Online Algorithms

Susanne Albers[*] Stefano Leonardi[†]

Over the past twelve years, online algorithms have received considerable research interest. Online problems had been investigated already in the seventies and early eighties but an extensive, systematic study started only when Sleator and Tarjan [41] suggested comparing an online algorithm to an optimal offline algorithm and Karlin, Manasse, Rudolph and Sleator [29] coined the term *competitive analysis*.

Foundations

An online algorithm receives the input incrementally, one piece at a time. In response to each input portion, the algorithm must generate output, not knowing future input. In a competitive analysis an online algorithm A is compared to an *optimal offline algorithm* OPT. An optimal offline algorithm knows the entire input sequence in advance and can process it optimally. Given an input sequence I, let $C_A(I)$ and $C_{OPT}(I)$ denote the costs incurred by A and OPT in processing I. Algorithm A is called c-competitive if there exists a constant a such that $C_A(I) \leq c \cdot C_{OPT}(I) + a$, for all input sequences I. An analogous definition can be given for online maximization problems. We note that a competitive algorithm must perform well on *all* input sequences.

In the above definition it is assumed that A is a deterministic algorithm. Randomization often allows online algorithms to obtain better competitive ratios than deterministic algorithms. Ben-David *et al.* [13] explored the power of randomization in online algorithms. Given a randomized online algorithm A, an input sequence is generated by an *adversary*. Ben-David *et al.* introduced different kinds of adversaries that, when generating a new input portion, may or may not see the outcome of the random choices made by A on previous input.

Competitive analysis has been applied successfully to many interesting problems.

Applications

In the late eighties and early nineties, three basic online problems were studied extensively, namely paging, the k-server problem and metrical task systems. The paging problem is to maintain a two-level memory system consisting of a small fast memory and a large slow memory. The goal is to serve a sequence of requests to memory pages so as to minimize the number of page faults incurred. The k-server problem, introduced by Manasse *et al.* [35], generalizes paging as well as more general caching problems. The problem consists in scheduling the motion of k mobile servers that reside on points of a metric space S. Requests are issued at points in S and,

---

[*] Max-Planck-Institut für Informatik, Im Stadtwald, D–66123 Saarbrücken, Germany. E-mail: albers@mpi-sb.mpg.de

[†] Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza", via Salaria 113, 00198, Roma, Italy. E-mail: leon@dis.uniroma1.it

in response to each request, one of the servers must be sent to that point. The goal is to minimize the total distance traveled by all the servers. Metrical task systems, introduced by Borodin *et al.* [16], can model a wide class of online problems. A metrical task system consists of a pair (S, d), where S is a set of n states and d is a cost matrix satisfying the triangle inequality. Entry $d(i, j)$ is the cost of changing from state i to state j. A task system must serve a sequence of *tasks* with low total cost. We refer the reader to the books [15, 26, 37] for excellent presentations of results on these problems.

During the past years, apart from the three basic problems, many online problems were investigated in application areas such as data structures, distributed data management, scheduling and load balancing, routing, robotics, financial games, graph theory, and a number of problems arising in computer systems. We concentrate on two class of problems that have been studied extensively by many researchers.

Scheduling and load balancing problems. The general situation in online *scheduling* is as follows. We are given a set of m machines. A sequence of jobs arrives online. Each job has a processing time that may or may not be known in advance. Each job must be scheduled immediately on one of the m machines, without knowledge of any future jobs. The goal is to optimize a given objective function. There are many problem variants, e.g., one can study various machine types and various objective functions.

One of the most basic scheduling problems was introduced by Graham [25] in 1966. Suppose that we are given m *identical* machines. Whenever a new job arrives, its processing time is known in advance. The goal is to minimize the makespan, i.e., the completion time of the last job that finishes. Graham [25] showed that the *List Scheduling* algorithm, which always assigns a new job to the least loaded machine is $(2 - \frac{1}{m})$-competitive. It was unknown for about 25 years whether or not there exist algorithms that achieve a competitive ratio c, $c < 2$, for all values of m. In 1992, Bartal *et al.* [10] gave an algorithm that is 1.986-competitive. Karger *et al.* [28] generalized the algorithm and proved an upper bound of 1.945. The best algorithm known so far achieves a competitive ratio of 1.923, see [1]. Gormley and Torng [24] showed that no deterministic online algorithm can be better than 1.853-competitive. Many problem variants of this basic scenario were studied where, e.g., jobs may be preempted, jobs may be rejected at a penalty, or online algorithms may use randomization. In addition, there are results for different machine types. There is also quite some work on online scheduling when job processing times are not known in advance, see e.g. [36, 40]. We refer the reader to [39] for a comprehensive survey on online scheduling.

