第二十二章-超越物理内存策略

Problem1

问题描述

使用以下参数生成随机地址：-s 0 -n 10,-s 1 -n 10 和-s 2 -n 10。将策略从 FIFO 更改为 LRU,并将其更改为 OPT。计算所述地址追踪中的每个访问是否命中或未命中。

问题解答

FIFO

```
> python2 paging-policy.py -s 0 -n 10
ARG policy FIFO
ARG cachesize 3
....

Access: 8  Hit/Miss?  State of Memory?
Access: 7  Hit/Miss?  State of Memory?
Access: 4  Hit/Miss?  State of Memory?
Access: 2  Hit/Miss?  State of Memory?
Access: 5  Hit/Miss?  State of Memory?
Access: 4  Hit/Miss?  State of Memory?
Access: 7  Hit/Miss?  State of Memory?
Access: 3  Hit/Miss?  State of Memory?
Access: 4  Hit/Miss?  State of Memory?
Access: 5  Hit/Miss?  State of Memory?
```

```
Access: 8  Miss  State of Memory:[8]
Access: 7  Miss  State of Memory:[8,7]
Access: 4  Miss  State of Memory:[8,7,4]
Access: 2  Miss  State of Memory:[7,4,2]
Access: 5  Miss  State of Memory:[4,2,5]
Access: 4  Hit   State of Memory:[4,2,5]
Access: 7  Miss  State of Memory:[2,5,7]
Access: 3  Miss  State of Memory:[5,7,3]
Access: 4  Miss  State of Memory:[7,3,4]
Access: 5  Miss  State of Memory:[3,4,5]
```

```
> python2 paging-policy.py -s 1 -n 10
ARG policy FIFO
ARG cachesize 3

...
```

```
Access: 1  Hit/Miss?  State of Memory?
Access: 8  Hit/Miss?  State of Memory?
Access: 7  Hit/Miss?  State of Memory?
Access: 2  Hit/Miss?  State of Memory?
Access: 4  Hit/Miss?  State of Memory?
Access: 4  Hit/Miss?  State of Memory?
Access: 6  Hit/Miss?  State of Memory?
Access: 7  Hit/Miss?  State of Memory?
Access: 0  Hit/Miss?  State of Memory?
Access: 0  Hit/Miss?  State of Memory?
```

```
Access: 1  Miss  State of Memory:[1]
Access: 8  Miss  State of Memory:[1,8]
Access: 7  Miss  State of Memory:[1,8,7]
Access: 2  Miss  State of Memory:[8,7,2]
Access: 4  Miss  State of Memory:[7,2,4]
Access: 4  Hit   State of Memory:[7,2,4]
Access: 6  Miss  State of Memory:[2,4,6]
Access: 7  Miss  State of Memory:[4,6,7]
Access: 0  Miss  State of Memory:[6,7,0]
Access: 0  Hit   State of Memory:[6,7,0]
```

```
> python2 paging-policy.py -s 2 -n 10
ARG policy FIFO
ARG cachesize 3

......

Access: 9  Hit/Miss?  State of Memory?
Access: 9  Hit/Miss?  State of Memory?
Access: 0  Hit/Miss?  State of Memory?
Access: 0  Hit/Miss?  State of Memory?
Access: 8  Hit/Miss?  State of Memory?
Access: 7  Hit/Miss?  State of Memory?
Access: 6  Hit/Miss?  State of Memory?
Access: 3  Hit/Miss?  State of Memory?
Access: 6  Hit/Miss?  State of Memory?
Access: 6  Hit/Miss?  State of Memory?
```

```
Access: 9  Miss  State of Memory:[9]
Access: 9  Hit   State of Memory:[9]
Access: 0  Miss  State of Memory:[9,0]
Access: 0  Hit   State of Memory:[9,0]
Access: 8  Miss  State of Memory:[9,0,8]
Access: 7  Miss  State of Memory:[0,8,7]
Access: 6  Miss  State of Memory:[8,7,6]
```

```
Access: 3  Miss  State of Memory:[7,6,3]
Access: 6  Hit   State of Memory:[7,6,3]
Access: 6  Hit   State of Memory:[7,6,3]
```

