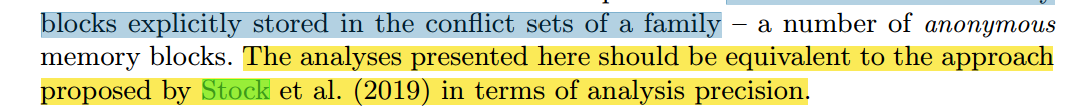
**精确持久性分析：**

2019年RTSS：<Cache_Persistence_Analysis_Finally_Exact.pdf>

**精确Must和持久性分析（LRU）：**

2019年：

2022年RTS：<Precise, Efficient, and Context-Sensitive Cache Analysis.pdf> 这篇是2020年RTNS的拓刊：<Precise and Efficient Analysis of Contex-Sensitive Cache Conflict Sets.pdf>相比较2022年这篇主要做时间优化：



**多核WCET分析：**

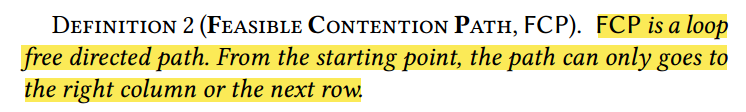
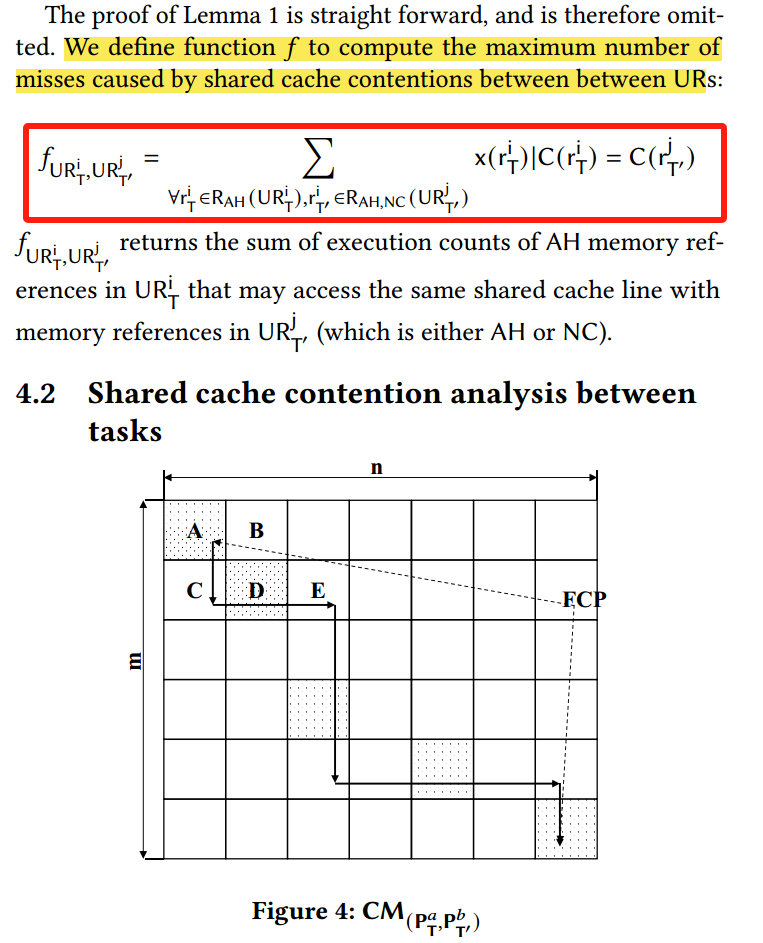
2012年RTS：<Timing Analysis of Concurrent Programs Running on.pdf>

2016年TECS：<Fast and Precise Worst-Case Interference Placement for Shared Cache Analysis.pdf>

2022年DAC：<1_Precise and Scalable.pdf>

2024年RTS：<Timing‑aware analysis of shared cache interference .pdf>

2022年DAC待讨论的问题：



F

**第一个问题：f的计算**

箭头并行任务的访存与分析任务的访存(riT)映射到了同一个缓存集合（即发生冲突）,riT被视为因为冲突造成被逐出（AH->NC）。显然这里没有考虑关联度k和riT的age，更精确的做法是当并行任务有> k-age(riT)个内存引用与riT争用同一个缓存集则riT（AH->NC）。

