Page Migration in Dynamic Networks

Dissertation

by

Marcin Bienkowski

Fakultät für Elektrotechnik, Informatik und Mathematik
Institut für Informatik und Heinz Nixdorf Institut
Universität Paderborn

Paderborn, July 2005

Reviewers:

- Prof. Dr. Friedhelm Meyer auf der Heide, Universität Paderborn, Germany
- Prof. Dr. Burkhard Monien, Universität Paderborn, Germany
- Prof. Dr. Susanne Albers, Universität Freiburg, Germany

For Ewa and
little Joanna

Acknowledgements

I am deeply grateful to my advisor, Prof. Friedhelm Meyer auf der Heide for his continuous encouragement during all the stages of this thesis. It is not common for advisors to be that supportive, but it is a rarity to be simultaneously supportive and unobtrusive. While he showed me the possible paths, I never felt forced to follow any.

I am also truly indebted to the people with whom I collaborated, mainly the members of the working group "Algorithms and Complexity". As the life in graduate school does not consist of having brilliant ideas only, I am thankful to all of them who were willing to listen to all these unproven theories and false theorems that crossed my mind. While I profited from their scientific help and they were all fun to work with, their moral support, and daily conversations about life, universe and everything, made the life in the rainy town Paderborn as cheerful as it could possibly be. In alphabetic order they are André Brinkmann, Jarek Byrka, Valentina Damerow, Gereon Frahling, Mirek Korzeniowski, Jarek Kutyłowski, Peter Mahlmann, Harald Räcke, Kay Salzwedel, Christian Schindelhauer, Christian Sohler, and Martin Ziegler.

Additional credits I owe to Mirek Korzeniowski and Jarek Kutyłowski for reading parts of these thesis. Their comments helped me to improve the readability of this work.

Last but not least, I thank my wife, Ewa Dacko, for being here for all these years, proofreading all my papers, keeping the companion and cheering me up. Without her love nothing would have been possible.

Contents

CHAPTER 1

Introduction

In the last couple of decades, network connected systems have gradually replaced centralized parallel computing machines. Current computational challenges like protein folding, weather prediction, theorem proving, or even search of extraterrestrial intelligence in space require computing power, which neither can be delivered by a single mainframe, nor is easily affordable. On the other hand, vast computing resources are within reach by means of the Internet and other large, unstructured networks.

In contrast to traditional parallel computers, networks of workstations deliver computing power which is relatively cheap, but scattered and distributed. In consequence, writing a network application that runs in a distributed environment is substantially more difficult than writing an analogous program running on a single multi-processor machine. *Basic services* which are taken for granted in the latter case do not exist per se in a distributed network. One of the most crucial services used in every single parallel program is providing an application with a transparent access to variables, databases, memory pages, or files, which are shared by the instances of programs running at nodes of the network.

An implementation of the variable sharing is essential to the performance of a distributed application. However, the traditional approach of storing the shared data in one or a few central repositories does not scale up well with the increase of the network size and is therefore inherently inefficient. One of the most straightforward, yet imprecise solution, is to abandon these central storage systems and use local memories of the nodes to store the shared objects.

In this thesis we investigate data management strategies that try to exploit *topological locality*, i.e., try to migrate the shared data in the network in such a way that a node accessing a data item finds it "nearby" in the network. Accesses to the shared data can be modelled as an online problem. While we briefly discuss several such models, in this work we deal with the classical, most basic one, called *Page Migration*.

However, in contrast to previous works on data management in networks, we focus on the page migration in a *dynamic* setting. We assume that the network is no longer static, but is subject to change, and the costs of communication between nodes may change with time. Such a situation is typical in mobile ad-hoc networks, but occurs also in large distributed systems, which are used concurrently by many applications and users. Thus, we have to deal with two sources of online events, namely the requests from nodes to data items and the changes in the network. The new challenges, both for *modelling* and *algorithm design and analysis*, arising from the combination of data management with the network dynamics are the main topic of this thesis.

1.1 Static networks

In many applications, access patterns to a shared object change frequently. This is common, for example, in parallel pipelined data processing, where the set of processors accessing shared variables changes in the runtime. In these cases, any static placement of the object copies is inefficient. Moreover, the knowledge of the future accesses to the objects is in reality either partial or completely non-existing, which renders any solution based on static placement infeasible. Instead, a data management strategy should migrate the copies to further exploit the locality of accesses. This poses an algorithmic problem, central to this thesis.

> Without knowledge of the future accesses to the shared objects, decide, whether it is worth to change the positions of their copies.

To keep the bookkeeping overhead small, it is often required that only *one copy* of each object is stored in the system. Additionally, in a typical situation in the parallel environments, shared objects are usually bigger than the part of their data that is being accessed at one time. Usually, processors want to read or change only one single unit of data from the object, or one record from a database. On the other hand, the data of one object should be kept in one place to reduce the maintenance overhead. This leads to a so-called *non-uniform model*, where migrating or copying the whole object is much more expensive than accessing one unit of data from it.

This traditional paradigm, called *Page Migration* (PM) was introduced by Black and Sleator [BS89]. It models an underlying network as a connected, undirected graph, where each edge e has an associated cost $c(e)$ of sending one unit of data over the corresponding communication channel. In case of wired networks, this cost might represent the load induced by sending data through this communication link. The cost of sending one unit of data between two nodes v_a and v_b is defined as the sum of costs of edges on

the cheapest path between v_a and v_b. There is only *one copy* of *one single object* of size D, which is further called a *(memory) page*, stored initially at one fixed node in the network.

A PM problem instance is a sequence of nodes $(\sigma_t)_t$, which want to access (read or write) one unit of data from the page. In one step t, one node σ_t issues a request to the node holding the page and appropriate data is sent back. For such a request, an algorithm for PM is charged a cost of sending one unit of data between σ_t and the node holding the page. At the end of each time step the algorithm may move the page to an arbitrary node. Such a transaction incurs a cost which is D times greater than the cost of sending one unit of data between these two nodes.

The goal is to compute a schedule of page movements to minimize the total cost. Furthermore, computing an optimal schedule *offline*, i.e., on the basis of the *whole* input sequence $\mathcal{I} = (\sigma_t)_t$, is an easy task, which can be performed in polynomial time. Thus, the main effort was placed on constructing online algorithms, i.e., ones which have to make decision in time step t solely on the part of the input up to step t.

1.1.1 Competitive analysis

To measure the performance of online strategies for the PM problem, the competitive analysis (see, e.g., [ST85, BE98]) was used. This kind of evaluation, primarily introduced by Sleator and Tarjan [ST85], compares the cost of an online solution to the cost of the optimal offline strategy. In the following we assume that an optimal solution is denoted by Opt, and for any algorithm Alg, $C_{\text{ALG}}(\mathcal{I})$ denotes the cost of this algorithm on input sequence $\mathcal{I} = (\sigma_t)_t$.

An online deterministic algorithm Alg is $\mathcal{R}$-*competitive*, if there exists a constant A, such that for any input sequence $\mathcal{I}$, it holds that

$$C_{\text{ALG}}(\mathcal{I}) \;\leq\; \mathcal{R} \cdot C_{\text{OPT}}(\mathcal{I}) + A \;\; . \tag{1.1}$$

If $A = 0$, then we call Alg *strictly* $\mathcal{R}$-competitive. For a randomized algorithm Alg we replace its cost in the definition above by its expectation $\mathbf{E}[C_{\text{ALG}}(\mathcal{I})]$. The expected value is taken over all possible random choices made by Alg.

However, in the randomized case, the power given to the adversary has to be further specified. Following Ben-David et al. [BBK+90], we distinguish between three types of adversaries: *oblivious*, *adaptive-online* and *adaptive-offline*. An *oblivious* adversary has to construct the whole input sequence in advance, not taking into account the random bits used by an algorithm. The other two types are adaptive ones; they may decide about the next requests upon seeing the algorithm's current page position. Since they are dependent on the algorithm's random choices, we have to replace $C_{\text{OPT}}(\mathcal{I})$ by its expectation (taken over these random choices). These two adaptive types differ, however, in the way they construct an optimal solution, which is later compared with the solution

of ALG. An *adaptive-online* adversary must provide an *answering entity*, which creates an "optimal" solution parallelly to ALG. This solution may not be changed afterwards. An *adaptive-offline* adversary may construct an optimal solution at the end, knowing the whole input sequence.

The power of these adversaries can be related as shown in [BBK+90]. Let $\mathcal{R}_{\text{OBL}}$, $\mathcal{R}_{\text{AD–ONL}}$, $\mathcal{R}_{\text{AD–OFF}}$ are the best competitive ratios for randomized algorithms against oblivious, adaptive-online, and adaptive-offline adversaries, respectively. Let $\mathcal{R}_{\text{DET}}$ be the best possible ratio for deterministic algorithm. Then

$$\mathcal{R}_{\text{OBL}} \leq \mathcal{R}_{\text{AD–ONL}} \leq \mathcal{R}_{\text{AD–OFF}} = \mathcal{R}_{\text{DET}} \ . \tag{1.2}$$

This relation implies that the randomization does not help against the adaptive-offline adversary. Hence in this thesis, we focus on the other three types of adversaries.

While we give complete overview of the algorithms for Page Migration and related problems later, here we mention only that this area has been explored thoroughly by many researchers [BS89, Wes92, ABF93a, CLRW93, LRWY99, BCI01], and algorithms achieving constant competitive ratios were designed for all the types of adversaries presented above.

1.2 Dynamic networks

In the past, an application executed on a parallel machine was running in a virtually static and invariable environment and one could safely assume that the interconnecting network is predictable and reliable. Such assumptions, which substantially reduced the complexity of the basic services design, ceased to hold when applications started to run in open and unknown networks.

First of all, the network has begun to be prone to link failures or bandwidth shortages. Second, other applications running in the network might behave completely unpredictably or even antagonistically, creating high loads on particular links, e.g., by flooding them with messages. Third, if the network consists of mobile stations, its topology may be changed due to nodes mobility.

In our considerations we do not take into account the dynamics induced by nodes joining and leaving the network. In fact, a model where nodes may become active and inactive was already investigated by Awerbuch, Bartal, and Fiat [ABF98] in context of a data management subproblem, a file allocation.

Basic services for mobile wireless networks and dynamically changing wired networks are a relatively new area. The *topology control* (the problem of computing and maintaining a connected topology stretched on the network nodes) and *routing* in wireless mobile networks received attention in a past few years. See [Raj02] for a survey by Rajaraman.

Some effort was also placed on constructing routing algorithms for wired networks, where an adversarial entity not only injects packets to be routed but also may arbitrarily deactivate/activate any links. Surprisingly, for this very strong model, where the adversary is essentially able to destroy existing connections between each pair of nodes, Awerbuch, Brinkmann, and Scheideler [ABS03] were able to construct the robust routing (both path selection and packet switching) algorithm. The model assumed that dropping packets is admissible, and thus the algorithm was able to cope with the situations of routing packets with unreachable destinations. However, the fraction of delivered (not dropped) packets was provably close to the number of packets which an optimal algorithm could route. Related models of network dynamics are further discussed in a survey by Scheideler [Sch02].

In comparison, basic services related to data management problems in dynamically changing networks are still in their infancy. Till recently, no theoretical analysis or even experimental evaluation was present in this area, which might have been influenced by the fact that no reasonable model of network changes was proposed. In particular, any model similar to the one described in [ABS03], with possible adversarial link failures, would be too strong for any data management scheme. This follows from the observation that it is relatively easy to construct a sequence of accesses to a shared object, which eventually forces any competitive algorithm to move all the copies of this object to one node. Afterwards, the link failures may disconnect this node from the rest of the network, leaving the algorithm no chance to access or migrate the data now or in the future.

Hence, for theoretical modelling dynamics of networks, we assume that an adversary may modify the costs of point-to-point communication arbitrarily, as long as the pace of these changes is restricted by, say, an additive constant per step. Intuitively, this gives the data management algorithm time to react to the changes. Such a model can be motivated by a reality-close *pedestrian model* by Schindelhauer, Lukovszki, Rührup, and Volbert [SLRV03], where mobile stations might be moved arbitrarily by an adversary, as long their speed is bounded. The model of slow changes in the communication costs, formally defined in the next section, tries also to capture slow changes in bandwidth available in wired networks, which are inherently induced by other programs running or users using (not abusing) the network.

1.2.1 Our model

To model the Page Migration problem in dynamic networks we make the following assumptions. The network is modelled as a set of n mobile nodes (processors) labelled

$v_1, v_2, \ldots, v_n$. These nodes are placed in a metric space $(\mathcal{X}, d)$, where the distance between any pair of points from $\mathcal{X}$ is given by the metric d.

Time is discrete and slotted into time steps $t = 0, 1, 2, \ldots$. To model dynamics we assume that the position of each node is a function of t, i.e., $p_t(v)$ denotes the position of v in time step t. As a natural consequence, the distance between a pair of nodes may also change with time. The distance between any pair of nodes v_a and v_b in time step t is denoted by

$$d_t(v_a, v_b) := d(p_t(v_a), p_t(v_b)) \ . \tag{1.3}$$

Note that such a distance can be equal to zero in two different cases. The first one occurs, if v_a and v_b are different nodes occupying the same position in $\mathcal{X}$. The second one occurs when $a = b$, in which case we are dealing with a single node (and we write $v_a \equiv v_b$).

A tuple describing the positions of all the nodes in time step t is called *configuration* in step t, and is denoted by C_t.[1] A configuration sequence $(C_t)_{t=0}^{T}$ contains the configurations in the first $T + 1$ time steps, beginning with the *initial configuration* C_0.

The changes in nodes' positions over time are arbitrary, as long as the nodes move with a *bounded speed*, as mentioned in the previous section. Formally, for any node v_i, its positions in two consecutive time steps t and $t + 1$ cannot be too far apart, i.e.,

$$d(p_t(v_i), p_{t+1}(v_i)) \leq \delta \ , \tag{1.4}$$

for some fixed constant δ. An adversarial entity creating sequence of configurations is called δ-restricted, if it obeys the inequality above. Furthermore, if $\mathcal{X}$ is a bounded metric space, then let λ denote its *diameter*, i.e., the maximum possible distance between two points from $\mathcal{X}$. For an unbounded space, $\lambda = \infty$. We call λ the *(maximum) extent* of the network.

Any two nodes are able to communicate directly with each other. Essentially, the communication cost is proportional to the distance between these two nodes, plus a constant overhead. This overhead represents the startup cost for establishing connection. Precisely, the cost of sending a unit of data from node v_a to v_b at time step t is defined by a *cost function* $c_t(v_a, v_b)$, defined as

$$c_t(v_a, v_b) := d_t(v_a, v_b) + 1 \ , \tag{1.5}$$

if v_a and v_b are different nodes. The communication within one node is free, i.e., if $v_a \equiv v_b$, then $c_t(v_a, v_b) = 0$.

[1] The actual representation of these positions in complicated metric spaces is not relevant. The only requirement is that the distances between any pair of nodes in time t are computable on the basis of C_t.

Naturally, the changes in the network themselves (described by the $(C_t)_{t=0}^T$ sequence) do not constitute a problem of its own. According to the described Page Migration model, a copy of memory page of size D is stored at one of the network's nodes, initially at v_1. In each time step $t \geq 1$, exactly one node, denoted by σ_t, tries to access one unit of data from the page. Since the model assumes that there is only one copy of the object stored in the system, there is no need of making distinction between read and write accesses. We refer to them as *accesses* or *requests* and we call σ_t the *requesting node*. The requests create the sequence $(\sigma_t)_{t=1}^T$, complementary to the configuration sequence $(C_t)_{t=0}^T$.[2]

In each step an algorithm for the Page Migration in dynamic networks has to serve the request, and then to decide, whether it wants to migrate the page to some other node. Precisely, for any algorithm Alg the following stages happen in time step $t \geq 1$.

1. The positions of the nodes in the current step are defined by C_t.
2. A node σ_t wants to access one single unit of data from the page. It sends a write or a read request to $P_{\mathrm{ALG}}(t)$, the node holding Alg's page in the current step.
3. Alg serves this request, i.e., it sends a confirmation in case of write, or a requested unit of data in case of read. This transaction incurs a cost $c_t(P_{\mathrm{ALG}}(t), \sigma_t)$.
4. Alg optionally moves the page to another node of its choice, called a *jump candidate*. A movement to $P'_{\mathrm{ALG}}(t)$ incurs a cost $D \cdot c_t(P_{\mathrm{ALG}}(t), P'_{\mathrm{ALG}}(t))$.

In fact, the only part which Alg may influence is choosing a new node $P'_{\mathrm{ALG}}(t)$ in the fourth stage. The problem, to which we further refer as *Dynamic Page Migration* (DPM), is to construct a schedule of page movements to minimize the total cost of communication for any pair of sequences $(C_t)_t, (\sigma_t)_t$. We will usually abbreviate this notion to $(C_t, \sigma_t)_t$.

