمان‌بندی (گانت چارت) گراف با اندازه ۱۰ به شکل مصور
- نمودار Makespan نهایی سیستم به ازای هر کدام از متغیرهای برنامه. توجه کنید که به ازای هر کدام از تعداد پردازنده‌ها، باید چهار نمودار جدا که محور افقی در هر کدام یکی از متغیرهای n, fat, regularity, density است باشد.
- در صورت پیاده‌سازی تکنیک‌های تحمل‌پذیری اشکال، به ازای نرخ اشکال 0.001 و 0.0001، باید نمودار Makespan های بخش قبل به ازای همه گراف‌ها طراحی شود.

فاز ۱:

- پیاده‌سازی الگوریتم تولید وظایف و تولید سه گراف مختلف با آن
- گزارشی مکتوب در حد دو صفحه، که صفحه اول به توضیح الگوریتم تولید وظایف پرداخته و صفحه دوم در مورد نحوه کارکرد الگوریتم PEFT توضیح داده شده باشد.

فاز ۲:

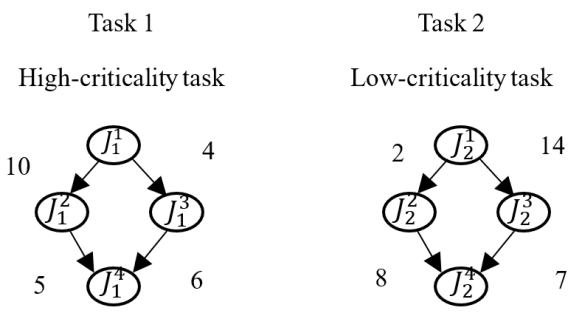
- پیاده‌سازی الگوریتم PEFT
- تولید تمامی خروجی‌های خواسته شده
- گزارش نهایی پروژه

پروژه شماره ۵ **درس:** سیستم‌های بی‌درنگ

عنوان: زمان‌بندی وظایف بحرانی-مختلط مربوط به یک خودرو خودران با استفاده از الگوریتم Criticality EDF و پروتکل ACP **دستیار آموزشی:** خانم جعفری

خودروهای خودران نیاز دارند که داده‌های ارسال شده از طرف حسگر (sensor) را پردازش کنند و بر اساس آن داده‌ها، تصمیم‌گیری سریع و به موقعی را انجام دهند. از آنجایی که در یک خوردو خودران مجموعه‌ای از وظایف بحرانی در کنار وظایف با درجه بحرانی پایین با هم به فعالیت می‌پردازند، بنابراین سامانه مورد استفاده در این وسایل خودران، یک سامانه بحرانی-مختلط می‌باشد.

این پروژه سامانه بحرانی-مختلط تک هسته‌ای است و شامل وظایف LC و HC می‌باشد که هر وظیفه در قالب یک گراف، همانند **Error! Reference source not found.** نمایش داده می‌شود. وظایف LC دارای یک بدترین زمان اجرا هستند در حالی که وظایف HC دارای دو بدترین زمان اجرا هستند. سامانه در حالت نرمال شروع به کار می‌کند و در صورتی که یکی از وظایف HC نتواند کار خود را تا پایان زمان اجرای کوچک خود به پایان برساند، سامانه وارد حالت overrun می‌شود که در این حالت وظایف HC با زمان اجرای بزرگ خود اجرا می‌شوند و وظایف LC، drop می‌شوند و تا زمانی که اجرای وظایف HC به پایان نرسند اجازه اجرا ندارند. شما به عنوان یک مهندس سامانه‌های بی‌درنگ باید وظایف مورد نظر را تحت پروتکل ACP (absolute-time ceiling protocol) و توسط الگوریتم زمان‌بندی Criticality EDF زمان‌بندی کنید. به این صورت که در هنگام زمان‌بندی، وظایف HC اولویت بالاتری نسبت به وظایف LC دارند. در راستای این پیاده‌سازی منابع را به گونه ای به وظایف اختصاص دهید که دسترسی سریالی (non-nested) به منابع وجود داشته باشد.۱- هر DAG را با استفاده از الگوریتم Fast Fourier Transformation به دست بیاورید.
۲- سطح بحرانی هر وظیفه (task) را به صورت تصادفی تولید کنید.



شکل ۱، نمایی از گراف مورد نظر

خروجی‌های مورد نیاز:

- **رسم نمودار برای حالتی که همه وظایف در حالت نرمال باشند:**
  - نمودار زمان‌بند پذیری وظایف با speedup factor = 1 برای حالتی که نسبت وظایف HC به LC برابر یک باشد.
  - نمودار کیفیت خدمات برای حالتی که نسبت وظایف HC به LC برابر یک باشد.
  - نمودار زمان‌بند پذیری وظایف به ازای speedup factorهای ۱.۵، ۱.۶، ۱.۷، ۲.
- **رسم نمودار برای حالتی که ۳۰ درصد از وظایف دچار overrun شوند:**
  - نمودار زمان‌بند پذیری وظایف با speedup factor = 1 برای حالتی که نسبت وظایف HC به LC برابر یک باشد.
  - نمودار کیفیت خدمات برای حالتی که نسبت وظایف HC به LC برابر یک باشد.
  - نمودار زمان‌بند پذیری وظایف به ازای speedup factorهای ۱.۵، ۱.۶، ۱.۷، ۲.

فاز اول: تولید وظایف با استفاده از الگوریتم FFT، پیاده‌سازی الگوریتم پیشنهادی برای تولید و نگاشت منابع، تخصیص بخش بحرانی (critical section) به وظایف

فاز دوم: زمان‌بندی وظایف بر روی یک هسته پردازشی و گزارش تمامی نمودارها

پروژه شماره ۶ **درس:** سیستم‌های بی‌درنگ

عنوان: زمان‌بندی وظایف با استفاده از الگوریتم GSN-EDF و پروتکل FMLP **دستیار آموزشی:** آقای مهدی زاده

یکی از پروتکل‌های ارائه شـده برای مدیریت منابع مشـترک، پروتکل FMLP یا flexible multiprocessor locking protocol اسـت. در این پروتکل، وظایف می‌توانند دسترسی تودرتو یا nested به منابع نیز داشته باشند. هم‌چنین برای زمان‌بندی وظایف از الگوریتم تحت عنوان GSN-EDF استفاده شده است که گسترشی بر الگوریتم G-EDF می‌باشد. در این پروژه باید این پروتکل را به همراه الگوریتم زمان‌بندی GSN-EDF برای یک پردازنده ۴ هسته‌ای پیاده‌سازی کنید. برای تولید وظایف نیز از الگوریتم Uunifast استفاده کنید. برای شبیه‌سازی دسترسی‌های تودرتو، پارامتری تحت عنوان Nesting factor معرفی شده است که احتمال داشتن ۱ یا ۲ دسترسی تودرتو را در وظایف مشخص می‌کند و عددی بین ۰ و ۰.۱ است.