LRU

```
> python2 paging-policy.py -s 0 -n 10 -p LRU
ARG policy LRU
ARG cachesize 3
....

Access: 8  Hit/Miss?  State of Memory?
Access: 7  Hit/Miss?  State of Memory?
Access: 4  Hit/Miss?  State of Memory?
Access: 2  Hit/Miss?  State of Memory?
Access: 5  Hit/Miss?  State of Memory?
Access: 4  Hit/Miss?  State of Memory?
Access: 7  Hit/Miss?  State of Memory?
Access: 3  Hit/Miss?  State of Memory?
Access: 4  Hit/Miss?  State of Memory?
Access: 5  Hit/Miss?  State of Memory?
```

```
Access: 8  Miss  State of Memory:[8]
Access: 7  Miss  State of Memory:[8,7]
Access: 4  Miss  State of Memory:[8,7,4]
Access: 2  Miss  State of Memory:[7,4,2]
Access: 5  Miss  State of Memory:[4,2,5]
Access: 4  Hit   State of Memory:[2,5,4]
Access: 7  Miss  State of Memory:[5,4,7]
Access: 3  Miss  State of Memory:[4,7,3]
Access: 4  Hit   State of Memory:[7,3,4]
Access: 5  Miss  State of Memory:[3,4,5]
```

```
> python2 paging-policy.py -s 1 -n 10 -p LRU
ARG policy LRU
ARG cachesize 3

...

Access: 1  Hit/Miss?  State of Memory?
Access: 8  Hit/Miss?  State of Memory?
Access: 7  Hit/Miss?  State of Memory?
Access: 2  Hit/Miss?  State of Memory?
Access: 4  Hit/Miss?  State of Memory?
Access: 4  Hit/Miss?  State of Memory?
Access: 6  Hit/Miss?  State of Memory?
```

```
Access: 7  Hit/Miss?  State of Memory?
Access: 0  Hit/Miss?  State of Memory?
Access: 0  Hit/Miss?  State of Memory?
```

```
Access: 1  Miss  State of Memory:[1]
Access: 8  Miss  State of Memory:[1,8]
Access: 7  Miss  State of Memory:[1,8,7]
Access: 2  Miss  State of Memory:[8,7,2]
Access: 4  Miss  State of Memory:[7,2,4]
Access: 4  Hit   State of Memory:[7,2,4]
Access: 6  Miss  State of Memory:[2,4,6]
Access: 7  Miss  State of Memory:[4,6,7]
Access: 0  Miss  State of Memory:[6,7,0]
Access: 0  Hit   State of Memory:[6,7,0]
```

```
> python2 paging-policy.py -s 2 -n 10 -p LRU
ARG policy LRU
ARG cachesize 3

......

Access: 9  Hit/Miss?  State of Memory?
Access: 9  Hit/Miss?  State of Memory?
Access: 0  Hit/Miss?  State of Memory?
Access: 0  Hit/Miss?  State of Memory?
Access: 8  Hit/Miss?  State of Memory?
Access: 7  Hit/Miss?  State of Memory?
Access: 6  Hit/Miss?  State of Memory?
Access: 3  Hit/Miss?  State of Memory?
Access: 6  Hit/Miss?  State of Memory?
Access: 6  Hit/Miss?  State of Memory?
```

```
Access: 9  Miss  State of Memory:[9]
Access: 9  Hit   State of Memory:[9]
Access: 0  Miss  State of Memory:[9,0]
Access: 0  Hit   State of Memory:[9,0]
Access: 8  Miss  State of Memory:[9,0,8]
Access: 7  Miss  State of Memory:[0,8,7]
Access: 6  Miss  State of Memory:[8,7,6]
Access: 3  Miss  State of Memory:[7,6,3]
Access: 6  Hit   State of Memory:[7,3,6]
Access: 6  Hit   State of Memory:[7,3,6]
```

OPT

```
> python2 paging-policy.py -s 0 -n 10 -p OPT
ARG policy OPT
ARG cachesize 3
....

Access: 8  Hit/Miss?  State of Memory?
Access: 7  Hit/Miss?  State of Memory?
Access: 4  Hit/Miss?  State of Memory?
Access: 2  Hit/Miss?  State of Memory?
Access: 5  Hit/Miss?  State of Memory?
Access: 4  Hit/Miss?  State of Memory?
Access: 7  Hit/Miss?  State of Memory?
Access: 3  Hit/Miss?  State of Memory?
Access: 4  Hit/Miss?  State of Memory?
Access: 5  Hit/Miss?  State of Memory?
```

```
Access: 8  Miss  State of Memory:[8]
Access: 7  Miss  State of Memory:[8,7]
Access: 4  Miss  State of Memory:[8,7,4]
Access: 2  Miss  State of Memory:[7,4,2]
Access: 5  Miss  State of Memory:[7,4,5]
Access: 4  Hit   State of Memory:[7,4,5]
Access: 7  Hit   State of Memory:[7,4,5]
Access: 3  Miss  State of Memory:[4,5,3]
Access: 4  Hit   State of Memory:[4,5,3]
Access: 5  Hit   State of Memory:[4,5,3]
```

```
> python2 paging-policy.py -s 1 -n 10 -p OPT
ARG policy OPT
ARG cachesize 3

...

Access: 1  Hit/Miss?  State of Memory?
Access: 8  Hit/Miss?  State of Memory?
Access: 7  Hit/Miss?  State of Memory?
Access: 2  Hit/Miss?  State of Memory?
Access: 4  Hit/Miss?  State of Memory?
Access: 4  Hit/Miss?  State of Memory?
Access: 6  Hit/Miss?  State of Memory?
Access: 7  Hit/Miss?  State of Memory?
Access: 0  Hit/Miss?  State of Memory?
Access: 0  Hit/Miss?  State of Memory?
```

```
Access: 1  Miss  State of Memory:[1]
Access: 8  Miss  State of Memory:[1,8]
```

```
Access: 7  Miss  State of Memory:[1,8,7]
Access: 2  Miss  State of Memory:[8,7,2]
Access: 4  Miss  State of Memory:[7,2,4]
Access: 4  Hit   State of Memory:[7,2,4]
Access: 6  Miss  State of Memory:[7,4,6]
Access: 7  Hit   State of Memory:[7,4,6]
Access: 0  Miss  State of Memory:[4,6,0]
Access: 0  Hit   State of Memory:[4,6,0]
```

```
> python2 paging-policy.py -s 2 -n 10 -p OPT
ARG policy OPT
ARG cachesize 3

......

Access: 9  Hit/Miss?  State of Memory?
Access: 9  Hit/Miss?  State of Memory?
Access: 0  Hit/Miss?  State of Memory?
Access: 0  Hit/Miss?  State of Memory?
Access: 8  Hit/Miss?  State of Memory?
Access: 7  Hit/Miss?  State of Memory?
Access: 6  Hit/Miss?  State of Memory?
Access: 3  Hit/Miss?  State of Memory?
Access: 6  Hit/Miss?  State of Memory?
Access: 6  Hit/Miss?  State of Memory?
```

```
Access: 9  Miss  State of Memory:[9]
Access: 9  Hit   State of Memory:[9]
Access: 0  Miss  State of Memory:[9,0]
Access: 0  Hit   State of Memory:[9,0]
Access: 8  Miss  State of Memory:[9,0,8]
Access: 7  Miss  State of Memory:[0,8,7]
Access: 6  Miss  State of Memory:[8,7,6]
Access: 3  Miss  State of Memory:[7,6,3]
Access: 6  Hit   State of Memory:[7,6,3]
Access: 6  Hit   State of Memory:[7,6,3]
```