Before we proceed with the considerations on the complexity of the DPM problem, we point out that the DPM model is more general than Page Migration itself. If the network is static, i.e., $C_t = C_{t-1}$ for all $t \geq 1$, and we neglect the constant overhead in the cost function definition, then DPM is capable of modelling any situation, in which cost function satisfies the triangle inequality. Note that even if the underlying graphs are not metrical, the page migration algorithm chooses shortest paths instead of direct connections, and thus the triangle inequality is fulfilled.

It is also straightforward, that the constant overhead may be neglected, if the minimum cost of communication in the network is large. For the Page Migration problem in a static network we may assume this property, since, without loss of generality, the costs defined by any instance of the problem might be scaled up by any factor.

[2] Note that nodes issue requests from the first step. The initial configuration in time step 0 is introduced to simplify the notation only.

1.2.2 Our contribution

Like in the Page Migration case, the problem of minimizing the total cost incurred is relatively easy, if both $(C_t)_t$ and $(\sigma_t)_t$ are given in *offline* setting, i.e., if an algorithm may read the whole input beforehand. In fact, we present an easy dynamic programming approach, which is able to find an optimal schedule of page movements for any instance of the DPM problem consisting of T steps, using $O(T \cdot n^2)$ operations and $O(T \cdot n)$ additional space.

However, as mentioned earlier, DPM has to be primarily solved in an *online* setting, where an algorithm must make its decisions (where to move the page) in time step t, exclusively on the sequence $C_0, C_1, \sigma_1, C_2, \sigma_2, \ldots, C_t, \sigma_t$. To evaluate an online strategy for the DPM problem, we use competitive analysis. Since the input sequence consists of two practically independent streams, $(\sigma_t)_t$ describing the request patterns and $(C_t)_t$ reflecting the changes in network topology, it is reasonable to assume that they are created by two separate adversarial entities, a *request adversary* and a *configuration (network) adversary*.

In this thesis, we design algorithms for different powers of adversaries, and rigorously analyze them using competitive analysis and its variants. However, not only we have to precise the power of a single adversary, but also we have to decide whether these adversaries cooperate in creating an input sequence. This yields different scenarios depending on the ways in which these adversaries interact.

Adversarial (cooperative) scenario

The most straightforward modelling creates a task, which is most difficult to solve. It arises when both adversaries may cooperate to create the combined input sequence. In fact, this is equivalent to having one adversary capable of constructing the whole input sequence and brings the problem back to the classical formulation of online analysis.

We investigate this scenario in Chapter 2 and Chapter 3. We construct deterministic and randomized memoryless strategies, which are $O(\min\{n \cdot \sqrt{D}, D, \lambda\})$-competitive. As main non-trivial building blocks, they use $O(n \cdot \sqrt{D})$-competitive algorithms MARK and DIST, respectively. In the randomized case, the ratio is attained against an adaptive-online adversary. Recall that λ denotes the maximum distance that occurs between two nodes during runtime. Furthermore, these algorithms are up to a constant factor optimal, due to the presented matching lower bound for adaptive-online adversaries.

Further, we show how to randomize the deterministic algorithm MARK to get a competitive ratio of $O(\sqrt{D \cdot \log n}, D, \lambda\})$ against an oblivious adversary. This result is asymptotically optimal in the case of $D \geq \log^3 n$, due to the presented lower bound of $\Omega(\min\{\sqrt{D \cdot \log n}, D^{2/3}, \lambda\})$. All the presented competitive ratios are strict, which means that the constant A occurring in (1.1) is equal to zero.

The competitive ratios of the best possible algorithms for DPM problem are large, even against the weakest, oblivious adversaries. It can be inferred that the poor performance of algorithms for this scenario is caused by the fact that the network and request adversaries might combine and synchronize their efforts in order to destroy our algorithm. If cooperation between them was forbidden, then one might hope for a provably better performance. However, it is semantically not clear what non-cooperativeness means. Therefore we propose that the DPM problem could be analyzed in another extreme case, where one of the adversaries is replaced by a stochastic process. This leads to another two scenarios.

Brownian motion scenario

In this scenario, presented in Chapter 4, requests are given by the adversary, but the movement of nodes is random. Precisely, the mobile nodes perform a random walk on a bounded area of diameter B, and the request adversary dictates which nodes issue requests during runtime. By *area* we mean a d-dimensional discrete torus or mesh of diameter B, where d is a constant.

The request adversary is "oblivious", i.e., it has to create the whole request sequence $(\sigma_t)_t$ in advance, without knowledge of the actual configuration sequence $(C_t)_t$ induced by a random walk. The definition of competitiveness has to be adapted appropriately to reflect the fact that the input sequence is created both by an adversary and a stochastic process. Motivated by the research in the smoothed analysis of online algorithms, and the corresponding performance metric called *smoothed competitive ratio* [BLM+03, SS04], we introduce the following notions.

A deterministic algorithm Alg is $\mathcal{R}$-competitive with probability p, if there exists a constant A, such that for all request sequences $(\sigma_t)_t$ it holds that

$$\mathbf{Pr}_{(C_t)_t}\Big[\, C_{\mathrm{ALG}}((C_t,\sigma_t)_t) \le \mathcal{R}\cdot C_{\mathrm{OPT}}((C_t,\sigma_t)_t) + A \,\Big] \ge p \ , \tag{1.6}$$

where the probability is taken over all possible configuration sequences generated by the random movement.

A deterministic algorithm Alg is $\mathcal{R}$-competitive in expectation (achieves expected competitive ratio of $\mathcal{R}$), if there exists a constant A, such that for all request sequences $(\sigma_t)_t$ it holds that

$$\mathbf{E}_{(C_t)_t}\left[\, \frac{C_{\mathrm{ALG}}((C_t,\sigma_t)_t) - A}{C_{\mathrm{OPT}}((C_t,\sigma_t)_t)} \,\right] \le \mathcal{R} \ . \tag{1.7}$$

We emphasize that our notion of expected competitive ratio is usually stronger and gives more realistic estimates than the similar $\mathbf{E}[C_{\mathrm{ALG}}]/\mathbf{E}[C_{\mathrm{OPT}}]$ ratio introduced by Koutsoupias and Papadimitriou [KP00b].

We present an algorithm MAJ, which is $O(\min\{\sqrt[4]{D}, n\} \cdot \text{polylog}(B, D, n))$-competitive in this scenario. This result holds for 1-dimensional areas if $\sqrt[3]{D} \leq B \leq \sqrt{D}$, or for any constant-dimensional areas if $B \geq \sqrt{D}$. The ratio is achieved with both in expectation and with high probability. In this context high probability means that the probability p occurring in (1.6) can be amplified to $1 - (B \cdot D)^{-\gamma}$, by setting $A = \gamma \cdot A_0$ for a fixed constant A_0.

Stochastic requests scenario

This is a scenario, discussed in Chapter 5, which is symmetric to the Brownian motion one. It assumes that requests appear with some given frequencies, i.e., in step t, σ_t is a node chosen randomly according to a fixed probability distribution π. Analogously, a deterministic algorithm ALG is $\mathcal{R}$-competitive with probability p, if there exists a constant A, such that for all possible network topology changes (configuration sequences) $(C_t)_t$ and all possible probability distributions π, it holds that

$$\mathbf{Pr}_{(\sigma_t)_t}\Big[\, C_{\text{ALG}}((C_t, \sigma_t)_t) \leq \mathcal{R} \cdot C_{\text{OPT}}((C_t, \sigma_t)_t) + A \,\Big] \geq p \quad . \tag{1.8}$$

The probability is taken over all possible request sequences $(\sigma_t)_t$ generated according to π. We define the expected competitive ratio in the same manner as in the Brownian requests scenario.

We present an algorithm MOVE-TO-FIRST-REQUEST, achieving strict $O(1)$-competitive ratio, in expectation and with high probability. In this context, high probability means that for any γ, one can achieve probability $1 - D^{-\gamma}$, if the input sequence is sufficiently long. Moreover, the algorithm can be slightly modified to also handle the following cost function

$$c_t(v_a, v_b) = [d_t(v_a, v_b)]^{\alpha} + 1 \quad , \tag{1.9}$$

for any constant α, still remaining $O(1)$-competitive. For the case of wireless radio networks, one can choose the parameter α to respect a *propagation exponent* of the medium (see, e.g., [Rap96]). For example by setting $\alpha = 2$, the cost definition reflects the energy consumption used to send a message in the ideally free space along a given distance. Thus, this result minimizes, up to a constant factor, the total energy used in the system.

1.3 Related work

To our best knowledge, the only work that exists in the area of data management in dynamically changing networks is the paper by Awerbuch, Bartal, and Fiat on distributed paging [ABF98]. However, they consider a setting, in which nodes may appear and disappear, which is in fact orthogonal to our model. In particular, their results, which we present in one of the following sections, are inapplicable in our scenario.

On the other hand, the area of data management in static networks has been successfully explored in the past years by numerous researchers. Since we build the thesis on top of these algorithms, we briefly state some of their results below.

Page migration

The Page Migration problem was thoroughly investigated for different types of adversaries. While we shortly state the results below, for a gentle introduction to the algorithms mentioned here, we refer the reader to the survey by Bartal [Bar96a].

First randomized solutions presented by Westbrook [Wes92] were a memoryless algorithm which was 3-competitive against an adaptive-online adversary and a phase-based algorithm whose competitive ratio against an oblivious adversary tends to 2.618 as D goes to infinity. The former result was proven to be tight by Bartal, Fiat, and Rabani [BFR95, Bar96a]. The lower bound construction was a slight modification of the analogous lower bound for deterministic algorithms by Black and Sleator [BS89]. On the other hand, the exact competitive ratio against an oblivious adversary is not a completely settled issue. The currently best known lower bound, $2 + \frac{1}{2D}$, is due to Chrobak, Larmore, Reingold, and Westbrook [CLRW93]. It is matched only for certain topologies, like trees or uniform networks (see [CLRW93] and [LRWY99], respectively).

The first deterministic, phase-based, 7-competitive algorithm Move-To-Min was given by Awerbuch, Bartal, and Fiat [ABF93a]. The result was subsequently improved by the Move-To-Local-Min algorithm [BCI01] attaining competitive ratio of 4.086. On the other hand, [CLRW93] showed a network with a lower bound of approximately 3.148.

Data management

In this part we give a brief overview of extensions of Page Migration that allow more flexible data management in networks. One of the possible generalizations of PM is allowing more than one copy of an object to exist in the network. This poses new interesting algorithmic questions which have to be resolved by a *data management* scheme.

- How many copies of shared objects should be created?
- Which accesses to shared objects should be handled by which copies?

A basic version of this problem, where only one shared object is present in the system, called *file allocation*, was first examined in the framework of competitive analysis by Bartal, Fiat, and Rabani [BFR95]. They presented a randomized strategy that achieves an asymptotically optimal competitive ratio of $O(\log n)$ against an adaptive-online adversary, by a reduction to the online Steiner tree problem [IW91, AA92]. Additionally,

they showed how to get rid of the central control (which is useful for example for locating the nearest copy of the object) and create $O(\log^4 n)$-competitive algorithm, which works in a distributed fashion. Awerbuch, Bartal, and Fiat [ABF93a] showed that the randomization is not crucial by constructing deterministic algorithms (centralized and distributed ones) for file allocation problem attaining asymptotically the same ratios.

For uniform topologies, Bartal, Fiat, and Rabani [BFR95] showed an optimal deterministic 3-competitive algorithm. Lund, Reingold, Westbrook, and Yan [LRWY99] gave a 3-competitive algorithm for trees, which is based on *work functions* technique.

If the shared data is read-only, then the file allocation becomes a *page replication* problem. It was also introduced by Black and Sleator [BS89]. Unlike page migration, in general networks, this problem reduces to online Steiner tree and one cannot hope for a competitive ratio better than $\Omega(\log n)$. Therefore the research on page replication conducted by Albers and Koga [Kog93, AK98], and by Fleisher, Głazek, and Seiden [Gła99, FS00, Gła01, FGS04] concentrated on particular topologies like trees, uniform networks, and rings. For all these topologies $O(1)$-competitive deterministic and randomized algorithms were given; the ratios for trees and uniform networks are optimal.

Memory constraints

If multiple objects are present in the network and the local memory capacity at nodes is limited, then running a file allocation scheme independently for each single object in the network might encounter some problems. Above all, it is not possible to copy an object into a node's memory, if it is already full. Possibly, some other object copies have to be dropped, which induces problems, if they were the last copies present in the network. This leads to a so called *distributed paging* problem, where file allocation solutions have to be combined with schemes known from *uni-processor paging* (see for example [ST85, KMRS88, FKL+91, MS91, ACN00]).

For uniform networks, Bartal, Fiat, Rabani [BFR95] presented the deterministic $O(m)$-competitive Distributed-Flush-When-Full algorithm, where m denotes the total number of copies that can be stored within the network. They also proved that this bound is tight by showing a lower bound of $\Omega(m)$ for competitiveness against an adaptive-online adversary. Awerbuch, Bartal, and Fiat [ABF93b] used randomized uni-processor paging algorithms [FKL+91, MS91, ACN00] to get an up to a constant factor optimal algorithm HEAT & DUMP, which is $O(\max\{\log(m-f), \log k\})$-competitive against an oblivious adversary. In this context, f is the number of different objects in the network, and k is the maximum number of files that can be stored at any node. If we again restrict the number of object's copies to one, it results in a problem called *page migration with memory constraints*. Albers and Koga [AK95] presented deterministic and randomized algorithms

for this problem, which are much simpler than their distributed paging counterparts, and attain competitive ratios $O(n)$ and $O(\log n)$, respectively.

For general networks Awerbuch, Bartal, and Fiat [ABF98] adopted the model suggested primarily for uniprocessor paging [ST85], which goes beyond pure competitive analysis. In order to compensate the optimal offline algorithm advantage of knowing the future, Sleator and Tarjan [ST85] proposed limiting the memory capacity that the optimal algorithm has at its disposal. This extension, which is sometimes referred to as *resource augmentation*, allowed the authors of [ABF98] to present a deterministic $O(\text{polylog } n)$-competitive algorithm, under the assumption that the online algorithm has $O(\text{polylog } n)$ times more memory than the optimal algorithm.

For the case of page migration in general networks with memory constraints, Bartal [Bar96b, Bar98] gave an $O(\log m \cdot \log n \cdot \log \log n)$-competitive randomized algorithm. The construction uses the solution for a tree, and probabilistically approximates any graph metric by a tree metric.

Optimizing congestion

In case of wired networks the communication cost between a pair of nodes might be measured in terms of the load generated by sending the data through a communication link. All the algorithms presented above were designed to minimize the total communication load. A more challenging task it to derive fine-grained algorithms, whose objective is to minimize congestion, i.e., the maximum load on each single link.

Maggs, Meyer auf der Heide, Vöcking, and Westermann [MMVW97] developed a distributed data management strategy for tree networks, which was 3-competitive for the uniform model (the size of object equal to 1). The aforementioned 3-competitive algorithm for trees by Lund, Reingold, Westbrook, and Yan [LRWY99] was proven to be also competitive with respect to congestion minimization, and worked for the non-uniform model. However, as it was based on computing work-functions, it was inherently centralized. Meyer auf der Heide, Vöcking, and Westermann [MVW99] fixed this deficiency, presenting a deterministic 3-competitive distributed strategy for trees for the non-uniform model.

However, the main result of [MMVW97] was a *bisimulation technique*. It was shown that for some regular networks like meshes of clustered networks, the original problem instance can be, without enlarging congestion, mapped into a virtual network, a so called *access tree*. As mentioned above, solving the problem on a tree is relatively easy. Finally, the virtual tree was randomly mapped back into the original network, so that the congestion increases at most by a factor of $O(\log n)$, with high probability. This yields a randomized algorithm, which is $O(\log n)$-competitive against an oblivious adversary.

Similar results for fat trees and hypercubic networks, as well as $O(1)$-competitive algorithms for uniform networks, were presented in [MVW00, Wes00] and experimentally evaluated in [KMR+02]. Finally, Räcke [Räc02, Räc03] proved the existence of access trees for any network topology, showing an $O(\log^3 n)$-competitive construction. Bienkowski, Korzeniowski, and Räcke [BKR03] showed that these access trees are computable in polynomial time, losing an additional $O(\log n)$ factor in the ratio. The result of [Räc02] was independently improved to $O(\log^2 n \cdot \log\log n)$ (which was also a polynomial-time construction) by Harrelson, Hildrum, and Rao [HHR03].

Furthermore, Meyer auf der Heide, Vöcking, and Westermann [MVW00] and later Westermann [Wes00] showed how to extend these strategies to respect the capacity constraints on the local memory modules. Their algorithms also exploit the paradigm of resource augmentation, giving to the online algorithm $O(\log n)$ times more memory than to the offline strategy. The competitive ratios are asymptotically the same as in the case without memory capacity restrictions.

1.4 Bibliographical notes

Most of the results presented in this thesis have been published in a preliminary form previously. The DPM problem was defined in [BKM04], where the possible scenarios were discussed, and performance metrics were proposed.

