The Value of Cooperation: From AIMD to Flipped Classroom Teaching

Chee Wei Tan
Nautilus Software Technologies
City University of Hong Kong

ABSTRACT

The well-known Additive Increase-Multiplicative Decrease (AIMD) abstraction for network congestion control was first published by Dah-Ming Chiu and Raj Jain in their seminal work [4] in 1989 and soon played a prominent part in TCP algorithm design for the Internet. The ingenuity of AIMD lies in the abstraction of Internet congestion control, and ever since its inception has also been a staple part of teaching curriculum for performance evaluation and computer networking courses at universities worldwide. In this paper, we describe teaching examples for university students to appreciate the AIMD abstraction from the theoretical aspects such as convex optimization and Perron-Frobenius theory to the data science aspect. The essence of cooperation encompassed by AIMD reverberates even in teaching networks formed by students and educators, giving rise to online classroom flipping teaching tools and data analytics to close the gap between teachers and students.

Keywords

Performance Analysis, AIMD, Data Science, Teaching and Learning, Classroom Flipping

1. INTRODUCTION

In the past, the performance analysis of a computer system often depends on a mathematical model. This performance model serves as a first approximation to the underlying real-world system, which may be too complex to analyze. Over the years, some elegant and useful performance models such as the queuing models and stochastic models have emerged [15, 12, 14, 16, 9]. To the practitioners, this performance model choice is often a trade-off between mathematical tractability and its relevance to the real-world system. Today's computer systems are becoming more complex, and finding an appropriate model becomes even more crucial. The right "abstraction" helps to focus on the most interesting and crucial aspect of a complex problem, and opens the door to a variety of performance analysis techniques. It allows one abstraction to relate to other abstractions, and is useful for exploring the trade-off between tractability and approximation.

The Additive-Increase-Multiplicative-Decrease (AIMD) algorithm abstraction proposed by Dah-Ming Chiu and Raj Jain in [6] is one of the most influential work on Internet congestion control in the field of networking.[1] In the early days of the Internet, there were many different models to study congestion control, some from the viewpoint of a single flow while others assume synchronous events or fluid flows. There was even little understanding of the convergence and stability properties of the then existing congestion control algorithms for the Transmission Control Protocol (TCP) protocol in the Internet.

Performance '21: 39th IFIP WG 7.3 International Symposium on Computer Performance, Modeling, Measurements and Evaluation 2021 International Workshop on Teaching Performance Analysis of Computer Systems, November 08–12, 2021, Milan, Italy

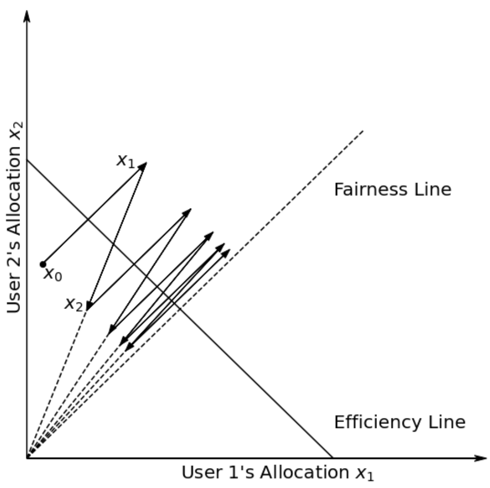
The most important contribution of the AIMD abstraction is to pose the congestion control problem as a problem with social consequences, whereby the parties encountering the same congestion need to work cooperatively to arrive at a good solution. The AIMD abstraction simplifies the problem of congestion control by considering only two flows, which is enough to address the issues of feedback-driven algorithm design. The AIMD abstraction has also been a staple part of performance evaluation and computer networking courses at colleges and universities worldwide due to its coverage in popular textbooks (e.g., see [17, 31]).

