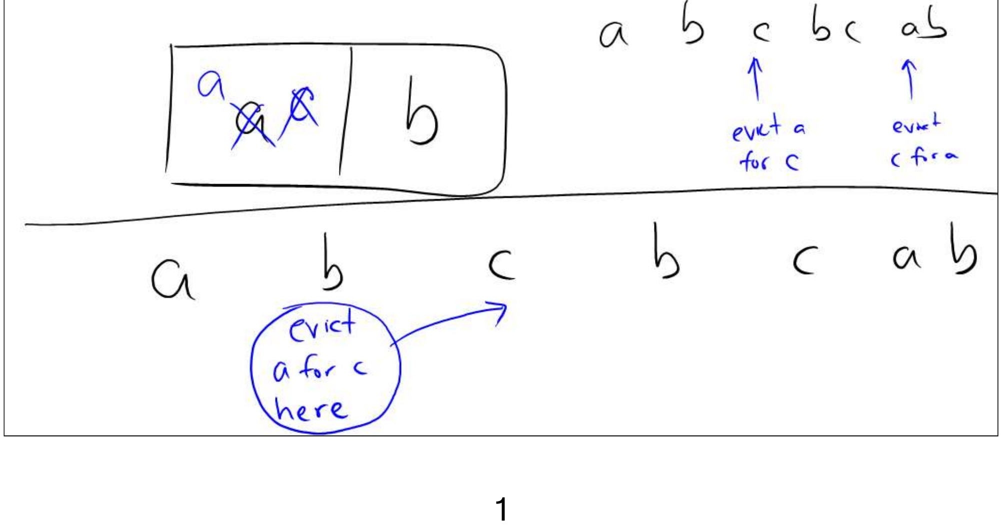
Computer Science
Spring 2021 Lecture 24:
Greedy Algorithms:
Optimal [Offline] Caching

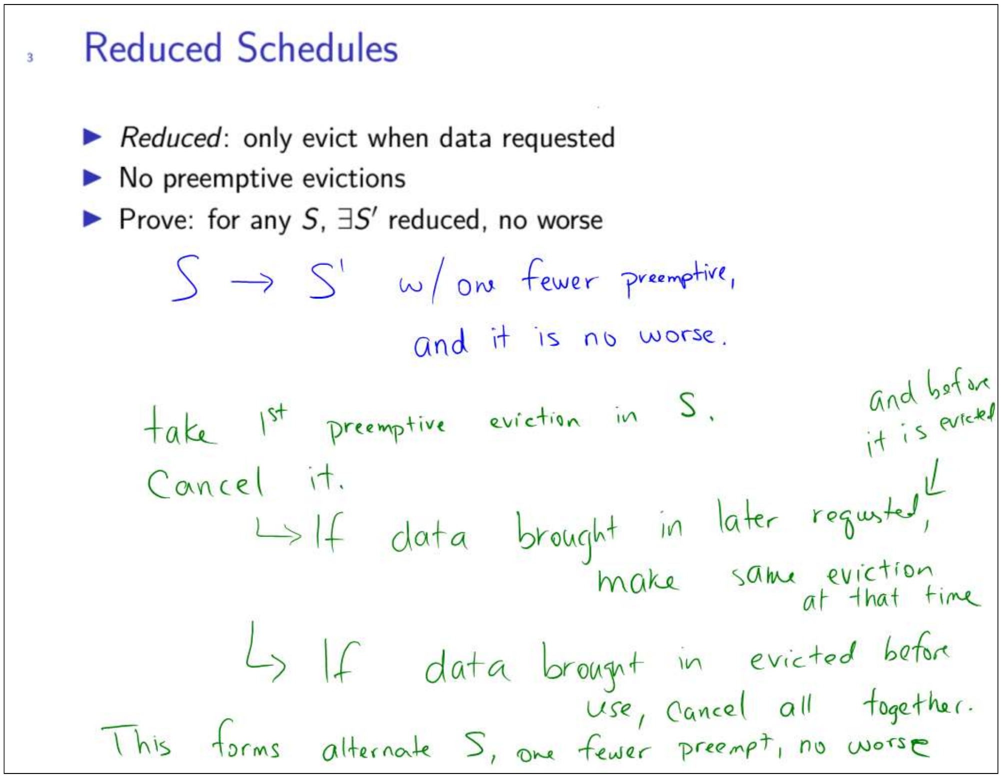
2

Caching Example

- Cache size $k = 2$
- Requests: a, b, c, b, c, a, b
- Sequence of evictions:

evictions: 11
(two)





3

Reduced Schedules

- *Reduced*: only evict when data requested
- No preemptive evictions
- Prove: for any S, $\exists S'$ reduced, no worse

$S \rightarrow S'$ w/ one fewer preemptive,
and it is no worse.