The preliminary version of the randomized memoryless $O(\min\{n \cdot \sqrt{D}, D, \lambda\})$-competitive strategy, together with the matching lower bound, presented in Chapter 3, were constructed in [BKM04]. The deterministic algorithm MARK was given in [BDK05]. In the same paper a simple randomization R-MARK was presented. This randomization was subsequently improved to the $O(\sqrt{D \cdot \log n})$-competitive strategy EBM in [BB05].

The results for the Brownian motion scenario in Chapter 4 extend the paper [BK05], which is, in turn, a generalization of a preliminary result of [BKM04]. The results of the complementary, stochastic requests scenario presented in Chapter 5 were previously published in [Bie05].

Finally, the results presented in this thesis were briefly described in a joint survey on Page Migration in Dynamic Networks [BM05].

CHAPTER 2

Basics

In this chapter we analyze the adversarial model of DPM and present a polynomial-time optimal offline solution. As in this thesis we consider only constant-restricted network adversaries, we prove that all such adversaries are up to a constant factor equivalent.

Further, we turn our attention to basic properties of online algorithms for DPM. To give a gentle introduction to the online algorithms for general networks presented in the next chapter, we analyze networks consisting only of two nodes here. We give a randomized algorithm EDGE, which attains the competitive ratio $O(\sqrt{D})$ against an adaptive-online adversary in such networks. We also present an $\Omega(\min\{\sqrt{D}, \lambda\})$ lower bound for any algorithm against an oblivious adversary, where λ is the maximum extent of the network (i.e., the maximum possible distance between two nodes). By relation between the power of adaptive-online and oblivious adversaries (see Section 1.1.1), we infer that EDGE is up to a constant factor optimal in networks with extent $\lambda = \Omega(\sqrt{D})$.

Although the results for two-node networks presented in this chapter are redundant, since in the next chapter we give algorithms for the general case, they may give the reader a deeper insight into the problem, and present techniques we are exploiting in the subsequent sections. In particular, we use potential function analysis of online algorithms (see [Tar85, ST85]) to analyze EDGE. For proving a lower bound we employ the Yao's min-max rule [Yao77, NM44].

Finally, we present two simple deterministic algorithms for general networks, and a randomized memoryless one, achieving competitive ratios $O(D)$, $O(\lambda)$, and $O(\lambda)$, respectively. We use them as building blocks in the next chapter.

2.1 Optimal offline solution

We show how to compute the optimal offline solution for any DPM problem instance in polynomial number of operations.

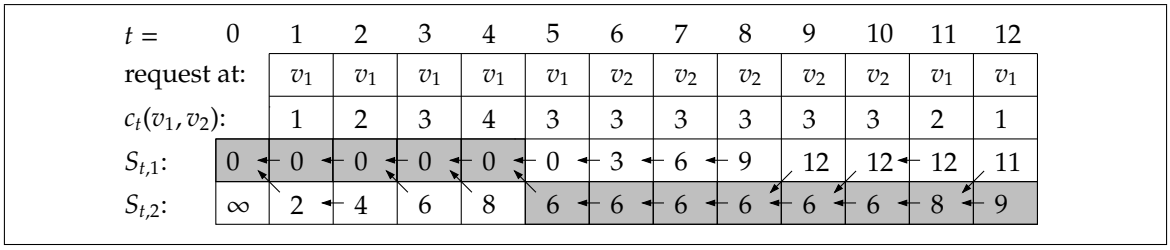

Figure 2.1: Example of constructing optimal schedule

Lemma 2.1. *For any T and for any input instance $(C_t, \sigma_t)_t$ of the DPM problem consisting of T steps, an optimal schedule of page movements can be computed using $O(T \cdot n^2)$ operations and $O(T \cdot n)$ additional memory.*

Proof. We compute the cost of the optimal solution by a straightforward dynamic programming. For any $0 \leq t \leq T$, $i \in [n]$ let $S_{t,i}$ be the cost of serving requests $\sigma_1, \ldots, \sigma_t$ and ending step t with the page at node v_i. Since at the beginning the page is at node v_1, we obtain boundary conditions: $S_{0,1} = 0$ and $S_{0,i} = \infty$ for any $i \in \{2, \ldots, n\}$. Using the recurrence relation

$$S_{t+1,i} = \min_{j \in [n]} \left\{ S_{t,j} + c_{t+1}(v_j, \sigma_{t+1}) + D \cdot c_{t+1}(v_j, v_i) \right\} \quad , \tag{2.1}$$

we can compute $S_{t+1,i}$ values from $S_{t,i}$ ones in time $O(n^2)$, using $O(n)$ additional memory. Thus, to fill the whole $S_{t,i}$ table we need $O(T \cdot n)$ space and $O(T \cdot n^2)$ time.

The cost of the optimal solution is given by $\min_{j \in [n]} S_{T,j}$. Assume that in addition to filling the $[S_{t,i}]$ table, for each entry $S_{t+1,i}$ we store also a pointer to the previous column

$$\textsc{Prev}_{t+1,i} = \arg\min_{j \in [n]} \left\{ S_{t,j} + c_{t+1}(v_j, \sigma_{t+1}) + D \cdot c_{t+1}(v_j, v_i) \right\} \quad , \tag{2.2}$$

with ties broken arbitrarily. We say that an entry $(t+1, i)$ points to $(t, \textsc{Prev}_{t+1,i})$. Then we can traverse the $[\textsc{Prev}_{t,i}]$ table, starting from the entry (T, k_T), where $k_T = \arg\min_{j \in [n]} \{S_{T,j}\}$, and in each step follow the pointers. This results in a sequence $(T, k_T), (T-1, k_{T-1}), \ldots, (1, k_1), (0, k_0)$, where $k_0 = 1$. The reversed part of this sequence, i.e., $k_0, k_1, \ldots, k_{T-1}$, denotes the indices of nodes holding the page in steps $1, 2, \ldots, T$, respectively, in an optimal solution. ■

If we are interested in the *cost* of an optimal schedule, and not in the schedule itself, it is sufficient to store only one column of the table $S_{t,i}$. This reduces the memory requirement to $O(n)$.

An example of the computed $S_{t,i}$ table, with $\textsc{Prev}_{t,i}$ pointers shown, is depicted in Figure 2.1. We considered a two-node network there, a page of size $D = 2$, and a sequence of 12 steps. The nodes issuing requests and the costs of communication between these two nodes are also depicted in this figure. We shaded an optimal schedule of page movements of cost 9 found by this approach.

2.2 Reduction Lemma

In this thesis we consider network adversaries which are δ-restricted, where δ is a constant. For convenience reasons, we usually choose $\delta = \frac{1}{2}$, since this assures that the distance between any two nodes can change only by 1 per time step. However, the presented theorems can be extended to any constant-restricted network adversary by means of the following Reduction Lemma.

Lemma 2.2 (Reduction Lemma). *Assume that there exists a k-competitive (possibly randomized) algorithm A against an a-restricted network adversary. Then A is k-competitive against a b-restricted network adversary for $b \leq a$. Additionally, for any $b \geq a$ there exists a (randomized) algorithm B, which is $\frac{b}{a} \cdot k$-competitive against any b-restricted network adversary.*

Proof. If $b \leq a$, then A is k-competitive against a b-restricted adversary, since it was k-competitive against a stronger a-restricted adversary.

If $b \geq a$, let $(C_t, \sigma_t)_t$ be any input sequence. B simulates the behavior of A on the sequence $(C'_t, \sigma_t)_t$, and repeats A's choices on $(C_t, \sigma_t)_t$. C'_t denotes the configuration C_t with all the original distances divided by b/a. Clearly, if (C_t) was created by a b-restricted adversary, then $(C'_t)_t$ sequence might be created by an a-restricted adversary. We have

$$\begin{aligned} C_B((C_t, \sigma_t)_t) &\leq \frac{b}{a} \cdot C_A((C'_t, \sigma_t)_t) \\ &\leq \frac{b}{a} \cdot k \cdot C_{\mathrm{OPT}}((C'_t, \sigma_t)_t) \\ &\leq \frac{b}{a} \cdot k \cdot C_{\mathrm{OPT}}((C_t, \sigma_t)_t) \ , \end{aligned}$$

and thus B is $\frac{b}{a} \cdot k$-competitive. For the proof for randomized algorithms we can replace the algorithm's cost by its expected value. ■

We note that the Reduction Lemma holds also if the request adversary is randomized, for example, in the stochastic requests scenario, presented in Chapter 5.

2.3 Two-node networks

In this section we present an algorithm and a lower bound for networks consisting of two nodes.

In the following we assume that the adversary is restricted to choosing the distance between v_1 and v_2 only from integers. Note that this simplification is not relevant, if we are interested in the asymptotic performance only, since we may consider an adversary which is not restricted in this manner, round the dictated distances up to the next integer, and thus increase the corresponding costs of communication at most twice. Besides, this simplification allows for much clearer presentation in this case.

Further, we assume that $D \geq 2$. If it is not the case, we may always use the $O(D)$-competitive algorithm JUMP presented in the Section 2.4.1 to get constant competitiveness.

2.3.1 Randomized algorithm EDGE

Let X_t denote the cost of sending one unit of data between our two nodes in time step t, i.e., $X_t = c_t(v_1, v_2)$. From the assumption above follows that X_t is an integer, greater than or equal to 1. One step of algorithm EDGE is described as follows. First, it serves the requests. Then, if the request was issued at the node opposite to the node holding the page of EDGE, the algorithm moves to that other node with probability $\mathcal{B}(X_t)$, where $\mathcal{B} : \mathbb{N}_+ \to \mathbb{R}$ is a function (called a *jump function*) defined as

$$\mathcal{B}(x) = 3 \cdot \frac{x}{k(x)} \ , \tag{2.3}$$

where

$$k(x) = \begin{cases} D - 1 + \sum_{i=1}^{x} i & \text{if } x \leq 2 \cdot D - 2 \ , \\ D \cdot x & \text{otherwise} \ . \end{cases} \tag{2.4}$$

The plot of $x/k(x)$ function is shown in Figure 2.2. In the remaining part of this section we prove the following theorem.

Theorem 2.3. EDGE *is strictly $O(\sqrt{D})$-competitive against an adaptive-online adversary in the adversarial scenario of the DPM.*

Potential functions and amortized analysis

We briefly present a very general potential functions technique (see for example [Tar85, ST85]), useful for proving competitiveness of online algorithms. See also [CLR97, chapter 18] for a detailed introduction to the amortized analysis method.

In our proofs, we follow the general scheme. We take any input sequence $\mathcal{I} = (C_t, \sigma_t)_t$ and we run our algorithm ALG together with an optimal algorithm OPT on the same input sequence. This is especially useful in analyzing algorithms playing against adaptive-online adversaries, since the optimal algorithm's moves are chosen then in each step by the adversary. In particular, in each step we know the current configuration of both ALG and OPT. In our case, the interesting part of algorithm configurations is the position of the memory page. We denote these positions by P_{ALG} and P_{OPT}, respectively.

We define a potential function Φ, which is a function of Alg and Opt configurations, e.g., a function of the distance between P_{ALG} and P_{OPT},[1] into the reals, having the following properties

1. At the beginning of input sequence, $\Phi = 0$.

2. For all configurations of Alg and Opt, which may occur, $\Phi \geq 0$.

We fix any time step t, and denote by $C_{\text{ALG}}(t)$ and $C_{\text{OPT}}(t)$ costs incurred on the algorithms Alg and Opt, respectively in time step t. By $\Delta\Phi(t)$ we denote the change in the potential Φ in this step t. If one can prove that for any step t

$$C_{\text{ALG}}(t) + \Delta\Phi(t) \;\leq\; \mathcal{R} \cdot C_{\text{OPT}}(t) \;, \tag{2.5}$$

then summing it over all time step of any input $\mathcal{I}$, and using two properties of the potential function, we get that the algorithm Alg is (strictly) $\mathcal{R}$-competitive. In the following, we usually omit t subscripts, when it does not lead to ambiguity.

For proving the competitiveness of randomized algorithms, we may replace the term $C_{\text{ALG}}(t) + \Delta\Phi(t)$ by its expected value $\mathbf{E}[C_{\text{ALG}}(t) + \Delta\Phi(t)]$.

Proof of competitiveness

Obviously, $x/k(x) \geq \frac{1}{D}$. Additionally, by $\mathcal{M}$ we denote the maximum value achieved by the fraction $x/k(x)$ (see Figure 2.2). We may prove the following upper bound on $\mathcal{M}$.

Lemma 2.4. *For $D \geq 2$ it holds that $\mathcal{M} \leq 1/\left(\sqrt{2\cdot(D-1)} + 1/2\right) = \mathcal{O}\left(1/\sqrt{D}\right)$.*

Proof. It is sufficient to show that $\frac{1}{\mathcal{M}}$ is at least $\sqrt{2\cdot(D-1)} + 1/2$.

$$\begin{aligned}
\frac{1}{\mathcal{M}} &= \min_{x\in\mathbb{N}_+}\left\{\frac{k(x)}{x}\right\} \\
&= \min\left\{\min_{\substack{x\in\mathbb{N}_+ \\ x\leq 2\cdot D-2}} \frac{D-1+\sum_{i=1}^{x} i}{x}, \min_{\substack{x\in\mathbb{N}_+ \\ x> 2\cdot D-2}} \frac{D\cdot x}{x}\right\} \\
&= \min\left\{\min_{\substack{x\in\mathbb{N}_+ \\ x\leq 2\cdot D-2}} \left\{\frac{D-1}{x} + \frac{x+1}{2}\right\},\; D\right\} \\
&\geq \min\left\{\min_{x\in\mathbb{R}_+} \left\{\frac{D-1}{x} + \frac{x+1}{2}\right\},\; D\right\}
\end{aligned}$$

[1] We sometimes abuse the notation, writing *distance between* Alg *and* Opt.

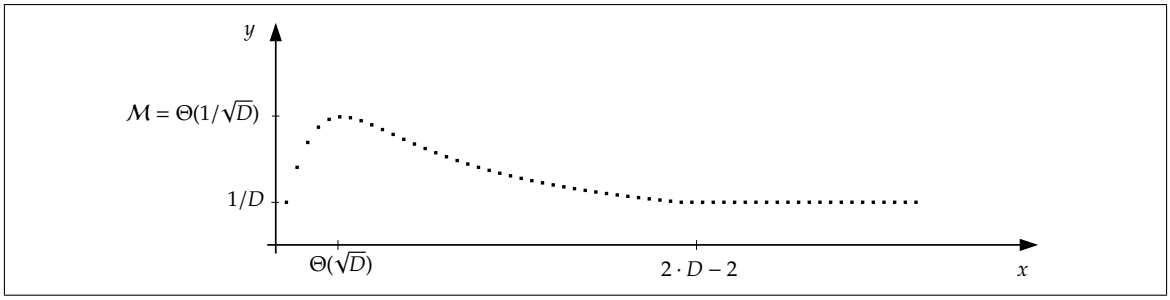

Figure 2.2: $y = x/k(x)$ function plot

We can analytically check that $\frac{D-1}{x} + \frac{x+1}{2}$ achieves its minimum for $x = \sqrt{2 \cdot (D-1)}$. Thus,

$$\begin{aligned} \frac{1}{\mathcal{M}} &\geq \min\left\{\sqrt{2 \cdot (D-1)} + \frac{1}{2},\ D\right\} \\ &\geq \sqrt{2 \cdot (D-1)} + \frac{1}{2}\ , \end{aligned}$$

where the last inequality follows from $D \geq 2$. ■

For the sake of the proof of competitiveness of EDGE, we conceptually divide each step t into two parts.

1. The adversary moves nodes to create configuration C_t. Both EDGE and OPT serve the request issued at σ_t. EDGE (optionally) moves the page.

2. OPT (optionally) moves the page.

Let L_c denote the cost of sending one unit of data between the nodes holding the pages of EDGE and OPT. We define the potential as

$$\Phi = \begin{cases} 6 \cdot \frac{\mathcal{M}}{\mathcal{B}(L_c)} \cdot D \cdot L_c & \text{if } L_c > 0\ , \\ 0 & \text{if } L_c = 0\ . \end{cases} \tag{2.6}$$

Clearly $\Phi = 0$ at the beginning of the input sequence, and is always non-negative. Thus, for proving the competitiveness of EDGE, it is sufficient to prove that for any time step t, for either of these two parts of the step, the amortized cost might be bounded using the optimal offline cost, i.e., $\mathbf{E}[C_{\text{EDGE}} + \Delta\Phi] \leq O(\sqrt{D}) \cdot C_{\text{OPT}}$.

Lemma 2.5. *For the first part of any step t, it holds that*

$$\mathbf{E}[C_{\text{EDGE}} + \Delta\Phi] \leq 10 \cdot \mathcal{M} \cdot D \cdot C_{\text{OPT}}\ .$$

Proof. Without loss of generality we can assume that EDGE is at v_1. We consider four cases depending on σ_t and the position of OPT's page.

