

دانشگاه صنعتي امیرکبیر (پلی تکنیک تهران)

دانشكده مهندسی کامپیوتر

پایان‌نامه کارشناسی

پیاده‌سازی چهارچوبی برای زمان‌بندی سیستم‌های بحرانی مختلط

نگارنده

مریم علی‌کرمی

استاد راهنما

دکتر حمیدرضا زرندی

فروردین ۱۴۰۳

اينجانب مریم علی‌کرمی متعهد مي‌شوم كه مطالب مندرج در اين پايان‌نامه حاصل كار پژوهشي اينجانب تحت نظارت و راهنمايي اساتيد دانشگاه صنعتي اميركبير بوده و به دستاوردهاي ديگران كه در اين پژوهش از آن‌ها استفاده شده است مطابق مقررات و روال متعارف ارجاع و در فهرست منابع و مآخذ ذكر گرديده است. اين پایان نامه قبلاً براي احراز هيچ مدرك هم‌سطح يا بالاتر ارائه نگرديده است.

در صورت اثبات تخلف در هر زمان، مدرك تحصيلي صادر شده توسط دانشگاه از درجه اعتبار ساقط بوده و دانشگاه حق پيگيري قانوني خواهد داشت.

كليه نتايج و حقوق حاصل از اين پایان نامه متعلق به دانشگاه صنعتي اميركبير مي‌باشد. هرگونه استفاده از نتايج علمي و عملي، واگذاري اطلاعات به ديگران يا چاپ و تكثير، نسخه‌برداري، ترجمه و اقتباس از اين پایان نامه بدون موافقت كتبي دانشگاه صنعتي اميركبير ممنوع است.

نقل مطالب با ذكر مآخذ بلامانع است.

مریم علی‌کرمی

امضا

سپاس‌گزاری

در این‌جا لازم می‌دانم که از استاد گرانقدر، جناب آقای دکتر حمیدرضا زرندی، که بدون راهنمایی‌های ارزشمند ایشان انجام این پروژه میسر نمی‌شد، نهایت تشکر را داشته باشم. همچنین از استاد گران‌مایه، جناب آقای دکتر احمد جوادی، که زحمت داوری این پایان‌نامه را برعهده داشتند، نهایت تشکر را دارم.

مریم علی‌کرمی

فروردین ۱۴۰۳

چکیده

متن چکیده

واژه‌های کلیدی:

سیستم‌های بحرانی مخلتط، زمان‌بندی، الگوریتم زمان‌بندی EDF-VD، سیستم‌های بی‌درنگ

**فهرست مطالب**

|  |  |
| --- | --- |
| عنوان | صفحه |

[1 - مقدمه 1](#_Toc163271443)

[2 - معرفی مفاهیم پایه و ابزار‌های مورد استفاده در پروژه 2](#_Toc163271444)

[2 - 1 سیستم‌های بی‌درنگ 2](#_Toc163271445)

[2 - 1 - 1 وظایف سیستم‌های بی‌درنگ 2](#_Toc163271446)

[2 - 1 - 2 زمان‌بندی سیستم‌های بی‌درنگ 3](#_Toc163271447)

[2 - 2 سیستم‌های بحرانی مختلط 4](#_Toc163271448)

[2 - 2 - 1 مدل سیستم‌های بحرانی مختلط 4](#_Toc163271449)

[2 - 2 - 2 دسته‌بندی تحقیقات انجام شده روی سیستم‌های بحرانی مختلط 5](#_Toc163271450)

[2 - 2 - 3 چالش‌های صنعتی‌سازی سیستم‌های بحرانی مختلط 7](#_Toc163271451)

[3 - زمان‌بندی سیستم‌های بحرانی مختلط 10](#_Toc163271452)

[3 - 1 چالش‌های زمان‌بندی سیستم‌های بحرانی مختلط 11](#_Toc163271453)

[3 - 2 زمان‌بندی اولویت ثابت 12](#_Toc163271454)

[3 - 2 - 1 روش‌های مبتنی بر تحلیل زمان پاسخگویی 12](#_Toc163271455)

[3 - 2 - 2 زمان‌بندی زمان اسلک 21](#_Toc163271456)

[3 - 2 - 3 زمان‌بندی با تبدیل تناوب 21](#_Toc163271457)

[3 - 3 زمان‌بندی اولویت متغیر با الگوریتم EDF 22](#_Toc163271458)

[3 - 3 - 1 گسترش EDF به EDF-VD 25](#_Toc163271459)

[3 - 3 - 2 تعریف MC-Schedulability 25](#_Toc163271460)

[3 - 3 - 3 شرط لازم برای MC-Schedulability 25](#_Toc163271461)

[3 - 3 - 4 نحوه عملکرد الگوریتم EDF-VD 26](#_Toc163271462)

[3 - 3 - 5 اثبات شرط کافی MC-Schedulability با الگوریتم EDF-VD 27](#_Toc163271463)

[3 - 3 - 6 مقایسه‌ی الگوریتم ‌EDF-VD با سایر الگوریتم‌ها 29](#_Toc163271464)

[3 - 3 - 7 محدودیت‌های EDF-VD 31](#_Toc163271465)

[3 - 4 زمان‌بندی چند هسته‌ای 31](#_Toc163271466)

[4 - پیاده‌سازی چهارچوب زمان‌بندی 33](#_Toc163271467)

[4 - 1 توسعه بخش نرم‌افزاری 33](#_Toc163271468)

[4 - 1 - 1 ابزارهای استفاده شده 33](#_Toc163271469)

[4 - 1 - 2 ساختمان سیمولاتور 33](#_Toc163271470)

[4 - 1 - 3 پیاده‌سازی بخش‌های مختلف برنامه 34](#_Toc163271471)

[4 - 1 - 4 تنظیمات شبیه‌سازی و ویژگی‌های قابل تغییر 35](#_Toc163271472)

[4 - 2 اجرای نرم‌افزار شبیه‌سازی و ارزیابی خروجی 36](#_Toc163271473)

[4 - 3 محدودیت‌ها و برنامه‌های آینده 36](#_Toc163271474)

[5 - ارزیابی پروژه 37](#_Toc163271475)

[5 - 1 استفاده از تحلیل مولفه‌های اصلی برای بررسی کیفیت شناسایی چهره 37](#_Toc163271476)

[5 - 2 تاثیر پیش‌پردازش پیش از بردار‌سازی در زمان تمرین مدل شناسایی چهره 37](#_Toc163271477)

[6 - جمع‌بندی، نتیجه‌گیری و کار‌های آینده 39](#_Toc163271478)

[6 - 1 جمع‌بندی و نتیجه‌گیری 39](#_Toc163271479)

[6 - 2 پیشنهادها 39](#_Toc163271480)

[7 - مراجع 41](#_Toc163271481)

فهرست شکل‌ها

|  |  |
| --- | --- |
| عنوان | صفحه |

[شکل ‏2‑1 ساختار یک پرسپترون Error! Bookmark not defined.](#_Toc132135306)

[شکل ‏2‑2 نمونه‌ای از یک شبکه عصبی با دو لایه میانی Error! Bookmark not defined.](#_Toc132135307)

[شکل ‏4‑1 کد بردار‌سازی چهره‌ها Error! Bookmark not defined.](#_Toc132135346)

[شکل ‏4‑2 کد مقایسه چهره‌ها Error! Bookmark not defined.](#_Toc132135347)

[شکل ‏4‑3 محاسبه فاصله میان چهره‌ها در کد Error! Bookmark not defined.](#_Toc132135348)

[شکل ‏4‑4 تصویر ترمینال شروع اجرای پروژه Error! Bookmark not defined.](#_Toc132135349)

فهرست جدول‌ها

|  |  |
| --- | --- |
| عنوان | صفحه |

[جدول ‏2‑1 چند نمونه تابع فعال‌سازی Error! Bookmark not defined.](#_Toc132135103)

[جدول ‏2‑2 مشخصات فنی بورد رزبری‌پای Error! Bookmark not defined.](#_Toc132135104)

[جدول ‏3‑1 حالات نمایش چهره‌های شناسایی شده در نمایش‌گر Error! Bookmark not defined.](#_Toc132135105)

[جدول ‏3‑2 نحوه جابجایی میان صفحات سامانه Error! Bookmark not defined.](#_Toc132135106)

[جدول ‏4‑1 توضیح برخی از فایل‌ها و پوشه‌های مهم پروژه 27](#_Toc132135107)

[جدول ‏4‑2 توضیحات فایل پیکره‌بندی تنظیمات Error! Bookmark not defined.](#_Toc132135108)

[جدول ‏4‑3 نحوه اتصال قطعات به بورد Error! Bookmark not defined.](#_Toc132135109)

# مقدمه

ته هر فصل یادت باشه section break بزنی که بره صفحه بعد

.

# معرفی مفاهیم پایه و ابزار‌های مورد استفاده در پروژه

معرفی مفاهیم پایه و ابزار‌های مورد استفاده در پروژه

در این پروژه از ابزار‌هایی استفاده شده که در ادامه به معرفی آن‌ها خواهیم پرداخت.

## سیستم‌های بی‌درنگ

تعریف سیستم بی‌درنگ.

انواع سیستم‌های بی‌درنگ (سافت و هارد و فرم)

### وظایف سیستم‌های بی‌درنگ

تعریف تسک چیه. مدل نمایش تسک چجوریه و مدل نمایش تسک ست. هر کدوم از مولفه‌هاش نشون‌دهنده‌ی چین. تعریف بهره‌وری. تعریف ریسپانس تایم.

تعریف جاب. مدل نمایش جاب.

انواع تسک‌ها بر اساس زمان فرارسیدن. پرودیک. آپرودیک و اسپورادیک.

ارتباط بین پریود و ددلاین (implicit, constrained, arbitary)

تعریف دابلیو ست و چالش‌های تخمین زدن آن. انواع مختلف هارد ور که می‌تونه دترمینیستیک یا نان دترمینیستک باشه. اور ران چیه.

اسلک چیه.

تعریف بحرانیت. و تعریف میشن کریتیکال و سیفتی کریتیکال.

### زمان‌بندی سیستم‌های بی‌درنگ

شمای کلی از نحوه‌ی اجرای تسک در سیستم (تعریف حالت‌های مختلف تسک و تعریف ردی کیو)

تعریف سیاست زمان‌بندی. Scheduling policy

تعریف زمان‌بندی. Schedule (عکس مدل اسکجول)

#### تفاوت زمان‌بندی‌پذیری و امکان‌پذیری

تسک امکان‌پذیر feasible task

زمان‌بندی امکان‌پذیرfeasible schedule

مجموعه تسک امکان پذیر.feasible task set

تعریف زمان‌بندی پذیری (schdulability)

تعریف و انواع تست زمان‌بندی پذیری (لازم. کافی. لازم و کافی)

پیچیدگی زمانی مساله‌ی زمان‌بندی (توضیح هر کدام از لول‌ها)\*

#### دسته‌‌بندی الگوریتم‌های زمان‌بندی

دسته‌بندی بر اساس زمان‌ بیکاری پردازنده (کانزرواتیو یا نان کانزرواتیو). پارامترهای تصمیم‌گیری (استاتیک یا داینامیک). زمان تصمیم گیری (آنلاین یا آفلاین) . بهینه‌سازی تابع هزینه(آپتیمال یا نان آپتیمال).

میتونن بر اساس مشخصه‌هایی مثل توجه به ددلاین، قبضه‌ای بودن و دارای اولویت‌بندی بودن و ... تقسیم‌بندی شوند.

#### شاخص‌ترین الگوریتم‌های زمان‌بندی دارای اولویت

دسته‌بندی الگوریتم‌های اولویت دار بر اساس نحوه‌ی تغییر اولویت (FTP, FJS, DP)

الگوریتم‌های اولویت ثابت (ددلاین مونوتونیک) (پالیسی هر کدوم چیه و اینکه آیا هر کدوم آپتیمال هست یا نه)

توضیح آنالیز ریسپانس تایم

الگوریتم‌های اولویت متغیر (الگوریتم ای دی اف به عنوان یک الگوریتم اولویت متغیر فلان شناخته می‌شود)

## سیستم‌های بحرانی مختلط

اینجا درباره‌ی سیستم‌های بحرانی مختلط توضیح میدیم.

تعریف استانداردهای سیستم‌های بی‌درنگ.

مدلی که وستال ارائه داد چی بود. توش اومد بیان کرد که دی‌ام و آرام برای زمان‌بندی مدلش مناسب نیستن و یه مدل تغییر یافته از تخصیص اولویت آدلیز رو استفاده کرد. بعد از اون خودش و باروآح اومدن مدل رو به سیستم اسپورادیک گسترش دادن و گفتن که ای دی اف هم برای این سیستم مناسب نیست. مدل‌های اولیه‌ی سیستم‌های میکسد کریتیکال بر پایه‌ی SMC بودن ولی مقاله‌ی باروآح در سال ۲۰۰۸ که اومدن RTA انجام دادن و مدل AMC رو ارائه کردن قدم خیلی بزرگی بود چون از اون جا به بعد دیگه تعریف قالب این شد که در صورت وقوع اورران بیان و تسک‌های لو رو حذف کنن.