In online *load balancing* we have again a set of m machines and a sequence of jobs that arrive online. However, each job has a *weight* and a *duration* that may or may not be known in advance. At any time the load of a machine is the sum of the weights of the jobs present on the machine at that time. The goal is to minimize the maximum load that occurs during the processing of the job sequence. Note that when all the jobs have an infinite duration, then the load balancing problem can be seen as a scheduling problem. In the following we concentrate on load balancing problems when jobs have unknown durations. For settings with m identical machines, Azar and Epstein [8] showed that the *Greedy* algorithm is $(2 - \frac{1}{m})$-competitive. Load balancing becomes more complicated with *restricted assignments*, i.e., each job can only be assigned to a subset of admissible machines. Azar *et al.* [6] proved that *Greedy* achieves a competitive ratio of $m^{2/3}(1 + o(1))$ and that no online algorithm can be better than $\Omega(\sqrt{m})$-competitive. In a subsequent paper, Azar *et al.* [9] gave a matching upper bound of $O(\sqrt{m})$. For load balancing on related machines, i.e. machines may have different speeds, Azar *et al.* [9] presented a 20-competitive algorithm and proved that no online algorithm can have a competitive ratio smaller than $3 - o(1)$. We refer the reader to [7] for a comprehensive survey on online load balancing.

Network routing problems. These problems have been considered in many different flavors and were studied starting from the *virtual circuit routing* problem. Consider a communication network where every link has a given maximum capacity. The input instance is formed by a sequence of communication requests. In response to each request, we must establish a virtual circuit on a path connecting two nodes of the network, at a given bandwidth, for a given duration. Aspnes *et al.* [2] gave an $O(\log n)$-competitive algorithm for the problem of minimizing the maximum load on a link of a network of n nodes, when connection requests have unlimited duration. The algorithm uses the idea of associating with every link a cost that is exponential in the fraction of the link capacity already assigned to ongoing circuits. The cost associated with a link of the network can be seen as the value assigned to a dual variable associated with the link itself in a linear programming formulation of the problem. The algorithm then routes every request on a minimum cost circuit.

The load balancing problem on networks is a generalization of the load balancing problem on unrelated parallel machines, from which a matching $\Omega(\log n)$ lower bounds is obtained [6]. The problem with limited duration has been considered in [9]. The virtual circuit routing problem has also been studied in its throughput version, the so called *call control* problem [22], where a benefit is associated with every request, requests can be accepted or discarded, while link capacities must not be exceeded. In [3] an algorithm with logarithmic competitive ratio is presented for maximizing the benefit obtained from accepted requests, under the assumption that the bandwidth request is at most a logarithmic fraction of any link capacity. For the case of bandwidth request exceeding a logarithmic fraction of the link capacity, the problem is basically reduced to the online version of the edge-disjoint path problem. In this case, deterministic algorithms fail to be competitive even for simple network topologies such as line or tree networks. A very high lower bound has been proved for general algorithms even if randomization is used [11]. However, randomized algorithms with competitive ratio $O(\log n)$ have been presented for a set of specific topologies like trees [4, 5], meshes [5, 30], and a class of planar graphs [30]. In [32] the problem of designing randomized algorithms combining a good competitive ratio and a good variance for the online edge-disjoint path problem has been considered.

A class of interesting routing problems also arises in wavelength division multiplexing (WDM) optical networks. In the basic case, every communication request must be assigned with a specific wavelength, obeying the constraint that communication requests assigned with same wavelength are routed on edge-disjoint paths. Competitive online algorithms for routing communications in optical networks have for instance been studied in [11, 12]. A large variety of other online network routing problems has been considered: calls can be preempted and/or rerouted at some later time, the benefit obtained from a call can be proportional to the amount of assigned resources, collective communication, e.g. multicast communication, has been addressed. For a comprehensive survey of the main techniques that are used and of the main results in the area we refer the reader to [15, 33].

Perspectives of online algorithms

Restricting an adversary: Competitive analysis is a strong worst-case performance measure. For some problems, such as paging, the competitive ratios of online algorithms are much higher than the corresponding performance ratios observed in practice. A line of research is concerned with evaluating online algorithms on restricted classes of request sequences. In other words, the power of an adversary is limited. In [17, 27], competitive paging algorithms with *access graph* are studied. Access graphs can model more realistic request sequences that exhibit locality of reference. It was shown that, using the access graph, it is possible to overcome some negative aspects of conventional competitive paging results [17, 20, 21, 27]. With respect to online financial games, Raghavan [38] introduced a *statistical adversary*: The input generated by the adversary must satisfy certain statistical assumptions. In [19], Chou *et al.* developed further results in this model. More generally, Koutsoupias and Papadimitriou [31] proposed the *diffuse adversary model*. An adversary must generate an input according to a probability distribution D that belongs to a class Δ of possible distributions known to the online algorithm.