Problem2

问题描述

对于大小为 5 的高速缓存，为以下每个策略生成最差情况的地址引用序列：FIFO、LRU 和 MRU(最差情况下的引用序列导致尽可能多的未命中）。对于最差情况下的引用序列，需要的缓存增大多少，才能大幅提高性能，并接近 OPT?

问题分析

FIFO和LRU在循环工作负载中达到最差情况，MRU在交替引用两个页面时达到最差情况。

问题解答

FIFO

```
>python2 paging-policy.py -a 0,1,2,3,4,5,0,1,2,3 -C 5 -c

......

Access: 0  MISS FirstIn ->          [0] <- Lastin  Replaced:- [Hits:0 Misses:1]
Access: 1  MISS FirstIn ->       [0, 1] <- Lastin  Replaced:- [Hits:0 Misses:2]
Access: 2  MISS FirstIn ->    [0, 1, 2] <- Lastin  Replaced:- [Hits:0 Misses:3]
Access: 3  MISS FirstIn -> [0, 1, 2, 3] <- Lastin  Replaced:- [Hits:0 Misses:4]
Access: 4  MISS FirstIn -> [0, 1, 2, 3, 4] <- Lastin  Replaced:- [Hits:0 Misses:5]
Access: 5  MISS FirstIn -> [1, 2, 3, 4, 5] <- Lastin  Replaced:0 [Hits:0 Misses:6]
Access: 0  MISS FirstIn -> [2, 3, 4, 5, 0] <- Lastin  Replaced:1 [Hits:0 Misses:7]
Access: 1  MISS FirstIn -> [3, 4, 5, 0, 1] <- Lastin  Replaced:2 [Hits:0 Misses:8]
Access: 2  MISS FirstIn -> [4, 5, 0, 1, 2] <- Lastin  Replaced:3 [Hits:0 Misses:9]
Access: 3  MISS FirstIn -> [5, 0, 1, 2, 3] <- Lastin  Replaced:4 [Hits:0
Misses:10]

FINALSTATS hits 0   misses 10   hitrate 0.00
```

LRU

```
>python2 paging-policy.py -a 0,1,2,3,4,5,0,1,2,3 -C 5 -p LRU -c

......

Access: 0  MISS LRU ->          [0] <- MRU Replaced:- [Hits:0 Misses:1]
Access: 1  MISS LRU ->       [0, 1] <- MRU Replaced:- [Hits:0 Misses:2]
Access: 2  MISS LRU ->    [0, 1, 2] <- MRU Replaced:- [Hits:0 Misses:3]
Access: 3  MISS LRU -> [0, 1, 2, 3] <- MRU Replaced:- [Hits:0 Misses:4]
Access: 4  MISS LRU -> [0, 1, 2, 3, 4] <- MRU Replaced:- [Hits:0 Misses:5]
Access: 5  MISS LRU -> [1, 2, 3, 4, 5] <- MRU Replaced:0 [Hits:0 Misses:6]
Access: 0  MISS LRU -> [2, 3, 4, 5, 0] <- MRU Replaced:1 [Hits:0 Misses:7]
Access: 1  MISS LRU -> [3, 4, 5, 0, 1] <- MRU Replaced:2 [Hits:0 Misses:8]
Access: 2  MISS LRU -> [4, 5, 0, 1, 2] <- MRU Replaced:3 [Hits:0 Misses:9]
Access: 3  MISS LRU -> [5, 0, 1, 2, 3] <- MRU Replaced:4 [Hits:0 Misses:10]

FINALSTATS hits 0   misses 10   hitrate 0.00
```