1. Opt is at v_1, and $\sigma_t = v_1$. In this case the lemma follows trivially, since

$$C_{\text{EDGE}} = \Delta\Phi = C_{\text{OPT}} = 0 \ .$$

2. Opt is at v_1 and $\sigma_t = v_2$. Both Edge and Opt pay X_t for serving the request. Since they are both at the same node, the adversarial change in the network topology, i.e., the change in the distance between v_1 and v_2, does not influence the potential. Additionally, with probability $\mathcal{B}(X_t)$, Edge moves to v_2. Thus, the expected cost of C_{EDGE} is

$$\begin{aligned}\mathbf{E}[C_{\text{EDGE}}] &= X_t + \mathcal{B}(X_t) \cdot D \cdot X_t \\ &\leq (1 + 3 \cdot \mathcal{M} \cdot D) \cdot X_t \\ &\leq 4 \cdot \mathcal{M} \cdot D \cdot X_t \ .\end{aligned}$$

Initially, the potential is equal to 0, and it raises to $6 \cdot \frac{\mathcal{M}}{\mathcal{B}(X_t)} \cdot D \cdot X_t$, if Edge moves (with probability $\mathcal{B}(X_t)$). Thus,

$$\begin{aligned}\mathbf{E}[\Delta\Phi] &= \mathcal{B}(X_t) \cdot 6 \cdot \frac{\mathcal{M}}{\mathcal{B}(X_t)} \cdot D \cdot X_t \\ &= 6 \cdot \mathcal{M} \cdot D \cdot X_t \ .\end{aligned}$$

Combining the two inequalities above,

$$\mathbf{E}[C_{\text{EDGE}} + \Delta\Phi] \leq 10 \cdot \mathcal{M} \cdot D \cdot C_{\text{OPT}} \ .$$

3. Opt is at v_2 and $\sigma_t = v_1$. Opt pays X_t for serving the request, while Edge pays nothing and does not move. However, the potential may change due to the change in the distance between Edge and Opt.

$$\begin{aligned}\mathbf{E}[\Delta\Phi] &= 6 \cdot \frac{\mathcal{M}}{\mathcal{B}(X_t)} \cdot D \cdot X_t - 6 \cdot \frac{\mathcal{M}}{\mathcal{B}(X_{t-1})} \cdot D \cdot X_{t-1} \\ &\leq 6 \cdot \mathcal{M} \cdot D \cdot \frac{1}{3} \cdot (k(X_t) - k(X_{t-1})) \\ &\leq 2 \cdot \mathcal{M} \cdot D \cdot X_t \\ &\leq 2 \cdot \mathcal{M} \cdot D \cdot C_{\text{OPT}} \ .\end{aligned}$$

4. Opt is at v_2 and $\sigma_t = v_2$. This is the hardest case, since $C_{\text{OPT}} = 0$ and the cost of the algorithm, which is the same as in case 2, i.e.,

$$\mathbf{E}[C_{\text{EDGE}}] \leq 4 \cdot \mathcal{M} \cdot D \cdot X_t \ ,$$

has to be amortized by the negative change in the potential. The change in the potential is twofold. First, it may increase due to the adversarial change in the network as in case 3. Second, with probability $\mathcal{B}(X_t)$ it decreases from $6 \cdot \frac{\mathcal{M}}{\mathcal{B}(X_t)} \cdot D \cdot X_t$ to 0. In total, we have

$$\mathbf{E}[\Delta\Phi] \leq 2 \cdot \mathcal{M} \cdot D \cdot X_t - \mathcal{B}(X_t) \cdot 6 \cdot \frac{\mathcal{M}}{\mathcal{B}(X_t)} \cdot D \cdot X_t$$

Thus,

$$\begin{aligned} \mathbf{E}[C_{\text{EDGE}} + \Delta\Phi] &\leq 4 \cdot \mathcal{M} \cdot D \cdot X_t - 4 \cdot \mathcal{M} \cdot D \cdot X_t \\ &= 0 \\ &= C_{\text{OPT}} \end{aligned}$$

In all four cases, it holds that $\mathbf{E}[C_{\text{EDGE}} + \Delta\Phi] \leq 10 \cdot \mathcal{M} \cdot D \cdot C_{\text{OPT}}$. ■

Lemma 2.6. *For the second part of any step t it holds that*

$$\mathbf{E}[C_{\text{EDGE}} + \Delta\Phi] \leq 2 \cdot \mathcal{M} \cdot D \cdot C_{\text{OPT}}$$

Proof. If OPT does not move then the lemma follows trivially. Otherwise, OPT moves between v_1 and v_2 paying $D \cdot X_t$. Since EDGE does nothing, $C_{\text{EDGE}} = 0$. On the other hand, the potential at the end of the second part is maximized if OPT and EDGE end at different nodes, in which case

$$\begin{aligned} \Phi &= 6 \cdot \frac{\mathcal{M}}{\mathcal{B}(X_t)} \cdot D \cdot X_t \\ &\leq 2 \cdot \mathcal{M} \cdot D \cdot D \cdot X_t \ . \end{aligned}$$

Since Φ is non-negative at the beginning of the second part,

$$\begin{aligned} C_{\text{EDGE}} + \Delta\Phi &\leq 0 + 2 \cdot \mathcal{M} \cdot D^2 \cdot X_t \\ &\leq 2 \cdot \mathcal{M} \cdot D \cdot C_{\text{OPT}} \ . \end{aligned}$$

This finishes the proof of the lemma. ■

Proof of Theorem 2.3. By applying $\mathcal{M} = \Theta(1/\sqrt{D})$ to Lemma 2.5 and Lemma 2.6, we get that for any step t, $\mathbf{E}[C_{\text{EDGE}} + \Delta\Phi] \leq O(\sqrt{D}) \cdot C_{\text{OPT}}$. Since Φ is a non-negative potential function, the $O(\sqrt{D})$-competitiveness of EDGE follows. ■

2.3.2 Lower bound for oblivious adversary

In this subsection we prove a lower bound for any algorithm for DPM problem in a two-node network with extent λ. We prove that for the oblivious adversary, any randomized algorithm is at least $\Omega(\min\{\sqrt{D}, \lambda\})$-competitive.

Yao min-max principle

In proving lower bounds for randomized algorithms the Yao min-max principle [Yao77, NM44] turned out to be a very useful tool. Consider any minimization problem $\mathcal{P}$, and let π be any distribution over the set of possible inputs. In essence, the principle claims that the expected cost of *the best* deterministic algorithm (which knows π), is a lower bound on the (expected) cost of any randomized algorithm for $\mathcal{P}$. The expected cost of the deterministic algorithm is taken over all possible inputs chosen according to the probability distribution π. A detailed discussion on this principle is presented in [MR95, chapter 2].

This principle can be also applied for constructing lower bounds for an online randomized algorithm playing against oblivious adversaries. The general discussion and proofs can be found in [BE98, chapter 7 and 8]. However, for our needs a simpler formulation and a straightforward proof presented by Chrobak, Larmore, Lund, and Reingold [CLLR97] is sufficient.

Lemma 2.7 (Yao min-max principle [CLLR97]). *Consider any cost minimization problem. Suppose that for arbitrarily large γ there exists a probability distribution π over request sequences $\mathcal{I}$, such that*

(i) $\mathbf{E}_\pi[C_{\text{OPT}}(\mathcal{I})] \geq \gamma$, *and*

(ii) For any deterministic algorithm Det *it holds that* $\mathbf{E}_\pi[C_{\text{DET}}(\mathcal{I})] \geq \mathcal{R} \cdot \mathbf{E}_\pi[C_{\text{OPT}}(\mathcal{I})]$.

Then no randomized online algorithm is $\mathcal{R}'$-competitive (even against an oblivious adversary) for any $\mathcal{R}' < \mathcal{R}$.

Lower bound

Let $B_{\text{exp}} = \min\{\sqrt{D}, \lambda\}$. First, we construct a probability distribution π over inputs of arbitrary length and prove that for any deterministic algorithm Det, which knows this distribution, it holds that $\mathbf{E}_\pi[C_{\text{DET}}(\mathcal{I})] \geq \Omega(B_{\text{exp}}) \cdot \mathbf{E}_\pi[C_{\text{OPT}}(\mathcal{I})]$.

We divide time into phases, each of length $D + 2 \cdot B_{\text{exp}}$ steps. Each phase consists of an *expanding part*, which lasts for B_{exp} steps, a *main part* lasting for D steps, and a *contracting part* lasting also for B_{exp} steps. Let R_t denote the distance between v_1 and v_2 in time step t. Each phase begins with R_t set to 0 (v_1 and v_2 occupy the same point in the space). Then within the expanding part, nodes are moved apart, so that in the t-th step of the expanding part $R_t = t - 1$. Throughout the whole main part $R_t = B_{\text{exp}}$. Finally, in the contracting part the nodes are moved closer to each other, so that in the t-th step of the contracting part $R_t = B_{\text{exp}} - t$. Note that the movements of the nodes are fixed deterministically. The distances in one phase are depicted in Figure 2.3.

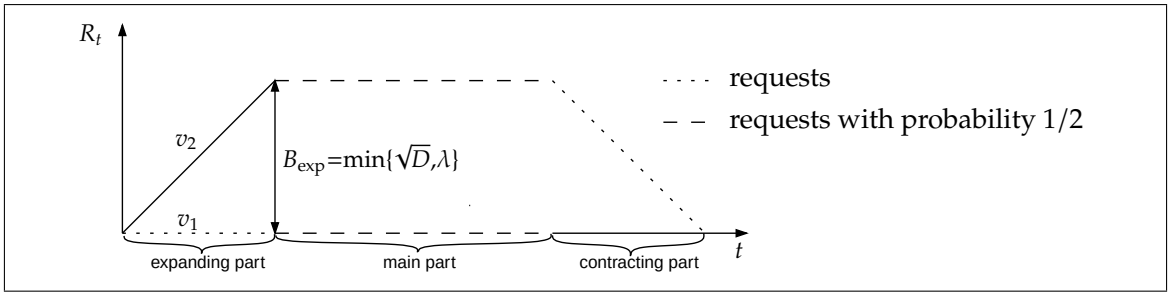

Figure 2.3: A lower bound for two-node network

Next, we have to explain where the requests are issued in each phase. In the expanding part all the requests are issued at v_1, and all the requests of the contracting part occur at v_2. Futher, in the main part, with probability 1/2, all the requests are issued at v_1, and, with probability 1/2, all the requests are issued at v_2.

Lemma 2.8. *Fix any T. Let $\mathcal{I}$ be a randomly generated input sequence consisting of T phases, where each phase is chosen as described above. Then for any deterministic algorithm* DET *for the DPM problem,*

$$\mathbf{E}_{\mathcal{I}}[C_{\mathrm{DET}}(\mathcal{I})] \geq \Omega(B_{\exp}) \cdot \mathbf{E}_{\mathcal{I}}[C_{\mathrm{OPT}}(\mathcal{I})] \ ,$$

where the expectation is taken over possible random choices of $\mathcal{I}$.

Proof. We concentrate on one single phase of $\mathcal{I}$. The optimal offline algorithm knows at which node the requests in the main part are issued. Assume that they are issued at v_2; if they are issued at v_1, the analysis is symmetric. Then OPT can move at the beginning of the phase to v_2 (if it is not already at v_2). Since at that time the nodes are in the same point of the space, the cost of this movement is D. Then OPT has to pay $\sum_{t=0}^{B_{\exp}-1}(t+1)$ for serving all the requests in the expanding part, paying nothing for the requests in the main or contracting part. Thus, the optimal cost for the whole input sequence $\mathcal{I}$ is at most

$$C_{\mathrm{OPT}}(\mathcal{I}) \ \leq \ T \cdot \left[D + \sum_{t=1}^{B_{\exp}} t \right] \ \leq \ 3 \cdot D \cdot T \ .$$

Note that we were able to bound $C_{\mathrm{OPT}}(\mathcal{I})$ for any randomly chosen $\mathcal{I}$, although a bound on $\mathbf{E}_{\mathcal{I}}[C_{\mathrm{OPT}}(\mathcal{I})]$ would be sufficient.

On the other hand, consider a deterministic online algorithm DET. It can only base its decisions on the past requests. In particular, in the last step of the expanding part it has to decide, whether to end this step at node v_1 or v_2. Independently of DET's choice, with probability 1/2, all the next D requests in the main part are given at the opposite node. In other words, at the beginning of the main part, DET is in a "wrong" node with probability 1/2. Assume that it is the case. If DET moves the page within the

main part, then it pays $D \cdot (B_{\text{exp}} + 1) > D \cdot B_{\text{exp}}$ for moving the page. Otherwise, it pays $D \cdot (B_{\text{exp}} + 1) > D \cdot B_{\text{exp}}$ for serving the requests during this part. Thus, the expected cost of Det in one phase is at least $\frac{1}{2} \cdot D \cdot B_{\text{exp}}$. Summing it over all phases, we get

$$\mathbf{E}_{\mathcal{I}}[C_{\text{DET}}(\mathcal{I})] \;\geq\; T \cdot \frac{1}{2} \cdot D \cdot B_{\text{exp}} \;\; .$$

Thus, the lemma follows. ■

Theorem 2.9. *Consider any randomized algorithm* Alg *which is c-competitive against an oblivious adversary for DPM problem in a two-node network. Then* $c = \Omega(\min\{\sqrt{D}, \lambda\})$*, where* λ *is the maximal extent of the network.*

Proof. First, note that in any phase Opt pays at least $\Omega(\min\{D, B_{\text{exp}}^2\})$. Indeed, if Opt moves within a phase, it pays D, otherwise it has to pay $\Omega(B_{\text{exp}}^2)$ either for serving all the requests in the expanding part or for all the requests in the contracting part. Thus, we can easily create a sequence with an arbitrarily high cost of the optimal schedule.

Hence, we may apply the Yao min-max principle to the result of the lemma above, which finally yields the theorem. ■

We note that the construction used the proofs above does not use non-integer distances R_t.

2.4 Trivial algorithms

In this section we present how to construct a simple deterministic algorithm achieving competitive ratio $O(D)$. We also show how to transform two algorithms for Page Migration, so that they work in dynamic networks, losing a factor of $O(\lambda)$ in the competitive ratio. What is more important, we can give explicitly their potential functions, and we may scale them up by any constant factor. This becomes important when we want to combine these algorithms with the algorithms presented in the next chapter.

2.4.1 Algorithm JUMP

Let Jump be a deterministic memoryless algorithm which upon receiving a request from the node σ_t serves this request and jumps to σ_t. We prove the following theorem

Theorem 2.10. Jump *is* $O(D)$*-competitive in the adversarial scenario of the DPM*

Proof. To maintain coherence with the notation used in the proof of competitiveness of the Edge, let L_c be the communication cost between P_{JUMP} and P_{OPT}. We define a potential

$$\Phi = 3 \cdot D \cdot L_c \;\; . \tag{2.7}$$

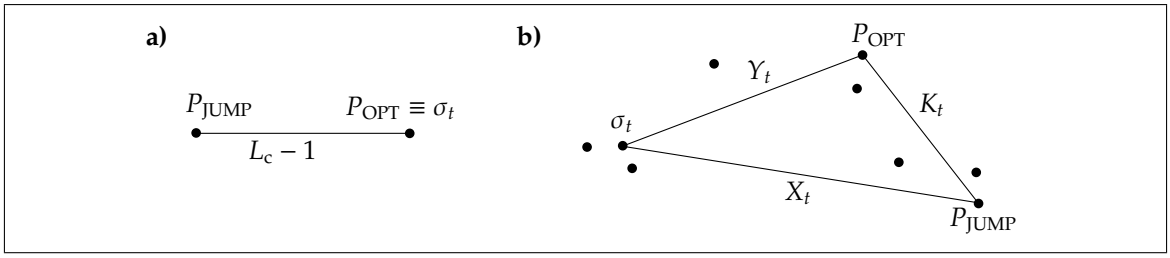

Figure 2.4: Illustration of the algorithm JUMP

It is sufficient to prove that in each step t, it holds that

$$C_{\text{JUMP}} + \Delta\Phi \leq O(D) \cdot P_{\text{OPT}} \ . \tag{2.8}$$

If, in step t, $P_{\text{JUMP}} \equiv P_{\text{OPT}} \equiv \sigma_t$ then (2.8) follows trivially. Otherwise, analogously to the proof of the competitiveness of EDGE, we divide each time step into two parts.

1. In the first part, the adversary changes the network to create configuration C_t out from C_{t-1}. This might increase L_c by at most 1, thus increasing the potential by D.