Learning theory: Concepts from computational learning theory have been applied recently to the area of online algorithms. The problem of *predicting from expert advice* in its simple form considers an algorithm that has to predict repeatedly the value of a $\{0,1\}$ function, e.g. if it will rain or or not rain on a sequence of days. The algorithm receives as input the advice of n experts. After having formulated its prediction, the algorithm is told the real answer. The future advice of every expert will be biased by the correctness of the advice received in the past. Several results were presented showing competitiveness against an expert that made a fewest number of mistakes [18, 34]. Interesting relations with competitive analysis also arise from the work on *learning from examples* (see [14] for a survey on online learning algorithms).

Game and decision theory: Concepts in game theory also have been related recently to competitive analysis. Borodin and El-Yaniv [15] study the relationship between mixed as well as behavioral strategies and randomized online algorithms proving that mixed randomized memoryless online algorithms can achieve strictly better competitive ratios than behavioral randomized algorithms. Borodin and El-Yaniv also compare competitive analysis with other possible optimality criteria developed during a number of decades in decision theory. For a comprehensive discussion of these subjects we refer to [15].

Experiments: Recently work has started to test experimentally the ideas that were developed in the area of competitive online algorithms. Paging algorithms in the access graph model often use the idea of evicting the page that is furthest in the graph to the page currently accessed. Fiat and Rosèn [21] implemented an algorithm based on a refinement of this idea. The access graph is not even part of the input but is learned from the sequence itself. The algorithm has been shown to beat LRU on traces observed in practice. Algorithms for online virtual circuit routing have also been implemented [23] and favorably compared with popular greedy strategies.

References

[1] S. Albers. Better bounds for online scheduling. In *Proc. 29th Annual ACM Symp. on Theory of Computing*, 130-139, 1997.

[2] J. Aspnes, Y. Azar, A. Fiat, S. Plotkin and O. Waarts. On-line routing of virtual circuits with applications to load balancing and machine scheduling. *Journal of the ACM*, 44:486–504, 1997.

[3] B. Awerbuch, Y. Azar and S. Plotkin. Throughput-competitive online routing. In *34th IEEE Symp. on Foundations of Computer Science*, 32–40, 1993.

[4] B. Awerbuch, Y. Bartal, A. Fiat and A. Rosén. Competitive non-preemptive call control. In *Proc. of 5th ACM-SIAM Symp. on Discrete Algorithms*, 312-320, 1994.

[5] B. Awerbuch, R. Gawlick, T. Leighton and Y. Rabani. On-line admission control and circuit routing for high performance computing and communication. In *Proc. of the 35th Annual IEEE Symp. on Foundations of Computer Science*, 412–423, 1994.

[6] Y. Azar, A. Broder and A. Karlin. On-line load balancing. *Proc. 36th IEEE Symp. on Foundations of Computer Science*, 218–225, 1992.

[7] Y. Azar. On-line load balancing. In *Online Algorithms: The State of the Art*, edited by A. Fiat and G. Woeginger, Springer LNCS 1442, 178–195, 1998.

[8] Y. Azar and L. Epstein. On-line load balancing with of temporary tasks on identical machines. In *Proc. 5th Israeli Symp. on Theory of Computing and Systems*, 119–125, 1997.

[9] Y. Azar, B. Kalyanasundaram, S. Plotkin. K. Pruhs amd O. Waarts. Online load balancing of temporary tasks. In *Proc. of the 3rd Workshop on Algorithms and Data Structures*, Springer LNCS, 119-130, 1993.

[10] Y. Bartal, A. Fiat, H. Karloff and R. Vohra. New algorithms for an ancient scheduling problem. *Journal of Computer and System Sciences*, 51:359–366, 1995.

[11] Y. Bartal, A. Fiat and S. Leonardi. Lower bounds for on-line graph problems with application to on-line circuit and optical routing. In *Proc. of the 28th ACM Symp. on Theory of Computing*, 531–540, 1996.

[12] Y. Bartal and S. Leonardi. On-line routing in all-optical networks. In *Proc. of the 24th International Colloqium on Automata, Languages and Programming*, Springer LNCS, Volume 1256, 516-526, 1997.

[13] S. Ben-David, A. Borodin, R.M. Karp, G. Tardos and A. Wigderson. On the power of randomization in on-line algorithms. *Algorithmica*, 11:2-14,1994.

[14] A. Blum. On-line algorithms in machine learning. In *Online Algorithms: The State of the Art*, edited by A. Fiat and G. Woeginger, Springer LNCS, Volume 1442, 306–325, 1998.

[15] A. Borodin and Ran El-Yaniv. *Online computation and competitive analysis.* Cambridge University Press, 1998.