مساله‌ی اصلی سیستم‌های بحرانی مختلط (تقابل کارایی و امنیت) – حالا بحث کارایی فقط به زمان مربوط نمیشه چون هزینه هم شامل زمان و توان پردازشیه هم شامل دما هم شامل مصرف انرژی و ...

از یه نگاه دیگه هم میشه به این مساله نگاه کرد البته. اینکه وریفیکشن و ولیدیشن سیستم دو مقوله‌ی جدا هستن. سیستم از نگاه سیستم دیزاینرها با دابلیو ست منطقی تره طراحی میشه ولی از نگاه استانداردها باید معیارهای خیلی سخت‌گیرانه‌تری روش اعمال بشه. مدل وستال سعی داره این دوتا هدف جدا از هم روی توی یک سیستم بگنجونه و راه حلی ارائه کنه که جفت این دوتا رو راضی نگه داره.

بحرانی مختلط یک کانسپته که میتونه روی معماری و زمان‌بندی و تخصیص منابع و مدیریت شکست اعمال بشه ولی ما توی این گزارش روی جنبه‌ی زمان‌بندیش تمرکز داریم.

قبل از بحرانی مختلط چی داشتیم؟ جدا کردن سیستم‌ها. دیزاین کانزرواتیو که همه‌ی تسک‌ها از اولویت بالا باشند. اختصاص بیش از حد منابع. سیستم‌های تایم تریگرد.

### مدل سیستم‌های بحرانی مختلط

نوشن اصلی سیستم‌های بحرانی مختلط اینه که به جای اینکه پارامترهای تسک‌ها فیکسد و مستقل باشه داره از یک حالت وابسته به کریتیکالیتی پیروی می‌کنه.

یک مدل خیلی کاملش اینه که یه سیستم از k تا کامپوننت که هر کدوم یک تسک ست اسپورادیک دارن که پارامترهای تی و فلان و فلان و فلان داره و هر کدوم از تسک‌ها میتونن بی‌نهایت جاب تولید کنن.

مدل اولیه‌ی وستال میاد می‌گه که دابلیو ست بر اساس لول مشخص بشه.

ولی مدل‌هایی بعد از وستال اومدن این وابستگی رو به پریود و ددلاین هم گسترش دادن که مدل‌های خیلی پیشرفته‌تریه.

چالش‌های اصلی پیاده‌ کردن این مدل چی هستن؟

یکی جداسازی – که هر کامپوننتی روی بقیه تاثیر نداشته باشه و اینا روی هم دخالت نداشته باشن

یکی دیگه اینکه جاب ها بیشتر از دابلیو ستشون اجرا نشن

یکی دیگه هم اینکه نزدیک تر از بازه‌ی پریودشون منتشر نشن.

مدلی که وستال ارائه داد در حقیقت یک مدل خیلی اکسپرسیو و قدرتمنده که باهاش میشه سیستم‌های زیادی رو مدل کرد. مثلا شما حتی اگه مقدار دابلیو ست یه تسک رو لو رو ندونید بر فرض هم میتونید این سیستم رو با مدل وستال مدل کنید.

### دسته‌بندی تحقیقات انجام شده روی سیستم‌های بحرانی مختلط

برای این دسته‌بندی‌ها اسم دقیقی وجود نداره ولی کلا بر اساس پارامترهاش ما میتونیم این دسته بندی رو ارائه بدیم.

دسته‌بندی بر اساس پارامتر متغیر: میتونه زمان اجرا باشه میتونه پریود باشه و حتی میتونه ددلاین باشه.

دسته‌بندی بر اساس لول: (از ۲ لول داریم به بالا. یه سریا مثلا تا ۱۴ تا هم رفتن ولی اگه بخوایم تعریف کریتیکالیتی رو بر اساس لول‌های استانداردها در نظر بگیریم تهش دیگه پنجه مثلا)

دسته‌بندی بر اساس مدل افزایش لول: میتونه سیستم لول باشه میتونه تسک لول باشه میتونه جاب لول باشه.

دسته‌بندی بر اساس زمان افزایش لول: میتونه در لحظه باشه میتونه بعد از یه سیکوئنس باشه میتونه بعد از کامل شدن اجرای تسک اورران شده باشه.

دسته‌بندی بر اساس دیگردیشن: میتونه تسک‌های لو رو حذف کنه. میتونه بعضی از تسکای لو رو حذف کنه و .... اجازه بدیم تسک های لوی شروع شده به پایان برسن. اولویت برخی از تسک‌های لو رو کاهش بدیم که قبل از تسک‌های های اجرا نشن. افزایش دادن تی یا دی برخی از تسک‌های لو که یعنی دیر به دیر تر بیان یعنی فرکانسشون رو کمتر کنیم. یا مثلا اجازه بدیم تسک‌های لو ددلاین میس داشته باشن هر از چندگاهی. یا مثلا از روش‌هایی استفاده کنیم که مدل وستال رو تغییر میدن یعنی برای لول بالاتر دابلیو ست کوچیکتری داشته باشیم به جای بزرگتر.

دسته‌بندی بر اساس تسک مدل: زمان‌بندی روی پرودیک یا اسپورادیک یا مجموعه‌ای از جاب‌ها. زمان‌بندی روی تسک‌های وابسته یا تسک‌های غیر وابسته.

دسته‌بندی بر اساس الگوریتم‌های زمان‌بندی انتخاب شده که خیلی زیاده... و توی بخش بعدی به تفصیل بهش می‌پردازیم و تمرکز اصلی این پروژه هم بر همین بخشه.

دسته‌بندی بر اساس بازگشت به حالت بحرانی اولیه. یه سریا کلا کانسیدر نمی‌کنن که باید به حالت اول برگشت و سیستم فقط به سمت بالا میره. یه سریا کانسیدر می‌کنن که برگرده به حالت اولیه ولی مکانیزم‌های بازگشتشون با همدیگه فرق می‌کنه.

دسته‌بندی بر اساس تک هسته‌ای یا چند هسته‌ای بودن که خود بحث پردازش چند هسته‌ای انقدر پارامتر‌های بیشتری ایجاد می‌کنه که دیوانه ‌می‌شوید. مثلا بحث‌های پارتیشنینگ و اساینمت و لود بالانسینگ و ...

دسته‌بندی بر اساس توجه به پارامتر‌های مصرف انرژی، کاهش فرکانس پردازنده ، کنترل دما و ... یعنی یه جورایی تمرکز کردن روی باقی بهینه‌سازی‌ها غیر از توان پردازشی. هم اینکه اونا چه تاثیری روی پردازش دارن هم اینکه پردازش چه تاثیری روی اونا داره و هم اینکه ام سی اس چجوری میتونه با اون هزینه‌ها آداپته بشه و کاهششون بده.

حالا بحث حتی فراتر هم میره و تحقیقات زیادی روی زمینه‌ی سخت‌افزار صورت می‌گیره. یا اینکه اصلا چقدر احتمال داره مود چینج صورت بگیره. هیمنطور تحقیقات زیادی روی روش تخصیص منابعی غیر از زمان و پردازنده صورت گرفته و همینطور روی روش‌های برقراری ارتباط بین تسک‌های لو و های که مثلا دیتا فلو میشه بینشون باشه یا نه. خصوصا که بحث اشتراک منابع خیلی میتونه روی بحث زمان‌بندی هم تاثیر بذاره به خاطر وجود لاک‌ها و پریاریتی اینورژن.

یه سری‌ها هم اومدن روی سیستم‌های FTS تمرکز کردن چون سیستم‌های MCS یک حالتی از سیستم‌های FTS حساب می‌شن که فالت توشون یعنی اورران کردن یا ددلاین میس کردن. خب از این نظر بخوای بهش نگاه کنی خیلی جا داره که بین این دوتا حوزه تحقیقات مشترک صورت بگیره یا اینکه از تحقیقات همدیگه استفاده کنن. یه سریا اومدن براش هاردور توسعه دادن. یه سریا هم اومدن روی جنبه‌ی سیستم‌عامل اضافه کردن و ویرچوآلایز کردنش کار کردن.

دسته‌بندی بر اساس خیلی چیزهای دیگه که هنوز چهارچوب بندی ثابتی ندارن و هر کسی توی مقاله‌ی خودش یه ابتکاری میزنه و یه تفاوتی ایجاد می‌کنه ولی یه مرجع مشخصی وجود نداره که این روش‌ها رو باهمدیگه مقایسه کنه یا یه تسک ثابتی نیست.

کلا هم تاریخچه‌اش به این صورت بوده که از مجموعه جاب‌های مستقل روی یک پردازنده‌ی تک هسته‌ای شروع شده و تا جاهای خیلی زیادی پیش رفته دیگه.

### چالش‌های صنعتی‌سازی سیستم‌های بحرانی مختلط

مثلا یه سری میگن بهتره بهشون بگی مالتی مدل سیستم تا اینکه بگیم میکسد کریتیکال.

دو دغدغه‌ی اساسی وجود داره:

‍۱. در حالت های نباید لو ها رو دیسکارد کنیم چون سیستم کلا ریل تایمه داداش. اونا هم کریتیکال هستن. حالا درسته سیفتی کریتیکال نیستن ولی کریتیکالن و برای کار ما مهمن. اونا رو حذف کنید انگار سیستم ما کار نمی‌کنه دیگه اصلا.

۲. سیستم‌‌هایی که طولانی مدت کار می‌کنن باید بتونن بعد از یه بازه‌ی زمانی به حالت اولیه برگردن.

حالا پوینت قضیه چیه؟

یک اینکه داداش گلم سیستم اصلا نباید وارد حالت های بشه و اون یه حالت خیلی بدبینانس که با احتمال خیلی کمی ممکنه پیش بیاد. ما فقط داریم از نظر تئوری بررسیش می‌کنیم. در عمل سیستم به احتمال خیلی زیاد تو همون حالت لو می‌مونه.

دوم هم اینکه بله توی سیستم‌هایی که کریتیکالیتی پایین‌تری دارن و تسک‌ها احتمال خطاشون بیشتره چون استاندارهایی که روی اون سیستمه اعمال میشه پایین تره (یعنی سیستم میکسد هست ولی در عین میکسد بودن حالت‌های هایش سیفتی کریتیکال نیستن). تو همچین سناریویی حرف مهندسا ولیده چون نیازی نیست وقتی سیستم میره توی حالت های تسک‌های لو حذف بشن.

باز یه پوینت دیگه هم هست که بعضی از سیستم‌ها هستن که تسک‌های هایشون برای اینکه سیستم ساستین کنه و هدفش رو هم به سرانجام برسونه کافین و تسک‌های لو دیگه خیلی ماینورن که روی فانکشنالیتی تاثیر خاصی ندارن.

با توجه به این موارد توی کنفرانس سال ۲۰۱۷ اعلام شد که مدل وستال دو وظیفه داره. یکی اینکه تایید کنه سیستم قابل اعتماده و میتونه استاندارد دریافت کنه. و وظیفه‌ی دیگش اینه که در زمان اجرا ساستین‌ابل باشه یعنی فانکشنالیتی خود سیستم به خاطر پالیسی‌های ارائه شده برای وریفیکشن از بین نرن. به بیان دیگه یعنی یک حالت گریسفول از کاهش سطح سرویس رو داشته باشیم. بر این اساس مدل‌هایی که میان تسک های لو رو زارتی حذف می‌کنن و تحقیقاتشون رو بر این پایه استوار کردن مورد قبول نیستن. از این نظر آپشن‌هایی خیلی زیادی هست که توی بخش قبلی به تحقیقاتی که روشون صورت گرفته اشاره کردیم.

یه چالش دیگه اینه که اگر سیستم ۷ تا لول داشته باشه لزوما برای تسک‌ها ۷ تا مقدار دابلیو ست نداریم.

یه چالش دیگه اینه که توی همه‌ی تحقیق‌هایی که روی این زمینه بوده مدل‌ها با هم یکی نبوده و فرق داشته و روی تعریف بحرانیت تجمیع نظر وجود نداره. یعنی اگر تعریف رو بنا به صنعت قرار بدیم اولویت بالا یعنی سیفتی کریتیکال ولی اگر بنا به استانداردها باشه اولویت بالا می‌تونه یه تسک غیر سیفتی کریتیکال باشه که تحمل خطای پایینی داره. یه چالش خیلی مهم اینه که بعضی از تسکای بحرانیت بالا فی‌الواقع تسک‌های سیفتی کریتیکال نیستن.