Interestingly, the ingenuity behind the AIMD abstraction can be appreciated through a classical illustration of two flows as shown in Figure 1a.[2] This illustration can be found in a number of popular computer networking textbooks (see, e.g., [17, 31]) to describe at length the engineering insight behind the AIMD abstraction. Indeed, an insightful illustration to accompany the right abstraction to a complex problem can be helpful to teaching university students to appreciate the problem. The AIMD abstraction in [6] has ushered in the development of robust TCP algorithms in [11] and new performance analysis techniques using convex optimization theory, Perron-Frobenius theory and many other mathematical tools. This AIMD abstraction has also found applications in defending against denial of service attacks [30], charging electric cars [7] and other large-scale distributed systems.

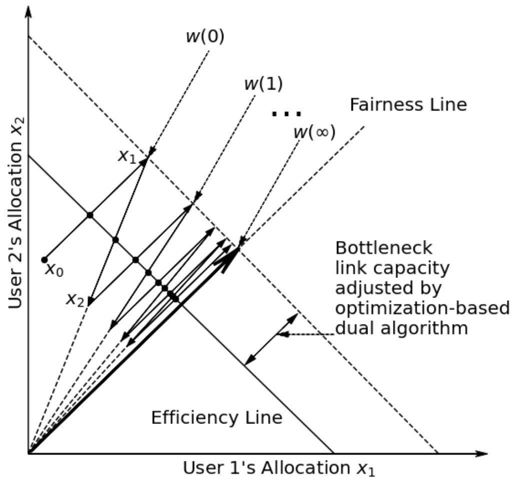
In the following, we discuss how the AIMD abstraction opens the door to two unique performance analysis techniques using convex optimization theory and Perron-Frobenius theory as well as teaching them in universities at the undergraduate and graduate levels. We then conclude the paper by drawing a parallel between the AIMD abstraction and the flipped classroom approach for teaching.

[1]This paper is in honor of Professor Dah-Ming Chiu on the occasion of his 70th birthday.

[2]Wittgenstein's principle of "*what can be shown, cannot be said*" applies fittingly to *illustrating an abstraction.*



(a) A visual proof of the AIMD abstraction showing that the rates of two users converge to the fairness line. When the rate iterates are below the efficiency line, the additive-increase mechanism is akin to a 45-degree increment plot. Otherwise, the multiplicative-decrease mechanism is akin to sliding to the midpoint between the current rate iterate and the origin. Photo courtesy of [6].



(b) Illustrating how the AIMD abstraction teaches convex optimization theory and Perron-Frobenius Theory. Optimization dual algorithm adapts the efficiency line to balance supply and demand (cf. Section 2). The rate trajectory along the efficiency line is analyzed using Perron-Frobenius theory, showing its convergence to the fairness line that is interpreted as a Perron-Frobenius right eigenvector (bold arrow), i.e., $\lim_{k\to\infty} w(k) \propto (1,\ 1)^T$ (cf. Sec. 3).

2. AN OPTIMIZATION THEORETIC PERSPECTIVE OF AIMD ABSTRACTION

Beginning in the late 1990s, a theoretical framework called Network Utility Maximization (NUM) has emerged that can analyze both the equilibrium and the dynamical nature of TCP algorithms [13, 28, 26, 22, 33]. The usefulness of this elegant framework brings tools and ideas from convex optimization theory [3] to bear on the design of internet congestion control algorithms. By leveraging Lagrange duality and gradient descent algorithms, TCP can be interpreted as a dual algorithm that maximizes the aggregate utilities in the network [13, 28, 26, 22, 33]. Specifically, AIMD can be viewed as solving an implicit network utility maximization where the sending source rates and network congestion measures are interpreted as primal variables and dual variables respectively.

The basic network utility maximization of n sending sources can be formulated as follows:

$$\text{maximize} \sum_{s=1}^{n} U_s(x_s) \ \text{ subject to } Rx \preceq c,\ x \succeq 0,$$

where x_s is the sending rate of source s given a routing matrix R with entries $R_{ls} = 1$ if source s uses link l with a link capacity c_l or 0 otherwise. At the tth iteration, source s solves:

$$x_s^*(q_s) = \text{argmax}\left\{U_s(x_s) - q_s x_s\right\},$$

and the lth link runs the algorithm:

$$p_l(t+1) = \max\left\{p_l(t) - \alpha(t)\left(c_l - \sum_{s:l\in L(s)} x_s^*(q_s(t))\right), 0\right\}.$$

