DS 320 Algorithms for Data Science **Lecture #8 Worksheet**
Spring 2025 Prof. Kira Goldner and Jeffrey Considine

Optimal Caching and Greedy Exchange

In the optimal caching problem, our computer has a main memory of size n, a cache of size k, and we are presented with a sequence of data $D = d_1, d_2, \ldots, d_m$ that we must process. When an item is not in the cache, we have a cache miss, and must bring the item into the cache and evict something else if the cache is full. Our goal is to give an algorithm that minimizes the number of misses.

Example 1:	a, b, c, b, c, a, b	$k = 2$	cache $= \{a, b\}$
Example 2:	$a, b, c, d, a, d, e, a, d, b, c$	$k = 3$	cache $= \{a, b, c\}$

Definition 1. A schedule is *reduced* if it does the minimal amount of work necessary in a given step.

Lemma 1. *For every non-reduced schedule, there is an equally good reduced schedule (that brings in at most as many items as the original schedule).*

Prove this by construction.

Hint: You might *charge* a miss from one schedule to a miss in another schedule to show that it doesn't have any extra misses.

Observation 1. *For any reduced schedule, the number of items that are brought in is exactly the number of misses.*

A Greedy Algorithm for Optimal Caching

Determine a cache maintenance algorithm by coming up with an eviction schedule.

Proof Brainstorming

Given S_O as the reduced schedule from some unspecified (possibly optimal) algorithm, and S_A as an algorithm that we want to prove optimal, how can we prove $cost(S_A) \leq cost(S_O)$?

Proving the Greedy Exchange

Let S_{FF} be the schedule created by the farthest-in-future algorithm and let S be an arbitrary reduced schedule.

Lemma 2. *Suppose S is a reduced schedule that makes the same eviction decisions as S_{FF} through the first j items in the sequence for some j. Then there exists a reduced schedule S' that makes the same eviction decisions as S_{FF} through the first $j+1$ items and incurs no more misses in total than S does in total.*

Prove this by constructing S'. *This is an **exchange** argument.*

a. What happens if the $j+1^{st}$ item is in cache?

b. What happens if the $j+1^{st}$ item isn't in cache, but S evicts the same item as S_{FF}?

c. What happens if the $j+1^{st}$ item isn't in cache, and S evicts a different item as S_{FF}? What should S' do?

d. How can you get S''s cache back to the same as S's without incurring more total misses?

e. How do we know that S' is a reduced schedule?

f. Sanity check: Are all parts of the lemma true?

Theorem 2. *S_{FF} incurs no more misses than any other schedule S^* and hence is optimal.*

Recap: Proof by Greedy Exchange

Step 1: Label. Label your algorithm's solution ($A = \{a_1, a_2, \ldots, a_k\}$), and a general solution ($O = \{o_1, o_2, \ldots, o_m\}$).

Step 2: Compare. Compare greedy with the other solution. Assume that they're not the same and isolate some difference.

Suppose they are the same through the first j items. Then we show in the following lemma, by an exchange argument, that we can modify them to be the same through the first $j + 1$ items.

Step 3: Exchange. Swap the elements in O without making the solution worse. Argue that swapping a finite number of times will result in A. Hence, greedy is just as good as *any* optimal or arbitrary solution.

After k exchanges, O has been transformed into A without increasing the cost.