F

为什么这么做？

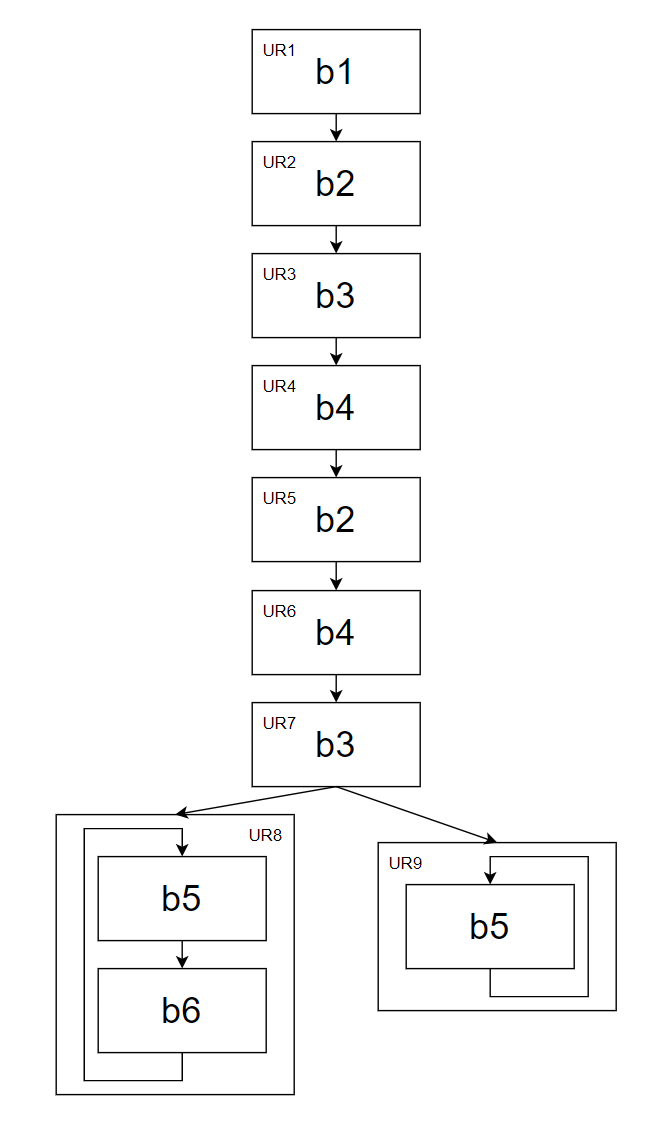
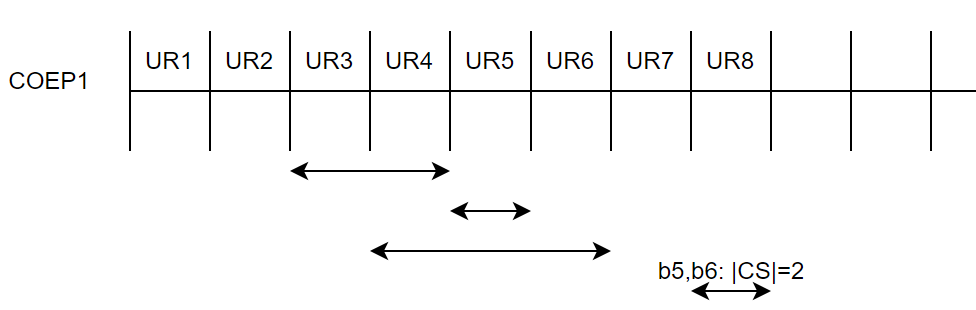
答：张伟的做法是一种安全的上界，虽然论文里面没有明确说明，但我认为他这样做的原因是为了让这个问题能用动态规划求解。