یه بحث خیلی خیلی مهم دیگه فالت تالرنس کامپوننت‌هاست چون اگه میخوایم تسک‌هایی که توی این کامپوننت‌ها اجرا میشن کیفیت تاپ ناچ نداشته باشن، کامپوننت‌هامون باید کیفیت تاپ ناچ داشته باشن که فالت‌های تسک‌های لو باعث به هم ریخته شدن تسک‌های های نشه. در غیر این صورت همه‌ی تسک‌هاتون باید های باشن. کلا جداسازی زمانی (پردازشی) و مکانی (مربوط به باس و مموری و اینا) باید توی کامپوننت‌های سیستم شما تضمین شده باشه وگرنه زرشک.

یه چالش دیگه هم اینه که این همه الگوریتم ارائه شده که هیچ کدومشون آپتیمال نیستن و معیار دقیقی برای مقایسه‌ی این الگوریتم‌ها وجود نداره.

#### 

# زمان‌بندی سیستم‌های بحرانی مختلط

در این فصل، با تمرکز عمده بر مجموعه‌ای از وظایف اسپورادیک، به بررسی جزئیات و چالش‌های مربوط به زمان‌بندی سیستم‌های بحرانی مختلط می‌پردازیم و برخی از الگوریتم‌های مطرح در این زمینه را مورد بررسی قرار می‌دهیم. توجه داشته باشید که توضیح مدل وظایف ارائه‌شده در این فصل و معنی واژگان مختصر، پیش‌تر در بخش \* آمده‌است.

## چالش‌های زمان‌بندی سیستم‌های بحرانی مختلط

پیش از هر چیز لازم است چند چالش عمده در زمینه‌ی زمان‌بندی سیستم‌های بحرانی-مختلط را به شمار بیاوریم و با در نظر داشتن این موارد به تحلیل الگوریتم‌های ارائه‌شده در این فصل بپردازیم:

* یک سیستم بحرانی مختلط ممکن است در هر یک از سطوح تعریف‌شده برای آن قابل زمان‌بندی باشد. اما، در صورتی که سیستم قابلیت تغییر سطح بحرانیت را داشته باشد، ممکن است در زمان انتقال از سطح بحرانی پایین‌تر به سطح بالاتر، قابل زمان‌بندی نباشد. به همین دلیل، تحلیل‌های ارائه‌شده برای بررسی زمان‌بندی این سیستم‌ها باید تمام حالت‌های ممکن برای تغییر سطح سیستم را در نظر بگیرند و ثابت کنند که سیستم در آن حالت‌ها نیز قابل زمان‌بندی است.
* مسئله زمان‌بندی سیستم‌های بحرانی مختلط یک مسئله NP-Hard است، حتی اگر سیستم تنها دارای دو سطح بحرانی باشد. به همین دلیل، تحلیل‌های ارائه‌شده برای زمان‌بندی‌پذیری این سیستم‌ها همگی دارای شروط کافی هستند و تاکنون هیچ تحلیل دقیقی ارائه نشده است [1].
* بسیاری از الگوریتم‌های زمان‌بندی برای سیستم‌های بحرانی مختلط به یک سازوکار نظارت در زمان اجرا[[1]](#footnote-1) نیاز دارند. با این حال، تحقیقات اندکی بر روی نحوه پیاده‌سازی این سازوکارها صورت گرفته است. علاوه بر این، معمولاً در محاسبات، از سربار[[2]](#footnote-2) ناشی از این سازوکارها چشم‌پوشی می‌شود. در تحقیقی که توسط سیگریست[[3]](#footnote-3) و همکاران انجام شده است، آن‌ها نشان دادند که این سازوکارها می‌توانند سرباری ۹۷ درصدی داشته باشند. آن‌ها توصیه می‌کنند که تمام مدل‌های زمان‌بندی باید گسترش یابند تا شامل مولفه‌هایی باشند که تأثیر سربار در زمان اجرا را در نظر بگیرند [1].

## زمان‌بندی اولویت ثابت

الگوریتم‌های اولویت ثابت قبضه‌ای[[4]](#footnote-4)، یکی از دسته‌های محبوب الگوریتم‌های زمان‌بندی در سیستم‌های بی‌درنگ هستند. بنابراین، تحقیقات اولیه در حوزه زمان‌بندی سیستم‌های بحرانی مختلط با مطالعه الگوریتم‌های اولویت ثابت آغاز شده و آن‌ها را برای مدل‌سازی وظایف بحرانی مختلط توسعه داده‌اند. در بیشتر این پژوهش‌ها، مسئله تخصیص اولویت و مسئله تحلیل زمان‌بندی مجموعه وظایف با استفاده از اولویت‌های تعیین‌شده بررسی می‌شوند. در مسئله تخصیص اولویت، محققان به دنبال الگوریتمی هستند که بر اساس منطقی ثابت به هر وظیفه اولویتی اختصاص دهد. در مسئله تحلیل زمان‌بندی نیز با یک مدل تغییر یافته از تحلیل‌های زمان‌بندی بی‌درنگ برای الگوریتم خود آزمون زمان‌بندی طراحی می‌کنند.

در این بخش، ابتدا به بررسی رشد و تکامل الگوریتم‌های اولویت ثابت برای سیستم‌های بحرانی مختلط می‌پردازیم که با استفاده از روش‌های مبتنی بر تحلیل زمان پاسخگویی انجام شده است. در ادامه، به سایر روش‌ها از جمله روش‌های مبتنی بر زمان اضافی[[5]](#footnote-5) و همچنین تغییر دوره تناوب[[6]](#footnote-6) اشاره خواهیم کرد.

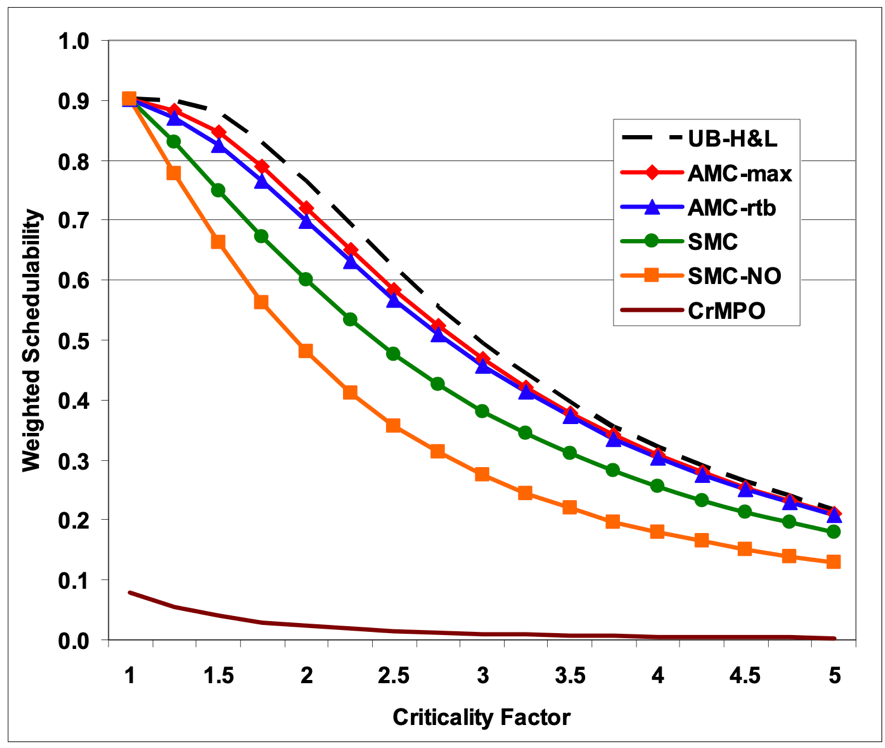
### روش‌های مبتنی بر تحلیل زمان پاسخگویی

در این روش‌ها، از تحلیل زمان پاسخگویی، که به اختصار RSA نامیده می‌شود، برای تخصیص اولویت و همچنین طراحی آزمون‌های زمان‌بندی استفاده می‌شود. یکی از دلایل گسترده استفاده محققان از تحلیل‌های مبتنی بر زمان پاسخگویی آن است که در حوزه زمان‌بندی سیستم‌های بی‌درنگ سنتی با الگوریتم‌های زمان‌بندی اولویت ثابت، این تحلیل یک آزمون زمان‌بندی دقیق را ارائه می‌دهد [2]. از این رو بدیهی‌ است که این روش‌ها برای سیستم‌های بحرانی مختلط نیز توسعه داده‌ شوند.

#### الگوریتمCrMPO[[7]](#footnote-7)

الگوریتم CrMPO یکی از ساده‌ترین و بدیهی‌ترین الگوریتم‌های تخصیص اولویت به وظایف بحرانی مختلط است. این الگوریتم جزو دسته‌ی الگوریتم‌های [[8]](#footnote-8)CM یا PC[[9]](#footnote-9) قرار می‌گیرد. در الگوریتم‌های CM، اولویت‌ها بر اساس سطح بحرانیت هر وظیفه اختصاص داده می‌شوند، به طوری که وظایف با بحرانیت بالاتر همواره اولویتی بالاتر از وظایف با بحرانیت پایین‌تر خواهند داشت. در تخصیص اولویت CrMPO، وظایف ابتدا بر اساس درجه‌ی بحرانیت مرتب می‌شوند و در صورت یکسان بودن درجه‌ی بحرانیت، مرتب‌سازی بر اساس الگوریتم DM انجام می‌گیرد.

مزیت این نوع الگوریتم‌ها در آن است که بروز خطای زمانی در وظایف با درجه‌ی بحرانیت پایین‌تر، موجب انتقال خطا به وظایف با درجه‌ی بحرانیت بالاتر نمی‌شود، چرا که این وظایف همواره قبل از وظایف با درجه‌ی بحرانیت پایین‌تر اجرا شده‌اند. از این منظر، اینگونه الگوریتم‌ها دغدغه‌ی Separation را به خوبی در یک سیستم بحرانی مختلط برآورده می‌کنند. علاوه بر آن، الگوریتم‌های CM نیازی به نظارت در زمان اجرا ندارند. اما همان‌طور که در شکل ‏3‑1 نشان داده شده است، این الگوریتم‌ها توانایی بسیار پایینی در زمان‌بندی مجموعه‌های وظایف بحرانی دارند، به این معنا که تعداد بسیار کمی از مجموعه‌های وظایف بحرانی را می‌توانند زمان‌بندی کنند [3].



شکل ‏3‑1 مقایسه‌ی انواع الگوریتم‌های زمان‌بندی اولویت ثابت در سیستم‌های بحرانی مختلط

{بعد تر ها یه آقایی میاد معیار ساستین‌ابل بودن رو مطرح می‌کنه و میگه کریتیکالیتی مونوتونیک اصلا ساستین ابل نیست داداشا}\*

#### تحلیل چند-بحرانیتی وستال و ناکارآمدی الگوریتم DM

در سیستم‌های بی‌درنگ سنتی که مهلت انجام وظیفه برابر یا کمتر از تناوب آن وظیفه است، الگوریتم DM به عنوان یک الگوریتم بهینه برای تخصیص اولویت به وظایف شناخته می‌شود. وستال در اولین مقاله خود پس از ارائه مدل سیستم‌های بحرانی مختلط، اثبات می‌کند که الگوریتم DM برای زمان‌بندی مجموعه وظایف بحرانی مختلط، بهینه نیست. به این منظور، مجموعه وظایف نمایش‌داده‌شده در جدول ‏3‑1 را به عنوان نمونه ارائه می‌دهد [4].

جدول ‏3‑1 نمونه مجوعه وظایف بحرانی مختلط ارائه‌شده توسط وستال برای اثبات ناکارآمدی الگوریتم DM [4]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **C(HI)** | **C(LO)** | **L** | **T** | **D** | **i** |
| 2 | 1 | B (LO) | 2 | 2 | 1 |
| 1 | 1 | A (HI) | 4 | 4 | 2 |

در این نمونه، اگر به وظیفه 1 بالاترین اولویت اختصاص یابد، آنگاه وظیفه 2 یک پردازنده‌ای را می‌بیند که قبلاً 100% از زمان موجود سطح A خود را برای وظیفه 1 کنار گذاشته است. با این حال، اگر به وظیفه 2 بالاترین اولویت اختصاص داده شود طبق تحلیل چند-بحرانیتی رابطه ۳-۱، سیستم قابل اجرا است [4].

|  |  |
| --- | --- |
|  | 1-3) |

رابطه‌ي ۳-۱ معادله‌ای تغییر یافته‌ از الگوریتم محاسبه‌ی بدترین زمان پاسخگویی جوزف-پاندیا[[10]](#footnote-10) است که وستال آن را برای بررسی زمان‌بندی سیستم‌های بحرانی مختلط ارائه داد. این رابطه یک رابطه‌ی بازگشتی‌ است که مقدار Ri در آن از محاسبه‌ی بازگشتی سمت راست به‌ روز‌رسانی می‌شود. معادله از Ri=CiLi آغاز می‌شود و تا زمانی که Ri > Di یا Ri دیگر تغییر نکند ادامه می‌یابد. در حالت اول، مجموعه وظایف با اولویت‌های ρ قابل زمان‌بندی نیست [4]. رابطه‌ی ۳-۱ در اصل صورتی تغییر یافته از RSA است که وستال برای تحلیل زمان‌بندی سیستم‌های بحرانی مختلط ارائه داد. طبق این رابطه، امکان زمان‌بندی‌پذیری هر وظیفه با توجه به سطح بحرانی همان وظیفه سنجیده می‌شود.