   If $P_{\text{OPT}} \equiv \sigma_t$, then JUMP pays L_c for the request and $D \cdot L_c$ for moving to σ_t. This is depicted in Figure 2.4a. Due to the movement, the potential decreases from $3 \cdot D \cdot L_c$ to 0. Thus, in total

   $$\begin{aligned} C_{\text{JUMP}} + \Delta\Phi &\leq L_c \cdot (1 + D) + D + (0 - 3 \cdot D \cdot L_c) \\ &\leq 0 \ . \end{aligned}$$

   It $P_{\text{OPT}} \not\equiv \sigma_t$ then we introduce some notation. We denote the distances in step t between P_{JUMP} and σ_t, between P_{OPT} and σ_t, and between P_{JUMP} and P_{OPT}, by X_t, Y_t, and K_t, respectively. The situation is depicted in Figure 2.4b. Again $C_{\text{JUMP}} = L_c \cdot (1 + D)$, where, by the definition of costs, $L_c \leq 1 + X_t$. As $P_{\text{OPT}} \not\equiv \sigma_t$, $C_{\text{OPT}} = 1 + Y_t$. The change in the potential due to the JUMP's movement from P_{JUMP} to σ_t is at most $3 \cdot D \cdot (1 + Y_t) - 3 \cdot D \cdot K_t$. Thus, summing up

   $$\begin{aligned} C_{\text{ALG}} + \Delta\Phi &\leq (1 + X_t) \cdot (1 + D) \ + \ D \ + \ 3 \cdot D \cdot (1 + Y_t) - 3 \cdot D \cdot K_t \\ &\leq 2 \cdot D \cdot (1 + X_t) + D - 3 \cdot D \cdot K_t + 3 \cdot D \cdot C_{\text{OPT}} \\ &\leq 3 \cdot D + (2 \cdot D \cdot X_t - 3 \cdot D \cdot K_t) + 3 \cdot D \cdot C_{\text{OPT}} \ . \end{aligned}$$

   Finally, we use the the triangle inequality for the term in brackets

   $$\begin{aligned} C_{\text{ALG}} + \Delta\Phi &\leq 3 \cdot D + 2 \cdot D \cdot Y_t + 3 \cdot D \cdot C_{\text{OPT}} \\ &\leq 6 \cdot D \cdot C_{\text{OPT}} \ . \end{aligned}$$

2. In the second part of the step t, the optimal algorithm potentially moves to a new destination along a distance of Y'_t. Obviously, $C_{\text{JUMP}} = 0$ in this case, and again by the triangle inequality follows that the distance between C_{JUMP} and C_{OPT} increases by at most Y'_t. Thus, the increase in the potential is bounded by

$$\Delta\Phi \;\leq\; 3 \cdot D \cdot (Y'_t + 1) \;=\; 3 \cdot C_{\text{OPT}} \;.$$

Hence, in both parts Inequality 2.8 holds, and thus the theorem follows. ■

2.4.2 Reusing Page Migration algorithms

It is relatively easy to show that if the maximum extent of the network is λ, then using any $O(1)$-competitive algorithm for Page Migration as a black box, we may construct $O(\lambda)$-competitive algorithms for the DPM problem. However, to combine these $O(\lambda)$-competitive algorithms with the algorithms given in the next chapter, we will need explicitly the potential functions used in their proofs. Moreover, these potential functions must be sufficiently large. Thus, we present a general simple scheme of transforming some Page Migration algorithms into DPM ones.

Lemma 2.11. *Let* Alg *be a strictly c-competitive algorithm for Page Migration in uniform networks. Then there exists an algorithm* Alg$'$ *for DPM, which is at most strictly $((\lambda + 1) \cdot c)$-competitive in the adversarial scenario.*

Proof. Let $\mathcal{I} = (C_t, \sigma_t)_t$ be any input for DPM problem. Alg$'$ simulates Alg on the uniform graph with edges of length 1 on the sequence $(\sigma_t)_t$, and repeats its movements on $\mathcal{I}$. Since the communication costs during $\mathcal{I}$ are at most $\lambda+1$, the cost increases at most by factor $\lambda + 1$. Let Opt be an optimal solution for Page Migration on $(\sigma_t)_t$. Obviously, it constitutes a lower bound for an optimal solution Opt$'$ for instance $\mathcal{I}$ of DPM problem, since all communication costs (between two different nodes) during $\mathcal{I}$ are at least 1. Combining these observations, we conclude

$$\begin{aligned} C_{\text{ALG}'}(\mathcal{I}) &\leq (\lambda + 1) \cdot C_{\text{ALG}}((\sigma_t)_t) \\ &\leq (\lambda + 1) \cdot c \cdot C_{\text{OPT}}((\sigma_t)_t) \\ &\leq (\lambda + 1) \cdot c \cdot C_{\text{OPT}'}(\mathcal{I}) \;. \end{aligned}$$

■

Step potential proofs

We would like to have algorithms, for which we may construct potential-based proofs with potential functions explicitly given. We introduce a notion of a *step potential proof*. Let Alg be any online algorithm, and B be any algorithm (possibly the optimal offline one) for Page Migration problem. Let P_B denote the node holding the page of B.

Additionally, let m denote the memory state of the algorithm ALG. In particular, it contains P_{ALG}, and if, for example, ALG uses counters, then m contains a tuple consisting of their current values. Let Φ be any non-negative function of P_{B} and m, which for $P_{\text{ALG}} \equiv P_{\text{B}}$ and initial state of m is equal to 0.

We call a memory state m *legal* if there exists a sequence $(\sigma_t)_t$, after serving which ALG is (in case of randomized algorithms, is with non-zero probability) in state m. Then we say that an algorithm ALG has a step potential proof of c-competitiveness, if there exists a potential function Φ, such that for any P_{B}, any legal m, and for any step request σ_t in any step t, it holds that

$$\mathbf{E}[C_{\text{ALG}}(t) + \Delta\Phi(t)] \leq c \cdot C_{\text{B}}(t) \tag{2.9}$$

If we sum Inequality 2.9 over all steps, we get that the cost of algorithm ALG is at most c times larger than the cost of any algorithm, in particular the optimal offline one. Thus, if an algorithm has a step potential proof of c-competitiveness, it is c-competitive.

Note that in (2.9) we compare algorithm ALG to *any* other algorithm, not necessarily the optimal one. Therefore, this definition might potentially exclude algorithms, which rely on some properties of the optimal solutions. However, this is not the case for the Page Migration algorithms we want to reuse. In particular, we have two facts.

Fact 2.12. *Strictly 3-competitive deterministic algorithm* M [BS89] *for the Page Migration problem in uniform networks has a step potential proof.*

Fact 2.13. *Strictly 3-competitive randomized memoryless algorithm* COIN-FLIPPING [Wes92] *for the Page Migration problem in uniform networks has a step potential proof. The algorithm is competitive against an adaptive-online adversary.*

Transformation from PM to DPM

By Lemma 2.11, the two facts above imply the existence of $3 \cdot (\lambda + 1)$-competitive algorithms for DPM problem. However, we could get a stronger result.

Lemma 2.14. *Let* ALG *be a strictly c-competitive algorithm with step potential proof for the Page Migration problem in uniform networks, and let Φ be the potential used. Fix any number $k \geq 1$. Then there exists an algorithm* ALG′ *for the DPM problem which is strictly $(k \cdot (\lambda + 1) \cdot c)$-competitive in the adversarial scenario, and also has a step potential proof. Moreover, the potential used in the latter proof is equal to $\Phi' = k \cdot (\lambda + 1) \cdot \Phi$.*

Proof. Let $\mathcal{I} = (C_t, \sigma_t)_t$ be any input for DPM problem. As in the previous construction, ALG′ simulates ALG on the uniform graph with edges of length 1 on the sequence $(\sigma_t)_t$, and repeats its movements on $\mathcal{I}$. Let $\Phi' = k \cdot (\lambda + 1) \cdot \Phi$. Let B' be any solution for $\mathcal{I}$,

and B be the same schedule of page movements, but on the uniform graph with edges of length 1. For any step t it holds that

$$\begin{aligned}\mathbf{E}[\textsc{Alg}'(t) + \Delta\Phi'(t)] &\leq k \cdot (\lambda + 1) \cdot \mathbf{E}[\textsc{Alg}(t) + \Delta\Phi(t)] \\ &\leq k \cdot (\lambda + 1) \cdot c \cdot B(t) \\ &\leq k \cdot (\lambda + 1) \cdot c \cdot B'(t) \ .\end{aligned}$$

Note that the last inequality would not necessarily hold, if we took a more straightforward approach and chose B' as an optimal algorithm for $\mathcal{I}$ and B as an optimal algorithm for sequence $(\sigma_t)_t$. ■

When we combine Lemma 2.14 with Fact 2.12 and Fact 2.13, and we look at the potential functions used in the proofs of algorithm M and Coin-Flipping, we immediately get the corollary below.

Corollary 2.15. *There exist a deterministic algorithm* Dyn-M *and a randomized memoryless algorithm* Dyn-CF *(competitive against an adaptive-online adversary), such that for any constant k, there exists a proof of their $O(\lambda)$-competitiveness, which uses a potential function Φ with the following properties. Assume that the algorithm is compared with an algorithm B, and they both start from the same node. Then*

(i) if the algorithm has the page in the node different than P_B, then $\Phi \geq k \cdot (\lambda + 1) \cdot D$,

(ii) if the algorithm has the page in P_B, then $\Phi \geq 0$,

(iii) at the beginning of input sequence, $\Phi = 0$.

Thus, we may place any multiplicative constants in the potential function without changing the asymptotic performance of the algorithms Dyn-M and Dyn-CF.

CHAPTER 3

Adversarial Scenario

In this chapter we thoroughly analyze the case where both configuration and request sequence are generated by an adversary.

First, we show how to generalize algorithm EDGE to n nodes. A new randomized, memoryless algorithm DISTRIBUTE (DIST) uses asymptotically the same jump function. However, due to the adversarial changes in the network, we cannot restrict potential page holders to the nodes which issued requests. Thus, DIST tries to place the page not exactly at the accessing node, but in its close neighborhood. Unfortunately, since such a neighborhood may contain all the nodes, the competitive ratio increases by n to $O(n \cdot \sqrt{D})$ (against an adaptive-online adversary). We define the paradigm of "catching OPT" and show how DIST follows its guidelines. Finally, we combine DIST with the $O(\lambda)$-competitive algorithm DYN-CF and the $O(D)$-competitive algorithm JUMP, proving an upper bound of $O(\min\{n \cdot \sqrt{D}, D, \lambda\})$ on the competitive ratio.

Second, we show how to get rid of the randomization and replace it by counters and marks. We construct a whole class of marking algorithms; a deterministic member of this class is a $O(n \cdot \sqrt{D})$-competitive algorithm MARK. Again it is possible to combine it with DYN-M and JUMP getting the deterministic upper bound of $O(\min\{n \cdot \sqrt{D}, D, \lambda\})$. We prove that these results are up to a constant factor optimal, by presenting a matching lower bound for any randomized algorithm against an adaptive-online adversary.

Thus, the only chance to beat this lower bound in the adversarial scenario of the DPM problem, is to consider oblivious adversaries. We show two randomizations of MARK: R-MARK and EBM, which still use marking information, but reduce the competitive ratio to $O(\sqrt{D} \cdot \log n)$ and to $O(\sqrt{D \cdot \log n})$, respectively. By combining the latter algorithm with the algorithms DYN-M and JUMP, we show that the competitive ratio in the oblivious case is $O(\min\{\sqrt{D \cdot \log n}, D, \lambda\})$. Finally, using Yao min-max principle, we present an almost matching lower bound of $\Omega(\sqrt{D \cdot \log n}, D^{2/3}, \lambda\})$.

We note that all competitive ratios are strict, i.e., we do not need an additive constant

A in the definition of the competitive ratio (1.1). We conclude with some open questions and conjectures.

General remarks

In this chapter we consider only $\frac{1}{2}$-restricted adversaries. By the Reduction Lemma (see Section 2.2, page 17) all constant-restricted adversaries are equivalent (up to a constant factor in competitive ratios) and our results hold asymptotically for any constant-restricted adversary.

By a *subsequence* we understand any contiguous time interval of the input sequence. For simplifying the notation, we also treat subsequences as sets of the corresponding time steps. For any subsequence $\mathcal{S}$ and any algorithm ALG, by $C_{\text{ALG}}(\mathcal{S})$ we denote the cost (of serving requests within $\mathcal{S}$ and moving the page) incurred by $\mathcal{S}$ on ALG. In particular, by OPT we denote the optimal offline algorithm, and by $C_{\text{OPT}}(\mathcal{S})$ its cost in $\mathcal{S}$.

3.1 Randomization against adaptive adversary

In this section we present a randomized online algorithm DIST, which achieves a competitive ratio $O(n \cdot \sqrt{D})$ against an adaptive-online adversary. We show that we can combine DIST with JUMP and DYN-CF to achieve the optimal competitive ratio $O(\min\{n \cdot \sqrt{D}, D, \lambda\})$ against an adaptive-online adversary. Although in the next section we prove asymptotically the same upper bound for the stronger, deterministic case, the algorithm presented in this section has an advantage of being memoryless, i.e., it does not have any state information. The ratio is provably asymptotically optimal, as we present an up to a constant factor matching lower bound in Section 3.3.1.

Last request based algorithms

Before we construct our algorithms for the general case, we argue, why the simple extension of the EDGE algorithm does not work. Consider a class of randomized algorithms which may move only to the node which issued a request in the current step. This class we call *last request based*. All previous randomized algorithms for Page Migration (presented for example in [Wes92, CLRW93, LRWY99]) as well as our JUMP algorithm fall into this category. We show that such an algorithm has no chance of being better than $\Omega(D)$-competitive against an adaptive-online adversary in networks with extent $\lambda = \infty$.

Consider a three-node network. ALG is any (possibly randomized) last request based algorithm. Without loss of generality, we may assume that ALG starts in v_1. The adversary chooses to keep OPT's all the time in v_3. Initially, all nodes occupy the same

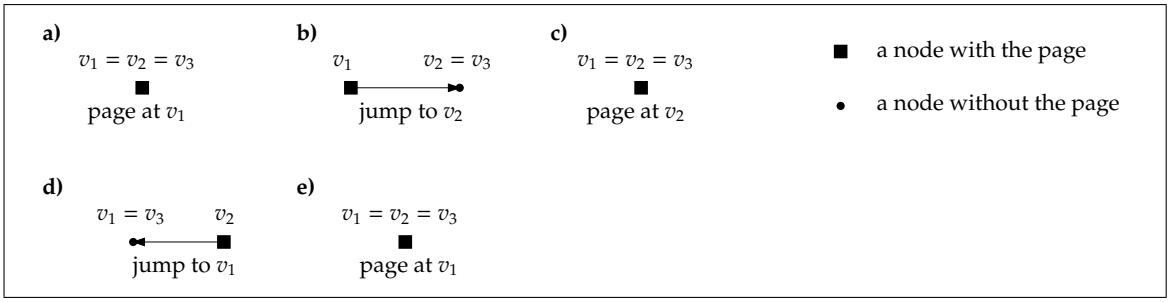

Figure 3.1: A lower bound for last request based algorithms

place in the space, which is depicted in Figure 3.1a. Assume also that the adversary is 1-restricted.

We divide the time into phases. In the first phase, requests are given at v_2, and v_1 is moved apart with the maximum possible speed, i.e., in step t of this phase, the distance between v_1 and v_2 (or v_3) is $t - 1$. At some point of the time Alg decides to move[1] (see Figure 3.1b). As a jump candidate it can only choose v_2.

Let X denote the distance between v_1 and v_2 at the moment of jump. In the next $X + 1$ steps requests are given at v_2, and nodes are contracted, i.e., v_1 is moved to v_2 and v_3 with the maximum possible speed, till it reaches the initial configuration (Figure 3.1c). After contracting, the phase ends, having lasted $2 \cdot (X + 1)$ steps. In this phase Opt pays $2 \cdot (X + 1)$ and Alg pays $D \cdot (X + 1) + \sum_{t=1}^{X}(t + 1) \geq D \cdot (X + 1)$.

We can continue this process for any number of phases. The even phases are symmetric to the first one, i.e., the roles of v_1 and v_2 are reversed (see Figure 3.1d and 3.1e). The odd phases follow the same rules as the first one. Thus, we conclude that Alg is at least $D/2 = \Omega(D)$-competitive.

Catching OPT

Hence, we take a step back and try to understand the reason why a last-request-based algorithm failed. If we run Opt and Alg, in parallel, on some input sequence, we could see the whole process as a "chasing game". Assume that at some time step Alg moves to the node which stores also Opt's page. We say that Alg has *caught* Opt. As long as Opt and Alg remain at one node, they pay the same for serving requests, which is obviously advantageous for our algorithm. Thus, in order to destroy our algorithm, the adversary has to separate its page from the algorithm's page, which leaves him, essentially, two options. The first one is to move to another node, paying at least D. The second one is to force Alg to move, by issuing requests at a different node than the one holding Opt and Alg page. In the latter case, though, Alg could easily incur a certain cost on Opt

[1] If Alg never moves, then trivially its competitive ratio is unbounded.

just by waiting before jumping to another node. Specifically, ALG may either wait till the cost of serving requests from the current place reaches some given threshold, or move to another node with a certain small probability. In either case, OPT has to pay (at least in expectation) a certain amount before the pages are separated.