MRU

```
>python2 paging-policy.py -a 0,1,2,3,4,5,4,5,4,5 -C 5 -p MRU -c
```

```
......

Access: 0  MISS LRU ->          [0] <- MRU Replaced:- [Hits:0 Misses:1]
Access: 1  MISS LRU ->       [0, 1] <- MRU Replaced:- [Hits:0 Misses:2]
Access: 2  MISS LRU ->    [0, 1, 2] <- MRU Replaced:- [Hits:0 Misses:3]
Access: 3  MISS LRU -> [0, 1, 2, 3] <- MRU Replaced:- [Hits:0 Misses:4]
Access: 4  MISS LRU -> [0, 1, 2, 3, 4] <- MRU Replaced:- [Hits:0 Misses:5]
Access: 5  MISS LRU -> [0, 1, 2, 3, 5] <- MRU Replaced:4 [Hits:0 Misses:6]
Access: 4  MISS LRU -> [0, 1, 2, 3, 4] <- MRU Replaced:5 [Hits:0 Misses:7]
Access: 5  MISS LRU -> [0, 1, 2, 3, 5] <- MRU Replaced:4 [Hits:0 Misses:8]
Access: 4  MISS LRU -> [0, 1, 2, 3, 4] <- MRU Replaced:5 [Hits:0 Misses:9]
Access: 5  MISS LRU -> [0, 1, 2, 3, 5] <- MRU Replaced:4 [Hits:0 Misses:10]

FINALSTATS hits 0   misses 10   hitrate 0.00
```

对于最差情况下的引用序列，需要缓存增大到与页面数相同，才能大幅提高性能，并接近 OPT。

Problem3

问题描述

生成一个随机追踪序列（使用 Python 或 Perl)。你预计不同的策略在这样的追踪序列上的表现如何?

问题分析

FIFO：非常简单的替换策略，先进先出。
LRU：换出最近最少使用的页。
OPT：替换最远的将来会访问的页，命中率是所有策略中最高的。
UNOPT：替换最近的将来会访问的页，命中率是所有策略中最低的。
RAND：完全看运气，可能会非常幸运，也可能非常不幸。
CLOCK：近似LRU，命中率应不超过LRU，且clockbits越多，命中率越高。

问题解答

FIFO

```
> python2 paging-policy.py -s 0 -n 10 -c -p FIFO

......

Access: 8  MISS FirstIn ->          [8] <- Lastin  Replaced:- [Hits:0 Misses:1]
Access: 7  MISS FirstIn ->       [8, 7] <- Lastin  Replaced:- [Hits:0 Misses:2]
Access: 4  MISS FirstIn ->    [8, 7, 4] <- Lastin  Replaced:- [Hits:0 Misses:3]
Access: 2  MISS FirstIn ->    [7, 4, 2] <- Lastin  Replaced:8 [Hits:0 Misses:4]
Access: 5  MISS FirstIn ->    [4, 2, 5] <- Lastin  Replaced:7 [Hits:0 Misses:5]
Access: 4  HIT  FirstIn ->    [4, 2, 5] <- Lastin  Replaced:- [Hits:1 Misses:5]
Access: 7  MISS FirstIn ->    [2, 5, 7] <- Lastin  Replaced:4 [Hits:1 Misses:6]
Access: 3  MISS FirstIn ->    [5, 7, 3] <- Lastin  Replaced:2 [Hits:1 Misses:7]
Access: 4  MISS FirstIn ->    [7, 3, 4] <- Lastin  Replaced:5 [Hits:1 Misses:8]
Access: 5  MISS FirstIn ->    [3, 4, 5] <- Lastin  Replaced:7 [Hits:1 Misses:9]
```

```
FINALSTATS hits 1   misses 9   hitrate 10.00
```

LRU

```
> python2 paging-policy.py -s 0 -n 10 -c -p LRU

......

Access: 8  MISS LRU ->          [8] <- MRU Replaced:- [Hits:0 Misses:1]
Access: 7  MISS LRU ->       [8, 7] <- MRU Replaced:- [Hits:0 Misses:2]
Access: 4  MISS LRU ->    [8, 7, 4] <- MRU Replaced:- [Hits:0 Misses:3]
Access: 2  MISS LRU ->    [7, 4, 2] <- MRU Replaced:8 [Hits:0 Misses:4]
Access: 5  MISS LRU ->    [4, 2, 5] <- MRU Replaced:7 [Hits:0 Misses:5]
Access: 4  HIT  LRU ->    [2, 5, 4] <- MRU Replaced:- [Hits:1 Misses:5]
Access: 7  MISS LRU ->    [5, 4, 7] <- MRU Replaced:2 [Hits:1 Misses:6]
Access: 3  MISS LRU ->    [4, 7, 3] <- MRU Replaced:5 [Hits:1 Misses:7]
Access: 4  HIT  LRU ->    [7, 3, 4] <- MRU Replaced:- [Hits:2 Misses:7]
Access: 5  MISS LRU ->    [3, 4, 5] <- MRU Replaced:7 [Hits:2 Misses:8]

FINALSTATS hits 2   misses 8   hitrate 20.00
```

