Online Algorithms

Online algorithms are an optimization paradigm where input is revealed sequentially and an algorithm has to make irrevocable decisions using only causal information. This is a growing area of research with great interest from the theoretical computer science community, having significant practical applications in operations research, big data analysis, design of communication networks, and so on. There are many different mathematical techniques that have been developed to analyse online algorithms, such as potential function arguments, primal–dual methods, and Yao's principle, to name a few. This textbook presents an easy but rigorous introduction to online algorithms for students. It starts with classical online paradigms like ski-rental, paging, list-accessing, and bin packing, where performance of the algorithms is studied under the worst-case input and moves on to newer paradigms like 'beyond worst case', where online algorithms are augmented with predictions using machine learning algorithms. Several other popular online problems, such as metrical task systems, which includes the popular k-server problem as a special case, secretary, knapsack, bipartite matching, load balancing, scheduling to minimize flow-time, facility location, k-means clustering, and travelling salesman, are also covered. A very useful technique for analysing online algorithms called the primal–dual schema is also included together with its application for multiple problems. The book goes on to cover multiple applied problems such as routing in communication networks, server provisioning in cloud systems, communication with energy harvested from renewable sources, and sub-modular partitioning. Finally, a wide range of solved examples and practice exercises are included, allowing hands-on exposure to the concepts. Each exercise has been broken down into simpler parts to provide a clear path towards the solution.

Rahul Vaze is Associate Professor at the School of Technology and Computer Science, Tata Institute of Fundamental Research, Mumbai, India, an institution he has been associated with since 2009. His research interests include online algorithms, wireless communication theory, resource allocation, and network information theory. He has previously published *Random Wireless Networks: An Information Theoretic Perspective* with Cambridge University Press in 2015.

Online Algorithms

Rahul Vaze

CAMBRIDGE
UNIVERSITY PRESS

Shaftesbury Road, Cambridge CB2 8EA, United Kingdom

One Liberty Plaza, 20th Floor, New York, NY 10006, USA

477 Williamstown Road, Port Melbourne, VIC 3207, Australia

314–321, 3rd Floor, Plot 3, Splendor Forum, Jasola District Centre, New Delhi – 110025, India

103 Penang Road, #05–06/07, Visioncrest Commercial, Singapore 238467

Cambridge University Press is part of Cambridge University Press & Assessment, a department of the University of Cambridge.

We share the University's mission to contribute to society through the pursuit of education, learning and research at the highest international levels of excellence.

www.cambridge.org
Information on this title: www.cambridge.org/9781009349185

First published 2023

A catalogue record for this publication is available from the British Library

ISBN 978-1-009-34918-5 Paperback

To all the COVID-19 victims and warriors.
This book was mostly written while working from home during full/partial lockdowns.

Contents

Preface

Let me begin with a disclaimer! Online algorithms, the subject topic of this book, have nothing to do with the internet or the online connected world. Online algorithms should really be called limited information algorithms or myopic algorithms that have to make decisions with limited information while being compared against the best algorithm in hindsight.

The simplest example of an online algorithm is the game of Tetris, where at each time, the player has to make a decision about where to place the newly arrived tile, given the current state of tile positions and the knowledge of only the newly revealed tile and the next upcoming tile, so as to make as many completed lines with similar colour disappear as possible. Clearly, if all the future arriving tiles were revealed at each time, the optimal placement of the newly arrived tile to maximize the number of completed lines can be computed. However, with the limited information setting of the game, the quest is to get as close to the optimal algorithm in hindsight.

To put online algorithms in perspective, let's ask a question: what does an algorithm usually do? Given a (full) input instance, it provides a routine to optimize an objective function, subject to a set of constraints. When the full input instance is known before the algorithm starts to execute, it is referred to as the *offline* setting.

For many optimization problems of interest, however, the input instance is revealed sequentially, and an algorithm has to execute or make irrevocable decisions sequentially with the partially revealed input amid uncertainty about the future input, e.g., as we discussed for the game of Tetris. This sequential decision setting is generally referred to as the **online** setting, and the corresponding algorithm as an **online algorithm**. Compared to the offline algorithm, an online algorithm's output is a function of its sequentially made decisions and the order in which input is revealed.

To contrast the offline versus the online setting, consider one of simplest problems of memory management in random access memory (RAM) of computing systems. RAM is a fast but limited sized memory, and files before processing have to be loaded in the RAM. At each step of computation, a file is requested. If the requested file is available in the RAM, then execution starts immediately. Otherwise, a fault is counted to model the delay, etc., for loading the requested file into the RAM before execution. Since RAM is of limited size, to load the newly requested file, some existing file has to be ejected. For fast processing, one needs to minimize the total number of faults, which in turn depends on the choice of the file being ejected on each fault by the algorithm. This problem is popularly known as the paging or the caching problem. In the offline setting, a simple algorithm that ejects the file whose next request is farthest in time is optimal. The online setting is more relevant but non-trivial, where an algorithm has to make a decision about which file to eject without knowing the set of files to be requested in future.

