

CSC2421: Online and Other Myopic Algorithms

Fall 2026

Allan Borodin

September 14, 2026

Lecture 2

We ended last week with a brief overview of the chapters.

I am not sure exactly which of the final Volume 2 chapters I didn't mention, so I will start by just briefly listing chapters 5 to 9 of Volume 2. I have now posted Volume 2 on Quercus. Please do not distribute. But please do tell me about any errors, or any suggestions that you have.

Then I will present a little more motivation for this course.

And then (at long last) we will proceed with some technical discussion of a few “classical” online problems.

Chapter 2 Chapters continued

Chapter 5 introduces the *real-time model*. In the literature of scheduling algorithms, the term online algorithm often refers to what I call a “real-time algorithm” where decisions about an input item arriving at *time* t can be made at any time $t' \geq t$. Real-time algorithms often allow for some form of *preemption/interruption*. We consider preemption with revoking, restarting, and resuming.

In Chapter 6 we study online algorithms with recourse; that is, algorithms that allow limited changing of previous decisions. This general area is called online algorithms with recourse. We also consider parallel and multi-pass online algorithms.

In Chapter 7, we discuss streaming algorithms and dynamic algorithms, topics that each deserve their own texts and courses. The emphasis here is on tradeoffs between resources (space for streaming, time for dynamic algorithms) and performance.

Final Chapters

In Chapter 8, we discuss offline greedy algorithms (and some extensions). This is formalized in terms of priority algorithms. Priority algorithms allow for some fixed or adaptive ordering of the input items but still when preprocessing the i^{th} input item, we do not have complete knowledge of the remaining items.

Chapter 9 discusses alternative measures of performance of an online algorithm. We conclude the text by returning to comments made at the start of Volume 1 about the paging problem. Namely, we observed that the basic competitive analysis model does not account for *locality of reference*. The Markov paging model (in Chapter 2) is one natural approach to model locality of reference. In Chapter 9 we consider the Working Set model originally proposed in the 1960s by Denning.

I will try to get to chapters in Volume 2 as fast as I can especially for chapters I am actively working on.

Why Study Online and Other Myopic Algorithms?

In many applications, there is an inherent online requirement. For example, *paging* (also called *caching*) algorithms have to decide what page to evict when the cache (or some level of the memory hierarchy) is full.

Some auctions need to be run in an online fashion. One very important example is online advertising (i.e. selling ads deriving from online search to advertisers).

In other applications it may not be a requirement but rather a desirable property. Moreover, online and other myopic algorithms tend to be **conceptually simple** and very efficient.

Even when online algorithms do not provide solutions that are “competitive” in performance with more complex algorithms, they can serve as a benchmark or initial solution when an initial solution is needed quickly.

The difference between online/myopic algorithm analysis and more standard algorithm analysis as in approximation algorithms and complexity theory.

The seminal work of Cook-Karp-Levin on NP -completeness strongly suggests that there are many related problems that cannot be optimally or approximately solved efficiently (i.e., in polynomial time). But this is a conjecture.

Although “we” strongly believe $P \neq NP$, we do not have explicit problems f in NP for which we can *prove* that f is not computable in quasi-linear time for a sufficiently general model of computation. Here I am hedging my bets rather than saying linear time.

In contrast, in the analysis of online and other myopic algorithms, we restrict the class of algorithms and prove negative results without any complexity (or even computability) assumptions. We use what are called *information theoretic* arguments to establish negative results. That is, because the algorithm is working with incomplete information, there are limitations to what such an algorithm can do.

Comments and disclaimers on the course perspective

- I consider this course to be a “foundational graduate course” even though online and other myopic algorithms are a small part of the field of *algorithm design and analysis*.
- Although the topic seems focused, the analysis of these algorithms often introduces analysis methods used more generally in algorithmic analysis.

Most graduate algorithms courses are biased towards some research perspective. Given that CS might be considered (to some extent) [The Science and Engineering of Algorithms](#), one cannot expect any comprehensive introduction to algorithm design and analysis in any course.

- I already mentioned that online algorithms is currently an active part of TCS. The reason is mainly that the number of online applications keeps expanding and new algorithmic ideas and results continue to be developed. Furthermore, there is increasing interest in the many variants of online algorithms and analysis that address current applications.

Rest of today's agenda

We begin with some basic problems studied implicitly or explicitly as online problems. This will provide a brief history of online algorithms.

Note: All these classic basic problems have been extended and studied in various models and applications.