Essentially, each link updates its congestion measure $p_l(t)$ (i.e., the shadow price) based on the subgradient algorithm in convex optimization theory. Each source has access to the total price incurred on the end-to-end route (ignoring the end-to-end delay) as $q_s(t) = \sum_l R_{ls} p_l(t)$. Now, certain choices of step size $\alpha(t)$ (e.g., $\alpha(t) = 1/t$) can guarantee the convergence to the globally optimal solution (x^*, p^*), indicating that the source and link algorithms jointly balance the supply and demand through pricing. In particular, using AIMD with an end-to-end marking probability q_s and a total delay D_s, TCP Reno has the utility function [22, 28, 26]:

$$\text{arctan utility} : U_s(x_s) = \frac{\sqrt{3/2}}{D_s} \arctan\left(\sqrt{2/3} x_s D_s\right).$$

This is due to the Karush-Kuhn-Tucker optimality condition: $\frac{d\,U_s(x_s)}{d\,x_s} = q_s(x_s)$ where $q_s = \frac{3}{2x_s^2 D_s^2 + 3}$, thus obtaining the above utility function after integration. This is also the mathematical basis for other fairness abstraction like proportional fairness in TCP Vegas and FAST TCP [22, 32]. Many other forms of TCP do not leverage Lagrange duality directly as they are instead interpreted as solutions to a penalty function formulation of the optimization problem [28, 26].

A teaching curriculum at the graduate student level could cover basic convex optimization theory like Lagrange duality, Karush-Kuhn-Tucker conditions [3] and optimization algorithm design. Students can appreciate the AIMD abstraction or other TCP algorithms as gradient-based optimization algorithms that implicitly cooperate to maximize the network utility. This abstraction ties in naturally with the idea of *layering as optimization decomposition* in [5] that establishes a mathematical theory of network architectures based on an optimization-theoretic perspective for cross-layer optimization [22, 4]. In essence, optimization theory is the *language* to mathematically quantify cooperation within and between *the layers of abstractions* of the OSI model network stack for the Internet.

Students can be taught using performance analysis and data science techniques to interpret measurement data under a variety of network congestion scenarios like a single bottleneck link with two flows or a network whose link capacity can be shaped by the abstraction of the physical or medium access control layer (e.g., algorithms for wireless network optimization [29]). Project-based learning can be incorporated in a graduate teaching curriculum to guide students to fine-tune optimization-based algorithms,[3] to design data analytics to reduce data-driven models into appropriate performance models with parameters to be optimized, and to validate different utility functions using convex optimization theory. In addition, the optimization perspective allows students to appreciate how the theory of primal and dual decomposition in distributed optimization applies to connecting abstractions across the network protocol layers.

3. A PERRON-FROBENIUS THEORETIC PERSPECTIVE OF AIMD ABSTRACTION

The AIMD abstraction can offer a glimpse to the theory of positive switched linear systems and dynamical systems theory. In particular, a positive discrete-time system can precisely describe the trajectory of rates at the bottleneck link as illustrated in Figure 1a. Following [1, 7], let $w_s(k)$ denote the congestion window size of source s immediately before the kth network congestion event is detected by all the sources as shown in Figure 1b. Let α_s and $0 < \beta_s < 1$ be the additive and multiplicative parameters of source s using the AIMD algorithm (that are conventionally set as 1 and 0.5) respectively. From the assumption of equal round-trip-times for each source (i.e., synchronous action of all sources), it can be seen that the window evolution under AIMD is completely defined over all time instants by knowledge of the $w_s(k)$ and the three time epochs corresponding to the time at which the number of unacknowledged packets in the pipe equals $\beta_s w_s(k)$, the time at which the pipe is full and the time when all the sources detect congestion simultaneously.