وستال مدل اختصاص اولویت آدلی[[11]](#footnote-11) را نیز با به‌کار‌گیری رابطه‌ی ۳-۱ برای مجموعه وظایف بحرانی مختلط تعمیم داد. در الگوریتم تخصیص اولویت وستال، ابتدا هیچ کدام از وظایف اولویتی ندارند و با به کار گرفتن رابطه‌ی ۳-۱، وظایف به ترتیب از پایین‌ترین اولویت به بالاترین اولویت دسته‌بندی می‌شوند. در اختصاص اولویت به وظیفه‌ی i، تصور می‌شود که تمامی وظایف دیگر که هنوز اولویت‌بندی نشده‌اند، اولویت بالاتری از i دارند. در این الگوریتم به هر وظیفه یک اولویت یکتا اختصاص داده می‌شود و اولویت هیچ دو وظیفه‌ای با هم برابر نیست. در صورتی که چند وظیفه با درجه‌ی بحرانی متفاوت، با درجه‌ی اولویت یکسانی قابل زمان‌بندی باشند، وظیفه با درجه‌ی بحرانی بالاتر، اولویت بالاتر را به خود اختصاص می‌دهد [4].

الگوریتم تخصیص اولویت وستال گرچه به نسبت الگوریتم‌هایی که بعد از آن ابداع شدند بدبینانه‌تر است اما، امکان رخداد واژگونی بحرانیت[[12]](#footnote-12) را در شرایطی محاسبه‌شده فراهم می‌آورد. واژگونی بحرانیت به این معناست که وظیفه‌ای با درجه‌ی بحرانی پایین‌تر از وظیفه‌ای با درجه‌ی بحرانی بالاتر، اولویت بالاتری داشته باشد [3]. بدیهی است که چنین تعبیری، به الگوریتم وستال اجازه می‌دهد تا توانایی زمان‌بندی تعداد وسیع‌تری از مجموعه وظایف را داشته باشد.

در پژوهش‌هایی که باروآح[[13]](#footnote-13) و همکارانش بعد از آن با اعمال تحلیل‌های زمان پاسخگویی بر انواع الگوریتم‌های اولویت ثابت انجام دادند، نشان داده شد که الگوریتم وستال با نام SMC-NO در زمان‌بندی مجموعه وظایف بحرانی مختلط از الگوریتم CrMPO بسیار توانمندتر است. نمونه‌ای از نتایج این تحلیل‌ها در شکل ‏3‑1 آورده شده‌است [3].

#### روش‌های AMC و درجه‌ی بحرانیت یک رفتار

باروآح و همکارانش در سال ۲۰۱۱ با بهره‌گیری از نظارت در زمان اجرا و تعریف مفهوم درجه‌ی بحرانیت یک رفتار[[14]](#footnote-14)، روش بحرانیت مختلط انطباق‌پذیر، به اختصار AMC، را ابداع کردند. این روش به صورت گسترده‌ای مورد تحقیق و بررسی قرار گرفت و بعد از آن تا حدودی به بخشی جدایی‌ناپذیر از تعریف سیستم‌های بحرانی مختلط تبدیل شد.

آن‌ها دسته‌بندی جدول ‏3‑2 را برای الگوریتم‌های اولویت‌ ثابت ارائه دادند:

جدول ‏3‑2 دسته‌بندی انواع سیستم‌های بحرانی مختلط اولویت ثابت [3]

|  |  |
| --- | --- |
| **نام روش** | **توضیح** |
| بحرانیت تقسیم‌بندی شده (PC) | یک طرح استاندارد که گاهی اوقات به نام تخصیص اولویت مونوتونیک بحرانیت (CM) نامیده می‌شود. در این روش به هر وظیفه اولویتی متناظر با درجه‌ی بحرانیت آن وظیفه اختصاص داده می‌شود. |
| بحرانیت مختلط ثابت (SMC) | طرحی منطبق بر مدل وستال که در آن هر کدام از کارهای مربوط به یک وظیفه می‌تواند تنها به اندازه‌ی بدترین زمان اجرای منسوب به درجه‌ بحرانیت وظیفه (CiLi) اجرا شود و بعد از آن بسته به مدل سیستم، آن کار متوقف شده یا برای بازیابی خطا در زمان دیگری زمان‌بندی می‌شود. |
| بحرانیت مختلط انطباق‌پذیر (AMC) | طرح نوآورانه‌ی باروآح و همکارانش که در آن در صورتی که هر کدام از کارهای مربوط به یک وظیفه از بدترین زمان اجرای منسوب به درجه بحرانیت وظیفه (CiLi) تجاوز کنند، تمامی کارهای تمامی وظایفی که درجه‌ی بحرانیتی هم‌اندازه با Li یا کمتر از آن دارند، متوقف شده و دیگر زمان‌بندی نمی‌شوند. |

{حالت میانی بین این دتا}

باروآح و همکارانش با دلایل مطرح‌شده در بخش ‏3 - 2 - 1 - 1 بیان کردند که الگوریتم‌های PC برای زمان‌بندی سیستم‌های بحرانی مختلط قدرت کافی ندارند. سپس با این استدلال که به‌کار‌گیری روش‌های نظارت در زمان اجرا در سیستم‌های بی‌درنگ روشی مرسوم است و در صنعت نیز مورد استفاده قرار می‌گیرد به ابداع روش نوآورانه‌ی AMC پرداختند. افزودن سازوکارهای نظارت در زمان اجرا و اعمال بودجه[[15]](#footnote-15)، به کارهای وظایف از درجه‌های بحرانیت متفاوت اجازه می‌دهد که اولویت‌هایی ترکیب‌شده داشته باشند و این قدرت زمان‌بندی الگوریتم‌ها را افزایش می‌دهد [3].

بعد از آن به تحلیل روش‌های بحرانیت مختلط ثابت می‌پردازند. روش پیشنهاد شده توسط وستال که در بخش ‏3 - 2 - 1 - 2 بیان شد با نام‌ SMC-NO بررسی می‌شود و نسخه‌ای از آن که به همراه نظارت‌های زمان اجرا کار می‌کند را SMC نامیدند. تحلیل‌های زمان پاسخگویی این دو الگوریتم در جدول ‏3‑3 آورده شده‌است. همانطور که مشاهده می‌کنید SMC با به کار‌گیری نظارت‌های زمان‌ اجرا می‌تواند با تبدیل Cj به Cj(min(Li, Lj)) بالاترین حد[[16]](#footnote-16) دقیق‌تری برای زمان پاسخگویی و در نتیجه زمان‌بندی‌پذیری بیشتری را فراهم کند. برای محاسبه‌ی تحلیل زمان‌ پاسخگویی SMC، هر سه حالت Li > Lj، Li < Lj و Li = Lj بررسی شده‌اند [3].

جدول ‏3‑3 تحلیل زمان پاسخگویی SMC و SMC-NO [3]

|  |  |
| --- | --- |
| SMC-NO | SMC |
|  |  |
| \* hp(i) نشان‌دهنده‌ی تمام وظایفی‌است که اولویتی بالاتر از وظیفه‌ی i دارند | |

**تعریف ۳-۱:** در اجراهای مختلف، هر سیستم وظیفه‌ای به طور کلی رفتارهای مختلفی را نشان می‌دهد: ممکن است وظایف مختلف در لحظات زمانی مختلف آغاز شوند و زمان اجرای واقعی متفاوتی داشته باشند. بیایید **درجه بحرانیت یک رفتار** را به عنوان کمترین سطح بحرانیت تعریف کنیم به طوری که هیچ وظیفه‌ای بیش از مقدار C خودش در این سطح بحرانیت، اجرا نشود. برای هر سطح بحرانیت L، تمام وظایف با بحرانیت ≥ L در هر رفتار با سطح بحرانیت L تا مهلت‌های مقررشان تکمیل خواهند شد [3].

با بهر‌ه‌گیری از تعریف ۳-۱، الگوریتم AMC برای یک سیستم دو بحرانیتی[[17]](#footnote-17) با بحرانیت‌های HI > LO، در ۵ قدم تعریف می‌شود [3]:

* قدم ۱: یک نشانگر سطح بحرانیت Γ وجود دارد که به LO مقداردهی اولیه می‌شود.
* قدم ۲: در حالی که (Γ ≡ LO)، در هر لحظه وظیفه‌ای که بالاترین اولویت را دارد برای اجرا انتخاب می‌شود.
* قدم ۳: اگر وظیفه‌ای که در حال اجرا است برای زمان اجرای بحرانیت پایین (LO) خود بدون نشان دادن اتمام کار، اجرا شود، آنگاه Γ ← HI.
* قدم ۴: هنگامی که (Γ ≡ HI)، وظایف با سطح بحرانیت ≡ LO اجرا نخواهند شد. بنابراین، از این پس، در هر لحظه وظیفه‌ای که توسط وظیفه با بحرانیت بالا (HI) و بالاترین اولویت تولید می‌شود برای اجرا انتخاب می‌شود.
* قدم ۵: یک قاعده اضافی می‌تواند شرایطی را مشخص کند که در آن Γ به LO بازنشانی می‌شود. این امر می‌تواند به عنوان مثال زمانی رخ دهد که هیچ وظیفه‌ای با بحرانیت بالا (HI) در یک لحظه خاص فعال نباشد.

البته این الگوریتم در ادامه مورد بررسی بیشتر و بهبود قرار گرفت. به عنوان مثال، از دو سطح بحرانیت به k سطح بحرانیت تعمیم داده شد. همچنین برای سناریوهای مختلفی از متوقف کردن وظایف با بحرانیت پایین، قبل یا بعد از تکمیل اجرای آن‌ها، بررسی‌هایی صورت گرفت که بحث در مورد آن‌ها از حوصله این گزارش خارج است [1].

تخصیص اولویت‌ها به وظایف در این الگوریتم نیز با روش آدلی انجام می‌شود. تحلیل زمان‌بندی سیستم در حالتی که Γ ≡ LO است طبق رابطه‌ی ۳-۲ و در حالتی که سیستم در حالت Γ ≡ HI است طبق رابطه‌ی ۳-۳ انجام می‌شود. در رابطه‌ی ۳-۳ منظور از hpH(i) وظایفی با اولویت بالاتر از وظیفه‌ی i هستند که Li = HI دارند [3].

|  |  |
| --- | --- |
| ۲-۳) |  |
| ۳-۳) |  |

همانطوری که در بخش ‏3 - 1 اشاره کردیم، مساله‌ی زمان‌بندی‌پذیر بودن مجموعه وظایف بحرانی مختلط در حالت تبدیل از سطح بحرانی پایین به سطح بحرانی بالا (قدم ۳ از الگوریتم AMC)، ‌یکی از چالش‌های اصلی زمان‌بندی سیستم‌های بحرانی مختلط است. تحلیل زمان‌بندی سیستم در حین این تغییر، به دو روش محاسبه می‌شود [3]:

##### شرط کافی AMC-rtb[[18]](#footnote-18)

در این تحلیل که یک تحلیل مبتنی بر زمان پاسخگویی می‌باشد، ابتدا رابطه‌ی SMC که در جدول ‏3‑3 آمده است به صورت رابطه‌ی ۳-۴ شکسته می‌شود. در رابطه‌ی ۳-۴ منظور از hpL(i) وظایفی با اولویت بالاتر از وظیفه‌ی i هستند که Li = LO دارند [3]:

|  |  |
| --- | --- |
| ۴-۳) |  |

از آنجایی که در حالت Γ ≡ HI تمامی وظایف با بدبینانه‌ترین بدترین زمان اجرای خود تحلیل می‌شوند و هدف بررسی فقط تسک‌هایی با بحرانیت HI است، رابطه‌ی ۳-۴ به صورت ۳-۵ تغییر می‌کند [3]:

|  |  |
| --- | --- |
| ۵-۳) |  |

رابطه‌ی ۳-۵ که در اصل مدلی تغییر کرده از تحلیل SMC است برای AMC بسیار سخت‌گیرانه است چون متوقف شدن وظایف با درجه بحرانیت LO رو در نظر نمی‌گیرد. تبدیل حالت رفتار سیستم از LO به HI حتما باید قبل از RiLO که در رابطه‌ی ۳-۲ محاسبه شد اتفاق بیافتد در غیر این صورت وظیفه‌ی i تحت تاثیر تبدیل حالت سیستم قرار نمی‌گیرد و پیش از آن به اتمام رسیده است. از این رو حالت دقیق‌تر رابطه‌ی ۳-۵ در رابطه‌ی ۳-۶ نمایش داده شده است. رابطه‌ی ۳-۶ همان تحلیل زمان‌بندی مبتنی بر زمان پاسخگویی AMC-rtb می‌باشد. از آنجایی که این رابطه با به احتساب آوردن RiLO به دست آمده است به آن حد زمان پاسخگویی (rtb) گفته می‌شود و R\*i مقدار بدترین زمان پاسخگویی وظیفه‌ی i در حالت تبدیل سیستم را به دست می‌دهد [3].

|  |  |
| --- | --- |
| ۶-۳) |  |

از نظر شهودی، از آن جایی که تحلیل AMC-rtb از تحلیل SMC دقیق‌تر است، انتظار می‌رود که قدرت زمان‌بندی AMC-rtb بیشتر از SMC باشد. ارزیابی‌ها نیز در شکل ‏3‑1 همین را اثبات می‌کنند.