Therefore, if only ALG was able to catch OPT, the cost of OPT could be raised. Note that the argument above does not work against an adaptive-offline adversary, since in that case the position of the optimal algorithm can be chosen afterwards (offline). Moreover, unlike Page Migration problem, it is not sufficient to have the page close to OPT's page, but an algorithm has to catch OPT frequently. The problem with the last request based approach is that it is possible (and it was the case in our example above) that OPT is never caught by the algorithm.

Hence, in order to construct an algorithm with a non-trivial competitive ratio, we have to employ a slightly different approach. We consider moving not only to a node which has just issued a request, but also to its nearest surroundings. In this way we are trying to stay close to the place where most of the requests are, but we increase our chances of catching OPT.

3.1.1 Algorithm DIST

The algorithm DIST (and its proof of competitiveness) is a generalization of the algorithm EDGE defined in Section 2.3.1, and is formally depicted in Figure 3.2. We explain the behavior of the algorithm in one fixed time step t. Let $X_t := d_t(P_{\text{DIST}}, \sigma_t)$ denote the distance between P_{DIST}, the node holding the DIST's page, and the request. If $P_{\text{DIST}} \equiv \sigma_t$, then the algorithm does nothing. Otherwise, DIST serves this request, and it moves to another node with probability $\mathcal{B}(X_t)$, where $\mathcal{B}(\cdot)$ is a *jump function* defined in Figure 3.2. $\mathcal{B}$ depends on a constant b, which will be defined later. We note that $\mathcal{B}$ is a continuous function and it differs only by a constant factor from the jump function of the algorithm EDGE. Naturally, we have to specify where DIST moves to. It chooses its destination (further called *a jump candidate*) uniformly at random from a so called *Jump Set*.

Definition 3.1. *For any time step t, let* Jump Set, *denoted by $J(t)$, be the set of all nodes which are at a distance of at most $\sqrt{D}$ from σ_t, i.e., $J(t) = \{v \in V : d_t(v, \sigma_t) \leq \sqrt{D}\}$.*

In other words, the algorithm DIST moves to any of the nodes from $J(t)$ with probability $\mathcal{B}(X_t)/|J(t)|$. Note that if $P_{\text{DIST}} \in J(t)$, then P_{DIST} may also be chosen as a jump candidate.

Amortized analysis

In this and the next two sections we prove the following theorem.

Let $\mathcal{B}(x) := \begin{cases} b \cdot \frac{1}{D} & \text{if } 0 \leq x < 1 \\ b \cdot \frac{x}{D} & \text{if } 1 \leq x < \sqrt{D} \\ b \cdot \frac{1}{x} & \text{if } \sqrt{D} \leq x \leq D \\ b \cdot \frac{1}{D} & \text{if } x > D \end{cases}$

Serve the request
Let $X_t = d_t(P_{\text{DIST}}, \sigma_t)$
if $P_{\text{DIST}} \not\equiv \sigma_t$ **then**
  With probability $1 - \mathcal{B}(X_t)$ **do** not move
  With probability $\mathcal{B}(X_t)$ **do**
    Choose uniformly at random one node P'_{DIST} from the set $J(t)$
      (each node with probability $\frac{1}{|J(t)|}$)
    Move the page to P'_{DIST}

Figure 3.2: Algorithm Dist in step t

Theorem 3.2. *There exists a constant b in the definition of the jump function $\mathcal{B}$, such that the algorithm* Dist *achieves a competitive ratio of $O(n \cdot \sqrt{D})$ against an adaptive-online adversary in the adversarial scenario of the DPM.*

In the following we set $b := 8$. To assure that the probability given by the jump function $\mathcal{B}(x)$ is valid, i.e., not greater than 1, we assume that $b \leq \sqrt{D}$. If it is not the case, then $D \leq 64$ and we may use the Jump algorithm to easily get a constant competitiveness.

In the analysis of the algorithm Dist we are not aiming at minimizing the constants but rather at the simplicity of the proof. Since we concentrate on a single step t only, for clarity we omit t subscripts in P_{ALG}, C_{ALG}, etc. Similarly as for the algorithm Edge, we concentrate on the analysis of the amortized cost and prove that such a cost in one step is bounded. However, now we define two potential functions: Φ, which measures the distance between Opt and Dist, and Θ, which puts emphasis on the fact that catching Opt is a desired event. Formally, in any moment, let L denote the distance between P_{OPT} and P_{DIST}. Then,

$$\Phi = f_1 \cdot \mathcal{F}(L), \quad \text{where} \quad \mathcal{F}(L) = L \cdot \sqrt{D} \cdot \min\left\{L, \sqrt{D}\right\} , \tag{3.1}$$

and

$$\Theta = \begin{cases} f_2 \cdot n \cdot D \cdot \sqrt{D} & \text{if } P_{\text{DIST}} \not\equiv P_{\text{OPT}} , \\ 0 & \text{if } P_{\text{DIST}} \equiv P_{\text{OPT}} , \end{cases} \tag{3.2}$$

where $f_1 = 32$ and $f_2 = f_1 + 22$ are constants. Clearly, Φ and Θ are initially 0 and are always non-negative. Thus, to prove Theorem 3.2 it is sufficient to prove that for any time step, it holds that

$$\mathbf{E}[C_{\text{ALG}}] + \mathbf{E}[\Delta\Phi] + \mathbf{E}[\Delta\Theta] \leq O\left(n \cdot \sqrt{D}\right) \cdot C_{\text{OPT}} \ , \tag{3.3}$$

where the expected value is taken over the random choices of the algorithm.

As in the proof of competitiveness of the algorithm EDGE (see Section 2.3.1), we conceptually divide each step t into two parts.

1. The adversary moves nodes to create configuration C_t. Both DIST and OPT serve the request issued at σ_t. DIST (optionally) moves the page.

2. OPT (optionally) moves the page.

In the next two sections, for each part separately, we prove that (3.3) holds.

Proof of Theorem 3.2. The competitiveness of DIST immediately follows by summing Inequality 3.3 over all time steps of the input, and by non-negativity of the potential functions Φ and Θ. Since at the beginning of the input $\Phi = \Theta = 0$, the achieved competitiveness is strict. ■

3.1.2 DIST in the first part of a step

In the first part of any step t the adversary does not move its page. If $P_{\text{DIST}} \equiv \sigma_t$, then (3.3) is trivially fulfilled,

$$C_{\text{DIST}} + \Delta\Phi + \Delta\Theta = 0 \leq C_{\text{OPT}} \ .$$

Hence, in the remaining part we assume that $P_{\text{DIST}} \not\equiv \sigma_t$. Consider the following thought experiment. Assume that DIST decides to move. Then instead of moving directly to a randomly chosen jump candidate $P'_{\text{DIST}} \in J(t)$, it moves first to σ_t, and then to P'_{DIST}. By the triangle inequality the cost of DIST may only increase through such an experiment. This experiment divides the first part of the step into two stages: a *jumping stage*, which contain actions of the algorithm including the movement to σ_t, and a *spreading stage*, which includes the jump from σ_t to P'_{DIST}. Thus, we may divide the cost of DIST as follows

1. Let $C^{\text{A}}_{\text{DIST}}$ be the cost of serving requests and moving to σ_t (with probability $\mathcal{B}(X_t)$).

2. Let $C^{\text{B}}_{\text{DIST}}$ be the cost of moving (only if we have just moved to σ_t) to a randomly chosen jump candidate P'_{DIST}.

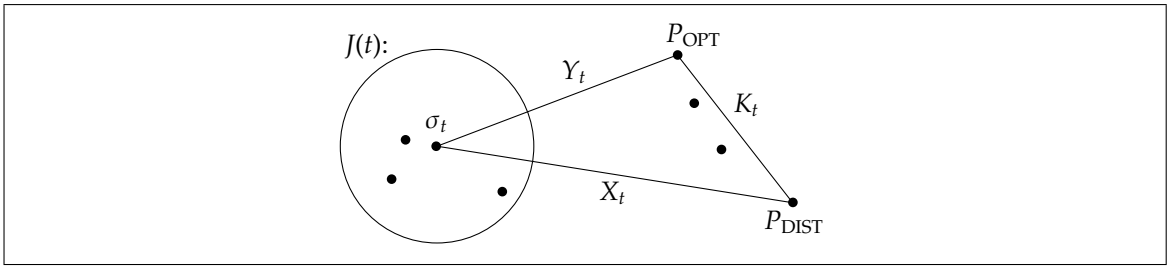

Figure 3.3: Illustration of the algorithm DIST

As mentioned above, $C_{DIST} \leq C^A_{DIST} + C^B_{DIST}$, and thus $\mathbf{E}[C_{DIST}] \leq \mathbf{E}[C^A_{DIST}] + \mathbf{E}[C^B_{DIST}]$. Analogously, we define the change in the potential induced by these two movements by $\Delta\Phi^A$ and $\Delta\Phi^B$; the actual expected change in the potential Φ is bounded by $\mathbf{E}[\Delta\Phi] \leq \mathbf{E}[\Delta\Phi^A] + \mathbf{E}[\Delta\Phi^B]$.

Let C_{OPT} be the cost of the optimal algorithm in the first part of the step. By half-amortized cost of some action of DIST we understand the cost of the algorithm plus the corresponding change in the potential Φ (not counting the change in the potential Θ). For showing that (3.3) holds in the first part of the step, we prove two lemmas. The first one bounds the half-amortized cost in the jumping stage, and the second one in the spreading stage. They both utilize the triangle inequality to show that these costs are both smaller than $O(\sqrt{D}) \cdot C_{OPT}$ plus some additive term of order $\mathcal{B}(X_t) \cdot D \cdot \sqrt{D}$.

Finally, we combine these two lemmas with the change in the potential Θ. We show that either C_{OPT} was large in the first part of the step, i.e., of order of the additive term $\mathcal{B}(X_t) \cdot D \cdot \sqrt{D}$, or DIST had a great chance of catching OPT, lowering in this way its potential Θ. In the latter case we show that the negative change in this potential majorizes the additive term above.

Half-amortized cost in the jumping stage

Lemma 3.3. *In the first part of any time step t, it holds that*

$$\mathbf{E}[C^A_{DIST}] + \mathbf{E}[\Delta\Phi^A] \leq O\left(\sqrt{D}\right) \cdot C_{OPT} + 21 \cdot \mathcal{B}(X_t) \cdot D \cdot \sqrt{D} \ .$$

Proof. Let Y_t and K_t be the distances between P_{OPT} and σ_t, and P_{DIST} and P_{OPT}, respectively (see Figure 3.3). Additionally K_{t-1} denotes the distance between P_{DIST} and P_{OPT} at the beginning of the time step, before the adversary moves the nodes, or equivalently, in the previous time step. We begin with computing the expected values of C^A_{DIST} and $\Delta\Phi^A$.

$$\mathbf{E}[C^A_{DIST}] \ = (X_t + 1) + \mathcal{B}(X_t) \cdot D \cdot (X_t + 1) \ .$$

Using $\mathcal{B}(X_t) \geq \frac{b}{D} \geq \frac{1}{D}$, and $\sqrt{D} \geq 2$, we get

$$\begin{aligned}\mathbf{E}[C^{\mathrm{A}}_{\mathrm{DIST}}] &\leq 2 \cdot \mathcal{B}(X_t) \cdot D \cdot (X_t + 1) \\ &\leq \underbrace{2 \cdot B(X_t) \cdot D \cdot X_t}_{=:C_{\mathrm{A}}} + \mathcal{B}(X_t) \cdot D \cdot \sqrt{D} \ .\end{aligned}$$

The first inequality above follows from $\mathcal{B}(X_t) \geq \frac{b}{D} \geq \frac{1}{D}$, and in the second one we use $\sqrt{D} \geq 2$. Later, we concentrate on restricting only the C_{A} part of the cost.

The change in the potential in the jumping stage is twofold as in case 4 in the proof of competitiveness of EDGE (see Lemma 2.5 on page 20). First, Φ may change due to the adversarial changes in the network. Second, with probability $\mathcal{B}(X_t)$, Φ changes because DIST moves its page. In total,

$$\mathbf{E}[\Delta\Phi^{\mathrm{A}}] \ = \ f_1 \cdot (\mathcal{F}(K_t) - \mathcal{F}(K_{t-1})) \ + \ f_1 \cdot \mathcal{B}(X_t) \cdot (\mathcal{F}(Y_t) - \mathcal{F}(K_t)) \ .$$

To upper-bound the first summand we may assume that $K_t \geq K_{t-1}$. The first derivative of $\mathcal{F}$ is equal to

$$\mathcal{F}'(L) \stackrel{\mathrm{def}}{=} \frac{\mathrm{d}\mathcal{F}(L)}{\mathrm{d}L} = \begin{cases} 2 \cdot \sqrt{D} \cdot L & \text{for } L < \sqrt{D} \ , \\ D & \text{for } L > \sqrt{D} \ . \end{cases}$$

For any L, $\mathcal{F}(L) \leq 2 \cdot D$, and thus $\mathcal{F}(K_t) - \mathcal{F}(K_{t-1}) \leq 2 \cdot D$. Additionally, for $K_t \leq \sqrt{D}$ and $L \in [K_{t-1}, K_t]$ holds $\mathcal{F}'(L) \leq 2 \cdot \sqrt{D} \cdot K_t$, and thus $\mathcal{F}(K_t) - \mathcal{F}(K_{t-1}) \leq 2 \cdot \sqrt{D} \cdot K_t$. We may conclude that

$$\mathbf{E}[\Delta\Phi^{\mathrm{A}}] \ \leq \ 2 \cdot f_1 \cdot \min\left\{K_t \cdot \sqrt{D},\, D\right\} \ + \ f_1 \cdot \mathcal{B}(X_t) \cdot (\mathcal{F}(Y_t) - \mathcal{F}(K_t)) \ . \tag{3.4}$$

We consider three cases. In each case we bound $C_{\mathrm{A}} + \mathbf{E}[\Delta\Phi^{\mathrm{A}}]$ either by $20 \cdot B(X_t) \cdot D \cdot \sqrt{D}$, or by $O(\sqrt{D}) \cdot C_{\mathrm{OPT}}$.

1. $Y_t \geq X_t/4$.

   This is the most straightforward case, since Y_t (and thus the cost paid by OPT) is large compared to X_t. Additionally, it follows from the triangle inequality that $K_t \leq Y_t + X_t \leq 5 \cdot Y_t$. Using $\mathcal{F}(x) \leq D \cdot x$ and $\mathcal{B}(X_t) \leq b/\sqrt{D}$, we obtain

   $$\begin{aligned} C_{\mathrm{A}} + \mathbf{E}[\Delta\Phi^{\mathrm{A}}] \ &\leq \ 2 \cdot \mathcal{B}(X_t) \cdot D \cdot X_t + 2 \cdot f_1 \cdot \sqrt{D} \cdot K_t + f_1 \cdot \mathcal{B}(X_t) \cdot \mathcal{F}(Y_t) \\ &\leq \ 8 \cdot \frac{b}{\sqrt{D}} \cdot D \cdot Y_t + 10 \cdot f_1 \cdot \sqrt{D} \cdot Y_t + f_1 \cdot \frac{b}{\sqrt{D}} \cdot D \cdot Y_t \\ &= \ O\left(\sqrt{D}\right) \cdot Y_t \\ &\leq \ O\left(\sqrt{D}\right) \cdot C_{\mathrm{OPT}} \end{aligned}$$

2. $Y_t < X_t/4$ and $X_t \leq \sqrt{D}$

   In this case the distances from P_{DIST} and P_{OPT} to the request σ_t are small, i.e., smaller than $\sqrt{D}$. Additionally, $K_t \leq X_t + Y_t \leq \frac{5}{4} \cdot X_t \leq \frac{5}{4} \cdot \sqrt{D}$. We use $B(X_t) \geq \frac{b}{D}$ to bound the half-amortized cost by term $O(1) \cdot \mathcal{B}(X_t) \cdot D \cdot \sqrt{D}$.

$$\begin{aligned} C_{\text{A}} + \mathbf{E}[\Delta\Phi^{\text{A}}] &\leq 2 \cdot \mathcal{B}(X_t) \cdot D \cdot X_t + 2 \cdot f_1 \cdot \sqrt{D} \cdot K_t + f_1 \cdot \mathcal{B}(X_t) \cdot \mathcal{F}(Y_t) \\ &\leq 2 \cdot \mathcal{B}(X_t) \cdot D \cdot \sqrt{D} + \frac{5}{2} \cdot f_1 \cdot \sqrt{D} \cdot X_t + f_1 \cdot \mathcal{B}(X_t) \cdot D \cdot Y_t \\ &\leq \left(2 + \frac{5}{2} \cdot f_1 \cdot \frac{1}{b} + \frac{1}{4} \cdot f_1\right) \cdot \mathcal{B}(X_t) \cdot D \cdot \sqrt{D} \\ &\leq 20 \cdot \mathcal{B}(X_t) \cdot D \cdot \sqrt{D} \end{aligned}$$

3. $Y_t < X_t/4$ and $X_t \geq \sqrt{D}$.