Let $q_{\max}$ and P be, respectively, the maximum queue length of the congested bottleneck link and the maximum instantaneous number of sent unacknowledged packets that are in transit (e.g., $P = q_{\max} + BT$ where B is the bottleneck link service rate in packets per second and T is the round-trip time). Therefore, we have [1]:

$$\sum_{i=1}^{n} w_i(k) = P + \sum_{i=1}^{n} \alpha_i, \ \ \forall\, k > 0.$$

[3] An example is writing the smallest linear program solver in Matlab, which is inspired by a convex optimization course at Stanford University, as described in the Appendix.

At the $(k+1)$th congestion event, the congestion window of source s satisfies

$$w_s(k+1) = \beta_s w_s(k) + \left(\frac{\alpha_s}{\sum_{i=1}^{n} \alpha_i}\right) \sum_{i=1}^{n} (1-\beta_i)\, w_i(k),$$

and, letting $w(k) = (w_1(k), \ldots, w_n(k))^T$, a positive linear system in matrix form is obtained as [1, 7]:

$$w(k+1) = Aw(k),$$

where

$$\begin{aligned} A \ = & \begin{bmatrix} \beta_1 & 0 & \cdots & 0 \\ 0 & \beta_2 & 0 & 0 \\ \vdots & 0 & \ddots & 0 \\ 0 & 0 & \cdots & \beta_n \end{bmatrix} \\ & + \ \frac{1}{\sum_{i=1}^{n} \alpha_i} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \cdots \\ \alpha_n \end{pmatrix} (1-\beta_1,\ 1-\beta_2,\ \cdots,\ 1-\beta_n)\,, \end{aligned}$$

whose spectrum (particularly, the Perron-Frobenius eigenvalue and eigenvectors) provides insights on fairness, rate of convergence and transient response [1]. In particular, we have

$$\lim_{k\to\infty} w(k) = \left(\frac{\alpha_1}{1-\beta_1}, \ldots, \frac{\alpha_n}{1-\beta_n}\right)^T,$$

which, if specialized to the case of $\alpha_s = 1$ and $\beta_s = 0.5$ for all s, is proportional to the all-ones vector (i.e., the fairness line in [6]) as it should be. From a performance modeling perspective, the Perron-Frobenius theoretic perspective accurately quantifies the effect of AIMD abstraction parameters on fairness, and allows TCP rate adaptation to be simulated by the classical power method algorithm.

Networking courses at the undergraduate student level often use Figure 1a in [6] to demonstrate that AIMD guarantees convergence to fairness due to its coverage in popular textbooks like [17, 31]. The undergraduate curriculum could be adapted to cover the classical linear Perron-Frobenius theory [7] and the aforementioned eigenvector interpretation of the fairness line in [6]. The linear Perron-Frobenius theory should be accessible to undergraduate students as they may have come across its other applications like the Google Pagerank and Markov chains in other courses. Performance analysis can be combined with data science techniques to study the transient behavior and convergence of flow rates in a bottleneck link using the iterative power method algorithm. Students can conduct experiments using data flows with different AIMD parameters $\{\alpha_s, \beta_s\}$ for all s to compete with the conventional TCP sources to empirically deduce that the *cooperative* configuration (i.e., "TCP-friendly") is such that $\alpha_s = 2(1-\beta_s)$ for all s. In other words, all these different flows eventually share the capacity equally regardless of their initial point conditions.

A teaching curriculum at the graduate student level can involve the nonlinear versions of the Perron-Frobenius theory (e.g., see [29, 25]) to analyze the AIMD abstraction for more general problems. To enhance the accessibility to the finite dimensional nonlinear Perron-Frobenius theory, the AIMD abstraction can be a useful illustrative example to introduce graduate students to more advanced mathematical tools in abstract problem solving. Graduate students can also learn to design data analytics based on TCP data measurement and apply machine learning algorithms to the AIMD abstraction or data-driven models to create computer-generated TCP algorithms (e.g., see recent similar ideas in [27], [35]). These data analytics can also be used to validate assumptions. For example, the AIMD abstraction in [6] can be validated with a single bottleneck data-driven model, and machine learning helps to automate this process when the bottleneck link may shift around in the network due to variable traffic conditions.