خروجی‌های مورد نیاز:

- نمودار زمان‌بندی‌پذیری با بهره‌وری ۰.۵ و ۰.۷۵ با Nesting factor برابر با ۰.۰۵ و ۰.۰۸ با وجود ۱۰۰ وظیفه و تعداد کل منابع برابر با ۱۰ و تعداد بخش‌های بحرانی هر وظیفه عدد تصادفی بین ۰ تا ۸.

- نمودار کیفیت خدمات با بهره‌وری ۰.۵ و ۰.۷۵ با Nesting factor برابر با ۰.۰۵ و ۰.۰۸ با وجود ۱۰۰ وظیفه و تعداد کل منابع برابر با ۱۰ و تعداد بخش‌های بحرانی هر وظیفه عدد تصادفی بین ۰ تا ۸.

فاز اول: پیاده‌سازی الگوریتم FMLP و GSN-EDF.

فاز دوم: تولید وظایف با توجه به Nesting factor به علاوه نمودارها و گزارش ارائه شود.

پروژه شماره ۷ **درس:** سیستم‌های بی‌درنگ

عنوان: An Adaptive Real-time Scheduler based of Periodic and Sporadic Tasks **دستیار آموزشی:** آقای احمدی-خانم غیبی

در این پروژه، به برنامه‌ریزی یک زمان‌بند preemptive سازگار با تغییرات می‌پردازیم. به‌طور کلی هدف این پروژه پیاده‌سازی یک سیستم زمان‌بندی انعطاف‌پذیر است که قادر به تنظیم پویای رفتار خود در واکنش به تغییرات محیط سیستم یا بار کاری است، می‌باشد. موارد قابل انتظار برای پیاده‌سازی در این پروژه به صورت زیر خواهند بود:

به‌طور کلی سیستم وظایف Sporadic را دریافت می‌کند که برای هرکدام از آن‌ها پارامترهای زمان اجرا(e)، زمان خاتمه(d) و حداقل فاصله‌ی میان دو مرحله ورود متوالی یک وظیفه (p) تعریف می‌شود. در طول اجرای این وظایف بر اساس APIهای سیستم، تغییراتی ممکن است اتفاق بیفتد که شامل تغییر deadline وظایف، تغییر اولویت یکی از وظایف برای مدت t ثانیه و تغییر نرخ ورود تسک‌های sporadic به سیستم می‌باشد.

فرض کنید که وظایف Sporadic می‌توانند به‌اندازه‌ی زمان T مانع از اجرای یک وظیفه‌ی با اولویت بالاتر از خود شوند. این زمان‌بند باید روش‌هایی برای تعیین میزان این زمان T برای هر وظیفه داشته باشد.

برای تعیین پارامتر T و زمان‌بندی نیاز است تا علاوه بر نظارت کردن (Profiling) بر تغییرات گفته‌شده روی سیستم، از روش‌های پیش‌بینی تغییرات نرخ ورود وظایف sporadic با استفاده از heuristicهای مختلف یا مدل‌های ML استفاده شود تا مقدار مناسب T برای هرکدام از وظایف دوره‌ای مشخص شود.

این پارامتر باید به دو صورت متفاوت برای هر وظیفه و به صورت یکسان برای کل سیستم (یک مقدار T برای کل سیستم) تنظیم شوند و نمودار نتایج این دو مورد مقایسه شوند.

پیاده‌سازی سیستم شما باید به صورت ماژولار و قابل آزمایش از سیستم خارجی یا تستر از طریق رابط‌های ساده API باشد.

ورودی‌ها:

- مجموعه‌ی وظایف دوره‌ای با دوره‌ی تناوب، زمان اجرا و ددلاین آن‌ها
- کل زمان اجرا و اطلاعات وظایف Sporadic و زمان ورود آن‌ها (زمان ورود این وظایف باید به‌صورت runtime مورد استفاده قرار گیرد و در زمان‌بندی اولیه‌ی شما نقشی نداشته باشد)

خروجی‌ها:

- زمان‌بندی دو تست شامل مجموعه وظایف Periodic و Sporadic، هرکدام در دو حالت T مجزا برای هر وظیفه و T یکسان برای کل سیستم نشان داده شوند.
- یک تست داده‌شده توسط دستیاران آموزشی شامل مجموعه وظایف Periodic و Sporadic
- ۳ نمودار تغییرات ممکن بر روی سیستم در زمان اجرا
- نمودار اختلاف نرخ ورود وظایف sporadic پیش‌بینی‌شده (به‌وسیله‌ی روش پیش‌بینی انتخابی طبق موارد بالا) با مقدار واقعی پس از اجرای وظایف (ارزیابی کیفیت پیش‌بینی شما با این نمودار انجام خواهد شد)

فاز اول:

- نمودار زمان‌بندی وظایف Periodic و Sporadic
- نظارت و ذخیره‌ی تغییرات روی سیستم

فاز دوم:

- پیاده‌سازی روش‌های heuristic یا مبتنی بر مدل‌های ML برای پیش‌بینی مقدار T
- تمامی نمودارها و خروجی‌های مورد نیاز

پروژه شماره ۸ **درس:** سیستم‌های بی‌درنگ

عنوان: سامانه هدایت پهپاد **دستیار آموزشی:** آقای حسینی

در این پروژه می‌خواهیم سامانه زمان‌بندی وظایف هدایت خودکار یک پهپاد را طراحی کنیم. این سامانه چندهسته‌ای، دو نوع وظیفه دارد. وظایف با درجه بحرانی بالا (HC) مثل وظایف مربوط به کنترل پرواز و تعادل، که اهمیت زیادی دارند و اگر که در موعد خود اجرا نشوند، ممکن است پهپاد تصادف یا سقوط کند و وظایف با درجه بحرانی پایین (LC) که اجرا شدن آنها بعد از موعد، همچنان مفید است اما کیفیت خدمات سامانه را کاهش می‌دهد. مثل ارسال گزارش برای کاربر. از آنجا که وظایف HC اهمیت بسیار زیادی دارند، برای اطمینان از درستی پاسخ آن‌ها، از روش تحمل‌پذیری اشکال رای اکثریت TMR استفاده می‌کنیم. برخی از وظایف هم ممکن است با خطا مواجه شوند؛ در این صورت باید آن وظایف را دوباره اجرا کرد.