   This is the most complex case, as we have to amortize the cost of our algorithm by the change in the potential induced by a jump of DIST. Luckily, since $Y_t \leq \frac{1}{4} \cdot X_t$, and therefore $K_t \geq X_t - Y_t \geq \frac{3}{4} \cdot X_t$, the term $\mathcal{F}(Y_t) - \mathcal{F}(K_t)$ occurring in (3.4) is negative and equal to $-\Theta(D \cdot X_t)$, as follows from the two inequalities below.

$$\begin{aligned} \mathcal{F}(K_t) &\geq \mathcal{F}((3/4) \cdot X_t) \\ &= (3/4) \cdot X_t \cdot \sqrt{D} \cdot \min\left\{(3/4) \cdot X_t, \sqrt{D}\right\} \\ &\geq (3/4) \cdot X_t \cdot \sqrt{D} \cdot (3/4) \cdot \sqrt{D} \ , \end{aligned}$$

   and

$$\begin{aligned} \mathcal{F}(Y_t) &\leq \mathcal{F}((1/4) \cdot X_t) \\ &= (1/4) \cdot X_t \cdot \sqrt{D} \cdot \min\left\{(1/4) \cdot X_t, \sqrt{D}\right\} \\ &\leq (1/4) \cdot X_t \cdot \sqrt{D} \cdot \sqrt{D} \ . \end{aligned}$$

   Thus, $\mathcal{F}(Y_t) - \mathcal{F}(K_t) \leq -\frac{5}{16} \cdot D \cdot X_t$. Additionally, for $X_t \geq \sqrt{D}$ it holds that $\frac{1}{b} \cdot \mathcal{B}(X_t) \cdot X_t \geq 1$. Thus, we get

$$\begin{aligned} C_{\text{A}} + \mathbf{E}[\Delta\Phi^{\text{A}}] &\leq 2 \cdot \mathcal{B}(X_t) \cdot D \cdot X_t + 2 \cdot f_1 \cdot D + f_1 \cdot \mathcal{B}(X_t) \cdot (\mathcal{F}(Y_t) - \mathcal{F}(K_t)) \\ &\leq \left(2 + 2 \cdot f_1 \cdot \frac{1}{b} - \frac{5}{16} \cdot f_1\right) \cdot \mathcal{B}(X_t) \cdot D \cdot X_t \\ &= 0 \ . \end{aligned}$$

In either case we have

$$\begin{aligned}\mathbf{E}[C^{\mathrm{A}}_{\mathrm{DIST}} + \Delta\Phi^{\mathrm{A}}] &\leq C_{\mathrm{A}} + \mathcal{B}(X_t) \cdot D \cdot \sqrt{D} + \mathbf{E}[\Delta\Phi^{\mathrm{A}}] \\ &\leq O\left(\sqrt{D}\right) \cdot C_{\mathrm{OPT}} + 21 \cdot \mathcal{B}(X_t) \cdot D \cdot X_t \ .\end{aligned}$$

This finishes the proof of the lemma. ■

Half-amortized cost in the spreading stage

Lemma 3.4. *In the first part of any time step t, it holds that*

$$\mathbf{E}[C^{\mathrm{B}}_{\mathrm{DIST}}] + \mathbf{E}[\Delta\Phi^{\mathrm{B}}] \leq (1 + f_1) \cdot \mathcal{B}(X_t) \cdot D \cdot \sqrt{D} \ .$$

Proof. We bound the amortized cost incurred on DIST in the spreading stage as follows. $C^{\mathrm{B}}_{\mathrm{DIST}}$, the cost of movement from σ_t to $P'_{\mathrm{DIST}} \in J(t)$ is at most $D \cdot \sqrt{D}$. We note that we do not consider a constant overhead for the communication (i.e., we have not written $D \cdot (\sqrt{D} + 1)$, as it was already taken into account while analyzing the jump stage. Additionally, the movement in the spreading part happens with probability $\mathcal{B}(X_t)$, and thus

$$\mathbf{E}[C^{\mathrm{B}}_{\mathrm{DIST}}] \leq \mathcal{B}(X_t) \cdot D \cdot \sqrt{D} \ . \tag{3.5}$$

We can bound $\Delta\Phi^{\mathrm{B}}$ analogously. As DIST moves (with probability $\mathcal{B}(X_t)$) along a distance of at most $\sqrt{D}$, the distance between P_{DIST} and P_{OPT} can increase only by $\sqrt{D}$. Therefore,

$$\mathbf{E}[\Delta\Phi^{\mathrm{B}}] \leq f_1 \cdot \mathcal{B}(X_t) \cdot D \cdot \sqrt{D} \ . \tag{3.6}$$

By summing up (3.5) with (3.6) we get the lemma. ■

Total amortized cost in the first part of the step

Lemma 3.5. *In the first part of any time step t, it holds that*

$$\mathbf{E}[C_{\mathrm{DIST}}] + \mathbf{E}[\Delta\Phi] + \mathbf{E}[\Delta\Theta] \leq O(n \cdot \sqrt{D}) \cdot C_{\mathrm{OPT}} \ .$$

Proof. First, we can combine Lemma 3.3 with Lemma 3.4 to get

$$\begin{aligned}\mathbf{E}[C_{\mathrm{DIST}}] + \mathbf{E}[\Delta\Phi] &\leq O\left(\sqrt{D}\right) \cdot C_{\mathrm{OPT}} + (22 + f_1) \cdot \mathcal{B}(X_t) \cdot D \cdot \sqrt{D} \\ &= O\left(\sqrt{D}\right) \cdot C_{\mathrm{OPT}} + f_2 \cdot \mathcal{B}(X_t) \cdot D \cdot \sqrt{D} \ .\end{aligned} \tag{3.7}$$

Thus, it remains to bound the term $f_2 \cdot \mathcal{B}(X_t) \cdot D \cdot \sqrt{D}$ using the change in the potential Θ. The brief idea is as follows. We may prove that either P_{OPT} is inside a jump set $J(t)$ and DIST catches it with a certain probability, substantially lowering the potential Θ, or P_{OPT} is outside $J(t)$, and thus OPT pays at least $\sqrt{D}$ for serving request in this time step. We consider two cases

1. $P_{\text{OPT}} \notin J(t)$ or $P_{\text{OPT}} \equiv P_{\text{DIST}}$.

   In this case we can prove that

   $$\mathcal{B}(X_t) \cdot D \leq b \cdot C_{\text{OPT}} \ . \tag{3.8}$$

   If $P_{\text{OPT}} \notin J(t)$, then $d_t(P_{\text{OPT}}, \sigma_t) > \sqrt{D}$, and thus $C_{\text{OPT}} > \sqrt{D}$. This, in turn, implies that $\mathcal{B}(X_t) \cdot D \leq (b/\sqrt{D}) \cdot D \leq b \cdot C_{\text{OPT}}$. On the other hand, if $P_{\text{OPT}} \in J(t)$, then from the case assumption $P_{\text{OPT}} \equiv P_{\text{DIST}}$. In this case $X_t \leq \sqrt{D}$ is also the distance between P_{OPT} and the request. Thus, $\mathcal{B}(X_t) \cdot D = b \cdot \frac{X_t}{D} \cdot D = b \cdot X_t \leq b \cdot C_{\text{OPT}}$. In either case Inequality 3.8 follows.

   Now, the increase in the potential Θ may be at most $f_2 \cdot n \cdot D \cdot \sqrt{D}$ and occurs with probability $\mathcal{B}(X_t)$. Thus,

   $$\mathbf{E}[\Delta\Theta] \leq f_2 \cdot n \cdot B(X_t) \cdot D \cdot \sqrt{D} \ .$$

   Summing this up with Inequality 3.7, we get

   $$\begin{aligned} \mathbf{E}[C_{\text{DIST}} + \Delta\Phi + \Delta\Theta] &\leq O\left(\sqrt{D}\right) \cdot C_{\text{OPT}} + 2 \cdot f_2 \cdot n \cdot \mathcal{B}(X_t) \cdot D \cdot \sqrt{D} \\ &\leq O\left(\sqrt{D}\right) \cdot C_{\text{OPT}} + 2 \cdot f_2 \cdot n \cdot \sqrt{D} \cdot b \cdot C_{\text{OPT}} \\ &\leq O\left(n \cdot \sqrt{D}\right) \cdot C_{\text{OPT}} \end{aligned}$$

2. $P_{\text{OPT}} \in J(t)$ and $P_{\text{OPT}} \not\equiv P_{\text{DIST}}$

   In this case, at the beginning of the step $\Theta = f_2 \cdot n \cdot D \cdot \sqrt{D}$. This potential remains unchanged, if DIST does not move, or if it moves to a node different from P_{OPT}. However, with probability $\mathcal{B}(X_t)/|J(t)|$, DIST moves to P_{OPT} lowering Θ to 0. Thus,

   $$\begin{aligned} \mathbf{E}[\Delta\Theta] &\leq -\frac{\mathcal{B}(X_t)}{|J(t)|} \cdot f_2 \cdot n \cdot D \cdot \sqrt{D} \\ &\leq -f_2 \cdot \mathcal{B}(X_t) \cdot D \cdot \sqrt{D} \ , \end{aligned}$$

   where the second inequality follows from $|J(t)| \leq n$. Summing this with (3.7), we get

   $$\begin{aligned} \mathbf{E}[C_{\text{DIST}} + \Delta\Phi + \Delta\Theta] &\leq O\left(\sqrt{D}\right) \cdot C_{\text{OPT}} + (f_2 - f_2) \cdot \mathcal{B}(X_t) \cdot D \cdot \sqrt{D} \\ &= O\left(\sqrt{D}\right) \cdot C_{\text{OPT}} \ . \end{aligned}$$

This finishes the proof of the Lemma 3.5. ■

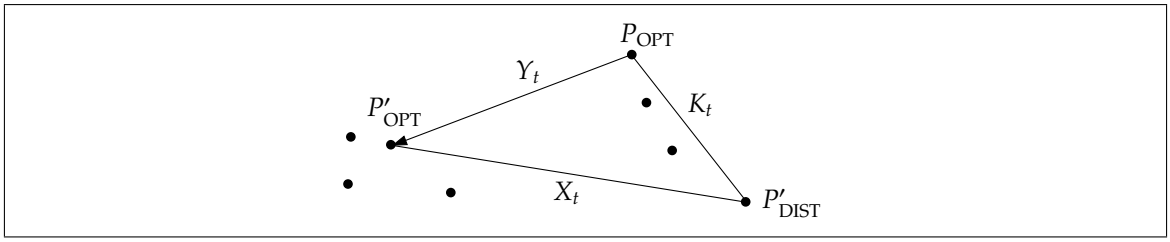

Figure 3.4: Algorithm DIST. OPT moves its page.

3.1.3 DIST in the second part of a step

Now we prove that the Inequality 3.3 holds also in the second part of each time step, in which the adversary moves the page.

Lemma 3.6. *In the second part of any time step t, it holds that*

$$\mathbf{E}[C_{DIST}] + \mathbf{E}[\Delta\Phi] + \mathbf{E}[\Delta\Theta] \leq O(n \cdot \sqrt{D}) \cdot C_{OPT} \ .$$

Proof. In this part $C_{DIST} = 0$ and we have already taken into account the changes in the potential caused by the adversarial changes in the network topology. Algorithm DIST already moved to a new position, called P'_{DIST} (possibly $P'_{DIST} \equiv P_{DIST}$). Assume that the adversary moves its page to a node P'_{OPT}. We reuse notation from the previous proof, i.e., let K_t, X_t and Y_t denote the distances between P'_{DIST} and P_{OPT}, P'_{DIST} and P'_{OPT}, and P_{OPT} and P'_{OPT}, respectively. See Figure 3.4 for an illustration.

Since the adversary does not jump to the same node it is already at, $C_{OPT} = D \cdot (Y_t + 1)$. We have to show only that the change in the potentials Φ and Θ can be bounded appropriately. Naturally, the total increase in Θ is bounded by $f_2 \cdot n \cdot D \cdot \sqrt{D}$. Thus,

$$\begin{aligned} \Delta\Theta &\leq f_2 \cdot n \cdot D \cdot \sqrt{D} \\ &\leq O\left(n \cdot \sqrt{D}\right) \cdot C_{OPT} \ . \end{aligned}$$

On the other hand, $\Delta\Phi = \mathcal{F}(X_t) - \mathcal{F}(K_t)$. We have to consider only the cases in which $X_t > K_t$, otherwise $\Delta\Phi \leq 0$ and the lemma would trivially follow. We use the triangle inequality and consider two cases. If $K_t \leq \sqrt{D}$, then

$$\begin{aligned} \Delta\Phi &\leq \mathcal{F}(X_t) \\ &\leq X_t \cdot D \\ &\leq (Y_t + K_t) \cdot D \\ &\leq (Y_t + \sqrt{D}) \cdot D \\ &\leq C_{OPT} + \sqrt{D} \cdot C_{OPT} \\ &\leq O(\sqrt{D}) \cdot C_{OPT} \ . \end{aligned}$$

Otherwise, if $K_t > \sqrt{D}$, then

$$\begin{aligned}\Delta\Phi &= \mathcal{F}(X_t) - \mathcal{F}(K_t)\\ &= X_t \cdot D - K_t \cdot D\\ &\leq Y_t \cdot D\\ &\leq C_{\text{OPT}} \ .\end{aligned}$$

Thus, in either case it holds that $C_{\text{DIST}} + \Delta\Phi + \Delta\Theta \leq O(n \cdot \sqrt{D}) \cdot C_{\text{OPT}}$. ■

3.1.4 Combining DIST with other algorithms

We showed that Dist is a memoryless randomized algorithm achieving competitive ratio of $O(n \cdot \sqrt{D})$. If we know the exact values of D, n and λ, then at the beginning we may choose either Dist, or $O(D)$-competitive Jump, or $O(\lambda)$-competitive Dyn-CF, which yields the upper bound of $O(\min\{n \cdot \sqrt{D}, D, \lambda\})$ for randomized memoryless algorithms.

While the knowledge of n and D seems reasonable, and our algorithms use these values anyway, it appears that we can get the strict $O(\min\{n \cdot \sqrt{D}, D, \lambda\})$-competitiveness without prior knowledge of maximum network extent λ.

First, we start $O(\lambda)$-competitive algorithm Dyn-CF. If at some point of the time the distance between any pair of nodes is equal of greater than D (or $n \cdot \sqrt{D}$, whichever is smaller), then algorithm starts using algorithm Jump (or, respectively, Dist) from this step. We have to show that this strategy is strictly $O(\min\{D, \lambda\})$-competitive (or respectively strictly $O(\min\{n \cdot \sqrt{D}, \lambda\})$-competitive).

Lemma 3.7. *If we combine* Dyn-CF *and* Jump *in this way, then the resulting strategy is strictly* $O(\min\{D, \lambda\})$*-competitive.*

Proof. Consider any input sequence $\mathcal{I}$. If the switching to Jump never occurs, then $\lambda < D$. Since Dyn-CF is strictly $O(\lambda)$-competitive, the lemma follows.

