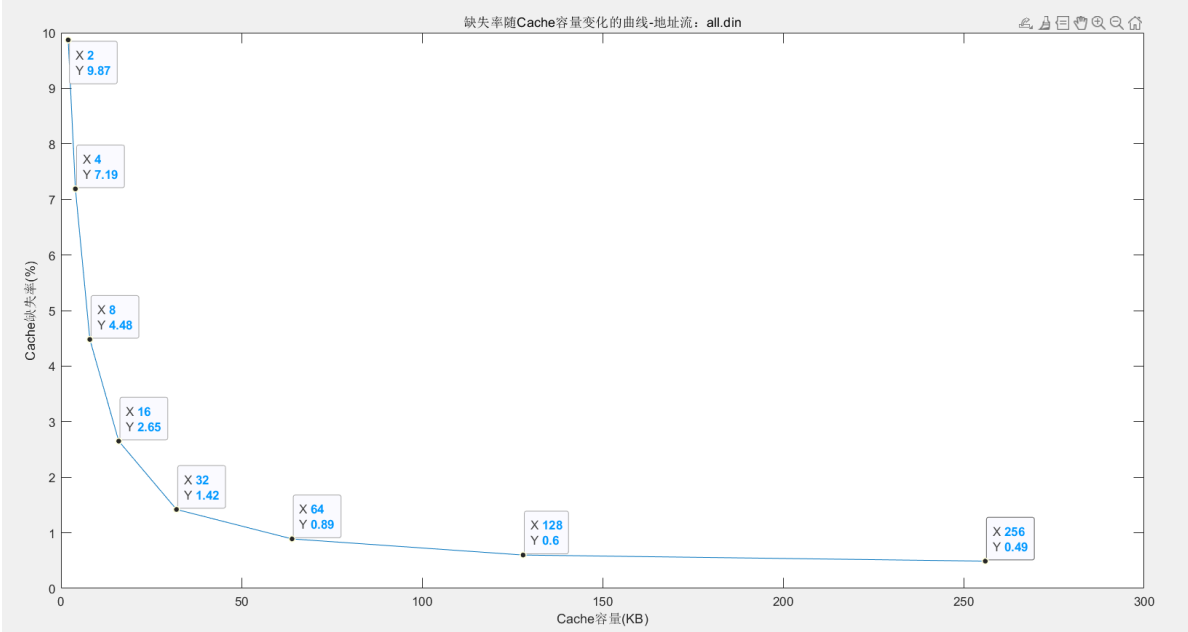
Cache容量对缺失率的影响

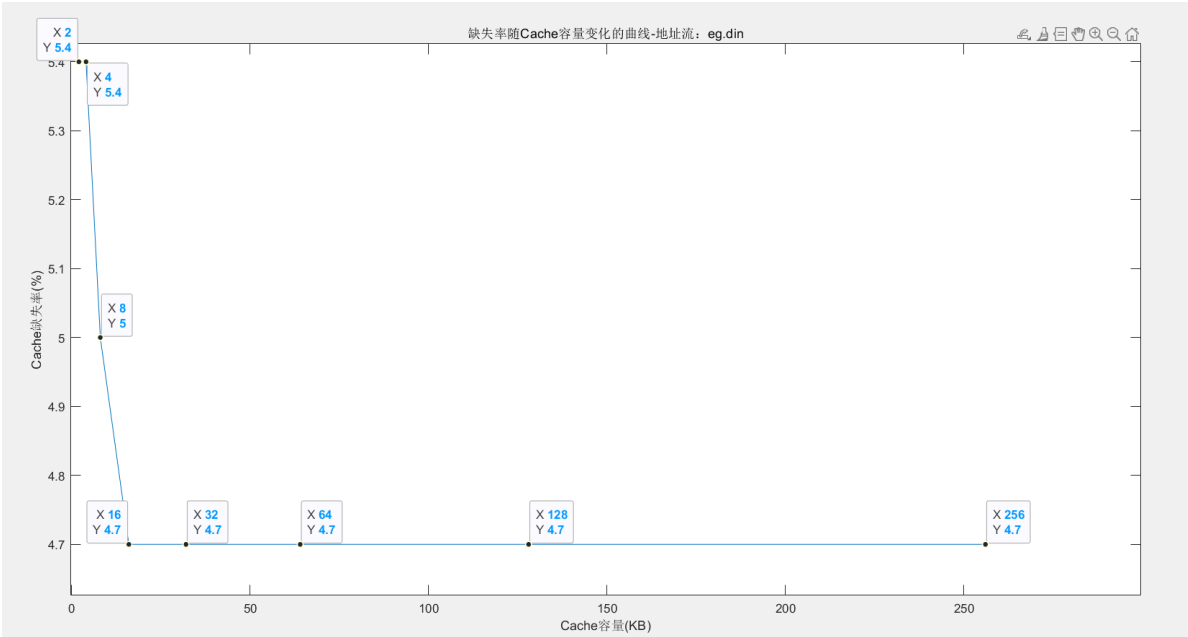
表2.1 不同容量时的Cache缺失率

Cache容量(KB)	2	4	8	16	32	64	128	256
缺失率	9.87%	7.19%	4.48%	2.65%	1.42%	0.89%	0.60%	0.49%

地址流文件名： all.din



结论：单独提升Cache容量对减小缺失率的影响具有边际效应，每次Cache容量翻倍后提升是逐渐减少的，如图可以看出斜率在逐渐减少，后续已经快趋于0。可见在只考虑Cache容量情况下，Cache容量的值应保持一定水平后不必提升，一味提升其容量并不意味着一定带来很好的性能提升，反而可能因成本因素得不偿失。事实上，如果换成eg.din的地址流文件，Cache容量在16KB后就再无提升。