4. THE POWER OF COOPERATION IN CLASSROOM TEACHING

In this section, we describe a flipped classroom teaching approach in [19] that leverages the power of cooperation in teaching the mathematical theories and data science skillsets mentioned in the previous sections. In traditional university teaching, the teacher tends to just give lectures and hand out homework to students – there is less cooperation between teacher and students as well as between students. A flipped classroom approach leverages the power of cooperation to improve the interaction between the teacher and students using feedback [10]. Can we engineer flipped classroom teaching tools to enhance teacher-student interaction and cultivate the spirit of cooperation between the teacher and students?

Interestingly, some ideas in flipped classroom teaching can be regarded as analogous to that of AIMD/TCP in distributed networking. Classroom flipping often requires the teacher to actively gather some form of information related to how students are learning. Let us draw a parallel between AIMD/TCP and flipped classroom teaching.[4] In AIMD/TCP, the users may wish to send as much data packets as possible without knowing the link capacity ahead in time. In teaching, what is the "*content capacity*" (related to teachers asking "*am I teaching too much?*" or "*am I teaching this not enough?*")? Is there a back-off when teachers realize that they are teaching too much content? In AIMD/TCP, the actual information of link capacity and degree of fairness are not known to the users sending the data packets. In teaching, what is the "*comprehension capacity*" (related to teachers asking "*are some students being overwhelmed and dropping behind?*" or "*is this material accessible to all the students?*")? Is there a slow start when teachers introduce new or more advanced concepts in class?

How to facilitate feedback in teaching to actively gather this information is of essence in any flipped classroom approach for teaching. Let us briefly describe how this information is obtained by short quizzes in two different classroom flipping approaches, namely Peer Instruction [34, 18, 23] and Just-In-Time Teaching [24]. Roughly speaking, quizzes are issued to students in class for Peer-Instruction and relies on the use of clickers – a kind of audience response system – to query students in classes and actively encourage them to seek out peers with different perspectives on a question to discuss before giving them the correct solution. Typically, students are first given a question and asked to vote individually before they get to see a whole-class response in the form of statistical display like histograms or pie-charts, and then students are asked to reflect on their votes and to engage in peer discussions. This poll-quiz routine is a unique feature in Peer Instruction [34, 18, 23].

The basic idea of Just-In-Time Teaching is for teachers to adapt class instruction by using some form of quiz feedback before students come to class [24]. Teachers can use the diagnostic results as talking points in class to engage students, and these quizzes can double as a low-stakes assessment when they are carefully designed. The purpose of quiz feedback is therefore instrumental to allow teachers and students to cooperate implicitly, and should be optimized in order to close the gap between teachers and students. The quiz feedback can be implemented in mobile software as students are likely to have personal mobile devices like a smartphone or tablet. We have developed mobile chatbot software technologies in [19] to blend together these two classroom flipping pedagogical methods for in-person or remote instructions. Beyond digitizing the clicker, the mobile chatbot software in [19] allows teachers to use the poll-quiz routine to regulate content delivery for the whole class to meet the "*content capacity*" and to use outside-class quizzes of varying difficulty levels to meet the "*comprehension capacity*" of individual students.

Let us describe the poll-quiz routine in [19] used for online teaching in 2020. The teacher first issues a poll whose instantaneous response outcome can be observed by the entire class, and a short discussion (e.g., peer discussions) ensues. This is then concluded by a time-limited quiz whose content is related to the prior poll. The polls and quizzes are typically multiple-choice questions but they can be enriched with multimedia contents, automated hints or interactive human-computer input like touch-screen annotation with auto-grading capability in [20, 21]. Figure 2a shows the mobile chatbot software interface with a poll assessing students on using an optimization software in [8] to solve a network utility maximization problem for rapid programming synthesis as shown in Figure 2b. An optimization framework is proposed in [19] to optimize the frequency and difficulty levels of quizzes, and data analytics can collate the data from the series of poll-quiz routines to identify students' weakness in learning and the instructors' blind spots in online teaching.