OPT

```
> python2 paging-policy.py -s 0 -n 10 -c -p OPT

......

Access: 8  MISS Left  ->          [8] <- Right Replaced:- [Hits:0 Misses:1]
Access: 7  MISS Left  ->       [8, 7] <- Right Replaced:- [Hits:0 Misses:2]
Access: 4  MISS Left  ->    [8, 7, 4] <- Right Replaced:- [Hits:0 Misses:3]
Access: 2  MISS Left  ->    [7, 4, 2] <- Right Replaced:8 [Hits:0 Misses:4]
Access: 5  MISS Left  ->    [7, 4, 5] <- Right Replaced:2 [Hits:0 Misses:5]
Access: 4  HIT  Left  ->    [7, 4, 5] <- Right Replaced:- [Hits:1 Misses:5]
Access: 7  HIT  Left  ->    [7, 4, 5] <- Right Replaced:- [Hits:2 Misses:5]
Access: 3  MISS Left  ->    [4, 5, 3] <- Right Replaced:7 [Hits:2 Misses:6]
Access: 4  HIT  Left  ->    [4, 5, 3] <- Right Replaced:- [Hits:3 Misses:6]
Access: 5  HIT  Left  ->    [4, 5, 3] <- Right Replaced:- [Hits:4 Misses:6]

FINALSTATS hits 4   misses 6   hitrate 40.00
```

UNOPT

```
> python2 paging-policy.py -s 0 -n 10 -c -p UNOPT

......
```

```
Access: 8  MISS Left  ->          [8] <- Right Replaced:- [Hits:0 Misses:1]
Access: 7  MISS Left  ->       [8, 7] <- Right Replaced:- [Hits:0 Misses:2]
Access: 4  MISS Left  ->    [8, 7, 4] <- Right Replaced:- [Hits:0 Misses:3]
Access: 2  MISS Left  ->    [8, 7, 2] <- Right Replaced:4 [Hits:0 Misses:4]
Access: 5  MISS Left  ->    [8, 2, 5] <- Right Replaced:7 [Hits:0 Misses:5]
Access: 4  MISS Left  ->    [8, 2, 4] <- Right Replaced:5 [Hits:0 Misses:6]
Access: 7  MISS Left  ->    [8, 2, 7] <- Right Replaced:4 [Hits:0 Misses:7]
Access: 3  MISS Left  ->    [2, 7, 3] <- Right Replaced:8 [Hits:0 Misses:8]
Access: 4  MISS Left  ->    [7, 3, 4] <- Right Replaced:2 [Hits:0 Misses:9]
Access: 5  MISS Left  ->    [3, 4, 5] <- Right Replaced:7 [Hits:0 Misses:10]

FINALSTATS hits 0   misses 10   hitrate 0.00
```

RAND

```
> python2 paging-policy.py -s 0 -n 10 -c -p RAND

......

Access: 8  MISS Left  ->          [8] <- Right Replaced:- [Hits:0 Misses:1]
Access: 7  MISS Left  ->       [8, 7] <- Right Replaced:- [Hits:0 Misses:2]
Access: 4  MISS Left  ->    [8, 7, 4] <- Right Replaced:- [Hits:0 Misses:3]
Access: 2  MISS Left  ->    [8, 7, 2] <- Right Replaced:4 [Hits:0 Misses:4]
Access: 5  MISS Left  ->    [8, 2, 5] <- Right Replaced:7 [Hits:0 Misses:5]
Access: 4  MISS Left  ->    [2, 5, 4] <- Right Replaced:8 [Hits:0 Misses:6]
Access: 7  MISS Left  ->    [2, 5, 7] <- Right Replaced:4 [Hits:0 Misses:7]
Access: 3  MISS Left  ->    [2, 7, 3] <- Right Replaced:5 [Hits:0 Misses:8]
Access: 4  MISS Left  ->    [7, 3, 4] <- Right Replaced:2 [Hits:0 Misses:9]
Access: 5  MISS Left  ->    [7, 3, 5] <- Right Replaced:4 [Hits:0 Misses:10]

FINALSTATS hits 0   misses 10   hitrate 0.00
```