We will then consider the identical machine makespan problem and discuss Graham's online greedy algorithm.

We will also consider Graham's LPT offline greedy algorithm for the makespan problem.

We will provide the basic template for deterministic and randomized online algorithms. These templates will be extended as we consider more general online algorithms.

Finally, we will give the basic template for priority algorithms, a precise model for greedy and "greedy-like" algorithms.

Some history of online problems and algorithms

Note: Please see the Historical Notes at the end of each Chapter for references. In particular, see Chapters 1 and 2 of the following problems. The stated dates are for what I think is the first published reference. In general dates may sometimes not reflect the actual chronology of results.

The Secretary Problem We are given a sequence of n candidates (e.g. secretaries) being considered in *random order* and the decision maker must either irrevocably reject a candidate or accept a candidate (and then not consider any more candidates). When each candidate arrives, the algorithm learns the value of the candidate. The objective is to select a candidate with the highest value. The problem was first explicitly introduced in a Scientific American article by Martin Gardner in 1960 but the problem is said to go back centuries (and is sometimes called the Sultan's Marriage Problem, and other names). Gardner called it the The Game of Googol. See Chapter 2 in Volume 2. The term Googol (for the number 10^{100}) was coined in 1920 by a 9 year old boy; see Wikipedia.

History continued

The line search problem There is one point x^* on the real line where a treasure is hidden. A navigator starting at $x_0 =$ is trying to find the point x^* travelling as little distance as possible.

Note: The line search problem doesn't fit into the general class of online problems (called request-answer games) we consider in our text. It is an online navigation problem and has some relation to other online problems we consider. The problem was introduced by Beck in 1964

The rest of the following problems will assume that the sequence of input items are arriving in *adversarial order*.

The identical machines makespan problem We are given n jobs where each job has a load or processing time. The objective is to minimize the maximum load assigned to any machine. The problem was introduced by Graham in 1966 (with a followup paper in 1969).

More history

The paging problem Hard to say who first explicitly considered the basic paging problem but we can at least say that Denning studied the problem in 1968. It is one of the two problems explicitly studied in the seminal Sleator and Tarjan 1985 paper which is often considered as the start of online competitive analysis. There is a relatively small cache consisting of k fast pages, and a large number N of slow pages. Whenever there is a request for a slow page, it must be brought into the cache evicting some page from the cache when it is full. There is a cost of 1 for every cache miss, and the objective is to minimize the number of misses.

The bin packing problem As in the makespan problem, we are sequentially given n jobs each having a load or size $s_j \leq C$. We have machines or bins each having capacity C . The objective is to schedule the jobs on a minimum number of bins. The problem was extensively studied by Johnson in his 1973 thesis. The problem was also studied in 1980 by Yao who explicitly considered lower bounds (i.e. negative bounds) for the online problem.

End of brief history

List accessing The Sleator and Tarjan 1985 paper also studied the list accessing problem. In the list accessing problem we have a linked list of items. In the static problem, we have an initial list of items and we are given a sequence of requests and we need to determine if the request is one of the items in the list. (We can assume once we find an item in the list there is associated information that can then be retrieved.)

In the dynamic version of the problem, a request can be a search, or an insert or a delete. After each request, the algorithm is allowed to rearrange the list. The objective is to minimize the cost of satisfy requests plus the cost of reordering. There are different cost measures that are considered.

The ski rental problem

This problem was implicitly introduced in a Karlin et al 1988 paper on “snoopy caching”. There is a resource (e.g., a pair of skis) that can be purchased on any day or rented each day. Each day, the operator of the resource determines if this is the last day the resource is available. An algorithm must determine if and when to purchase the skis.

Greedy online algorithm for the identical machine makespan problem

- Let's start with this online greedy algorithm that dates back to Graham's 1966 and 1969 papers. Graham also considered an offline greedy algorithm called Longest Processing Time LPT in which the jobs are first sorted so $p_1 \geq p_2 \dots \geq p_n$ and then scheduled greedily.
- These are good starting points since (preceding NP-completeness) Graham conjectured that these are hard (requiring exponential time) problems to compute optimally but for which there were “good” worst case approximation ratios (although he didn't use that terminology).
- This might then be called the start of worst case approximation algorithms.
- There are some general concepts to be observed in Graham's work and (even after more than 50 years) still open questions concerning the initial makespan problem and the many variants of makespan problems.

We ended the Monday class just mentioning the makespan problem. This is where we will continue the next class.