Other prominent examples of sequential decision problems include: job scheduling in data/call centres, where jobs arrive sequentially and processing decisions are made causally, the (bin) packing problem, where items of different sizes have to be packed in the smallest number

of bins, where each bin has a fixed capacity, e.g., storing differently sized data files on finite sized disks, an ad allocation problem in web-advertising, where on a user's arrival to a webpage an online decision has to be made about which ads to display, and many others.

Because of the sequential input setting, quantifying the performance of an online algorithm is difficult, i.e., directly defining or finding an optimal online algorithm is hard. Hence, a 'strong' performance measure called the *competitive ratio* is used, which is defined as the maximum of the ratio of the objective function of an online algorithm and the optimal offline algorithm (which has access to full input to begin with) over all possible input instances, where inputs can even be chosen adversarially. Thus, the goal is to design online algorithms with small competitive ratios. Even though holding an online algorithm to the standard of the offline optimal algorithm via competitive ratio appears unfair, surprisingly for most of the optimization problems of interest, online algorithms with small competitive ratios have been derived. Thus, even with a limited view of the input, online algorithms can remain 'close' to the offline optimal algorithm, which is remarkable.

Designing optimal or near-optimal online algorithms is not only of theoretical interest but also of profound practical significance. For example, the web advertising business is worth billions of dollars, where on arrival of a user to a webpage, a decision is made about which ads to display depending on the user profile. Each advertiser pays a fixed revenue to the web platform to display an ad and accrues a payoff or utility from the user depending on its profile. The ad display decision has to be made online, as and when a user arrives, without knowing the profiles (consequently the utilities) of users arriving in the future. Other paradigms where sequential decisions are made with progressively accumulating data include cloud computing and big data problems, both of which are multi-billion dollar businesses.

Theoretical results on online algorithms began with the consideration of the classical problems, such as list accessing, bin packing, paging, load balancing, and job scheduling. These are examples of canonical online problems with tremendous impact in the area of operations research, operating systems, and large-scale computing in data centres. As the area evolved, rich combinatorial problems with wide applications were considered, such as the facility location, the travelling salesman, and the knapsack.

Problems motivated by more modern applications such as web advertising (AdWords problem), server allocations in cloud computing (convex optimization with switching costs), and machine learning (k-means clustering) have been considered in depth more recently. Many applied online paradigms have also been studied extensively, e.g., green communication networks, where nodes are powered by harvesting energy from renewable sources, and the transmission decisions have to be made without any knowledge of future harvested energy.

Since online algorithms are compared against optimal offline algorithms, which is a somewhat pessimistic metric, online algorithms endowed with a little more 'power' are also of interest. Typically, this power is either in terms of (i) resource augmentation, where the online algorithm is allowed more resources than the offline algorithm, (ii) lookahead, where the online algorithm is allowed to know the future input of finite length, and (iii) recourse, where the online algorithm is allowed to change a finite number of its prior decisions. For different online problems one or more of these possibilities are relevant.

With the advancement of the machine learning models, for many online problems, future inputs can be reasonably predicted. For example, the arriving user profiles in the web advertising model. Following this, the most recent trend in the area of online algorithms is to design online

algorithms that are arbitrarily close to the optimal offline algorithm if the prediction is accurate, while the same online algorithm has a small competitive ratio even if the prediction is inaccurate. Importantly, this needs to be accomplished without knowing the accuracy level of the machine learning model.

BRIEF DESCRIPTION OF THE BOOK

There are many different mathematical techniques that have been developed to analyse online algorithms, such as potential function arguments, primal–dual methods, Yao's principle, and beyond the worst case analysis with machine learning predicted input, to name a few. We present a comprehensive view of most of these techniques and illustrate their use via multiple examples. The book is presented in a cohesive and easy-to-follow manner but without losing the mathematical rigour. The treatment is such that the book is accessible to anyone with mathematical maturity without any necessary advanced prerequisites. Sufficient background and critical details are provided for the advanced mathematical concepts required for solving the considered problems.

In the first part of the book, we cover most of the classical online paradigms, such as the ski-rental, metrical task systems, list accessing, bin packing, metrical task system, k-server, paging, secretary, knapsack, bipartite matching, and load balancing. Next, we discuss a generic technique called the primal–dual schema, and present algorithms built using the primal–dual philosophy for important online problems such as set cover, AdWords, etc. The two related and important problems called the facility location and k-means clustering (a fundamental problem in machine learning) are discussed next. Thereafter, we present three prominent and interrelated scheduling problems in sufficient detail.