CLOCK

```
> python2 paging-policy.py -s 0 -n 10 -c -p CLOCK

......

Access: 8  MISS Left  ->          [8] <- Right Replaced:- [Hits:0 Misses:1]
Access: 7  MISS Left  ->       [8, 7] <- Right Replaced:- [Hits:0 Misses:2]
Access: 4  MISS Left  ->    [8, 7, 4] <- Right Replaced:- [Hits:0 Misses:3]
Access: 2  MISS Left  ->    [8, 7, 2] <- Right Replaced:4 [Hits:0 Misses:4]
Access: 5  MISS Left  ->    [8, 2, 5] <- Right Replaced:7 [Hits:0 Misses:5]
Access: 4  MISS Left  ->    [2, 5, 4] <- Right Replaced:8 [Hits:0 Misses:6]
Access: 7  MISS Left  ->    [2, 5, 7] <- Right Replaced:4 [Hits:0 Misses:7]
Access: 3  MISS Left  ->    [2, 5, 3] <- Right Replaced:7 [Hits:0 Misses:8]
Access: 4  MISS Left  ->    [2, 5, 4] <- Right Replaced:3 [Hits:0 Misses:9]
```

```
Access: 5  HIT  Left  ->    [2, 5, 4] <- Right Replaced:- [Hits:1 Misses:9]

FINALSTATS hits 1   misses 9   hitrate 10.00
```

Problem4

问题描述

现在生成一些局部性追踪序列。如何能够产生这样的追踪序列？LRU 表现如何？RAND 比 LRU 好多少？CLOCK 表现如何？CLOCK 使用不同数量的时钟位，表现如何？

问题解答

tool.py

```
import random
import sys

numAddr = 10

# 空间局部性
def generate_spatial_locality_trace():
    trace = [random.randint(0, numAddr)]
    for i in range(8):
        l = trace[-1]
        rand = [l, (l + 1) % numAddr, (l - 1) % numAddr, random.randint(0,
numAddr)]
        trace.append(random.choice(rand))
    # 问题给的paging-policy.py -a参数里，逗号后不能空格,因此拼接再打印
    print(','.join([str(i) for i in trace]))

# 时间局部性
def generate_temporal_locality_trace():
    trace = [random.randint(0, numAddr)]
    for i in range(8):
        rand = [random.randint(0, numAddr), random.choice(trace)]
        trace.append(random.choice(rand))
    print(','.join([str(i) for i in trace]))

if len(sys.argv) != 1:
    if sys.argv[1] == '-t':
        generate_temporal_locality_trace()
    elif sys.argv[1] == '-s':
        generate_spatial_locality_trace()
```

用法：

```
> python3 tool.py -s #产生具有空间局部性的序列
> python3 tool.py -t #产生具有时间局部性的序列
```

LRU

```
> python2 paging-policy.py -p LRU -c -a  $(python3 tool.py -t)
ARG addresses 5,5,2,0,5,0,5,0,2
ARG addressfile
ARG numaddrs 10
ARG policy LRU
ARG clockbits 2
ARG cachesize 3
ARG maxpage 10
ARG seed 0
ARG notrace False

Solving...

Access: 5  MISS LRU ->          [5] <- MRU Replaced:- [Hits:0 Misses:1]
Access: 5  HIT  LRU ->          [5] <- MRU Replaced:- [Hits:1 Misses:1]
Access: 2  MISS LRU ->       [5, 2] <- MRU Replaced:- [Hits:1 Misses:2]
Access: 0  MISS LRU ->    [5, 2, 0] <- MRU Replaced:- [Hits:1 Misses:3]
Access: 5  HIT  LRU ->    [2, 0, 5] <- MRU Replaced:- [Hits:2 Misses:3]
Access: 0  HIT  LRU ->    [2, 5, 0] <- MRU Replaced:- [Hits:3 Misses:3]
Access: 5  HIT  LRU ->    [2, 0, 5] <- MRU Replaced:- [Hits:4 Misses:3]
Access: 0  HIT  LRU ->    [2, 5, 0] <- MRU Replaced:- [Hits:5 Misses:3]
Access: 2  HIT  LRU ->    [5, 0, 2] <- MRU Replaced:- [Hits:6 Misses:3]

FINALSTATS hits 6   misses 3   hitrate 66.67
```

```
> python2 paging-policy.py -p LRU -c -a  $(python3 tool.py -s)
ARG addresses 6,5,5,4,4,5,10,6,3
ARG addressfile
ARG numaddrs 10
ARG policy LRU
ARG clockbits 2
ARG cachesize 3
ARG maxpage 10
ARG seed 0
ARG notrace False

Solving...

Access: 6  MISS LRU ->          [6] <- MRU Replaced:- [Hits:0 Misses:1]
Access: 5  MISS LRU ->       [6, 5] <- MRU Replaced:- [Hits:0 Misses:2]
Access: 5  HIT  LRU ->       [6, 5] <- MRU Replaced:- [Hits:1 Misses:2]
Access: 4  MISS LRU ->    [6, 5, 4] <- MRU Replaced:- [Hits:1 Misses:3]
Access: 4  HIT  LRU ->    [6, 5, 4] <- MRU Replaced:- [Hits:2 Misses:3]
Access: 5  HIT  LRU ->    [6, 4, 5] <- MRU Replaced:- [Hits:3 Misses:3]
Access: 10  MISS LRU ->   [4, 5, 10] <- MRU Replaced:6 [Hits:3 Misses:4]
Access: 6  MISS LRU ->   [5, 10, 6] <- MRU Replaced:4 [Hits:3 Misses:5]
```

```
Access: 3  MISS LRU ->   [10, 6, 3] <- MRU Replaced:5 [Hits:3 Misses:6]

FINALSTATS hits 3   misses 6   hitrate 33.33
```

