CS 471 Operating Systems

Yue Cheng

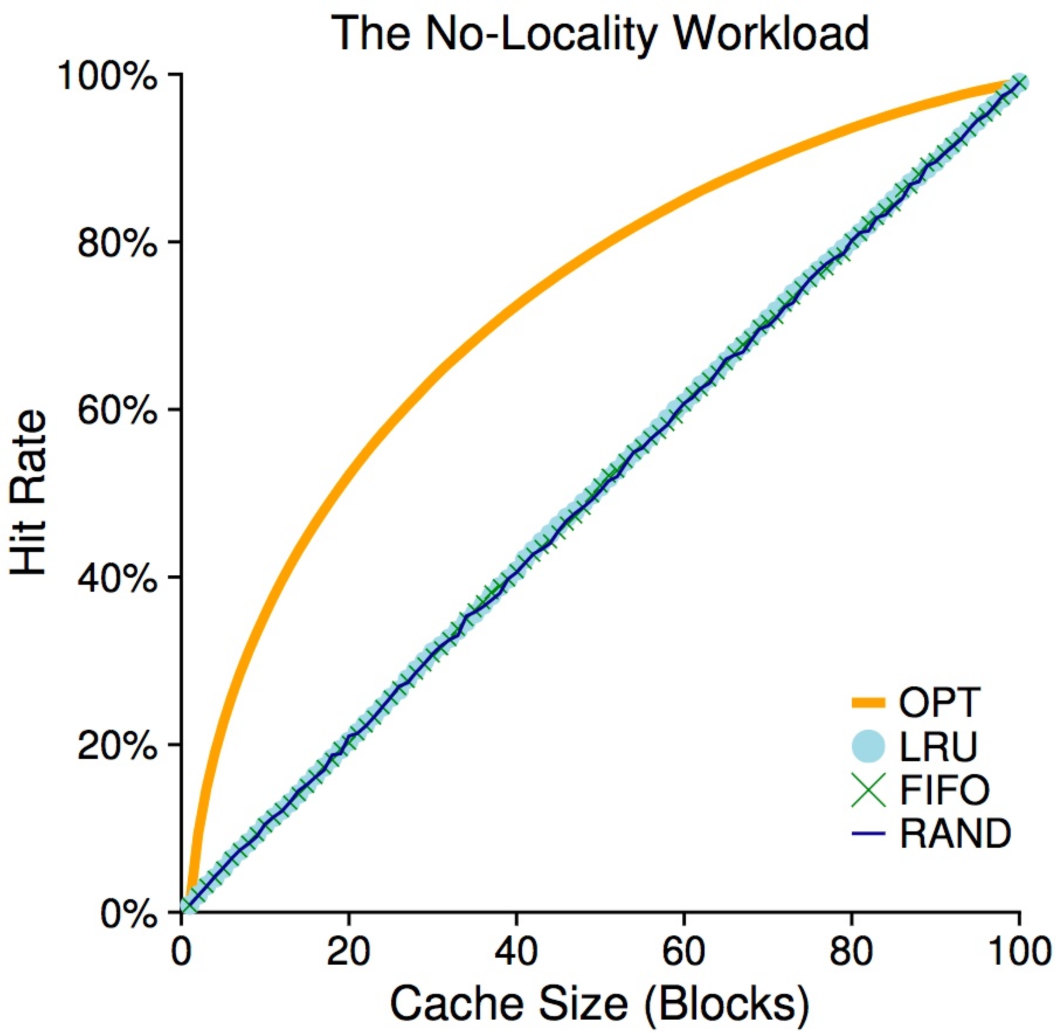
George Mason University
Spring 2019

Page Replacement Workload Examples

Workload Examples

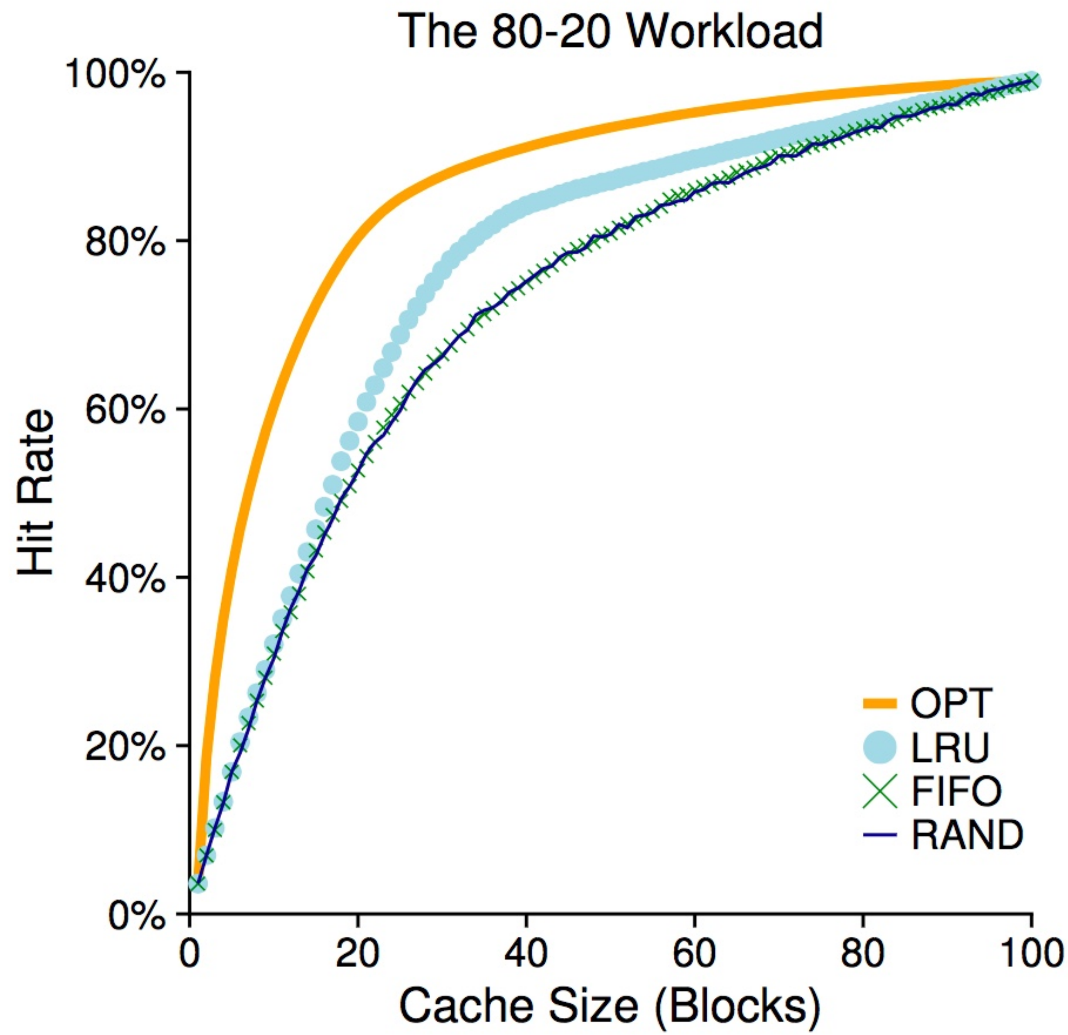
- A simple workload
  - Workload consists of a working set of 100 pages
  - Workload issues 10,000 access requests
- Four replacement policies
  - OPT: The optimal
  - LRU: Least-recently used
  - FIFO: First-in first-out
  - RAND: Random