##### شرط کافی AMC-max[[19]](#footnote-19)

در این تحلیل تلاش می‌شود که قسمت‌هایی که در تحلیل AMC-rtb فرضیاتی سخت‌گیرانه داشته‌اند، بهبود یابند و حد بالای دقیق‌تری از بدترین زمان پاسخگویی وظایف به دست بیاید. واژه‌ی max در این نام‌گذاری، به بیشترین میزان تداخل- میان وظایف LO و HI- اشاره دارد.

فرض می‌شود که اگر تغییر در حالت سیستم در لحظه‌ی s توسط وظیفه s صورت گیرد، تداخلی که وظیفه s برای وظیفه‌ی i از درجه‌ی بحرانی بالا ایجاد می‌کند، چقدر است. این مقدار با Rsi نمایش داده می‌شود که نحوه‌ی محاسبه‌ی آن در رابطه ‌۳-۷ آورده شده‌است و برابر است با بیشترین زمان پاسخگویی وظیفه‌ی i در صورتی که تبدیل حالت در لحظه‌ی s اتفاق بیافتد. بیشترین مقدار ممکن برای Rsi، حد بالای زمان پاسخگویی وظیفه‌ی i را به دست می‌دهد که در رابطه‌ی ۳-۸ با R\*i نشان داده شده است [3].

|  |  |
| --- | --- |
| ۷-۳) | where |
| ۸-۳) |  |

بررسی‌های باروآح و همکارانش برای دقیق‌تر کردن رابطه‌ی ۳-۷ از حیطه‌ی این گزارش خارج است اما می‌توان آن را به صورت کلی توضیح داد؛ برای محاسبه‌ی Rsi میزان تداخل وظایف LO و میزان تداخل وظایف HI بر روی وظیفه‌ی i به طور جداگانه محاسبه می‌شود. در محاسبه‌ی تداخل وظایف LO فرض می‌شود که وظیفه‌ی s که باعث تبدیل حالت شده‌است، قبل از تبدیل حالت سیستم به طور کامل اجرا می‌شود. در محاسبه‌ی تداخل وظایف HI نیز تلاش شده‌است با ارائه‌ی فرمول M(k,s,t) تعداد دقیق وظایفی که قبل از لحظه‌ی s اجرای آن‌ها کامل می‌شود و در نتیجه با C(LO) اجرا می‌شوند از تعداد وظایفی که بعد از لحظه‌ی s و با C(HI) اجرا می‌شوند، تمییز داده شود. مقادیری از s که می‌توانند ‌Rsi را بیشینه‌ کنند، احتمالا زمان‌های انتشار وظایف hpL(i) هستند [3].

تحلیل AMC-max که رابطه‌ی ۳-۸ نشان داده شد، از تحلیل AMC-rtb بسیار دقیق‌تر است اما همچنان فرضیات سخت‌گیرانه‌ای دارد که باعث می‌شود این تحلیل نیز یک شرط کافی باشد و نتواند آزمون دقیقی برای بررسی زمان‌بندی‌پذیری مجموعه وظایف بحرانی مختلط با الگوریتم ‌AMC ارائه دهد [3].

### زمان‌بندی زمان اسلک

یه سری کصخلم هستن که میگن بیایید از زمان اسلک تسکای های استفاده کنیم تا لو ها رو اسکجول کنیم. اسلک هم تو دو حالت پیش میاد یا های ها کمتر از زمانی که فکر کردیم استفاده میکنن یا اینکه با فواصل زمانی بیشتری از اون چیزی که فکر کردیم ظاهر میشن. که جفت اینا برای اینکه لوها رو توش اسکجول کنیم مناسبن. ولی خب یه سری چالش داره مثلا وقتی یه تسک های مدتیه نیومده و شما میخوای از اسلکش استفاده کنی از کجا مطمئن باشی نمی‌رینی و اون تایمی که میدی به تسک لو خطرناک نیست؟ هان؟! برای همین اطمینان از زمان‌بندی پذیری های‌ها توی همچین سیستم‌هایی قابل اتکا نیست.

(میتونم بگم جی پی تی بقیش رو از مقاله‌ی ریویو خلاصه کنه و بگه)

### زمان‌بندی با تبدیل تناوب

اینم روش جالبیه که یه سری آدما که جوز میزنن ازش استفاده میکنن. روشش اینطوریه که یه تسک رو میشکونن به ان تا تسک و اون ان تا تسک رو با آر ام یا دی ام زمان‌بندی می‌کنن. این حالت برای وقتیه که دی با تی برابر باشه ولی میتونن به دی کوچکتر از تی هم تعمیمش بدن. این روش باعث میشه که تسک های های حتما قبل از لو ها اجرا بشن. ولی خب دردسرش اینجاست که تعداد کانتکست سوییچ هاش خیلی خیلی زیاده و همینطور برای سیستم‌هایی که بیشتر از دوتا لول کریتیکالیتی دارن میتونه خیلی سخت تر باشه آداپته کردن ددلاین‌ها و زمان‌های اجرا. چون یه دور باید لو ها رو با میدها هماهنگ کنن بعد میدها رو با های ها هماهنگ کنن. هر چی تعداد لول بره بالاتر تعداد ترنسفورم‌ها بیشتر می‌شه و نتیجتا تعداد تسک‌های ریز شده خیلی زیاد میشه و انقدر کانتکست سوییچش زیاد میشه که عملا ممکنه از حالت عملی خارج بشه.

یه چالش دیگه هم داره این زمان‌بندی اونم این که تسک رو باید از کجا بشکونن. یا باید به حالت استاتیک باشه که خود پرگرمر محل‌های شکست کد رو مشخص می‌کنه که این میتونه جلوی برنامه‌ریزی ماژولار رو بگیره و کیفیت کد رو کاهش بده و فلکسیبیلیتی سیستم رو کاهش بده یا باید به صورت ران‌تایم سرور باشه. که در حالت ران‌تایم سرور کافیه که فقط تسک‌های های مانیتور بشن که اورران می‌کنن یا نه و نیازی نیست لو ها رو مانیتور کنیم چون به هر جهت بعد از های‌ها اجرا میشن. این مزیت زمان‌بندی تبدیل تناوبه چون توی حالت‌های دیگه ران‌تایم سرورمون هم باید های‌ها رو مانتیور کنه هم لو ها رو مانیتور کنه و زمان‌ اجرا رو روشون انفورس کنه.

## زمان‌بندی اولویت متغیر با الگوریتم EDF

می‌دانیم که الگوریتم EDF در زمان‌بندی سیستم‌های بی‌درنگ سنتی یک الگوریتم بهینه است و آزمون دقیقی بر اساس میزان بهره‌وری وظایف دارد [2]، به همین دلیل بررسی این الگوریتم برای زمان‌بندی سیستم‌های بحرانی مختلط نیز انجام شده‌‌است. در سیستم‌های بی‌درنگ سنتی، الگوریتم‌های اولویت متغیری چون EDF توانایی بالاتری برای زمان‌بندی مجموعه وظایف دارند. از این رو انتظار می‌رفت که این الگوریتم در زمان‌بندی سیستم‌های بحرانی مختلط نیز عملکرد بهتری از الگوریتم‌های اولویت ثابت داشته باشد. وستال و باروآح در تحقیقی نشان دادند که ترکیبی از الگوریتم اولویت ثابت و EDF برای زمان‌بندی بحرانی‌ مختلط، عملکرد بهتری از SMC-NO که پیشتر توسط وستال ارائه شده بود دارد اما این الگوریتم که آن را زمان‌بندی اولویت-ترکیبی[[20]](#footnote-20) نامیده بودند نیز الگوریتم بهینه‌ای نبود و نمونه‌هایی از مجموعه وظایف امکان‌پذیر وجود داشتند که با این الگوریتم قابل زمان‌بندی نبودند [5].

در این بررسی با فرض اینکه برای زمان‌بندی‌پذیر بودن یک سیستم بحرانی مختلط، حالت سنتی آن نیز باید امکان‌پذیر باشد، وستال و باروآح با مجموعه وظایف نمونه‌ی جدول ‏3‑4 نشان دادند که الگوریتم EDF برای زمان‌بندی سیستم‌های بحرانی مختلط بهینه نیست.

جدول ‏3‑4 نمونه مجوعه وظایف بحرانی مختلط ارائه‌شده برای اثبات ناکارآمدی الگوریتم EDF [5]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **C(HI)** | **C(LO)** | **L** | **T** | **D** | **i** |
| 2 | 2 | 2 (HI) | 4 | 4 | 1 |
| 5 | 2 | 1 (LO) | 7 | 7 | 2 |

وظیفه ۱ یک وظیفه با بحرانیت بالاتر است. با اختصاص دادن اولویت بیشتر به وظیفه ۱ نسبت به وظیفه ۲، می‌توان تأیید کرد که هر دو وظیفه در سطوح بحرانیت مورد نظر خود مهلت‌های خود را برآورده می‌کنند؛ بنابراین، سیستم با یک الگوریتم اولویت ثابت قابل زمان‌بندی‌ است [5].

اکنون EDF را در نظر بگیرید و بیایید روی کار دوم از وظیفه ۱ تمرکز کنیم. به یاد داشته باشید که معنای مدل وظیفه چند بحرانیتی ایجاب می‌کند که تمام مقادیر WCET مورد استفاده برای تضمین اینکه یک وظیفه مهلت خود را برآورده می‌کند، همان سطح اطمینان مورد نیاز توسط وظیفه را داشته باشد. وظایفی که اولویت بیشتری نسبت به کار دوم از وظیفه ۱ دارند شامل کار اول از وظیفه ۲ و به طور غیر مستقیم، کار اول از وظیفه ۱ هستند. در یک سیستم بحرانی مختلط، باید از مقادیر WCET های C1(2) و C2(2) برای تعیین اینکه آیا وظیفه ۱ مهلت‌های خود را برآورده می‌کند، استفاده شود. اما در شبیه‌سازی EDF با این تخمین‌های WCET، به راحتی می‌توان دید که کار دوم از وظیفه ۱ در لحظه زمانی 8 مهلت خود را از دست می‌دهد (از آنجا که کار اول از وظیفه ۱ در بازه [0, 2) و کار اول از وظیفه ۲ در بازه [2, 7) اجرا می‌شوند، در نتیجه فقط یک واحد اجرایی، به جای دو واحد مورد نیاز، برای کار دوم از وظیفه ۱ قبل از مهلتش باقی می‌ماند) [5].

پس از آن، بررسی‌های بیشتری روی الگوریتم EDF انجام شد. به عنوان مثال، پارک و کیم[[21]](#footnote-21) نسخه تغییر یافته‌ای از آن را تحت عنوان CBEDF[[22]](#footnote-22) ابداع کردند. سپس، اکبرگ و یی[[23]](#footnote-23) مدلی تغییر یافته از الگوریتم EDF را ارائه دادند که شباهت‌های زیادی به EDF-VD داشت. این الگوریتم که بر پایه اولویت‌بندی ثابت استوار است، برای هر وظیفه HI، دو اولویت واقعی و مجازی را در نظر می‌گیرد. در حالت عادی فعالیت سیستم، وظایف HI با اولویت مجازی زمان‌بندی می‌شوند و در صورت تغییر حالت سیستم، وظایف LO حذف شده و وظایف HI با اولویت واقعی زمان‌بندی می‌شوند. این الگوریتم برای بیش از دو سطح بحرانیت و همچنین برای مدل‌های وظیفه‌ای که در آن‌ها عواملی به جز WCET تغییرپذیر بودند، تعمیم داده شد. پس از بررسی‌های بیشتر روی الگوریتم‌های ابداعی و همچنین ویرایش آزمون زمان‌بندی EDF و ارائه الگوریتم‌هایی مانند QPA[[24]](#footnote-24) که همگی بر اساس ایجاد مهلت‌های مجازی برای وظایف بودند، باروآح و همکارانش در سال 2011، الگوریتم EDF-VD را برای زمان‌بندی سیستم‌های بحرانی مختلط دو بحرانیتی با مهلت ضمنی ارائه دادند. پس از آن، تحقیقات بر روی EDF-VD ادامه یافت و برای مدل‌های وظیفه‌ای با مهلت و تناوب متغیر و همچنین k سطح بحرانیت تعمیم یافت [1].