相联度对缺失率的影响

表2.2 容量为64KB、不同相联度时的Cache缺失率

相联度	1	2	4	8	16	32
缺失率	1.97%	1.15%	0.99%	0.93%	0.92%	0.91%

地址流文件名：cc1.din

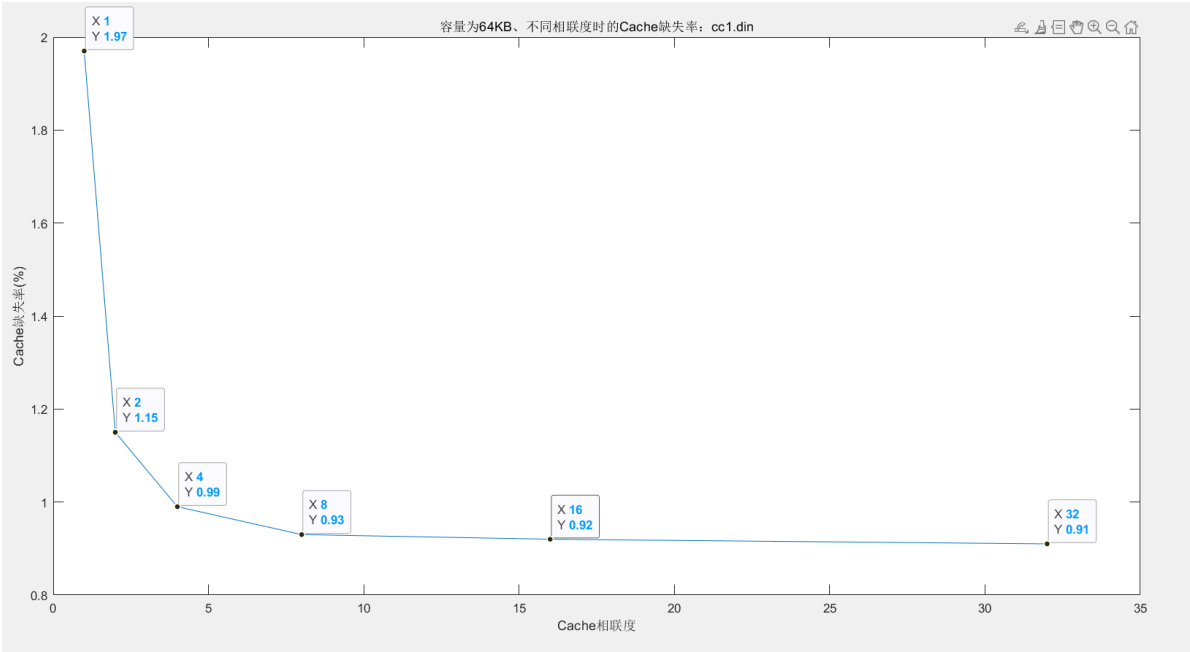
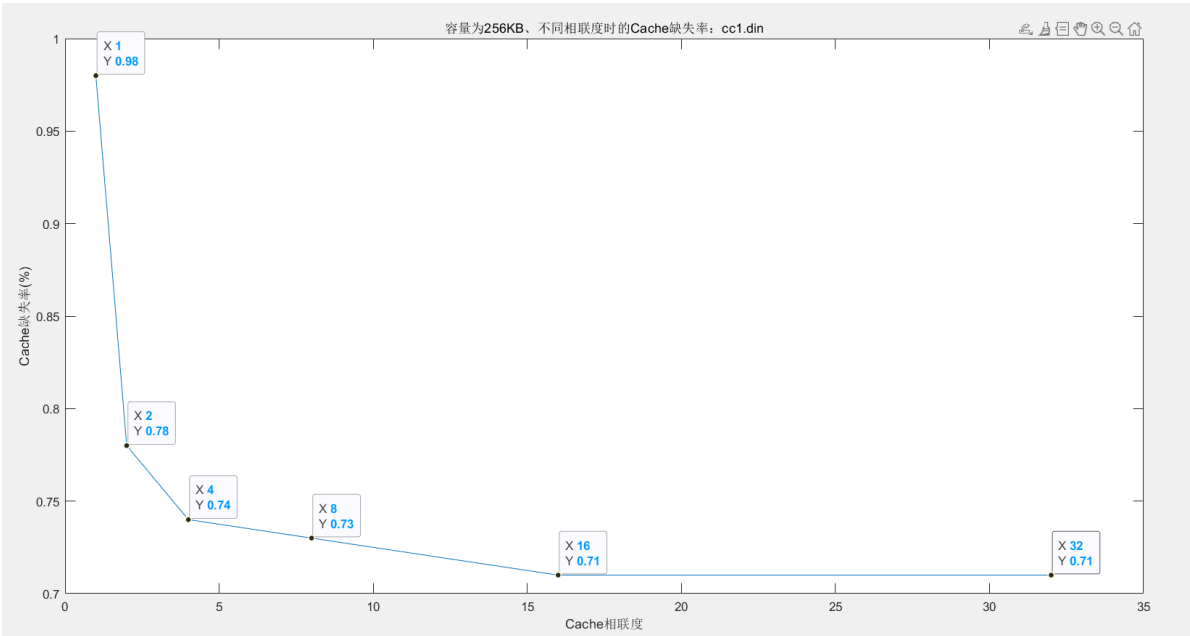


表2.3 容量为256KB、不同相联度时的Cache缺失率

相联度	1	2	4	8	16	32
缺失率	0.98%	0.78%	0.74%	0.73%	0.71%	0.71%

地址流文件名：cc1.din



结论：由**“表2.2 容量为64KB、不同相联度时的Cache缺失率”**，不难理解课件上如①往上的相联度与8相联度的约等关系，进而推论出与全相联的约等关系。

当然在all.din里也有相同结论。

相联度	1	2	4	8	16	32
缺失率	0.89%	0.53%	0.47%	0.45%	0.44%	0.44%

地址流文件名：all.din

由**"表2.3 容量为256KB、不同相联度时的Cache缺失率"**，不清楚是不是巧合，256KB的直接映像与64KB的4路组相连差不多，尽管此时与②中前提$S_{Cache} \leq 128KB$并不相符。

① S_{Cache}不变，$F_{8\text{路}} \approx F_{\text{全相联}}$

② $S_{\text{Cache}} \leqslant 128\text{KB}$，$S_{\text{Cache}}$的$F_{1\text{路}} \approx S_{\text{Cache}}/2$的$F_{2\text{路}}$

可见，相联度的提升同样面临边际效应，应当选取适当的相联度，不过不同于Cache容量的是，可以推出这一具体的值可能不应大于8路。在Cache容量足够大的情况下，可以适当考虑降低相联度以为提升其他方面的性能让步。