The No-Locality Workload



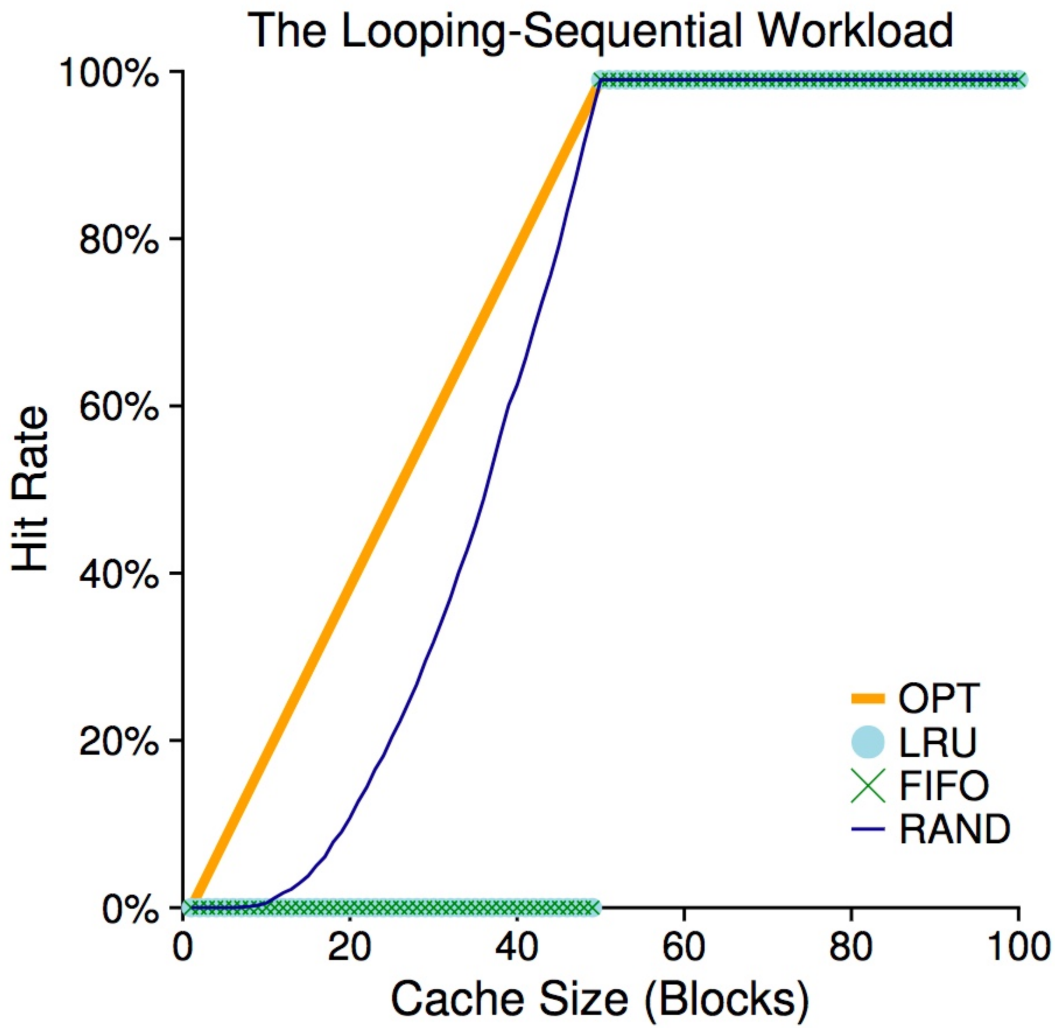
Each reference is to a random page within the set of accessed pages