شما باید با استفاده از الگوریتم UUnifast، مجموعه وظایفی برای این سامانه تولید کنید و آن را با الگوریتم Worst fit Deceasing (WFD) و First Fit Decreasing (FFD) روی پردازنده‌ها نگاشت کنید. همچنین باید تست زمان‌بندی‌پذیری را بر روی سامانه اجرا کنید و در صورتی که زمان‌بندی قابل انجام نباشد، پیغام خطا چاپ کنید. سپس زمان‌بندی کل سامانه باید با الگوریتم ER-EDF انجام شود. در هنگام زمان‌بندی، ممکن است که تعدادی از هسته‌ها وارد حالت سرریز (overrun) شوند. این حالت تا زمانی که به اتمام Hyper period برسیم ادامه خواهد داشت. در این سامانه، از سربار preempt کردن صرف نظر نمی‌شود و برای کاهش این هزینه‌ی سربار، از cooperative scheduling استفاده می‌کنیم. وظیفه‌ی ما این است که بازه‌های مناسب برای درج preemption pointها را به دست آوریم به طوری‌که به سامانه‌ای برسیم که بهترین زمان‌بندی‌پذیری و بیشترین کیفیت خدمات را داشته باشد. (فرض کنید که فاصله بین هر preemption point تا نقطه‌ی بعدش یکسان است).

در این سامانه می‌توانید دوره‌ی وظایف را از بین اعداد [۱۰، ۲۰، ۵۰، ۱۰۰، ۲۰۰، ۵۰۰] انتخاب کنید.

خروجی‌های مورد نیاز:

الف) سامانه ۴ هسته‌ای با بهره‌وری ۵۰ درصد برای هر هسته، که بین وظایف LC و HC به مساوات تقسیم می‌شود.

ب) سامانه ۴ هسته‌ای با بهره‌وری ۸۰ درصد برای هر هسته، که بین وظایف LC و HC به مساوات تقسیم می‌شود.

ج) سامانه ۴ هسته‌ای با بهره‌وری ۷۵ درصد برای هر هسته، که بین وظایف LC و HC به مساوات تقسیم می‌شود. در این سامانه ۲ تا از هسته‌ها وارد حالت سرریز خواهند شد.

د) سامانه ۴ هسته‌ای با بهره‌وری ۷۵ درصد برای هر هسته، که بین وظایف LC و HC به مساوات تقسیم می‌شود. در این سامانه ۲۰٪ کل وظایف با اشکال مواجه می‌شوند.

ه) سامانه ۸ هسته‌ای با بهره‌وری ۸۰ درصد برای هر هسته، که بین وظایف LC و HC به مساوات تقسیم می‌شود. در این سامانه ۲۰٪ کل وظایف با اشکال مواجه می‌شوند و ۴ تا از هسته‌ها هم وارد حالت سرریز می‌شوند.

فاز اول: نتایج تولید وظایف، نگاشت وظایف به هسته‌ها، و تست زمان‌بندی را به ازای هر یک از سامانه‌های زیر ارائه دهید.

فاز دوم: به ازای هر کدام از سامانه‌های گفته شده در بالا، نتیجه‌ی زمان‌بندی را ارائه دهید. همچنین به ازای طول‌های مختلف فاصله بین preemption pointها، نمودار کیفیت خدمات و نمودار زمان‌بندی‌پذیری را رسم کنید.

پروژه شماره ۹ **درس:** سیستم‌های بی‌درنگ

عنوان: زمان‌بندی و نگاشت آگاه به انسداد وظایف بحرانی-مختلط در سامانه‌های چندهسته‌ای **دستیار آموزشی:** خانم ملکی

در یک خودرو مدرن، واحدهای مختلفی مانند واحد کنترل موتور (سطح بحرانی بالا) و سیستم اطلاعات-سرگرمی (سطح بحرانی پایین) وجود دارد. این سامانه‌ها باید منابع را به اشتراک بگذارند و همزمان با اطمینان از ایمنی عملکردهای حیاتی، عمل کنند. درنتیجه، سامانه بحرانی-مختلط دوسطحی نوعی از سامانه‌های بحرانی-مختلط است که فقط دو سطح بحرانی دارد و شامل وظایف دوره‌ای LC و HC می‌باشد که وظایف LC دارای یک بدترین زمان اجرا و وظایف HC دارای دو بدترین زمان اجرا هستند. سامانه در حالت عادی (Normal) شروع به کار می‌کند و در صورتی که یکی از وظایف HC زمان اجرای کوچک خود را رد کند و هنوز کامل نشده باشد سامانه وارد حالت سرریز (Overrun) می‌شود.

شما به‌عنوان مهندس طراح این سامانه، باید وظایف موردنظر را توسط یک الگوریتم ابتکاری به هسته‌ها نگاشت کنید و تحت پروتکل Multi-processor Stack Resource Policy (MSRP) و توسط الگوریتم زمان‌بندی EDF-VD، وظایف را زمان‌بندی کنید. این الگوریتم ابتکاری از دو پارامتر **ازدحام منابع** و **بهره‌وری هسته‌ها** برای نگاشت استفاده می‌کند. ازدحام منابع به معنی میزان بهره‌وری هر وظیفه در هنگام دسترسی به یک منبع خاص است. برای نگاشت هر وظیفه، تمام منابعی که به آنها دسترسی دارد در نظر گرفته می‌شود و ازدحام این منابع بر روی هر هسته محاسبه می‌شود. اولویت انتخاب با هسته‌ای است که ازدحام همه منابع روی آن بیشترین باشد، در غیر این صورت هسته‌ای انتخاب می‌شود که بیشترین ازدحام را برای یک یا چند منبع داشته باشد (باید وظایف براساس پارامترهای ازدحام در یک صف مرتب شوند و همچنین اگر بهره‌وری باقی‌مانده از هسته کمتر از بهره‌وری وظیفه در حال پردازش باشد، هسته بعدی در صف انتخاب می‌شود). حال اگر شرایط برای دو یا چند هسته یکسان بود، هسته‌ای انتخاب می‌شود که بیشترین بهره‌وری روی آن قرار گرفته باشد (WFD). برای ارزیابی این روش، شما باید به کمک الگوریتم Uunifast مجموعه‌ای از وظایف مصنوعی تولید کنید و پارامترهای خواسته شده زیر را به کمک هر دو **الگوریتم نگاشت ارائه شده** و **WFD مقایسه** کنید و نتایج را ثبت و گزارش کنید.