In addition to the classical worst-case input setting, we also consider the new paradigm of 'beyond worst case', where online algorithms are augmented with predictions using machine learning algorithms to get competitive ratio results that degrade smoothly with prediction error, without any prior information on the quality of the prediction. We consider this prediction paradigm in depth for multiple problems, such as ski-rental, paging, secretary, and scheduling with deadlines.

The final part of the book covers applied problems such as routing in networks, server provisioning in cloud systems (convex optimization with switching constraints), communication with energy harvested from renewable sources, and submodular partitioning.

As mentioned before, online algorithms with some slightly more power than the optimal offline algorithm, known as resource augmentation, are also of interest. We discuss three different regimes of resource augmentation for the relevant problems at appropriate places in the book mainly via exercises.

TARGET AUDIENCE

Online algorithms is a growing area of research with great interest from the theoretical computer science community, in the field of operations research and combinatorial optimization, for the

design of communication networks, and solving resource allocation problems. The book is targeted at interested graduate students and first-time readers looking for an easy and rigorous introduction to the area of online algorithms.

REASONS FOR WRITING THE BOOK

Research on online algorithms primarily began in the 1980s when classical problems like list accessing, bin packing, ski-rental, and paging were considered. The area gained more interest in the 1990s with a lot of work reported on load balancing and scheduling, in addition to the classical problems. In the following two decades, 2000–2020, attention to online algorithms increased rapidly because of the advent of motivating problems from web advertising, machine learning, cloud computing, etc.

The aim of this book is to review this extensive work and present the material in a 'textbook' form that is readily accessible to senior undergraduate and graduate students. There are lecture notes available on the web covering specific topics; however, this book takes a unified view, and presents a cohesive treatment of interrelated concepts, and brings out connections between different problems of interest.

This book is born from the lecture notes that I have been developing while teaching this course for graduate students in 2017, 2019, 2021, and 2022, at the Tata Institute of Fundamental Research, Mumbai. The said lecture notes have been used by the students, and their feedback has been incorporated to improve the readability of the book. Most of the exercises have also been solved by them via homework or exam problems, providing a good gauge of their accessibility. I also maintain video course lectures for the full course at www.youtube.com/playlist?list=PLTtM9ThZ2L-c638AjqTskivmXwVTzTYnl which will keep getting regular updates.

For most of the chapters in the book, the focus is on describing the basic ideas that capture the key concepts that are useful for the considered problem, together with elegant analysis. In the interest of ease of exposition, for certain problems an algorithm with a weaker competitive ratio guarantee is discussed compared to the best-known algorithm. Exercises are presented at the end of each chapter allowing a hands-on exposure to the basic concepts covered in the chapter. Most of the exercises are broken down into several (simpler) parts to provide a clear path towards the solution. Some algorithms are also introduced and analysed via exercises that are not covered in the main body due to space constraints. To really make good use of the book, solving a large fraction of exercises is highly recommended.

The book is much longer than initially planned and even then many important online formulations could not be accommodated. Notable exclusions include discrepancy, graph colouring, disjoint set cover, minimum spanning tree, matching for general graphs, crowdsourcing, network throughput maximization, scheduling to minimize the age of information, and several others. All these problems are important in their own right and should be studied in detail.

HOW TO USE THE BOOK

For instructors

The book is longer than what can be covered in a usual full semester course of roughly 25–26 lectures. For my last two full course offerings, I was able to cover around 12 chapters each time. For a full-semester course, I would recommend covering Chapters 1–6 always, in addition to Chapter 10 on the primal–dual technique. Thereafter, depending on the instructor's taste and students' interests, one can choose either the combinatorial suite, Chapters 7–9, or the scheduling group, Chapters 12–15. Other more applied chapters can be prescribed as student reading.

For students

Essential reading for easy access to all parts of the book should include Chapter 1, Chapter 2 on ski-rental (for Yao's principle), and Chapter 7 (for the secretarial input model). All other chapters are self-contained with these three chapters as pre-requisites.

Classical online problems are reviewed in Chapters 2–6. These chapters should also give the basic flavour of results as well as generic analysis techniques. For students interested in scheduling, they can focus on Chapters 12–15, while for subset selection or combinatorial problems Chapters 7–9 and 20 should be of interest. The most versatile tool used for analysing online algorithms is the primal–dual technique that is reviewed in Chapter 10. Other chapters should be referenced for more applied problems.