Now, creating a pair of poll and quiz for such online classroom flipping is an art. Patrick H. Winston who was an AI expert noted for his pioneering work in teaching excellence at MIT explained technology-enabled polling [36]: *One obvious advantage is that clicker polling does not embarrass shy students fearful of ridicule if they choose the wrong answer. One not-so-obvious advantage is that instructors who choose to have clickers feel obligated to use them, and so must conceive interesting and informing polling questions (Patrick H. Winston [36]).* Indeed, equipping instructors who are motivated to find interesting and engaging questions to synchronize in-class teaching with real-time feedback is important. Such a technology-enabled pedagogy can create new forms of online instruction by allowing teachers to find suitable operating points in the "*content capacity*" and to understand limitations due to the "*comprehension capacity*". This can be especially useful for a fully online instruction setting such as one necessitated by the COVID-19 pandemic, where the authors in [2] at Stanford University pointed out that teachers at the Computer Science department encountered problems to "read a room" while teaching remotely and highlighted a need for further research on blended forms of instruction.

[4] The author thanks Geoffrey M. Voelker for suggesting the parallel between AIMD/TCP and classroom teaching.

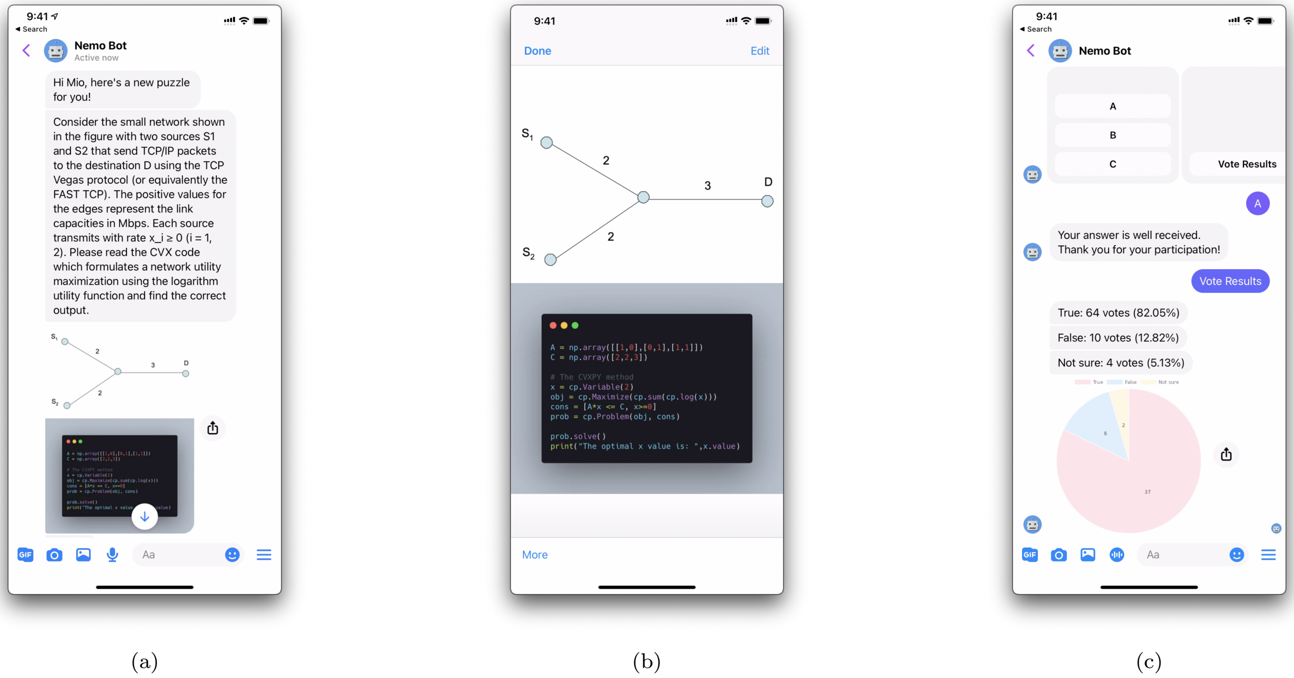


Figure 2: Students receive a poll via a mobile chatbot software on the topic of network utility maximization in (a) testing their understanding of the CVX optimization software [8] in (b). In (c), the student once having voted on the poll gets to see the whole class response to the poll before engaging in peer discussions and answering a timed quiz.