۱. **نمودار اول: زمانبندپذیری در هر دو حالت عملیاتی سامانه.**
   - ✓ میانگین زمان‌بندپذیری ۴۰۰ وظیفه با نسبت تعداد وظایف HC به LC برابر با یک در ۱۰ بار اجرا (وظایف متفاوت در هر بار اجرا) ، انتخاب تعداد منابع به صورت تصادفی از بازه [۶-۲]، تعداد هسته برابر با ۲ و بهره‌وری ۰.۲۵ به ازای هر هسته.
   - ✓ میانگین زمان‌بندپذیری ۴۰۰ وظیفه با نسبت تعداد وظایف HC به LC برابر با یک در ۱۰ بار اجرا (وظایف متفاوت در هر بار اجرا) ، انتخاب تعداد منابع به صورت تصادفی از بازه [۶-۲]، تعداد هسته برابر با ۴ و بهره‌وری ۰.۲۵ به ازای هر هسته.
   - ✓ میانگین زمان‌بندپذیری ۴۰۰ وظیفه با نسبت تعداد وظایف HC به LC برابر با یک در ۱۰ بار اجرا (وظایف متفاوت در هر بار اجرا) ، انتخاب تعداد منابع به صورت تصادفی از بازه [۶-۲]، تعداد هسته برابر با ۸ و بهره‌وری ۰.۲۵ به ازای هر هسته.
۲. **نمودار دوم: قابلیت نگاشت برای بهره‌وری ۰.۲۵ و ۰.۵ برای هر حالت.**
   - ✓ میانگین قابلیت نگاشت ۴۰۰ وظیفه با نسبت تعداد وظایف HC به LC برابر با یک در ۱۰ بار تولید وظایف (وظایف متفاوت در هر بار اجرا) ، انتخاب تعداد منابع به صورت تصادفی از بازه [۶-۲] و تعداد هسته برابر با ۲.
   - ✓ میانگین قابلیت نگاشت ۴۰۰ وظیفه با نسبت تعداد وظایف HC به LC برابر با یک در ۱۰ بار تولید وظایف (وظایف متفاوت در هر بار اجرا) ، انتخاب تعداد منابع به صورت تصادفی از بازه [۶-۲] و تعداد هسته برابر با ۴.
   - ✓ میانگین قابلیت نگاشت ۴۰۰ وظیفه با نسبت تعداد وظایف HC به LC برابر با یک در ۱۰ بار تولید وظایف (وظایف متفاوت در هر بار اجرا) ، انتخاب تعداد منابع به صورت تصادفی از بازه [۶-۲] و تعداد هسته برابر با ۸.

فازبندی پروژه:

فاز اول: در فاز اول **پیاده‌سازی الگوریتم تولید وظایف (به همراه گزارش وظایف تولید شده)، تولید و نگاشت منابع، تخصیص بخش بحرانی به وظایف، تعیین سطوح قبضگی تمام وظایف (براساس پروتکل SRP)** و **نمودار های قابلیت نگاشت** به عنوان نتایج خروجی گزارش شوند.

فاز دوم: در فاز نهایی، **زمان‌بندی** به صورت کامل **پیاده‌سازی** شود تمام **نمودارهای خواسته شده** گزارش شوند.

پروژه شماره ۱۰ **درس:** سیستم‌های بی‌درنگ

عنوان: الگوریتم زمان‌بندی برای وظایف واقعی در سامانه‌ی بی‌درنگ نرم و سخت **دستیار آموزشی:** آقای اثنی عشری

در این پروژه، به زمان‌بندی وظایف متناوب نرم و سخت[۹] در سامانه‌های چندهسته‌ای همگن می‌پردازیم. برای ارائه یک تجربه عملی، از بسته‌محک[۱۰] PARSEC استفاده می‌شود. بسته‌محک PARSEC را بر روی gem 5 اجرا کرده و پردازنده‌ی X86 را مدنظر قرار دهید. سپس وظایف فوق را با استفاده از سیاست زمان‌بندی BFD[11] بر روی یک سامانه‌ی ۱۶ هسته‌ای نگاشت کنید. در انتها وظایف نرم و سخت را به چهار روش زیر زمان‌بندی کرده و نتایج به‌دست‌آمده از هرکدام را با یکدیگر مقایسه کنید:

روش اول) تمامی وظایف نرم و سخت با سیاست نزدیک‌ترین موعد زمانی نخست[۱۲] زمان‌بندی شوند.

روش دوم) تمامی وظایف نرم و سخت با سیاست دورترین موعد زمانی نخست[۱۳] زمان‌بندی شوند.

روش سوم) وظایف نرم با سیاست EDF و وظایف سخت با سیاست LDF زمان‌بندی شوند.

روش چهارم) وظایف نرم با سیاست LDF و وظایف سخت با سیاست EDF زمان‌بندی شوند.

در انتها باید خروجی‌های زیر را بر روی سامانه‌ی ۸ و ۱۶ هسته‌ای همگن زمانی که میزان بهره‌وری هر هسته ۰.۲۵، ۰.۵ است، ارائه کنید:

- نمودار میزان کیفیت خدمات وظایف در هر چهار روش
- نمودار زمان‌بندی‌پذیری وظایف در هر چهار روش
- جدول وظایف و مشخصات آن‌ها

فاز اول پروژه:

اجرای بسته‌محک PARSEC بر روی gem 5 و تولید جدول وظایف و مشخصات آن‌ها.

فاز دوم پروژه:

پیاده‌سازی تمامی موارد گفته‌شده، تولید تمام خروجی‌های خواسته‌شده و تهیه گزارش پایانی.

[9] Soft and Hard periodic Real-Time Tasks
[10] Benchmark
[11] Best Fit Decreasing
[12] Earliest Deadline First (EDF)
[13] Latest Deadline First (LDF)

پروژه شماره ۱۱ **درس:** سیستم‌های بی‌درنگ

عنوان: زمان بندی گراف‌های وظایف در یک سامانه ابری **دستیار آموزشی:** آقای طوقانی

در یک سامانه ابری، جریان‌های کاری داریم که باید به موقع انجام گیرند. هر جریان کاری را به صورت مجموعه از وظایف نمایش می‌دهیم که این وظایف به یکدیگر وابستگی دارند، این وابستگی را با گراف وظایف مدل‌سازی می‌کنیم. هدف از این پروژه ارائه یک زمان‌بند برای چنین سامانه‌ای می‌باشد به گونه‌ای تمام جریان‌های کاری به موقع انجام شوند.