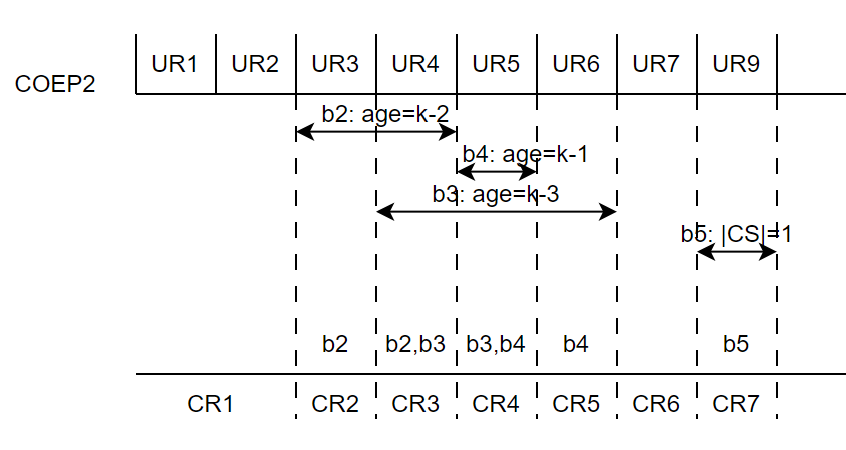
动态规划求解：

如果考虑k和age，以上图为例子，k-age(r2T)=2, E发生是否会造成r2T被逐出，是受D是否发生影响的，前面选择影响后面的选择，这样的问题是不能用动态规划求解的。

**第二个问题：**FCP是否是一个最坏的情况？

FCP是如上图所示只能向下和向右走的路。F在路径上，但是F左边的两个块（用红色箭头标注）也是可能会发生的，但FCP却没有计算。我们用下图来解释一个干扰块会造成多个冲突发生。如果干扰块在UR4执行期间发生，那么它将会与b2,b3发生争用，





假设有两个连续的干扰块x,y在UR4执行期间发生那么路径图应该如下，FCP显然是不能覆盖这种情况的。

|  |  |  |
| --- | --- | --- |
|  | x | y |
| UR5(b2) | 1 | 1 |
| UR6(b4) | - | - |
| UR7(b3) | 1 | 1 |

**换一个方式组织UR：**

UR带的信息太少了，按COEP2图例中CR的方式来组织会让CR块带上信息。如CR1，在CR1执行期间，并行程序的执行不会造成任何块被逐出。而在CR2执行期间，如果并行程序与b2争用缓存集合则可能会造成b2被逐出（AH->NC的条件与之前一样）。好处是，CR和UR一样具有时序信息（即，CR(i+1)一定在CRi之后发生），仍然可以通过时间上的互斥性去收紧WCET。

**问题建模**

已知争用集M(T,c)的定义（我们定义的）是干扰任务T争用缓存集c的block。现改变一下以适应性新的模型，M(T\_UR,c)，T\_UR是干扰任务T中时间上可连续发生的UR集合。我们需要求解这个集合以找到最坏情况：

1. T\_UR中的UR在时间上是可连续发生的。
2. T\_UR的集合是两两互斥的
3. ...

**算法建模**

有m张卡片，每张卡片具有几种属性，一共有s种属性。

有n个领奖人，每个人有x张兑奖凭证，兑奖凭证上告诉他们需要集齐y张某种属性的卡片才能兑换一定积分（a）的奖励。注意，领奖人之间可能拥有相同的兑奖凭证，但一张兑奖凭证只能被领取一次。同一个人可以对不同的兑奖凭证进行卡片的复用。

m张卡片被按顺序摆放在桌子上，为了得到奖励，他们现在排队并按规则去领取卡片。领取卡片的规则：1.对每张卡片可以选择拿取或者丢弃2.拿取或丢弃某张卡片时，必须保证这张卡片之前的卡片已经被拿去或者丢弃了。

现要求解怎样拿取卡片才能让领奖人总积分最高。

m,n<=1000

0<=x<=5

1<=y<=8

1<=s<=64 （注：大多数的卡片只有一种属性）

1<=a<=1000 （注：大多数的积分都是1）