لیپاری و بوتازو[[25]](#footnote-25) با استفاده از روش‌های مبتنی بر رزرواسیون[[26]](#footnote-26) و تغییر الگوریتم EDF، الگوریتمی را ارائه می‌دهند که مهلت وظایف HI را به گونه‌ای تغییر می‌دهد که ظرفیت باقی‌مانده برای اجرای وظایف LO را بیشینه کند. سو[[27]](#footnote-27) و همکارانش نیز روش‌هایی مبتنی بر مدل وظیفه‌ی ارتجاعی را معرفی کرده است. در این روش‌ها، برای افزایش قابلیت زمان‌بندی‌پذیری مجموعه وظایف، ممکن است در یک یا چند ویژگی آن‌ها تغییراتی اعمال شود. سو در روش خود، با افزایش تناوب وظایف LO (کاهش فرکانس آن‌ها)، سطح سرویس حداقلی را برای وظایف LO در نظر می‌گیرد. در صورتی که پس از اجرای وظایف HI، زمان اضافی باقی بماند، فرکانس وظایف LO افزایش می‌یابد. تحلیل جایگزین برای سیستم‌های بحرانی مختلط زمان‌بندی شده توسط EDF توسط مهدیانی و مسرور و همچنین سانتینلی[[28]](#footnote-28) و همکاران ارائه شده است. آن‌ها از چندین منحنی محدودیت تقاضا[[29]](#footnote-29) استفاده می‌کنند تا تحلیل حساسیتی را ارائه دهند که می‌تواند برای ایجاد تعادل بین استفاده از منابع و قابلیت زمان‌بندی اعمال شود. اشمیت[[30]](#footnote-30) و همکاران نیز از مقیاس‌بندی غیر یکنواخت مهلت‌ها[[31]](#footnote-31) برای بهبود کیفیت خدمات سیستم‌های زمان‌بندی شده توسط EDF استفاده می‌کنند [1].

### گسترش EDF به EDF-VD

از بین تمامی الگوریتم‌های ارائه شده بر اساس EDF که در بالا به آن‌ها اشاره شد، الگوریتم EDF-VD الگوریتمی شناخته‌شده‌تر است که به دلیل کیفیت بهتر آن نسبت به سایرین، توجه بیشتری را دریافت کرده‌است [1]. این الگوریتم ابتدا برای مجموعه‌ای از کارهای مستقل ارائه شد [6]و پس از آن برای مجموعه‌ای از وظایف بی‌درنگ اسپورادیک که تعداد نامحدودی کار را منتشر می‌کنند، تعمیم داده شد [7]. در ادامه به بررسی این الگوریتم بر روی مجموعه‌ای از وظایف اسپورادیک می‌پردازیم.

### تعریف MC-Schedulability

ابتدا تعاریف زمان‌بندی‌ را برای یک مجموعه وظایف بحرانی مختلط گسترش می‌دهیم.

* **قابلیت زمان‌بندی در سیستم‌های بحرانی مختلط (MC-Schedulability):** یک الگوریتم (برخط) یک سیستم وظیفه‌ای اسپورادیک τ را به طور صحیح زمان‌بندی می‌کند اگر بتواند هر دنباله‌ی کار تولید شده توسط τ را به گونه‌ای زمان‌بندی کند که اگر سطح بحرانیت سناریوی مربوطه Γ باشد، آنگاه تمام وظایف با سطح حداقل Γ بین زمان انتشار و مهلت خود تکمیل شوند [7].
* یک سیستم **قابل زمان‌بندی در بحرانیت مختلط (MC-Schedulable)** نامیده می‌شود اگر الگوریتم صحیحی برای زمان‌بندی آن وجود داشته باشد [7].

### شرط لازم برای MC-Schedulability

از آنجایی که الگوریتم EDF دارای تحلیلی مبتنی بر بهره‌وری سیستم می‌باشد، لازم است تا تعاریف بهره‌وری را نیز برای سیستم‌های بحرانی مختلط گسترش دهیم.

* **بهره‌وری وظیفه** j در سیستمی با سطح بحرانیت k، از طریق رابطه‌ي ۳-۹ محاسبه می‌شود [7]:

|  |  |
| --- | --- |
| ۹-۳) |  |

* **بهره‌وری مجموعه‌ی وظایف** با درجه‌ی بحرانیت i در سیستمی با سطح بحرانیت k از طریق رابطه‌ی ۳-۱۰ محاسبه می‌شود [7]:

|  |  |
| --- | --- |
| ۱۰-۳) |  |

می‌دانیم که در مورد یک سطح بحرانیت واحد (سیستم‌های سنتی)، یک مجموعه وظایف با مهلت ضمنی بر روی m پردازنده با سرعت σ قابل اجرا است اگر و تنها اگر U1(1) ≤ σm و برای همه‌ی j ها uj(1) ≤ σ باشد. با تعمیم این مفهوم به سیستم‌های بحرانی مختلط، می‌توانیم به **شرط لازم** **MC-Schedulability** برسیم. برای سیستمی بحرانی مختلط متشکل از K درجه‌ی بحرانیت، سیستم زمان‌بندی‌‌پذیر است اگر رابطه‌ی ۳-۱۱ برقرار باشد [7]:

|  |  |
| --- | --- |
| ۱۱-۳) |  |

### نحوه عملکرد الگوریتم EDF-VD

اگر رابطه‌ی ۳-۱۱ برقرار باشد به سراغ زمان‌بندی سیستم با الگوریتم EDF می‌رویم. الگوریتم EDF-VD که متغیری از الگوریتم EDF است برای سیستمی دو-بحرانیتی که روی پردازنده‌ی تک‌هسته‌ای با سرعت واحد اجرا می‌شود، به صورت زیر و در دو حالت عمل می‌کند [7]:

۱) اگر ، آنگاه الگوریتم EDF را برای مهلت‌های تغییر نیافته (اصلی) وظایف اعمال کنید. به محض رسیدن سیستم به سطح 2 (HI)، یعنی اجرای یک وظیفه بیش از WCET خود در سطح 1 (LO)، وظایف موجود در سطح ۱ (LO) را لغو کنید.

۲) اگر حالت ۱ برقرار نباشد و رابطه‌ی \* برقرار باشد، آنگاه \* را تنظیم کنید. سپس در حالی که سیستم در سطح 1 (LO) است: برای وظایف سطح ۲ (HI)، \* را تعریف کنید (مهلت مجازی وظایف سطح ۲)؛ مهلت‌ها را با افزودن \* به زمان انتشار هر کار از وظایف سطح ۲ (HI) را بازتعریف کنید، مهلت‌های کارهای وظایف موجود در سطح ۱ (LO) را همانطور که بوده‌اند باقی بگذارید و EDF را برای مهلت‌های تغییر یافته اعمال کنید.

به محض رسیدن سیستم به سطح 2 (HI)، وظایف موجود در سطح ۱ (LO) را لغو کنید؛ مهلت‌های هر کار از وظایف سطح ۲ (HI) را با افزودن pi به زمان انتشار آن‌ها بازنشانی کنید؛ EDF را برای این مهلت‌های (اصلی) اعمال کنید.

بنابراین شرط کافی برای آنکه یک سیستم دو-بحرانیتی با استفاده از الگوریتم EDF-VD روی پردازنده‌ای تک‌هسته‌ای با سرعت واحد قابل زمان‌بندی یا  MC-Schedulable باشد، به شکل رابطه‌ی ۳-۱۲ تعریف می‌شود [7].

|  |  |
| --- | --- |
| ۱۲-۳) |  |

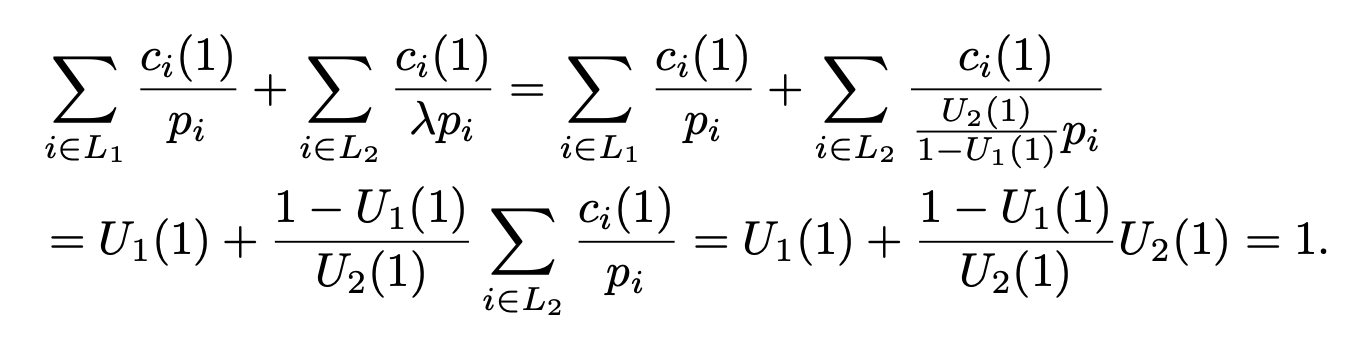
باروآح و همکارانش این رابطه را برای k سطح بحرانیت نیز تعمیم داده‌اند و آن را با نام EDF-VD(K) معرفی می‌کنند اما بررسی آن الگوریتم از شرح این گزارش خارج است [7].

### اثبات شرط کافی MC-Schedulability با الگوریتم EDF-VD

در اینجا رابطه‌ی ۳-۱۲ را به عنوان یک شرط کافی برای تحلیل زمان‌بندی‌پذیری یک مجموعه وظایف دو-بحرانیتی بر روی پردازنده‌ای تک‌هسته با سرعت واحد است، اثبات می‌کنیم. برای اثبات این رابطه باید آن را برای هر دو حالت الگوریتم EDF-VD اثبات کنیم اما حالت اول که در آن یک حالت بدیهی‌ است، پس به سراغ اثبات حالت – در سناریو‌های مختلف می‌رویم [7].

**سناریوی اول: سیستم در سطح ۱ (LO) قرار دارد**

در این سناریو باید نشان دهیم که سیستم با وجود اینکه مهلت وظایف سطح ۲ به – تغییر کرده‌است همچنان در سطح ۱ قابل زمان‌بندی می‌باشد. به این منظور ثابت می‌کنیم:



**سناریوی دوم: سیستم در سطح ۲ (‌HI) قرار دارد**

در این حالت، سیستم تنها شامل وظایف سطح ۲ می‌باشد که با مهلت اصلی خودشان زمان‌بندی شده‌اند پس باید نشان دهیم که . اثبات این مورد نیز از روی – بدیهی‌ است و با رسیدن به – می‌توانیم نتیجه بگیریم که .