محدودیت‌های پیاده‌سازی:

- سامانه چند هسته‌ای با واحدهای پردازشی ناهمگن است که شامل چندین جریان کاری پریودیک است.
- هر جریان کاری به صورت یک گراف وظایف (DAG) مدل می‌شود و دارای چندین وظیفه به هم وابسته است.
- هر جریان کاری، مهلت زمانی و دوره دارد که برای سادگی دوره را برابر مهلت زمانی آن در نظر بگیرید.
- برای هر گراف با استفاده از الگوریتم HEFT زمان‌بندی را انجام دهید. زمان اتمام این تک گراف با الگوریتم HEFT را makespan آن گراف در نظر بگیرید. سپس، براساس زمان‌بندی انجام شده مهلت زمانی هر وظیفه را بدست آورید. مهلت زمانی هر وظیفه برابر است با زمان اتمام آن وظیفه با رویکرد HEFT به علاوه فاصله مهلت زمانی کل گراف و makespan گراف.
- با رویکرد BFS[۱۴] وظایف را بر روی هسته‌های پردازشی نگاشت کرده و برای هر هسته با الگوریتم EDF (به صورت قبضه‌ای) زمان‌بندی را انجام دهید. توجه کنید که شما نباید یک وظیفه را دیرتر از زمان اتمام وظیفه پدرش (در گراف) اجرا کنید. همچنین بخشی از وظیفه پدر را نیز نمی‌توانید قبضه کرده و جلوتر از وظیفه فرزند آن اجرا کنید.

محدودیت‌های شبیه‌سازی:

- گراف‌ها را با استفاده از دو توپولوژی Gaussian Elimination و Fast Fourie Transform با m=10 ایجاد کنید.
- هزینه ارتباطی بین وظایف را در بازه [5, 50] به صورت تصادفی (با توزیع یکنواخت) انتخاب کنید.
- زمان اجرای هر وظیفه را در بازه [10, 100] به صورت تصادفی (توزیع یکنواخت) انتخاب کنید.
- توجه کنید که سامانه ناهمگن است، بنابراین زمان اجرا و هزینه ارتباطی بسته به پردازنده‌های مختلف، متفاوت است.

خروجی‌ها:

برای ۲، ۴، ۸ و ۱۶ هسته پردازشی، مهلت زمانی هر گراف را برابر مجموع بدترین زمان اجرای وظایف آن لحاظ کنید و برای این چهار حالت خروجی‌های زیر را بدست آورید:

- نمودار زمان‌بندی انجام شده برای حداقل ۴ گراف بدست آورید.
- نمودار زمان‌بندی‌پذیری سامانه برای ۲، ۴، ۸ و ۱۶ گراف بدست آورید.
- نمودار میانگین makespan هر گراف به ازای ، ۴، ۸ و ۱۶ گراف بدست آورید.

فازبندی:

۱. فاز اول: تولید حداقل ۸ گراف با استفاده از توپولوژی‌های FFT و GE، پیاده‌سازی پیمایش گراف BFS و نمایش چگونگی آن. تهیه گزارش از این فاز.
۲. فاز دوم: پیاده‌سازی تمام موارد خواسته شده، بدست آوردن تمام خروجی‌های گفته شده و آماده کردن گزارش نهایی

[14] Breadth-first search

پروژه شماره ۱۲ **درس:** سیستم‌های بی‌درنگ

عنوان: پیاده‌سازی مسئله Job-Shop Scheduling با الگوریتم‌های کلاسیک و یادگیری عمیق **دستیار آموزشی:** آقای نامجو

یکی از مسائل معروف حوزه زمان‌بندی، مسئله Job-Shop Scheduling است. در این مسئله n وظیفه و m ماشین برای انجام آن‌ها وجود دارد. هر وظیفه، از تعدادی عملیات تشکیل شده و هر عملیات توانایی اجرا شدن روی ماشین متفاوتی را دارد. عملیات‌های یک وظیفه نمی‌توانند همزمان انجام بشوند و همچنین یک ماشین تنها توانایی اجرای یک عملیات را در یک زمان مشخص دارد. مسئله بدین صورت است که ترتیبی از انجام عملیات‌ها روی ماشین‌ها ارائه بشود که Makespan را کمینه کند. می‌توان نشان داد که مسئله معروف فروشنده دوره‌گرد حالت خاصی از این این مسئله است و در نتیجه این مسئله NP-Hard است و الگوریتم چندجمله‌ای برای حل بهینه آن وجود ندارد. با این حال روش‌های مختلفی در راستای آن صورت گرفته است.

در فاز اول این پروژه، باید به کمک الگوریتم موسوم به الگوریتم جانسون اقدام به حل این مسئله بکنید. این الگوریتم برای حالتی که تنها دو ماشین داشته باشیم بهینه عمل می‌کند و برای تعداد بیش‌تر نیز به شکلی قابل تعمیم است اما لزوما بهینه عمل نمی‌کند. علاوه بر این در این فاز ، باید با استفاده از الگوریتم‌های هوش مصنوعی، اقدام به حل این مسئله بکنید. حالت مورد انتظار این است که با استفاده از یکی دو الگوریتم ژنتیک و یا Ant Colony این مسئله را حل کنید. انتخاب بین این دو نوع الگوریتم بر عهده خود شماست ولی باید این موضوع را با دستیار آموزشی پروژه نهایی کنید. در فاز دوم می‌بایست با استفاده از روش‌های یادگیری عمیق تقویتی (Deep Reinforcement Learning) به پیاده‌سازی روش حل این مسئله بپردازید.

خروجی‌های مورد انتظار:

- نتیجه زمان‌بندی چندین نمونه وظیفه مختلف با هر دو الگوریتم پیاده شده در فاز اول و فاز دوم
- نمودار میانگین تاخیر و میانگین زمان انتظار به ازای چندین نمونه وظیفه مختلف برای هر یک از الگوریتم‌ها. برای این مورد، به ازای هر الگوریتم باید حداقل ۴ نمودار تولید بشود. یک نمودار به ازای m ثابت و n متغیر برای میانگین تاخیر، یک نمودار به ازای m ثابت و n متغیر برای میانگین زمان انتظار، یک نمودار به ازای n ثابت و m متغیر برای میانگین تاخیر و یک نمودار به ازای n ثابت و m متغیر برای میانگین زمان انتظار
- کدهای هر دو فاز پروژه
- گزارشی از انجام پروژه که در آن به طور دقیق روش مورد استفاده در فاز دوم شرح داده شده باشد.