由上可知，LRU在处理具有时间局部性的追踪序列时表现较好。

RAND

```
> python2 paging-policy.py -p RAND -c -a  5,5,2,0,5,0,5,0,2
ARG addresses 5,5,2,0,5,0,5,0,2
ARG addressfile
ARG numaddrs 10
ARG policy RAND
ARG clockbits 2
ARG cachesize 3
ARG maxpage 10
ARG seed 0
ARG notrace False

Solving...

Access: 5  MISS Left  ->          [5] <- Right Replaced:- [Hits:0 Misses:1]
Access: 5  HIT  Left  ->          [5] <- Right Replaced:- [Hits:1 Misses:1]
Access: 2  MISS Left  ->       [5, 2] <- Right Replaced:- [Hits:1 Misses:2]
Access: 0  MISS Left  ->    [5, 2, 0] <- Right Replaced:- [Hits:1 Misses:3]
Access: 5  HIT  Left  ->    [5, 2, 0] <- Right Replaced:- [Hits:2 Misses:3]
Access: 0  HIT  Left  ->    [5, 2, 0] <- Right Replaced:- [Hits:3 Misses:3]
Access: 5  HIT  Left  ->    [5, 2, 0] <- Right Replaced:- [Hits:4 Misses:3]
Access: 0  HIT  Left  ->    [5, 2, 0] <- Right Replaced:- [Hits:5 Misses:3]
Access: 2  HIT  Left  ->    [5, 2, 0] <- Right Replaced:- [Hits:6 Misses:3]

FINALSTATS hits 6   misses 3   hitrate 66.67
```

```
> python2 paging-policy.py -p RAND -c -a  6,5,5,4,4,5,10,6,3
ARG addresses 6,5,5,4,4,5,10,6,3
ARG addressfile
ARG numaddrs 10
ARG policy RAND
ARG clockbits 2
ARG cachesize 3
ARG maxpage 10
ARG seed 0
ARG notrace False

Solving...

Access: 6  MISS Left  ->          [6] <- Right Replaced:- [Hits:0 Misses:1]
Access: 5  MISS Left  ->       [6, 5] <- Right Replaced:- [Hits:0 Misses:2]
Access: 5  HIT  Left  ->       [6, 5] <- Right Replaced:- [Hits:1 Misses:2]
```

```
Access: 4  MISS Left  ->    [6, 5, 4] <- Right Replaced:- [Hits:1 Misses:3]
Access: 4  HIT  Left  ->    [6, 5, 4] <- Right Replaced:- [Hits:2 Misses:3]
Access: 5  HIT  Left  ->    [6, 5, 4] <- Right Replaced:- [Hits:3 Misses:3]
Access: 10  MISS Left  ->   [6, 5, 10] <- Right Replaced:4 [Hits:3 Misses:4]
Access: 6  HIT  Left  ->   [6, 5, 10] <- Right Replaced:- [Hits:4 Misses:4]
Access: 3  MISS Left  ->    [6, 5, 3] <- Right Replaced:10 [Hits:4 Misses:5]

FINALSTATS hits 4   misses 5   hitrate 44.44
```