5. CONCLUSIONS

In conclusion, the principle of cooperation is the defining characteristic of the AIMD abstraction, leading to new insights on the performance evaluation of the Internet. Finding the right abstraction may take academic and industry collaboration to fully appreciate a problem. Data science analytics may be intermediate steps to strip down a complex problem to an appropriate abstraction that once fully analyzed can allow data science analytics to build back to a solution of the original problem. The data-driven models obtained in the process may lead to meaningful performance models that a performance analyst can interpret or optimize. Students can thus uncover a lot when they *learn to do abstraction* using both classical performance analysis and modern data science. Lastly, we draw a parallel between AIMD and the flipped classroom approach enabled by mobile software technologies to engender the spirit of cooperation between teachers and students.

6. ACKNOWLEDGEMENT

The author thanks Dah-Ming Chiu, Y. C. Tay, Vittoria de Nitto Persone, R. Srikant, Geoffrey M. Voelker and Lorenzo Alvisi for many useful and insightful discussions. The author is grateful to Mung Chiang, Steven Low and Stephen Boyd for their valuable advice on teaching convex optimization and networks. This work was funded in part by a Google faculty grant and UGC Teaching Award Proj. 6989041.

APPENDIX

In March 2010, Jacob Mattingley who was a TA for the Convex Optimization course EE364 at Stanford University put forth a challenge to write the *smallest* linear program solver in MATLAB, with the following code of seventy-five characters based on Dikin's interior point method :

```
for i=1:50,p=diag(x)∧ 2;r=p*(c-A'*(A*p*A'*p*c));
x=x-r*min(x./abs(r))/2;end
```

Zhonghao Zhang, a student in my class on optimization and networking, improved that to sixty-nine characters:

```
for i=1:50,X=diag(x);F=X*null(A*X);d=F*F'*c;
x=x-d/max(X(d))/2;end
```

with the latest version being forty-nine characters based on the algorithm of alternating direction method of multipliers and created by Borja Peleato.

The software aspect of convex optimization is interesting, but rarely covered in class. It can be a stimulating intellectual exercise for students to use their favorite programming language to come up with the tiniest possible versions of various optimization algorithms, e.g., smallest quadratic program solver in Python or C language, and then to test-drive their software on problems of increasingly massive size.

7. REFERENCES

[1] Abraham Berman, Robert Shorten, and Douglas Leith. Positive matrices associated with synchronised communication networks. *Linear Algebra and its Applications*, 393:47–54, 2004.

[2] Maxwell Bigman and John Mitchell. Teaching online in 2020: Experiments, empathy, discovery. In *Proceedings of the 7th Learning with MOOCS (LWMOOCS)*. IEEE, 2020.

[3] Stephen P Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[4] Mung Chiang. *Networked Life: 20 Questions and*

Answers. Cambridge University Press, 2012.

[5] Mung Chiang, Steven H Low, A Robert Calderbank, and John C Doyle. Layering as optimization decomposition: A mathematical theory of network architectures. *Proceedings of the IEEE*, 95(1):255–312, 2007.

[6] Dah-Ming Chiu and Raj Jain. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. *Computer Networks and ISDN Systems*, 17(1):1–14, 1989.

[7] Martin Corless, Christopher King, Robert Shorten, and Fabian Wirth. *AIMD Dynamics and Distributed Resource Allocation*. Society for Industrial and Applied Mathematics, 2016.

[8] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1 http://cvxr.com/cvx, March 2014.

[9] Mor Harchol-Balter. *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*. Prentice Hall Press, 2013.

[10] John Hattie and Helen Timperley. The power of feedback. *Review of Educational Research*, 77(1):81–112, 2007.

[11] Van Jacobson. Congestion avoidance and control. *ACM SIGCOMM computer communication review*, 18(4):314–329, 1988.

[12] Frank Kelly. *Reversibility and Stochastic Networks*. Wiley, 1979 and Cambridge University Press, 2011, 1979.