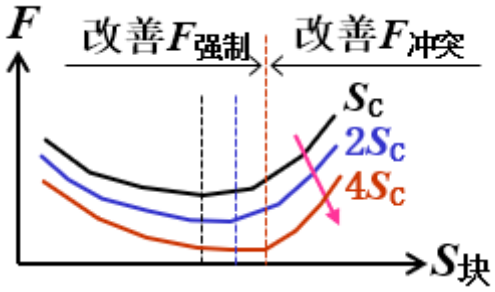
Cache块大小对缺失率的影响

表2.4 各种块大小时的Cache缺失率

----------	Cache容量（KB）				
块大小（B）	**2**	**8**	**32**	**128**	**512**
16	12.02%	5.79%	1.86%	0.95%	0.71%
32	9.87%	4.48%	1.42%	0.60%	0.42%
64	9.36%	4.03%	1.20%	0.43%	0.27%
128	10.49%	4.60%	1.08%	0.35%	0.20%
256	13.45%	5.35%	1.19%	0.34%	0.16%

地址流文件名：all.din

分析：在Cache容量不够时，将块大小提升的过大可能适得其反，如果能保证Cache容量足够大，适当提升块大小还是可以很好地帮助减小Cache缺失率。



例1：表中$T_{传输}$=2CLK/16B，$T_{命中}$=1CLK，选择各容量的$S_{块}$

$S_{块}$及$T_{缺失}$ S_C		4KB	16KB	64KB	256KB
16B	80+2	8.57%	3.94%	2.04%	1.09%
32B	80+4	7.24%	2.87%	1.35%	0.70%
64B	80+8	7.00%	2.64%	1.06%	0.51%
128B	80+16	7.78%	2.77%	1.02%	0.49%
256B	80+32	9.51%	3.29%	1.15%	0.49%

解：$T_A=T_{命中}+F\cdot T_{缺失}$，

$T_{4K/16}$=1+8.57%*82=8.027CLK，

$T_{4K/64}$=1+7.00%*88=7.160CLK，

得 4KB时$S_{块}$=32B，

16KB/64KB/256KB时$S_{块}$=64B

思考②：例1中80→40，结果会变化吗？

结果—$S_{块}$尽量大，值取决于下级MEM的延迟与带宽

└→保持$F\cdot T_{缺失}$不变→┴→$T_{调入}$较小时$S_{块}$较大

替换算法对缺失率的影响

表2.5 LRU和随机替换算法时的Cache缺失率

Cache 容量	相 联 度					
----------	2 路		4 路		8 路	
----------	LRU算法	随机算法	LRU算法	随机算法	LRU算法	随机算法
16KB	1.71%	2.24%	1.33%	2.41%	1.21%	2.84%
64KB	0.53%	0.68%	0.47%	0.72%	0.45%	0.83%
256KB	0.38%	0.39%	0.36%	0.37%	0.36%	0.36%
1MB	0.35%	0.35%	0.35%	0.35%	0.35%	0.35%

地址流文件名：all.din

分析：可以看出在Cache容量达到一定程度（当然也与块大小有关系），比如256KB，无论相联度如何，LRU与RAND间差距都显得比较小了，在1MB时更是没有任何区别。

在16KB，64KB时还可以看出LRU算法相对于RAND算法的优势，且LRU算法很明确随相联度提升会提升命中率，而RAND随机算法则是不确定的。

结合课件，由16KB，64KB对应LRU算法，可以看出随着n（相联度）增大，LRU算法的H（命中率）确实在增大（对应缺失率减小），而RAND算法与n并无明确关系，虽然16KB，64KB时由随n增大而缺失率有所增大，但256KB时却在减小。

*常见算法：

	状态的个数	状态更新的时机	牺牲行的选择	对H的影响
RAND	1个随机数/Cache	块替换时，产生随机数	随机数对应的行	H随机
FIFO	1个计数值/行	块调入时，更新n个值	(n个)值最大的行	H随机
LRU	1个计数值/行	块访问时，更新n个值	(n个)值最大的行	H随n增大
注：n—组相联的路数，即候选行的个数；计数值—刚调入/访问的行清零				