The 80-20 Workload



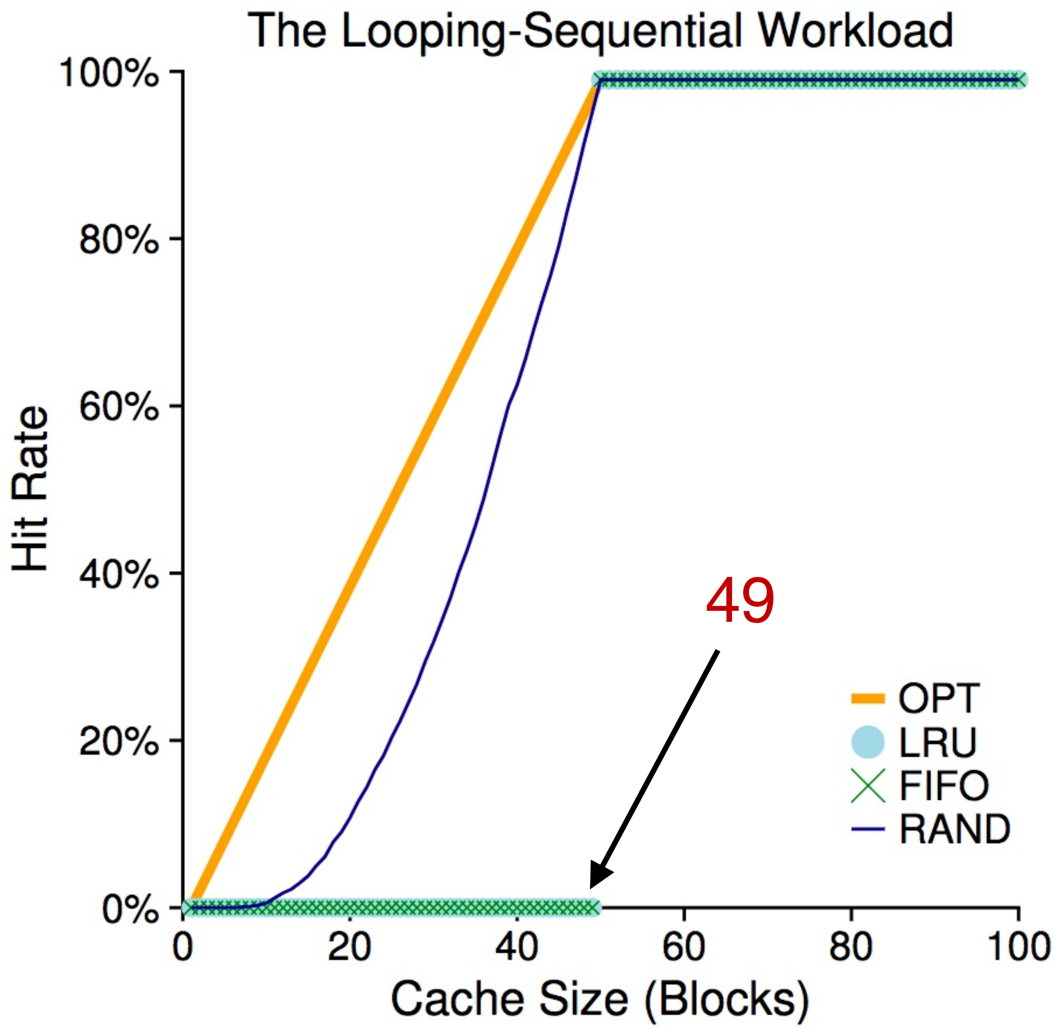
80-20: 80% of the refs are made to 20% of the pages ("**hot**" pages)