[13] Frank P Kelly, Aman K Maulloo, and David Kim Hong Tan. Rate control for communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research society*, 49(3):237–252, 1998.

[14] Leonard Kleinrock. *Queueing Systems: Volume 1*. Wiley-Interscience; 1st edition, 1975.

[15] Hisashi Kobayashi. *Modeling and Analysis: An Introduction to System Performance Evaluation Methodology*. Addison-Wesley, 1978.

[16] Hisashi Kobayashi and Brian L. Mark. *System Modeling and Analysis: Foundations of System Performance Evaluation*. Prentice Hall Press, 2009.

[17] Jim Kurose and Keith Ross. *Computer Networking: A Top-Down Approach, 8th edition*. Pearson, 2020.

[18] Nathaniel Lasry, Eric Mazur, and Jessica Watkins. Peer instruction: From Harvard to the two-year college. *American Journal of Physics*, 76(11):1066–1069, 2008.

[19] Jingting Li, Lin Ling, and Chee Wei Tan. Blending peer instruction with just-in-time teaching: Jointly optimal task scheduling with feedback for classroom flipping. *Proceedings of the Eighth ACM Conference on Learning Scale*, page 117–126, 2021.

[20] Lin Ling and Chee Wei Tan. Human-assisted computation for auto-grading. *Proceedings of the IEEE ICDM Workshop on Machine Learning for Education*, 2018.

[21] Lin Ling and Chee Wei Tan. Peer-grading at scale with rank aggregation. *Proceedings of the Eighth ACM Conference on Learning Scale*, 2021.

[22] Steven H Low. Analytical methods for network congestion control. *Synthesis Lectures on Communication Networks*, 10(1):1–213, 2017.

[23] Eric Mazur. *Peer Instruction: A User's Manual*. Pearson, ISBN 978-0135654415., 1997.

[24] G. N. Novak, E. T. Patterson, A. Gavrin, and W. Christian. *Just-in-Time Teaching: Blending active Learning and Web Technologys*. Prentice Hall, Saddle River, NJ, ISBN 0-13-085034-9., 1999.

[25] Roger D. Nussbaum and Bas Lemmens. *Nonlinear Perron-Frobenius Theory*. Cambridge University Press, 2012.

[26] S. Shakkottai and Rayadurgam Srikant. *Network Optimization and Control*. NoW Publishers Foundations and Trends in Networking, 2007.

[27] Anirudh Sivaraman, Keith Winstein, Pratiksha Thaker, and Hari Balakrishnan. An experimental study of the learnability of congestion control. *ACM SIGCOMM Computer Communication Review*, 44(4):479–490, 2014.

[28] Rayadurgam Srikant. *The mathematics of Internet congestion control*. Springer Science & Business Media, 2004.

[29] Chee Wei Tan. Wireless network optimization by perron-frobenius theory. *Foundations and Trends in Networking*, 9(2-3):107–218, 2015.

[30] Chee Wei Tan, Dah Ming Chiu, Chi-Shing Lui, and David Yau. A distributed throttling approach for handling high bandwidth aggregates. *IEEE Trans. on Parallel and Distributed Systems*, 18(7):983–995, 2007.

[31] Andrew Tanenbaum and David J. Wetherall. *Computer Networks, 5th edition*. Prentice Hall, 2010.

[32] David X. Wei, Cheng Jin, Steven H. Low, and Sanjay Hegde. Fast tcp: Motivation, architecture, algorithms, performance. *IEEE/ACM Trans. Netw.*, 14(6):1246–1259, dec 2006.

[33] Peter Whittle. *Networks: Optimisation and Evolution*. Cambridge University Press, 2007.

[34] Carl Wieman. *Improving How Universities Teach Science*. Harvard University Press, 2017.

[35] Keith Winstein and Hari Balakrishnan. Tcp ex machina: Computer-generated congestion control. *ACM SIGCOMM Computer Communication Review*, 43(4):123–134, 2013.

[36] Patrick H. Winston. *Make It Clear: Speak and Write to Persuade and Inform*. The MIT Press, ISBN 978-0262539388, 2020.