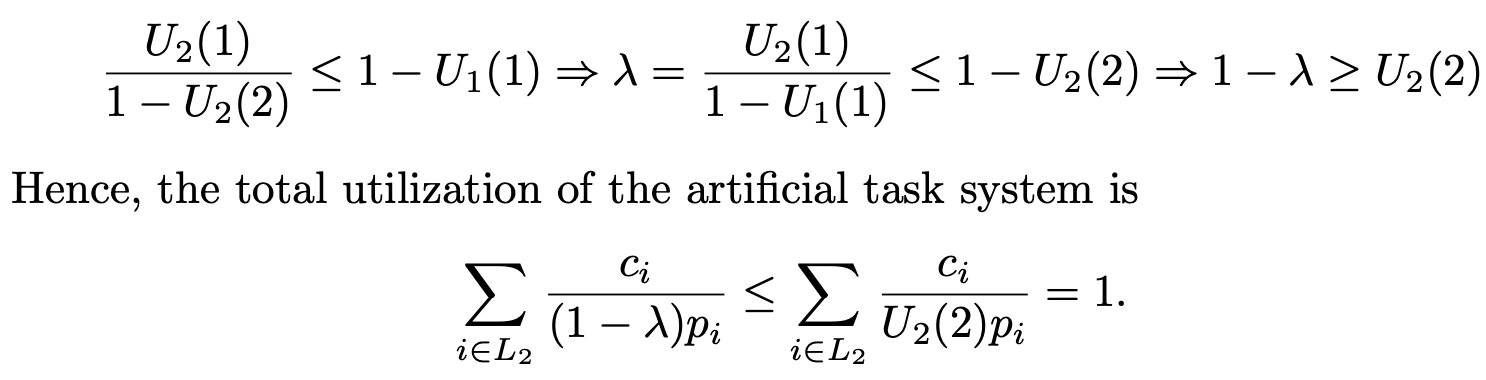
**سناریوی سوم: سیستم از حالت ۱ (LO) به حالت ۲ (HI) می‌رسد**

برای اثبات این سناریو باید نشان دهیم که مجموعه وظایفی که با مهلت‌های مجازی زمان‌بندی کرده‌ایم، با رسیدن سیستم به حالت ۲ در هر لحظه‌ی دلخواه t\*، زمان کافی برای اجرای کامل کارهای فعال (کارهایی که قبل از رسیدن به سطح ۲ منتشر شده‌اند و هنوز کامل نشده‌اند) از وظایف سطح ۲ قبل از فرارسیدن مهلتشان را دارد.

یک کار فعال دلخواه را باj نمایش می‌دهیم. مهلت اصلی این‌ کار برابر است با ‌dj = rj + Tj اما این کار با استفاده از مهلت مجازی d^j = rj + T^j زمان‌بندی شده‌است. از آنجایی که این کار در لحظه‌ی t\* فعال است می‌توان نتیجه گرفت که t\* <= d^j.

|  |  |
| --- | --- |
| ۱۳-۳) |  |

با توجه به رابطه‌ی ۳-۱۳ این بدان معناست که در زمان t∗ برای هر j حداقل = زمان باقی مانده است تا کارش را به موقع تمام کند. می‌توان نشان داد که سیستم وظایف مجازی فقط با وظایف سطح 2 (بنابراین فقط با یک سطح بحرانیت) که دارای دوره‌های -هستند، بهره‌وری ای کمتر از 1 دارد، که بدین معناست که این سیستم می‌تواند به درستی توسط EDF زمان‌بندی ‌شود، که بدین معناست که سیستم اصلی می‌تواند از زمان t∗ به بعد به درستی زمان‌بندی شود. از حالت دوم از رابطه‌ی ۳-۱۲ می‌توانیم ثابت کنیم:



### مقایسه‌ی الگوریتم ‌EDF-VD با سایر الگوریتم‌ها

همانطور که در بخش ‏3 - 1 اشاره شد، نشان داده شده است که مسئله قابلیت زمان‌بندی بحرانیت مختلط چه به صورت قبضه‌ای چه غیر از آن، حتی اگر فقط دو سطح بحرانیت وجود داشته باشد، به شدت NP-Hard است. بنابراین فقط تحلیل کافی و نه دقیق امکان‌پذیر است. گرچه در سال 2017 کاهیل[[32]](#footnote-32) و همکاران ادعا کرده‌اند که مثال نقضی برای اثبات اینکه مسئله بهینگی بحرانیت مختلط به کلاس NP تعلق دارد یافته‌اند اما هنوز هیچ الگوریتم بهینه‌ای برای زمان‌بندی سیستم‌های بحرانی مختلط یافت نشده‌است [1].

برای رویکردها و آزمایش‌هایی که فقط کافی هستند، ارزیابی کیفیت آن‌ها با ضریبی تحت عنوان ضریب افزایش سرعت امکان‌پذیر است.

#### ضریب افزایش سرعت

یک ضریب سرعت به میزان X به طوری که X بزرگ‌تر از ۱ باشد، برای آزمون قابلیت زمان‌بندی الگوریتم S به این معناست که مجموعه وظایفی که بر روی یک پردازنده با سرعت 1 قابل زمان‌بندی نیست، اگر سرعت پردازنده به X افزایش یابد، توسط S قابل زمان‌بندی تلقی خواهد شد [1].

البته، به طور کلی، امکان ندارد بدانیم که آیا مجموعه وظایف بر روی پردازنده اصلی با سرعت 1 قابل زمان‌بندی است چون این نیازمند یک آزمون دقیق است، اما یک طرح و آزمون زمان‌بندی واقعی با ضریب سرعت مثلاً 2 قطعاً بهتر از یکی با ضریب سرعت 10 است [1].

می‌توان این مفهوم را با یک مثال به شکل دقیق‌تری توضیح داد:

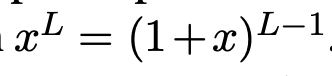
* زمان‌بندی یک مجموعه از کارها با الگوریتم PC دارای ضریب افزایش سرعت نامحدود است. این نتیجه به راحتی با در نظر گرفتن یک سیستم که دارای دو کار است نشان داده می‌شود.کار با بحرانیت LO زمان محاسبه کوچکی به اندازه‌ی 1، و مهلت 2 دارد. کار با بحرانیت HI زمان محاسبه عظیمی به اندازه G و مهلت G + 1 دارد. این دو کار هر دو مهلت‌های خود را برآورده می‌کنند اگر به وظیفه با بحرانیت LO بالاترین اولویت داده شود. اما طبق الگوریتم‌های PC، کار HI حتما باید اولویت بالاتری از کار LO داشته باشد. در این صورت سیستم تنها در حالتی قابل زمان‌بندی خواهد بود که پردازنده بتواند G + 1 کار را قبل از مهلت 2 انجام دهد. نتیجتا، ضریب افزایش سرعت باید (G + 1)/2 باشد. از آنجا که G می‌تواند به اندازه دلخواه بزرگ باشد، ضریب افزایش سرعت نیز به طور مؤثر نامحدود است [1].

#### ضریب افزایش سرعت EDF-VD

برای سیستم‌های زمان‌بندی شده توسط EDF، باروآح و همکاران ثابت می‌کنند که EDF-VD نیز بر روی یک پردازنده تک‌هسته‌ای که سرعت آن با ضریب φ= 1.618 (نسبت طلایی) افزایش یافته است، قابل زمان‌بندی است. آن‌ها همچنین نشان می‌دهند که یک مجموعه متناهی از کارهای مستقل که بر روی m پردازنده یکسان زمان‌بندی شده‌است، با ضریب سرعت φ + 1 + 1/m قابل زمان‌بندی است [7]. سپس نشان می‌دهند که در یک سیستم تقسیم‌شده[[33]](#footnote-33)، ضریب سرعت φ + ε برای هر مقدار ε > 0 قابل دست‌یابی است. در کارهای بعدی، این حد را از φ به 4/3 بهبود می‌بخشند. تحقیقات در زمینه‌ی ضریب افزایش سرعت EDF-VD توسط دیگر محققان نیز ادامه داده شده‌است [1].

* درج اثبات سرعت ای دی اف وی دی

#### ضریب افزایش سرعت سایر الگوریتم‌های قابل رقابت با EDF-VD

برای زمان‌بندی کارهای مستقل با الگوریتم اولویت ثابت، یک الگوریتم اختصاص اولویت و تحلیل آن با ضریب افزایش سرعت SL (برای L سطح بحرانیت) یافت شده است، که در آن SL ریشه معادله  است. برای سیستم‌های دو-بحرانیتی، نتیجه برابر با نسبت طلایی φ = 1.618 است که این مقدار با ضریب افزایش سرعت EDF-VD برابر است [1]. نکته‌ی حائز اهمیت در این مقایسه، آن است که EDF-VD علاوه ‌بر زمان‌بندی کارهای مستقل، قادر به زمان‌بندی یک مجموعه وظایف اسپورادیک نیز هست.

رویکرد MC-Fluid نیز دارای ضریب سرعت 4/3 است اما هنوز هیچ الگوریتم دیگری ارائه نشده است که ضریب افزایش سرعتی بهتر از 4/3 داشته باشد [1].

### محدودیت‌های EDF-VD

با وجود آنکه الگوریتم EDF-VD الگوریتم قدرتمندی است که با یک ضریب افزایش سرعت قابل‌توجه قادر به زمان‌بندی سیستم‌های بحرانی مختلط است، اما معایب و محدودیت‌هایی دارد که لازم است به شمار آورده شوند:

۱. در این الگوریتم فرض بر آن است که در صورت وقوع خطای زمانی، همه‌ی وظایف با درجه بحرانیت پایین، کنار گذاشته می‌شوند در صورتی که همانطور که در بخش \* به آن اشاره شد، چنین عملکردی برای صنعتی‌سازی زمان‌بندی سیستم‌های بحرانی مختلط مورد پذیرش نیست چرا که وظایف با درجه بحرانیت پایین نیز مهلت و هدف خاصی دارند و باید به گونه‌ای اجرا شوند.

۲. اگر کاری از هر کدام از وظایف از خوش‌بینانه‌ترین بدترین زمان اجرای خود (C(LO)) تجاوز کند، سیستم به حالت بحرانی بالا منتقل می‌شود و فرض می‌شود که تمامی وظایف با بحرانیت بالا رفتاری بدبینانه از خود بروز خواهند داد. این فرض نیز بیش از حد بدبینانه است؛ زیرا تغییرات حالت تمامی وظایف، مستقل از یکدیگر است.

۳. همچنین در حین تغییر حالت سیستم و رسیدن به درجه‌ی بحرانی بالا، فرض می‌شود که تمامی وظایف با بحرانیت بالا در بدبینانه‌ترین زمان اجرای خود (C(HI))، اجرا خواهند شد. این فرض نیز بسیار بدبینانه است چرا که در واقعیت به ندرت پیش می‌آید که یک وظیفه به این حد از زمان اجرا برسد. این امر موجب تخصیص بیش از حد منابع به وظایف با درجه‌ی بحرانیت بالا و هدر رفت منابع می‌شود، دقیقا برعکس هدفی که سیستم‌های بحرانی مختلط تلاش دارند به آن دست‌یابند.

تمامی حالات فوق بسیار بدبینانه هستند و از معایب الگوریتم EDF-VD به حساب می‌آیند. علت بروز این معایب آن است که این الگوریتم‌ سازوکارهای مشخصی برای کاهش نظام‌مند سطح سرویس وظایف از درجه‌ی بحرانیت پایین ندارد و آن‌ها را به یک‌باره از سیستم حذف می‌کند. همچنین سازوکار نظام‌مندی برای افزایش تدریجی منابع اختصاص داده شده به وظایف با درجه‌ی بحرانی بالا وجود ندارد. گرچه تمام این سازوکارها می‌توانند در زمان اجرا اعمال شوند، اما به‌کارگیری آن‌ها در تحلیل‌های زمان‌بندی EDF-VD تاثیرگذار خواهد بود و نیاز به محاسباتی بسیار دقیق‌تر و پیچیده‌تر دارد.

## زمان‌بندی چند هسته‌ای

خب تا الان تمرکزمون رو تک هسته‌ای بود ولی چند هسته‌ای هم داریما

* درج کسشر‌های چت جی‌پی‌تی درباره‌ی انواع رویکردهای چند هسته‌ای
* درج کسشر‌های شیمونی درباره‌ی چند هسته‌ای

# پیاده‌سازی چهارچوب زمان‌بندی

در این بخش، شیوه توسعه نرم‌افزار پروژه توضیح داده خواهد شد.

## مدل سیستم بحرانی مختلط مفروض

## ابزارهای استفاده شده

* زبان جی اس
* ران‌تایم نود جی اس
* کتابخانه ایکس ام‌ال جی اس
* کتابخانه‌ی تایپ اسکریپت
* نرم‌افزار وی اس کد
* یارن

### ساختمان سیمولاتور

#### نمودار کلاس‌ها

* درج عکس
* درج توضیحات

#### فلوچارت اجرای کد

* درج عکس
* درج توضیحات

#### ساختار ورودی‌های برنامه

* ایکس ام ال
* ساختار ایکس ام ال
* مدل تسک ست ورودی و ویژگی‌هاش

#### ساختار خروجی‌های برنامه

* ساختار ایکس ام ال خروجی
* کلاک به کلاک اجرای نرم‌افزار به همراه تک تک ایونت‌هایی که اتفاق افتاده
* نمایش زمان‌بندی تولید شده
* نمایش آمار زمان‌بندی تولید شده

### پیاده‌سازی بخش‌های مختلف برنامه

#### پیاده‌سازی تسک جنریتور

* درج الگوریتم ایجاد تسک
* توضیح الگوریتم یو یونی فست
* توضیح

#### پیاده‌سازی چهارچوب اجرای برنامه

* جنریت شدن تسک‌ها و نحوه‌ی محاسبه‌ی اکچوآل تایم
* نحوه‌ی کار کردن سی پی یو
* نحوه کار کردن ردی کیو
* ایجاد یک نسخه از سیمولاتور

در جدول ‏4‑1، لیستی از فایل‌ها و پوشه‌های مهم پروژه توضیح داده شده است. لازم به ذکر است که علاوه بر موارد ذکر شده در این جدول، فایل‌های دیگری نیز وجود دارند اما تلاش شده تا در این جدول، تنها به بخش‌های کلیدی اشاره شود.