فاز اول:

- تولید حداقل ۲۰ نمونه وظیفه مختلف برای انجام تست‌ها
- پیاده‌سازی فاز اول پروژه با الگوریتم جانسون
- خروجی‌های گفته شده در قسمت قبل برای الگوریتم جانسون

فاز دوم:

- پیاده‌سازی فاز دوم پروژه با الگوریتم‌های مربوط به هوش مصنوعی
- خروجی‌های گفته شده در قسمت قبل برای این الگوریتم و ترکیب (مقایسه) آن با خروجی‌های فاز اول
- گزارش کامل پروژه

پروژه شماره ۱۳ **درس:** سیستم‌های بی‌درنگ

عنوان: شبیه‌سازی یک معماری ۳ لایه پردازش مهی و حل مسئله با الگوریتم ژنتیک **دستیار آموزشی:** آقای مهدی زاده

فرض کنید که یک شهر هوشمند از یک معماری ۳ لایه برای پردازش وظایف دستگاه‌های کاربری استفاده می‌کند. وظایف در لایه اول ایجاد شده و برای اجرا می‌توانند به لایه بالاتر انتقال یابند. شبیه‌سازی ارائه کنید که ابتدا بتواند این معماری ۳ لایه را پیاده‌سازی کند. شبیه‌ساز باید تعداد دستگاه‌های کاربری و تعداد گره‌های مه را ورودی بگیرد و توپولوژی یا گراف شبکه را به صورت تصادفی ایجاد کند. (هر دستگاه کاربری به حداقل یک گره مهی دسترسی داشته باشد و گره‌های مهی به لایه ابر متصل باشند.) سپس گراف شبکه را تولید کرده و وظایف بتوانند به لایه بالاتر منتقل شوند.

در قدم بعدی باید مسئله انتقال وظایف را طوری حل کنید که یک زمان‌بندی درست (رعایت شدن ددلاین‌ها) برای وظایف ارائه دهد. به این منظور فرض کنید تمام وظایف دارای یک ددلاین سخت هستند. به هنگام ایجاد شدن آن‌ها روی دستگاه‌های کاربری، یک دستگاه می‌تواند تصمیم بگیرد که آن را روی پردازنده خودش اجرا کند یا به لایه بعدی انتقال دهد. (دقت شود که لایه بعدی انتخاب بین گره‌های مهی همسایه‌اش است.) در گره مه نیز دقیقا همین تصمیم گرفته می‌شود و وظیفه یا روی همان گره پردازش می‌شود یا به لایه ابر منتقل می‌شود.

برای اجرای وظایف فرض کنید که اجرا بر روی دستگاه‌های کاربری کندتر از گره‌های مهی است و هم‌چنین قدرت پردازش لایه ابر بی‌نهایت است. هم‌چنین مدت زمانی طول می‌کشد که وظایف از لایه کاربری به لایه مه و لایه ابر منتقل شوند که انتقال از لایه کاربری به مه کمتر از انتقال از مه به ابر طول می‌کشد. فرض کنید که الگوریتم زمان‌بندی دستگاه‌های کاربری و گره‌های مهی EDF است.

با استفاده از الگوریتم ژنتیک، راه‌حلی ارائه دهید که تصمیم انتقال یا ماندن وظایف را به طوری بگیرد که یک زمان‌بندی درست ارائه شود. دقت کنید که وظایف باید طوری به سیستم داده شوند که بدون استفاده از هر ۳ لایه، ارائه یک زمان‌بندی ممکن نباشد و زمان‌بندی ارائه شده توسط الگوریتم ژنتیک نیز باید در انتها از هر ۳ لایه استفاده کند.

در **فاز اول**، معماری ۳ لایه و الگوریتم EDF برای گره‌ها پیاده‌سازی شود. هم‌چنین ۳ روش پایه انتقال وظیفه نیز پیاده‌سازی شود. روش پایه اول، LocalOnly است. در این روش تمام دستگاه‌های کاربری به هنگام ایجاد وظایف انتقالی انجام نمی‌دهند و به صورت محلی اجرا زمان‌بندی می‌کنند. روش پایه دوم، FogOnly است. در این روش تمام وظایف از لایه کاربری به لایه مه انتقال داده می‌شوند و در آن‌جا زمان‌بندی می‌شوند. روش پایه سوم، CloudOnly است. در این روش تمام وظایف به لایه ابر منتقل می‌شود و در آن‌جا پردازش انجام می‌شود.

در **فاز دوم**، با استفاده از الگوریتم ژنتیک، مسئله انتقال وظایف را حل کنید و پارامترهای زمان‌بندی خروجی را با ۳ روش پایه مقایسه کنید. دقت شود که مجموعه وظایف ورودی به شبیه‌ساز باید با روش ژنتیک بهترین نتیجه را بگیرند. هم‌چنین گزارش و نمودارها نیز در این فاز ارائه شود.

نمودارهای کیفیت‌خدمات این ۴ روش را برای ۳ توپولوژی مختلف خروجی دهید.

- ۱۰ دستگاه کاربری و ۵ گره مهی
- ۱۰ دستگاه کاربری و ۱۰ گره مهی
- ۴۰ دستگاه کاربری و ۱۰ گره مهی

پروژه شماره ۱۴

درس: سیستم‌های بی‌درنگ

عنوان: Predicting Optimal Points for Task Splitting in a Limited-Preemption Scheduler **دستیار آموزشی:** آقای احمدی-خانم غیبی

در این پروژه، به پیاده‌سازی یک زمان‌بند limited-preemptive می‌پردازیم. این زمان‌بند باید با نظارت بر وظایف ورودی periodic, aperiodic نقاط بهینه‌ای را برای شکستن وظایف periodic به بخش‌های کوچک‌تر انتخاب کند که در انتهای هر کدام از این وظایف امکان preemption برای وظایف دوره‌ای وجود داشته باشد. موارد قابل انتظار برای پیاده‌سازی در این پروژه به صورت زیر خواهند بود:

- پیاده‌سازی زمان‌بندی وظایف periodic و aperiodic با استفاده از Deferrable Server Scheduling
- ساخت مجموعه وظایف با utilizationهای متفاوت و قابل زمان‌بندی با استفاده از الگوریتم UUnifast
- پیاده‌سازی الگوریتم یافتن نقاط بهینه (با کم‌ترین میزان miss-rate) برای شکستن وظایف دوره‌ای به بخش‌های کوچک‌تر در نقاط تعیین‌شده به‌گونه‌ای که بخش‌های کوچک‌تر به ددلاین خود برسند.
- پارامترهایی مانند ارتباط میان تسک‌ها و سربار حاصل از وقفه در انجام وظایف باید در انتخاب نقاط بهینه در نظر گرفته شود. گراف ارتباط میان تسک‌ها به صورت زیر به شما داده خواهد شد (در گراف زیر انجام‌شدن تسک ۲ وابسته به انجام‌شدن تسک ۱ می‌باشد):

t1->t2, t2->t3, …

- پیاده‌سازی سیستم شما باید به صورت ماژولار و قابل آزمایش از سیستم خارجی یا تستر از طریق رابط‌های ساده API باشد.