[16] A. Borodin, N. Linial and M. Saks. An optimal on-line algorithm for metrical task systems. *Journal of the ACM*, 39:745–763, 1992.

[17] A. Borodin, S. Irani, P. Raghavan and B. Schieber. Competitive paging with locality of reference. In *Proc. 23rd Annual ACM Symp. on Theory of Computing*, 249–259, 1991.

[18] N. Cesa-Bianchi, Y. Freund, D.P. Helbold, D. Haussler, R.E. Schapire and M.K. Warmuth. How to use expert advice. In *Proc. 25th Annual ACM Symp. on Theory of Computing*, 382–391, 1993.

[19] A. Chou, J. Cooperstock, R. El Yaniv, M. Klugerman and T. Leighton. The statistical adversary allows optimal money-making trading strategies. In *Proc. 6th Annual ACM-SIAM Symp. on Discrete Algorithms*, 467–476, 1995.

[20] A. Fiat and A.R. Karlin. Randomized and multipointer paging with locality of reference. In *Proc. 27th Annual ACM Symp. on Theory of Computing*, 626–634, 1995.

[21] A. Fiat and Z. Rosèn. Experimental studies of access graph based heuristics. In *Proc. 8th ACM-SIAM Symp. on Discrete Algorithms*, 1997.

[22] J. Garay, I.S. Gopal, S. Kutten, Y. Mansour and M. Yung. Efficient online call control algorithms. In *Proc. 2nd Israel Symp. on Theory of Computing and Systems*, 285–293, 1993.

[23] R. Gawlick, A. Kamath, S. Plotkin and K. Ramakrishnan. Routing and admission control of virtual circuits in general topology networks. Technical Report BL011212-940819-19TM AT&T Bell Laboratories, 1994.

[24] T. Gormley and E. Torng. Bounded online problems. Manuscript, 1998.

[25] R.L. Graham. Bounds for certain multiprocessor anomalies. *Bell System Technical Journal*, 45:1563–1581, 1966.

[26] D.S. Hochbaum (Editor). *Approximation Algorithms for NP-Hard Problems.* PWS Publishing Company, 1997.

[27] S. Irani, A.R. Karlin and S. Phillips. Strongly competitive algorithms for paging with locality of reference. In *Proc. 3rd Annual ACM-SIAM Symp. on Discrete Algorithms*, 228–236, 1992.

[28] D.R. Karger, S.J. Phillips and E. Torng. A better algorithm for an ancient scheduling problem. *Journal of Algorithms*, 20: 400–430, 1996.

[29] A. Karlin, M. Manasse, L. Rudolph and D.D. Sleator. Competitive snoopy caching, *Algorithmica*, 3:79–119, 1988.

[30] J. Kleinberg and È. Tardos. Disjoint paths in densely embedded graphs. In *Proc. of the 36th Annual IEEE Symp. on Foundations of Computer Science*, 52–61, 1995.

[31] E. Koutsoupias and C.H. Papadimitriou. Beyond competitive analysis. In *Proc. 34th Annual Symp. on Foundations of Computer Science*, 394–400, 1994.

[32] S. Leonardi, A. Marchetti-Spaccamela, A. Presciutti and A. Rosèn. On-line randomized call-control revisited. In *Proc. of the 9th ACM-SIAM Symp. on Discrete Algorithms*, 223-232, 1998.

[33] S. Leonardi. On-line network routing. In *Online Algorithms: The State of the Art*, edited by A. Fiat and G. Woeginger, Springer LNCS, Volume 1442, 242-267, 1998.

[34] N. Littlestone and M.K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108:212-261, 1994.

[35] M.S. Manasse, L.A. McGeoch and D.D. Sleator. Competitive algorithms for on-line problems. In *Proc. 20th Annual ACM Symp. on Theory of Computing*, 322-33, 1988.

[36] R. Motwani, S. Phillips and E. Torng. Non-clearvoyant scheduling. *Proc. 4th Annual ACM-SIAM Symp. on Discrete Algorithms*, 422–431, 1993.

[37] R. Motwani and P. Raghavan. *Randomized Algorithms.* Cambridge University Press, 1995.

[38] P. Raghavan. A statistical adversary for on-line algorithms. In *On-Line Algorithms*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 79–83, 1991.

[39] J. Sgall. On-line scheduling. In *Online Algorithms: The State of the Art*, edited by A. Fiat and G. Woeginger, Springer LNCS, Volume 1442, 198–231, 1998.

[40] D. Shmoys, J. Wein and D.P. Williamson. Scheduling parallel machines on-line. *SIAM Journal on Computing* 24:1313–1331, 1995.

[41] D.D. Sleator and R.E. Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28:202–208, 1985.