جدول ‏4‑1 توضیح برخی از فایل‌ها و پوشه‌های مهم پروژه

|  |  |
| --- | --- |
| نام فایل یا پوشه | توضیحات |
| admin\_panel/ | فایل سرور مربوط به پنل مدیر و همچنین قالب صفحات این پنل در این پوشه قرار دارند. |
| data/ | اطلاعات مربوط به کاربران سامانه مانند نام و عکس‌های مربوط به هر شخص در این پوشه ذخیره می‌شوند. |
| utils/ | مجموعه‌ای از توابع رایج استفاده شده در پروژه در این پوشه جمع آوری شده است. از انواع این توابع می‌توان به توابح مربوط به احراز هویت، توابع نمایش نوشته‌ها بر روی صفحه نمایش، کنترل قطعات سخت‌افزاری و مدیریت کاربران اشاره کرد. |
| configs.json | تنظیمات پیکره‌بندی سامانه در این فایل قابل مشاهده و تغییر می‌باشد. |
| main.py | این فایل، فایل اصلی پروژه می‌باشد که سامانه را اجرای آن شروع به کار می‌کند. |
| requirement.txt | لیستی از کتابخانه‌های لازم برای این پروژه در این فایل جمع‌آوری شده که با اجرای دستور pip install -r requirements.txt، این کتابخانه‌ها نصب می‌شوند. |

#### پیاده‌سازی الگوریتم زمان‌بندی EDF-VD

* پیاده‌سازی فیزیبیلیتی تست
* پیاده‌سازی الگوریتم ای دی اف
* پیاده‌سازی الگوریتم ای‌دی اف وی دی

### تنظیمات شبیه‌سازی و ویژگی‌های قابل تغییر

* فلان
* بیسار
* بهمان

بسیاری از سازمان‌های بزرگ در حیطه فناوری که در زمینه هوش مصنوعی فعالیت می‌کنند، مدل‌های از پیش تمرین داده شده را در اختیار عموم قرار می‌دهند. برای تشخیص چهره از دو مدل از پیش تمرین داده شده استفاده می‌کنیم.

## اجرای نرم‌افزار شبیه‌سازی و ارزیابی خروجی

* آماده‌سازی پیش از اجرا
  + نصب کتابخانه
  + تنظیم ویژگی‌های سیستم
* مراحل اجرا چنین است.
* نمونه‌ی ورودی چنین است.
* نمونه‌ی خروجی چنین است.

## محدودیت‌ها و برنامه‌های آینده

* فلان رو میشه اضافه کرد
* بیسار رو میشه اضافه کرد.
* تک هسته‌ایه
* وابستگی‌ها رو مدیریت نمی‌کنه

# ارزیابی پروژه

## استفاده از تحلیل مولفه‌های اصلی برای بررسی کیفیت شناسایی چهره

همانطور که گفته شد، هر چهره یک نقطه در فضای ۱۲۸ بعدی می‌باشد که انتظار می‌رود نقاط مربوط به یک شخص در این فضا در نزدیکی یکدیگر قرار داشته باشند. برای مصور‌سازی و درک بهتر این موضوع، نیاز داریم داده‌ها را به نحوی تغییر دهیم که قابل رسم باشد. برای این کار از  تحلیل مؤلفه‌های اصلی[[34]](#footnote-34) استفاده می‌کنیم که تبدیلی در فضای برداری است و می‌تواند مجموعه داده‌های بزرگ با تعداد بعد یا ویژگی زیاد را تحلیل کند، تفسیرپذیری داده‌ها با حفظ حداکثر مقدار اطلاعات انجام دهد و تجسم داده‌های چند بعدی را فراهم می‌کند. تحلیل مؤلفه‌های اصلی، در واقع یک تکنیک آماری برای کاهش ابعاد یک مجموعه داده است. این کار با تبدیل خطی داده‌ها به یک سیستم مختصات جدید انجام می‌شود که بیشتر تغییرات در داده‌ها را می‌توان با ابعاد کمتری نسبت به داده‌های اولیه توصیف کرد. در **Error! Reference source not found.**، خروجی تکنیک تحلیل مولفه‌های  اصلی برای دو بعد و سه بعد آورده شده است. همان‌طور که در این شکل مشاهده می‌شود، عکس‌هایی که از هر شخص گرفته می‌شوند با یک رنگ خاص نمایش داده شده‌اند و همانطور که انتظار می‌رود، نقاط متناظر با چهره‌های هر شخص در این فضا‌های برداری در مجاورت یکدیگر قرار دارند.

## تاثیر پیش‌پردازش پیش از بردار‌سازی در زمان تمرین مدل شناسایی چهره

همانطور که در **Error! Reference source not found.** گفته شد، دو تغییر روی عکس‌های چهره‌هایی که در سامانه ذخیره می‌شوند اعمال می‌شود:

# جمع‌بندی، نتیجه‌گیری و کار‌های آینده

## جمع‌بندی و نتیجه‌گیری

شناسایی چهره، یکی از چالش‌برانگیزترین مسائل دنیای امروز محسوب می‌شود چرا که معمولا از آن برای احراز هویت افراد استفاده می‌شود و احراز هویت غلط ممکن است در برخی موارد، موجب به وجود آمدن خسارت شود. اولین گام فرایند شناسایی چهره، تشخیص چهره می‌باشد. دو الگوریتم تشخیص چهره در این پروژه بیان شده است: الگوریتم هیستوگرام شیب‌های جهت‌دار و الگوریتم شبکه عصبی در هم تنیده.

طبق بررسی‌های انجام شده از عملکرد سامانه، نتایج زیر حاصل شد:

* الگوریتم هیستوگرام شیب‌های جهت‌دار نسبت به الگوریتم شبکه عصبی در هم تنیده در تشخیص چهره‌های با زاویه نسبت به دوربین عملکرد خوبی ندارد ولی سربار کمتری دارد و برای اجرا روی بورد رزبری‌پای که منابع محدودی دارد مناسب‌تر است.
* بر اساس ارزیابی‌های انجام شده، دقت شناسایی چهره در این سامانه "96.487٪ " می‌باشد که درصد قابل قبولی است.
* زیاد کردن پارامتر‌هایی که به دقت سامانه کمک می‌کنند، باعث کند شدن تصویر سامانه می‌شوند که نامطلوب است، در نتیجه باید از زیاد کردن بی‌رویه آن جلوگیری نمود.

## پیشنهادها

* یکی از بزرگترین ایراد‌های فرایند احراز هویت به کمک چهره، تشخیص مثبت نادرست[[35]](#footnote-35) است. به‌عنوان مثال، ممکن است تصویر یک شخص که عضو سامانه است را داخل یک تلفن همراه داشته باشیم و چهره شخص مورد نظر را از طریق تلفن مقابل سامانه قرار دهیم. در این صورت، سامانه متوجه این موضوع نخواهد شد و هویت او را تایید می‌کند. یک پیشنهاد، افزودن حسگری برای سنجیدن فاصله چهره تا دوربین است. اگر از کاربران خواسته شود که در فاصله‌ای معین از دوربین سامانه قرار گیرند، می‌توان گفت اندازه چهره‌های تشخیص داده شده در محدوده معینی قرار خواهد گرفت. اگر احراز هویت را در صورتی انجام دهیم که فاصله چهره از سامانه و همچنین اندازه چهره تشخیص داده شده معین باشد، در این صورت می‌توان از برخی موارد مثبت نادرست جلوگیری کرد.
* پیشنهاد می‌شود

[4] [5] [6] [4] [4] [6] [3] [7] [3] [3] [3] [1] [8] [9] [10] [11] [12] [2]

# مراجع

|  |  |
| --- | --- |
| [1] | A. Burns and R. I. Davis, Mixed Criticality Systems - A Review (13th Edition), York: The University of York, 2022, p. 97 (total pages). |
| [2] | G. Buttazzo, Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications, Springer US, 2010, p. 426 (total pages). |
| [3] | S. K. Baruah, A. Burns and R. I. Davis, "Response-time analysis for mixed criticality systems," in *IEEE Real-Time Systems Symposium (RTSS)*, 2011. |
| [4] | S. Vestal, "Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance," in *Real-Time Systems Symposium (RTSS)*, 2007. |
| [5] | M. Mahdiani, Advanced Scheduling Techniques for Mixed-Criticality Systems, Chemnitz: Technische Universitat, 2020. |
| [6] | S. K. Baruah, V. Bonifaci, G. d’Angelo, A. Marchetti-Spaccamela and S. van Der Ster, "Mixed-Criticality Scheduling of Sporadic Task Systems [DOI: 10.1007/978-3-642-23719-5\_47]," in *19th Annual European Symposium on Algorithms (ESA 2011)*, Saarbruecken, 2011. |
| [7] | S. K. Baruah and S. Vestal, "Schedulability analysis of sporadic tasks with multiple criticality specifications," in *ECRTS*, 2008. |
| [8] | K. O.R., A. H. and B. Zhao, "On partitioned scheduling of fixed-priority mixed-criticality task sets," in *International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2011. |
| [9] | C. Gu, N. Guan, Q. Deng and W. Yi, "Partitioned mixed-criticality scheduling on multiprocessor platforms," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2014. |
| [10] | P. Ekberg and W. Yi, "Bounding and shaping the demand of generalized mixed-criticality sporadic task systems," *Real-Time Systems,* vol. 50, no. 1, pp. 48-86, 2013. |
| [11] | B. A., T. Fleming and B. S., "Cyclic executives, multi-core platforms and mixed criticality applications," in *Euromicro Conference on Real-Time Systems (ECRTS)*, 2015. |
| [12] | S. Baruah, B. Chattopadhyay, H. Li and I. Shin, "Mixed-criticality scheduling on multiprocessors," *Real-Time Systems,* vol. 50, no. 1, pp. 142-177, 2014. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| شماره وظیفه (i) | مهلت نسبی انجام وظیفه (D) | دوره تناوب وظیفه (T) | سطح بحرانی وظیفه (L) | خوش‌بینانه‌ترین بدترین زمان اجرای وظیفه (C(LO)) | بدبینانه‌ترین بدترین زمان اجرای وظیفه (C(HI)) |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

1. Runtime Monitoring [↑](#footnote-ref-1)
2. Overhead [↑](#footnote-ref-2)
3. Sigrist [↑](#footnote-ref-3)
4. Preemptive Fixed Priority [↑](#footnote-ref-4)
5. Slack Time [↑](#footnote-ref-5)
6. Period Transformation [↑](#footnote-ref-6)
7. Criticality Monotonic Priority Ordering [↑](#footnote-ref-7)
8. Criticality Monotonic [↑](#footnote-ref-8)
9. Partitioned Criticality [↑](#footnote-ref-9)
10. Joseph-Pandya [↑](#footnote-ref-10)
11. Audsley [↑](#footnote-ref-11)
12. Criticality Inversion [↑](#footnote-ref-12)
13. Baruah [↑](#footnote-ref-13)
14. Criticality Level of a Behavior [↑](#footnote-ref-14)
15. Budget Enforcement [↑](#footnote-ref-15)
16. Upperbound [↑](#footnote-ref-16)
17. Dual Criticality [↑](#footnote-ref-17)
18. Adaptive Mixed Criticality – Response Time Bound [↑](#footnote-ref-18)
19. Adaptive Mixed Criticality – Maximum Interference [↑](#footnote-ref-19)
20. Hybrid-Priority [↑](#footnote-ref-20)
21. Park and Kim [↑](#footnote-ref-21)
22. Criticality Based Earliest Deadline First [↑](#footnote-ref-22)
23. Ekberg and Yi [↑](#footnote-ref-23)
24. Quick convergence Processor-Demand Analysis [↑](#footnote-ref-24)
25. Lipari and Buttazzo [↑](#footnote-ref-25)
26. Reservation-Based [↑](#footnote-ref-26)
27. Su [↑](#footnote-ref-27)
28. Santinelli [↑](#footnote-ref-28)
29. Demand-Bound Curve [↑](#footnote-ref-29)
30. Schmidt [↑](#footnote-ref-30)
31. Non-Uniform Deadline Scaling [↑](#footnote-ref-31)
32. Kahil [↑](#footnote-ref-32)
33. Partitioned [↑](#footnote-ref-33)
34. Principal Component Analysis [↑](#footnote-ref-34)
35. False Positive [↑](#footnote-ref-35)