ورودی‌ها:

- مجموعه‌ی وظایف دوره‌ای با دوره‌ی تناوب، زمان اجرا و ددلاین آن‌ها
- وابستگی‌های میان وظایف دوره‌ای
- کل زمان اجرا و زمان‌های ورود وظایف aperiodic (زمان ورود این وظایف باید به‌صورت runtime مورد استفاده قرار گیرد و در زمان‌بندی اولیه‌ی شما نقشی نداشته باشد)

خروجی‌ها:

- زمان‌بندی چهار تست شامل مجموعه وظایف Periodic و Aperiodic با مجموعه وظایف قابل زمان‌بندی ساخته‌شده توسط الگوریتم UUnifast با utilizationهای ۰.۵,۰.۶,۰.۷,۰.۸
- یک تست داده‌شده توسط دستیاران آموزشی شامل مجموعه وظایف Periodic و Sporadic
- ۲ نمودار میزان تغییر slack time و miss rate به‌ازای تغییرات در نقاط شکسته‌شدن در در تسک‌ها با utilization = ۰.۷,۰.۸

فاز اول:

- نمودارهای زمان‌بندی همزمان وظایف periodic و aperiodic

فاز دوم:

- تمامی نمودارها و خروجی‌های مورد نیاز
- پیاده‌سازی الگوریتم پیداکردن نقاط بهینه

پروژه شماره ۱۵ **درس:** سیستم‌های بی‌درنگ

عنوان: زمان‌بندی وظایف بی‌درنگ سخت در سامانه‌های چندهسته‌ای به کمک پروتکل MrsP **دستیار آموزشی:** خانم ملکی

سیستم های کنترل شبکه برق، عملکردهای حیاتی مانند متعادل سازی بار و تشخیص عیب را مدیریت می کنند. این سامانه‌ها دارای وظایف دوره‌ای بی‌درنگ سخت هستند که از منابع مشترکی استفاده می‌کنند و نباید موعد زمانی خود را نقض کنند. شما به‌عنوان مهندس طراح این سامانه، باید وظایف موردنظر را تحت دو پروتکل Multiprocessor resource sharing Protocol (MrsP) و Multi-processor Stack Resource Policy (MSRP) و توسط الگوریتم زمان‌بندی EDF زمان‌بندی کنید و همچنین برای نگاشت وظایف نیز از الگوریتم WFD استفاده کنید. تفاوت پروتکل MrsP با پروتکل MSRP در استفاده از زمان‌های انتظار مشغول برای اجرای دیگر وظایف موجود در هسته‌ای است که دچار انتظار مشغول شده است. در پروتکل MSRP، زمانی که یک وظیفه دچار انسداد از راه دور می‌شود، به دلیل انتظار مشغول هسته، هیچ وظیفه دیگری نمی‌تواند روی هسته اجرا شود. در پروتکل MrsP شما باید براساس اولویت و محدودیت‌های موجود، یک یا چند وظیفه را برای اجرا در این بازه زمانی انتخاب کنید به گونه ای که پس از پایان زمان انتظار مشغول، وظیفه اصلی بدون هیچ وقفه‌ای به ادامه‌ی اجرای خود بپردازد. برای ارزیابی این دو روش، شما باید به کمک الگوریتم Uunifast مجموعه‌ای از وظایف مصنوعی تولید کنید و پارامترهای خواسته شده زیر را به کمک هر دو پروتکل MSRP و MrsP مقایسه کنید و نتایج را ثبت و گزارش کنید.

خروجی‌های مورد نیاز:

نمودار زمان‌بندپذیری به ازای هر دوحالت: ۱)تعداد منابع برابر با تعداد هسته در هر حالت، ۲) تعداد منابع به صورت تصادفی از بازه [۶-۲]

۱. نمودار اول:
- ✓ میانگین زمان‌بندپذیری ۴۰۰ وظیفه در ۱۰ بار اجرا (وظایف متفاوت در هر بار اجرا) ، تعداد هسته برابر با ۲ و بهره‌وری ۰.۲۵ به ازای هر هسته.
- ✓ میانگین زمان‌بندپذیری ۴۰۰ وظیفه در ۱۰ بار اجرا (وظایف متفاوت در هر بار اجرا) ، تعداد هسته برابر با ۴ و بهره‌وری ۰.۲۵ به ازای هر هسته.
- ✓ میانگین زمان‌بندپذیری ۴۰۰ وظیفه در ۱۰ بار اجرا (وظایف متفاوت در هر بار اجرا) ، تعداد هسته برابر با ۸ و بهره‌وری ۰.۲۵ به ازای هر هسته.

۲. نمودار دوم:
- ✓ میانگین زمان‌بندپذیری ۴۰۰ وظیفه در ۱۰ بار اجرا (وظایف متفاوت در هر بار اجرا) ، تعداد هسته برابر با ۲ و بهره‌وری ۰.۵ به ازای هر هسته.
- ✓ میانگین زمان‌بندپذیری ۴۰۰ وظیفه در ۱۰ بار اجرا (وظایف متفاوت در هر بار اجرا) ، تعداد هسته برابر با ۴ و بهره‌وری ۰.۵ به ازای هر هسته.
- ✓ میانگین زمان‌بندپذیری ۴۰۰ وظیفه در ۱۰ بار اجرا (وظایف متفاوت در هر بار اجرا) ، تعداد هسته برابر با ۸ و بهره‌وری ۰.۵ به ازای هر هسته.

فازبندی پروژه:

فاز اول: در فاز اول **پیاده‌سازی الگوریتم تولید وظایف (به همراه گزارش وظایف تولید شده)، تولید و نگاشت منابع، تخصیص بخش بحرانی به وظایف، تعیین سطوح قبضگی تمام وظایف (براساس پروتکل SRP)** و **نگاشت وظایف به هسته‌ها** گزارش شوند.

فاز دوم: در فاز نهایی، **زمان‌بندی** به صورت کامل **پیاده‌سازی** شود و تمام **نمودارهای خواسته شده** گزارش شوند.

پروژه شماره ۱۶ **درس:** سیستم‌های بی‌درنگ

عنوان: زمان‌بندی وظایف در یک سامانه چند هسته‌ای مه **دستیار آموزشی:** آقای طوقانی

در این پروژه به زمان‌بندی وظایف برای یک سامانه چند هسته‌ای مه می‌پردازیم. در این سامانه وظایف به یکدیگر وابستگی دارند و با گراف وظایف مدل‌سازی می‌شوند. هدف زمان‌بندی این وظایف به گونه‌ای است که میزان بهره‌وری هسته‌های پردازشی کاهش یابد و تمام وظایف در مهلت زمانی خود اجرا شوند.