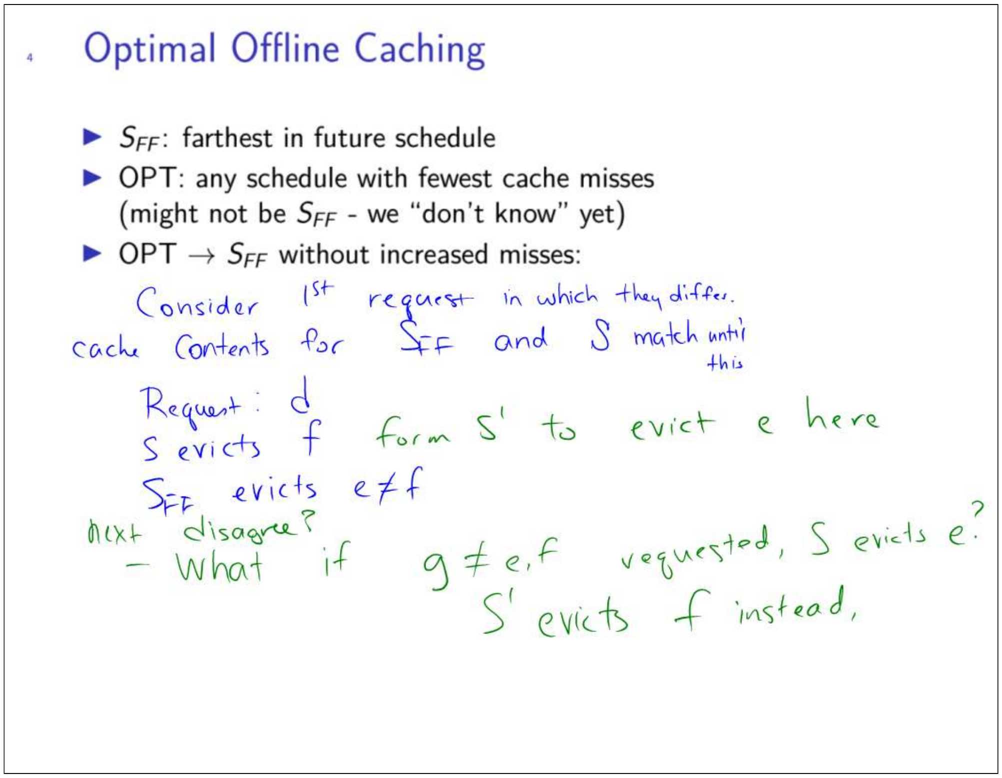
take 1st preemptive eviction in S.
Cancel it.

and before it is evicted

↳ If data brought in later requested,
make same eviction at that time

↳ If data brought in evicted before use, cancel all together.

This forms alternate S, one fewer preempt, no worse



4

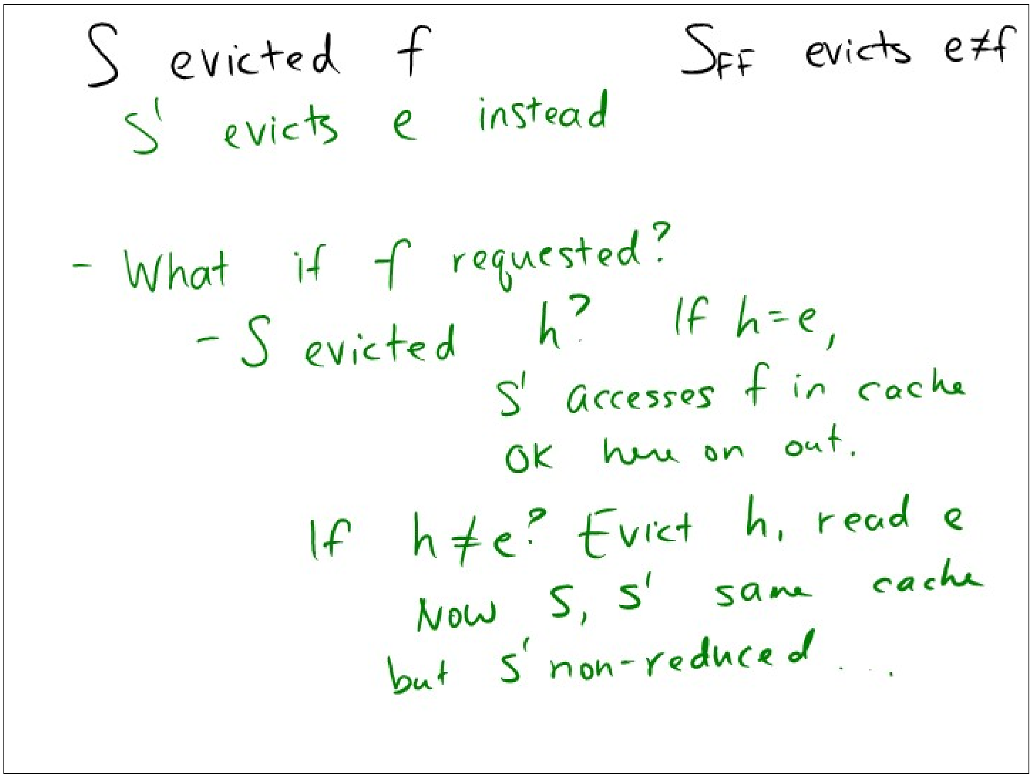
Optimal Offline Caching

- S_{FF}: farthest in future schedule
- OPT: any schedule with fewest cache misses (might not be S_{FF} - we "don't know" yet)
- OPT → S_{FF} without increased misses:

Consider 1st request in which they differ.
cache contents for S_{FF} and S match until this

Request: d
S evicts f   form S' to evict e here
S_{FF} evicts $e \neq f$

next disagree?
- What if $g \neq e, f$ requested, S evicts e?
S' evicts f instead,



S evicted f S_{FF} evicts $e \neq f$

S' evicts e instead

- What if f requested?
  - S evicted h? If $h = e$,

    S' accesses f in cache

    OK here on out.

    If $h \neq e$? Evict h, read e

    Now S, S' same cache

    but S' non-reduced . . .