RAND完全看运气，可能会非常幸运，也可能非常不幸。本例中时间局部性序列命中率等于LRU,空间局部性序列命中率高于LRU。

CLOCK(clockbits=2)

```
> python2 paging-policy.py -p CLOCK -c -a  5,5,2,0,5,0,5,0,2
ARG addresses 5,5,2,0,5,0,5,0,2
ARG addressfile
ARG numaddrs 10
ARG policy CLOCK
ARG clockbits 2
ARG cachesize 3
ARG maxpage 10
ARG seed 0
ARG notrace False

Solving...

Access: 5  MISS Left  ->          [5] <- Right Replaced:- [Hits:0 Misses:1]
Access: 5  HIT  Left  ->          [5] <- Right Replaced:- [Hits:1 Misses:1]
Access: 2  MISS Left  ->       [5, 2] <- Right Replaced:- [Hits:1 Misses:2]
Access: 0  MISS Left  ->    [5, 2, 0] <- Right Replaced:- [Hits:1 Misses:3]
Access: 5  HIT  Left  ->    [5, 2, 0] <- Right Replaced:- [Hits:2 Misses:3]
Access: 0  HIT  Left  ->    [5, 2, 0] <- Right Replaced:- [Hits:3 Misses:3]
Access: 5  HIT  Left  ->    [5, 2, 0] <- Right Replaced:- [Hits:4 Misses:3]
Access: 0  HIT  Left  ->    [5, 2, 0] <- Right Replaced:- [Hits:5 Misses:3]
Access: 2  HIT  Left  ->    [5, 2, 0] <- Right Replaced:- [Hits:6 Misses:3]

FINALSTATS hits 6   misses 3   hitrate 66.67
```

```
> python2 paging-policy.py -p CLOCK -c -a  6,5,5,4,4,5,10,6,3
ARG addresses 6,5,5,4,4,5,10,6,3
ARG addressfile
ARG numaddrs 10
ARG policy CLOCK
ARG clockbits 2
ARG cachesize 3
ARG maxpage 10
ARG seed 0
ARG notrace False
```

```
Solving...

Access: 6  MISS Left  ->          [6] <- Right Replaced:- [Hits:0 Misses:1]
Access: 5  MISS Left  ->       [6, 5] <- Right Replaced:- [Hits:0 Misses:2]
Access: 5  HIT  Left  ->       [6, 5] <- Right Replaced:- [Hits:1 Misses:2]
Access: 4  MISS Left  ->    [6, 5, 4] <- Right Replaced:- [Hits:1 Misses:3]
Access: 4  HIT  Left  ->    [6, 5, 4] <- Right Replaced:- [Hits:2 Misses:3]
Access: 5  HIT  Left  ->    [6, 5, 4] <- Right Replaced:- [Hits:3 Misses:3]
Access: 10  MISS Left  ->   [6, 4, 10] <- Right Replaced:5 [Hits:3 Misses:4]
Access: 6  HIT  Left  ->   [6, 4, 10] <- Right Replaced:- [Hits:4 Misses:4]
Access: 3  MISS Left  ->   [6, 10, 3] <- Right Replaced:4 [Hits:4 Misses:5]

FINALSTATS hits 4   misses 5   hitrate 44.44
```

CLOCK是近似LRU,本例中时间局部性序列命中率等于LRU,空间局部性序列命中率高于LRU。

CLOCK(clockbits=4)

```
> python2 paging-policy.py -p CLOCK -c -b 4 -a 5,5,2,0,5,0,5,0,2
ARG addresses 5,5,2,0,5,0,5,0,2
ARG addressfile
ARG numaddrs 10
ARG policy CLOCK
ARG clockbits 4
ARG cachesize 3
ARG maxpage 10
ARG seed 0
ARG notrace False

Solving...

Access: 5  MISS Left  ->          [5] <- Right Replaced:- [Hits:0 Misses:1]
Access: 5  HIT  Left  ->          [5] <- Right Replaced:- [Hits:1 Misses:1]
Access: 2  MISS Left  ->       [5, 2] <- Right Replaced:- [Hits:1 Misses:2]
Access: 0  MISS Left  ->    [5, 2, 0] <- Right Replaced:- [Hits:1 Misses:3]
Access: 5  HIT  Left  ->    [5, 2, 0] <- Right Replaced:- [Hits:2 Misses:3]
Access: 0  HIT  Left  ->    [5, 2, 0] <- Right Replaced:- [Hits:3 Misses:3]
Access: 5  HIT  Left  ->    [5, 2, 0] <- Right Replaced:- [Hits:4 Misses:3]
Access: 0  HIT  Left  ->    [5, 2, 0] <- Right Replaced:- [Hits:5 Misses:3]
Access: 2  HIT  Left  ->    [5, 2, 0] <- Right Replaced:- [Hits:6 Misses:3]

FINALSTATS hits 6   misses 3   hitrate 66.67
```

```
> python2 paging-policy.py -p CLOCK -c -b 4 -a 6,5,5,4,4,5,10,6,3
ARG addresses 6,5,5,4,4,5,10,6,3
ARG addressfile
ARG numaddrs 10
ARG policy CLOCK
```

```
ARG clockbits 4
ARG cachesize 3
ARG maxpage 10
ARG seed 0
ARG notrace False

Solving...

Access: 6  MISS Left  ->          [6] <- Right Replaced:- [Hits:0 Misses:1]
Access: 5  MISS Left  ->       [6, 5] <- Right Replaced:- [Hits:0 Misses:2]
Access: 5  HIT  Left  ->       [6, 5] <- Right Replaced:- [Hits:1 Misses:2]
Access: 4  MISS Left  ->    [6, 5, 4] <- Right Replaced:- [Hits:1 Misses:3]
Access: 4  HIT  Left  ->    [6, 5, 4] <- Right Replaced:- [Hits:2 Misses:3]
Access: 5  HIT  Left  ->    [6, 5, 4] <- Right Replaced:- [Hits:3 Misses:3]
Access: 10  MISS Left  ->   [6, 5, 10] <- Right Replaced:4 [Hits:3 Misses:4]
Access: 6  HIT  Left  ->   [6, 5, 10] <- Right Replaced:- [Hits:4 Misses:4]
Access: 3  MISS Left  ->   [6, 10, 3] <- Right Replaced:5 [Hits:4 Misses:5]

FINALSTATS hits 4   misses 5   hitrate 44.44
```

clockbits越多，命中率越高。本例中时间局部性序列命中率等于LRU,空间局部性序列命中率高于LRU。