محدودیت‌های پیاده‌سازی:

- سامانه چند هسته‌ای با واحدهای پردازشی ناهمگن است.
- وظایف با یکدیگر وابستگی دارند و به صورت چندین گراف وظیفه (DAG) مدل‌سازی می‌شوند که این گراف‌ها مبتنی بر قاب هستند.
- وظایف را ابتدا با الگوریتم ILP[15] بر روی هسته‌ها نگاشت کنید به گونه‌ای که دو هدف کاهش بهره‌وری و کاهش هزینه عملیاتی مدنظر قرار گیرد. هزینه عملیاتی عبارت است از تعداد دستورالعمل‌های اجرا شده بر روی هسته‌های پردازشی برای اجرای وظایف. سپس با استفاده از الگوریتم ژنتیک به زمان‌بندی این وظایف بر روی هر هسته بپردازید. شما باید هنگام زمان‌بندی وابستگی بین وظایف و مهلت زمانی کل سامانه را در نظر بگیرید.

محدودیت‌های شبیه‌سازی:

- مهلت زمانی کل گراف‌ها ۱۰۰۰ میلی ثانیه در نظر بگیرید.
- گراف‌ها را با استفاده از دو توپولوژی Gaussian Elimination و Fast Fourie Transform با m=5 ایجاد کنید.
- زمان اجرای هر وظیفه را در بازه [10, 100] میلی ثانیه در نظر بگیرید و هزینه ارتباطی بین وظایف را ۰ لحاظ کنید.
- برای هر هسته MIPS را در بازه [1, 10] به صورت تصادفی (با توزیع یکنواخت) انتخاب کنید.
- ماکسیمم تعداد اجراهای الگوریتم ژنتیک را ۱ میلیون نظر بگیرید.

خروجی‌ها:

- نمودار زمان‌بندی‌پذیری سامانه را به ازای ۴ گراف بر روی ۲، ۴، ۸ و ۱۶ هسته بدست آورید.
- نمودار زمان‌بندی‌پذیری سامانه را به ازای ۴ هسته پردازشی و برای ۲، ۴، ۸ و ۱۶ گراف بدست آورید.
- نمودار makespan سامانه را به ازای ۴ گراف بر روی ۲، ۴، ۸ و ۱۶ هسته بدست آورید.
- نمودار makespan سامانه را به ازای ۴ هسته پردازشی و برای ۲، ۴، ۸ و ۱۶ گراف بدست آورید.
- نمودار هزینه عملیاتی سامانه را به ازای ۴ گراف بر روی ۲، ۴، ۸ و ۱۶ هسته بدست آورید.
- نمودار هزینه عملیاتی سامانه را به ازای ۴ هسته پردازشی و برای ۲، ۴، ۸ و ۱۶ گراف بدست آورید.
- زمان‌بندی سامانه برای ۲ گراف بر روی ۸ هسته پردازشی را نمایش دهید که شامل زمان شروع، زمان اتمام و هسته نگاشت شده می‌باشد. این خروجی را می‌توانید به صورت json یا به صورت شماتیک نمایش دهید.

فازبندی:

۱. فاز اول: تولید حداقل ۸ گراف با استفاده از توپولوژی‌های FFT و GE، پیاده‌سازی اولیه الگوریتم ILP و حل مسئله‌ای دلخواه و ساده با آن. تهیه گزارش از این فاز.
۲. فاز دوم: پیاده‌سازی تمام موارد خواسته شده، بدست آوردن تمام خروجی‌های گفته شده و آماده کردن گزارش نهایی

[15] Integer linear Programming

پروژه شماره ۱۷ **درس:** سیستم‌های بی‌درنگ

عنوان: زمان‌بندی در GPU با هدف کاهش Deadline Miss Rate و انرژی مصرفی **دستیار آموزشی:** آقای احمدی-خانم غیبی

در این پروژه، به زمانبندی وظایف متناوب با هدف کاهش Deadline Miss-rate و انرژی مصرفی در سیستم‌های چندهسته‌ای همگن Non-preemptive می‌پردازیم. در این سیستم می‌توان با اختصاص دادن تعداد بیشتری هسته زمان اجرای وظیفه را کاهش داد و پس از اختصاص یک هسته به یک وظیفه می‌توان قبل از زمان پایان آن وظیفه، هسته را از آن گرفت. همچنین فرض می‌کنیم قابلیت اجرای همزمان حداکثر دو وظیفه وجود دارد و همه هسته‌ها باید بین دو وظیفه تقسیم شوند و یا بیکار بمانند.

الگوریتم **Cooperative**:

در این الگوریتم به ترتیب وظایف را اجرا و به هرکدام از آنها هسته‌ها را اختصاص می‌دهیم. این الگوریتم به عنوان Baseline در نظر گرفته شود.

الگوریتم **sBEET**:

در این الگوریتم فرض می‌کنیم قابلیت اجرای همزمان حداکثر دو وظیفه وجود دارد. وظایف را به ترتیب اولویت (که در اینجا مشابه الگوریتم RM است) مرتب می‌کنیم. سپس اولین وظیفه را انتخاب میکنیم و به ازای همه حالات اختصاص هسته‌ها به آن، زمانبندی‌ها را در نظر می‌گیریم.

سپس بین این زمانبندی‌ها آنهایی که منجر به از دست رفتن موعد زمانی(ددلاین) سایر وظایف در آینده نمی‌شوند را انتخاب می‌کنیم. اگر چندین زمانبندی از این نظر مشابه بودند بین آنها زمان‌بندی با کمترین انرژی مصرفی را انتخاب می‌کنیم. به این ترتیب ابتدا Deadline Miss-rate و سپس انرژی مصرفی را کاهش می‌دهیم.

ورودی‌ها:

- پروفایل شامل زمان اجرا، توان و انرژی مصرفی هر وظیفه به ازای تعداد هسته‌ها (حداکثر۶هسته)
- مجموعه وظایف شامل دوره تناوب و Utilization هر وظیفه

خروجی‌ها:

- نمودار زمانبندی الگوریتم‌ها
- نمودار توان الگوریتم‌ها
- نمودار مقایسه Deadline Miss rate با افزایش Utilization
- نمودار مقایسه انرژی مصرفی با افزایش Utilization
- نمودار Speedup نسبت به الگوریتم Cooperative با افزایش Utilization

فاز اول:

- نمودار زمان‌بندی الگوریتم‌ها

فاز دوم:

- تمامی نمودارها و خروجی‌های مورد نیاز