The Looping-Sequential Workload



Loop first 50 pages starting from 0 to 49 for a total of 10,000 accesses

The Looping-Sequential Workload



Loop first 50 pages starting from 0 to 49 for a total of 10,000 accesses

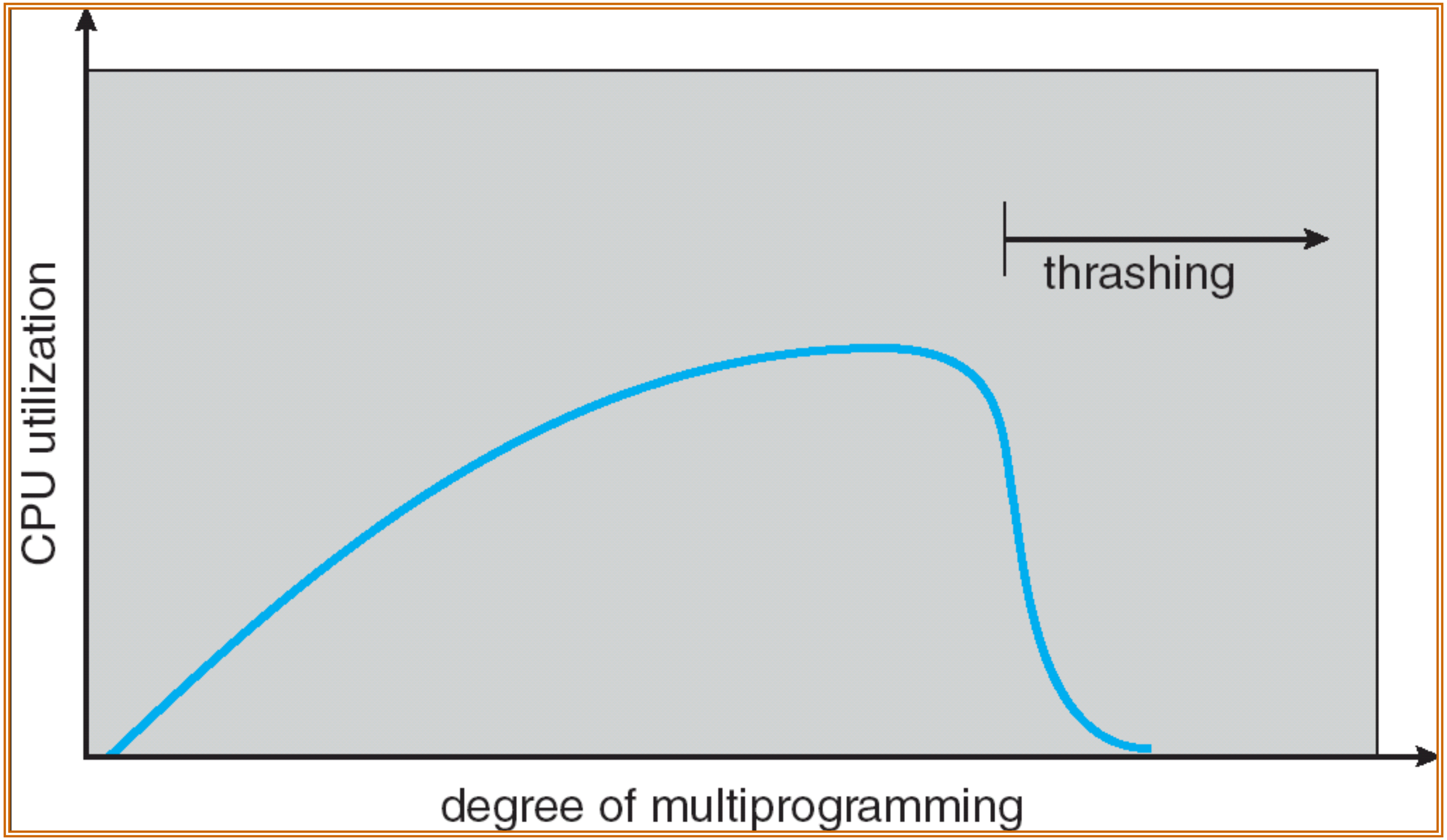
Thrashing

Thrashing

- High-paging activity: The system is spending more time paging than executing

- How can this happen?
  - OS observes low CPU utilization and increases the degree of multiprogramming
  - Global page-replacement algorithm is used, it takes away frames belonging to other processes
  - But these processes need those pages, they also cause page faults
  - Many processes join the waiting queue for the paging device, CPU utilization further decreases
  - OS introduces new processes, further increasing the paging activity

CPU Utilization vs. the Degree of Multiprogramming



How to Avoid Thrashing?

- To avoid thrashing, earlier OS did admission control to only run a subset of processes

- Some current OS takes more draconian approach
  - E.g., some Linux runs an out-of-memory killer to choose a memory-intensive process and kill it

Review: Demand Paging

- Bring a page into memory only when it is needed
  - Less I/O needed
  - Less memory needed
  - Faster response
  - Support more processes/users
- Page is needed $\Rightarrow$ use the reference to page
  - If not in memory $\Rightarrow$ must bring from the disk
- Demand paging versus swapping
  - Fetching the page in only on demand vs. kicking out one victim then paging in one under mem pressure

Demand Paging and Thrashing

- Why does demand paging work?
  Locality model
  - Process migrates from one locality to another
  - Localities may overlap

- **Why does thrashing occur?**
  `Σ size of locality > total memory size`
  `Or Σ working set size > total memory size`

- Definition of **working set size** (WSS): number of unique items that are accessed

Impact of Program Structures on Memory Performance

Impact of Program Structure on Memory Performance

- Consider an array named data with `128*128` elements
- Each row is stored in one page (of size 128 words)

Impact of Program Structure on Memory Performance

- Consider an array named data with `128*128` elements
- Each row is stored in one page (of size 128 words)
- **Program 1**

```
for (j = 0; j <128; j++)
  for (i = 0; i < 128; i++)
    data[i][j] = 0;
```

128 x 128 = **16,384** page faults

Impact of Program Structure on Memory Performance

- Consider an array named data with `128*128` elements
- Each row is stored in one page (of size 128 words)
- **Program 1**

```
for (j = 0; j <128; j++)
  for (i = 0; i < 128; i++)
    data[i][j] = 0;
```

128 x 128 = **16,384** page faults

- **Program 2**

```
for (i = 0; i < 128; i++)
  for (j = 0; j < 128; j++)
    data[i][j] = 0;
```

Only **128** page faults