FINAL REMARKS

I began the preface with a disclaimer; let me end with another. I am not a theoretical computer scientist working in the broad area of online algorithms. My own contributions to the area of online algorithms are mostly limited to scheduling problems motivated by applications in communication networks, which are only briefly reviewed in the book. However, I am deeply interested in the broad area of online algorithms and remain a sincere student. This book also reflects that passion with special emphasis on it being accessible to beginner students. Experts might find some parts too laborious but it is done keeping student interests in mind. For many problems covered in the book the original proofs are highly clever but very terse. With the aid of additional explanation, relevant examples, and illustrative figures, this book takes special care in presenting the proof ideas for simpler exposition for readers with limited background.

Acknowledgements

I would like to thank everyone who has made this book possible, especially my wife Rashmi, and son Niraad. All of us stuck together during the tough COVID period, during which most of the book was written.

I would like to thank all my colleagues who have made detailed comments on my various drafts that undoubtedly made the book more readable. Their critical comments have also shaped the structure and content of this book. Comments by Arindam Khan made me add a couple of chapters and helped reorganize the book. Discussions with Thomas Kesselheim about the corrected proofs of his papers on the weighted matching and knapsack problem have been very helpful. Comments by Rajshekhar Bhat, Abhishek Sinha, Mohit Sharma, Jaikumar Radhakrishnan, Kumar Appaiah, and Jayakrishnan Nair have helped immensely in polishing the writing. Thanks are also due to all the students who in the recent past have taken my course on online algorithms and helped in proofreading the manuscript. I would especially like to thank Spandan Senapati, Akhil Bhimaraju, Kumar Saurav, Kshitij Gajjar (who was a PhD student at TIFR, now an assistant professor at IIT-Jodhpur), Pranshu Gaba, Hari Krishnan P. A., Praveen C. V., Arghya Chakraborty, Malhar Managoli, Yeshwant Pandit, Agniv Bandyopadhyay, and Soumyajit Pyne.

Critical feedback from reviewers has also been helpful in smoothing the rough edges of the book and to keep a clear and sharp focus.

Notation

$\mathcal{A}$	A deterministic online algorithm
$\mathcal{R}$	A randomized online algorithm
OPT	Optimal offline algorithm
σ	input
$\mu_{\mathcal{A}} = \max_{\sigma} \frac{C_{\mathcal{A}}(\sigma)}{C_{\text{OPT}}(\sigma)}$	competitive ratio of a deterministic algorithm $\mathcal{A}$
$\mu_{\mathcal{R}} = \max_{\sigma} \frac{\mathbb{E}\{C_{\mathcal{R}}(\sigma)\}}{C_{\text{OPT}}(\sigma)}$	competitive ratio of a randomized algorithm $\mathcal{R}$
$\mathbb{P}(A)$	Probability of event A
$\mathbb{E}$	Expectation operator
$\mathbf{A}$	Matrix A
$\mathbf{A}(i,j)$	$(i,j)^{th}$ entry of matrix A
$\mathbf{a}$	vector a
$\mathbf{a}(i)$ or $\mathbf{a}_i$	i^{th} element of vector $\mathbf{a}$
$\mathbf{a}^T, \mathbf{A}^T$	Transpose of vector $\mathbf{a}$ or matrix $\mathbf{A}$
$\mathbb{R}, \mathbb{Z}, \mathbb{N}$	Set of real, integer, and natural numbers, respectively
$\mathbb{R}^+, \mathbb{Z}^+$	Set of non-negative real and integer numbers, respectively
$\mathbb{R}^{++}, \mathbb{Z}^{++}$	Set of positive real and integer numbers, respectively
$\mathbb{R}^d$	Set of real numbers in d dimensions
i.i.d.	independent and identically distributed
$\|A\|$	Number of elements in set A
For two sets A, B, $A \backslash B$	Set difference, elements of A that are not in B
$\|a\|$	Absolute value of for a real number a
$\mathbf{1}_E$	Indicator variable which is 1 if event E is true and 0 otherwise
∇f	derivative of f
t^+ and t^-	Just after time t and just before time t, respectively
$[n]$	the set $1, 2, \ldots, n$
$\binom{n}{k}$	all possible ways of choosing k elements out of total n elements

$f(n) = \Omega(g(n))$	If $\exists\ k > 0,\ n_0,\ \forall\ n > n_0, \lvert g(n)\rvert k \leq \lvert f(n)\rvert$
Big O $f(n) = \mathcal{O}(g(n))$	If $\exists\ k > 0,\ n_0,\ \forall\ n > n_0, \lvert f(n)\rvert \leq \lvert g(n)\rvert k$
$f(n) = \Theta(g(n))$	If $\exists\ k_1,\ k_2 > 0,\ n_0,\ \forall\ n > n_0, \lvert g(n)\rvert k_1 \leq \lvert f(n)\rvert \leq \lvert g(n)\rvert k_2$
Small O $f(n) = \mathrm{o}(g(n))$	If $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$