On the other hand, if the strategy switches to Jump at some point of the time, then $\lambda \geq D$. In this case, it is sufficient to prove that the strategy is $O(D)$-competitive. Let $\mathcal{I}_1$ and $\mathcal{I}_2$ be the parts of the input $\mathcal{I}$ processed by Dyn-CF and Jump, respectively. Let t_1 be the last step of $\mathcal{I}_1$, and t_2 be the last step of $\mathcal{I}_2$. Note that switching decision occurs in fact in step $t_1 + 1$, but since it is made before the node movement, we may assume for the analysis that this decision is made at the end of step t_1. Let $\Phi_{\text{DYN-CF}}$ and Φ_{JUMP} be the potential functions used in the proof of Dyn-CF and Jump competitiveness. Since the maximal distance occurring in $\mathcal{I}_1$ is at most D, we have

$$\mathbf{E}[C_{\text{DYN-CF}}(\mathcal{I}_1) + \Phi_{\text{DYN-CF}}(t_1) - \Phi_{\text{DYN-CF}}(0)] \ \leq \ O(D) \cdot \mathbf{E}[C_{\text{OPT}}(\mathcal{I}_1)] \ .$$

On the other hand, from the competitiveness of JUMP

$$\mathbf{E}[C_{\text{JUMP}}(\mathcal{I}_2) + \Phi_{\text{JUMP}}(t_2) - \Phi_{\text{JUMP}}(t_1)] \;\leq\; O(D) \cdot \mathbf{E}[C_{\text{OPT}}(\mathcal{I}_2)] \;.$$

Summing up, and taking into account that $\Phi_{\text{DYN-CF}}(0) = 0$, and $\Phi_{\text{JUMP}}(t_2) \geq 0$, we get

$$\mathbf{E}[C_{\text{DYN-CF}}(\mathcal{I}_1) + C_{\text{JUMP}}(\mathcal{I}_2) + \Phi_{\text{DYN-CF}}(t_1) - \Phi_{\text{JUMP}}(t_1)] \;\leq\; O(D) \cdot \mathbf{E}[C_{\text{OPT}}(\mathcal{I})] \;.$$

Thus, the strategy is $O(D)$-competitive. To show strict competitiveness, it suffices to show that the potential of DYN-CF at the moment of switch (i.e., at the end of time step t_1) between the algorithms is greater than the starting potential of JUMP. If at the end of time step t_1, the algorithm has its page in the same node as OPT, then by Corollary 2.15 (see page 29), $\Phi_{\text{DYN-CF}}(t_1) \geq 0 = \Phi_{\text{JUMP}}(t_1)$. Otherwise, by the same corollary, $\Phi_{\text{DYN-CF}}(t_1) \geq k \cdot (D+1) \cdot D$ for any chosen constant k. If we choose $k = 3$, then $\Phi_{\text{DYN-CF}}(t_1) \geq \Phi_{\text{JUMP}}(t_1)$, as the maximum distance at the end of t_1 is at most D. ■

Lemma 3.8. *If we combine* DYN-CF *and* DIST *in this way, then the resulting strategy is strictly* $O(\min\{n \cdot \sqrt{D}, \lambda\})$*-competitive.*

Proof. The proof follows an identical pattern as the previous one. Similarly, it suffices to show that the potential does not increase at the moment of switch from DYN-CF to DIST. If the algorithm has its page at the same node as OPT, then $\Phi_{\text{DYN-CF}}(t_1) \geq 0 = \Phi_{\text{DIST}}(t_1) + \Theta_{\text{DIST}}(t_1)$. Otherwise, by Corollary 2.15, $\Phi_{\text{DYN-CF}}(t_1) \geq k \cdot (n \cdot \sqrt{D} + 1) \cdot D$ for any chosen constant k. If we pick $k = f_1 + f_2$, then

$$\begin{aligned}\Phi_{\text{DYN-CF}}(t_1) &\geq f_1 \cdot n \cdot \sqrt{D} \cdot D + f_2 \cdot n \cdot D \cdot \sqrt{D} \\ &\geq \Phi_{\text{DIST}}(t_1) + \Theta_{\text{DIST}}(t_1) \;,\end{aligned}$$

which finishes the proof. ■

By combining the two lemmas above we get the following corollary.

Corollary 3.9. *There exists a randomized memoryless strategy, which upon knowing n and D (or at least their constant approximations), is* $O(\min\{n \cdot \sqrt{D}, D, \lambda\})$*-competitive against an adaptive-online adversary in the adversarial scenario of the DPM.*

3.2 Marking algorithms

In this section we show that the randomization is not crucial, presenting a deterministic algorithm MARK, which achieves asymptotically the same competitive ratio as the DIST algorithm described in the previous section.

Later on, we show that by adding a randomization to the MARK's core, we may trivially reduce its competitiveness to $O(\sqrt{D} \cdot \log n)$ against an oblivious adversary. By fine-tuning its parameters we get a randomized algorithm EBM, whose competitive ratio of $O(\sqrt{D \cdot \log n})$ beats the best possible ratio of $\Omega(\sqrt{D} \cdot n)$ against an adaptive adversary. Moreover, in Section 3.3.2 we show that this ratio is optimal for $D \geq \log^3 n$.

The algorithm MARK is partially inspired by the MOVE-TO-MIN algorithm by Awerbuch, Bartal and Fiat [ABF93a]. A brief idea the 7-competitive algorithm MOVE-TO-MIN is as follows. It works in chunks of length D, i.e., it divides the input sequence into chunks of such length, in each chunk serves all the requests, and *moves only at the end of chunks* to a so-called *gravity center*. A gravity center is a node, which would be the best place for the page in the last chunk, i.e., it minimizes the sum of distances to all the requests issued.

MARK takes the chunk-based approach after MOVE-TO-MIN. The chosen chunk's length must be long enough to allow amortization of the page movements against the cost incurred by serving requests, and short enough to make the network changes negligible. In this whole section, K denotes the length of the chunk. K is a parameter, different for different algorithms. For now we assume only that $K \leq 2 \cdot \sqrt{D}$.

In addition, on the beginning of the input sequence, the algorithm works in chunks of length $\tilde{K} := 2 \cdot \sqrt{D}$ for a short period. This run-up is needed to ascertain only that the marking algorithms using $K < 2 \cdot \sqrt{D}$ could be strictly competitive.[2]

Gravity centers

Since in our setting the distances can change with time, we have to be careful with defining gravity centers. Consider any subsequence $\mathcal{S}$ of length ℓ steps. We number this steps from 1 to ℓ. Let σ_i be the node which issues a request in the i-th step of I and $d_i(\cdot)$, $c_i(\cdot)$ be the distance and cost functions, respectively, in the i-th step.

Definition 3.10. *A gravity center for a subsequence $\mathcal{S}$ of length ℓ is a vertex v, which minimizes the sum $\sum_{i=1}^{\ell} d_\ell(v, \sigma_i)$. We denote it by $\mathcal{G}_\mathcal{S}$. If there is more than one such vertex, then the gravity center is the one labelled with the smallest index.*[3]

Design rationale

However, if we consider the lower bound for last request based algorithms presented in Figure 3.1, we see that it also applies for an algorithm which considers only gravity centers as jump candidates. Actually, v_3 from that example would never be chosen.

[2] This will be become clear after the proof of Lemma 3.23.

[3] We could apply any tie breaking method here. Moreover, the algorithms presented would still work, if we chose a node which minimizes $\sum_{i=1}^{\ell} d_i(v, \sigma_i)$ or $\sum_{i=1}^{\ell} c_i(v, \sigma_i)$.

Keeping in mind that storing the page close to the gravity center is, generally, a good idea, we strive for adapting the "chasing OPT game" to the deterministic setting. In other words, we have to ascertain that MARK frequently catches OPT. To achieve this, we introduce a marking scheme and build MARK on top of it.

A short motivation behind the scheme is following. MARK will be, in essence, a derandomization of the DIST algorithm. The gravity center tries to approximate the position, at which DIST would be after serving all the requests from one chunk. In particular, if all the requests within one chunk are given at a node v^*, whose distance to the node holding the page is roughly $\sqrt{D}$, then in each step DIST would move to the jump set of this node with probability $\Theta(1/\sqrt{D})$. If the length of the chunk is chosen close to $\sqrt{D}$, then at the end of such a chunk DIST would be in the neighborhood of v^* with a constant probability. Hence, we aim for this in constructing MARK, i.e., after a chunk I, it should move to the neighborhood of the gravity center $\mathcal{G}_I$.

Furthermore, DIST tries to catch OPT (assuming that OPT is in the jump set) by randomly choosing a node from the jump set. In order to simulate this, MARK has to distinguish between the potential jump candidates, not to choose the same node twice. This is the place, where the marking scheme comes to play. Generally, the nodes, in which MARK has been already are marked, and will not be visited by MARK again. Additionally, as mentioned in Section 3.1 (in the part about catching OPT), we have to assure that each time when MARK catches OPT, a certain cost is incurred on OPT. This is achieved by making the marking of a node dependent on the cost, which is incurred on the algorithm remaining for the whole time at this node. Thus, we introduce the following definition.

Definition 3.11. *For any subsequence $\mathcal{S}$, a* counter $A_i(\mathcal{S})$ *denotes the cost of serving the requests within $\mathcal{S}$ from node v_i. Equivalently, it is the cost of an algorithm, which remains at v_i for the whole $\mathcal{S}$ and does not move.*

The considerations above sketch the following rough picture of MARK. MARK works in chunks of length $K = 2 \cdot \sqrt{D}$. Chunks are grouped in epochs, each epoch begins with all nodes unmarked. In each epoch we track A_i counters for the part of the epoch seen so far. If such a counter exceeds D, then the corresponding node becomes marked. At the end of a chunk, in which all nodes are already marked, the current epoch ends, the scheme unmarks all nodes, and a new epoch begins. The marking process runs completely independently from any algorithm.

On the other hand, MARK uses this scheme in the following way. It remains in a node till the end of the chunk, in which this node gets marked and then moves to a node, which is not marked yet. In the proof we show that this approach guarantees that nodes chosen for jump candidates are close to the gravity centers.

Let M_i be the number of marks v_i has.

Run-up epoch:

Set $M_i := 0$ for all v_i /* unmark all the nodes */
$\mathcal{E}_0 := \emptyset$ /* $\mathcal{E}_0$ is the zeroth epoch */
while $(M_1 = 0)$ **do**
  $I :=$ next $\tilde{K}$ steps from $\mathcal{I}$
  $\mathcal{E}_0 := \mathcal{E}_0 \uplus I$
  $M_1 := \lfloor A_1(\mathcal{E}_0) / \left(\frac{1}{4} \cdot \tilde{K}^2\right) \rfloor$ /* compute current marks of v_1 */

Regular epochs:

Set $M_i := 0$ for all v_i /* unmark all the nodes */
Let $T := D / \left(\frac{1}{4} \cdot K^2\right)$
$(I_1, I_2, I_3, \ldots, I_m) := \mathcal{I} \setminus \mathcal{E}_0$ /* divide remaining part of $\mathcal{I}$ into chunks I_j of length K */
$\mathcal{E} := \emptyset$ /* $\mathcal{E}$ is the current epoch; the first epoch begins */

for j = 1 **to** m **do**
  $\mathcal{E} := \mathcal{E} \uplus I_j$
  for each $v_i \in V$
    $M_i := \lfloor A_i(\mathcal{E}) / \left(\frac{1}{4} \cdot K^2\right) \rfloor$ /* compute current marks */
    if $M_i \geq T$ for all v_i **then** /* if all nodes are marked at least T times */
      Set $M_i := 0$ for all v_i /* unmark all the nodes
      $\mathcal{E} := \emptyset$ a new epoch begins */

Figure 3.5: Marking scheme for input sequence $\mathcal{I}$ with regular chunks of length K

Marking scheme

While the Mark's description above is sufficient to derive a proof of its competitiveness, we make the marking scheme more general, and present Mark as an example of *marking-based* algorithms (a class of algorithms, which we define later). This allows us to capture the common parts of Mark and its randomized successors, R-Mark and Ebm, and reuse crucial lemmas. Above all, we parameterize the regular chunk length, allowing them to be shorter than $2 \cdot \sqrt{D}$, and we parameterize number of times each node is marked before an epoch ends.

We define the marking scheme in Figure 3.5. The marking scheme depends on one parameter — the value of K, the chunk length chosen by the algorithm. We divide

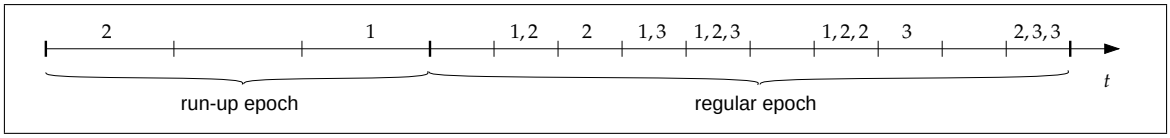

Figure 3.6: Example of marking scheme for $T = 4$

input sequence $\mathcal{I}$ into chunks. The initial ones (called *run-up chunks*) are of length $\tilde{K}$, the remaining ones (called *regular chunks*) are of length K. The last chunk possibly has a shorter length. As a byproduct, the marking scheme computes the division of $\mathcal{I}$ into epochs. Each epoch contains one or more chunks, i.e., the partition into chunks is a refinement of the partition into epochs. The zeroth epoch (called also *run-up epoch*) starts with the beginning of the input sequence $\mathcal{I}$ and contains all run-up chunks.

In each epoch $\mathcal{E}$ we are tracking the A_i counters for the part of the epoch $\mathcal{E}$ seen so far. Each node v_i has an associated counter M_i, denoting the number of marks it got. Initially, at the beginning of each epoch nodes begin unmarked, i.e., all M_i are set to 0. One can view marks as a rough approximation of the corresponding A_i counters, i.e., $M_i = \lfloor A_i(\mathcal{E})/(\frac{1}{4} \cdot K^2) \rfloor$. Additionally, marks are computed only at the end of each chunk. If after some chunk I a counter M_i increases, then we say that v_i *was marked in* I, or that I is a *marking chunk* for v_i.

Epoch zero ends with the chunk, in which v_1 was marked (this implies that $A_1(\mathcal{E}_0)$ counter exceeds D). Recall that any algorithm begins with the page initially at v_1. For the next epochs (called also *regular epochs*), their ending condition is slightly more complex. A node which is marked at least $T = D/(\frac{1}{4} \cdot K^2)$ times we call *saturated*. For a saturated node v_i the corresponding counter $A_i(\mathcal{E})$ exceeds D. If at the end of some chunk all nodes are saturated, then the current epoch ends.

When an epoch ends, before the next chunk begins, we unmark all the nodes, i.e., all M_i are set to 0. A new epoch begins with the next chunk. Possibly, at the end of the input sequence, there is one epoch which is not ended; such an epoch we call *unfinished*.

We note that the division into epochs and chunks as well as the marking scheme is independent of the algorithm and depends only on the input sequence and the value of K. For clarity of the presentation, we assume that K is so chosen, that both K and T are integers. If it is not the case, we can always increase D a little bit, so that K becomes an integer, and use a new value of D in the analysis of the algorithm. Since this increase changes D only by a constant factor, asymptotically all the bounds hold. Finally, we assume that $D \geq 4$. If it is not the case, we may use the algorithm JUMP to achieve constant competitiveness.

An example of marking scheme for $K = \sqrt{D}$ and a three-node graph is presented in Figure 3.6. The chosen value of K implies that $T = 4$. The figure contains one run-up epoch followed by one regular epoch; numbers above the chunks in this figure denote

```
Serve all requests in chunk I

if I is a finished chunk then
    if I is the last chunk in the current epoch
    then
        v* := G_I
        Move to v*
    else
        If P_MB is marked in I then
            Choose a jump candidate v*    /* placeholder */
            Move to v*
```

Figure 3.7: Framework of a marking-based algorithm MB

indices of nodes, which are marked in these chunks.

We have a trivial lower bound on Opt in any finished epoch.

Lemma 3.12. *For any finished epoch $\mathcal{E}$ it holds that $C_{\text{OPT}}(\mathcal{E}) \geq D$.*

Proof. If the optimal offline algorithm moves its page within $\mathcal{E}$ then it pays at least D. Otherwise, it remains for the whole epoch $\mathcal{E}$ in one node v_i, paying $A_i(\mathcal{E})$. For the zeroth epoch, Opt starts and remains in v_1, and therefore it pays $A_1(\mathcal{E}_0) \geq \frac{1}{4} \cdot \tilde{K}^2 = D$. For a regular epoch $\mathcal{E}$, any node v_i is marked at least T times within $\mathcal{E}$, and thus Opt pays $A_i(\mathcal{E}) \geq T \cdot \frac{1}{4} \cdot K^2 = D$. ■

Marking-based algorithms

Now we show how to use the marking information in our "chasing OPT" game. We can view the online problem as a request-answer game (see [BBK+90] or [BE98, chapter 7] for an introduction to such games), between the adversary (and Opt) and our algorithm. For this argument we consider a deterministic algorithm, although the rationale makes sense also for randomized ones.

As we have a lower bound for Opt guaranteed by Lemma 3.12, it is the role of our algorithm to force the adversary to end the epoch, i.e., to have all the nodes marked at least T times as quick as possible.

Note that the algorithm could hardly trigger that marking event, if it is at a saturated node. In that case the adversary may issue requests at a node with a low number of marks, deferring this way the end of the current epoch. In fact, the A_i counters for other not yet saturated nodes increase in this case, but it may happen that the increase

is only 1 per round, while the increase in our algorithm's cost could be huge. Therefore, the rationale of our algorithm would be to remain at a node with small number of marks, until it gets marked in some chunk I, and then move to another unsaturated node. Note that for $K = 2 \cdot \sqrt{D}$ a node is saturated when it is marked at least one time, and this justifies the MARK algorithm moving to any not yet marked node.

At the end of the last chunk of any epoch (for any regular epoch all nodes are saturated then) the algorithm could move to the gravity center. These considerations lead us to the following definition.

Definition 3.13. *An algorithm* MB *we call* marking-based *if* MB *moves only at the end of the chunk I, which*

(i) was a marking chunk for P_{MB}, the node holding the page of MB, *or*

(ii) was the last chunk in epoch.

Additionally, if condition (ii) is met, then MB *moves to $\mathcal{G}_I$.*

Clearly, each marking-based algorithm is also chunk-based. For making our arguments concise, we assume that after condition (i) or (ii) of Definition 3.13 occurs, the algorithm always move, although in the rare cases it may move to the same node it is currently in. The framework for any marking-based algorithm is depicted of Figure 3.7. We distinguish between choosing a jump candidate *inside epoch* and choosing a jump candidate *at the end of epoch*. In the latter case the jump candidate is always the gravity center of the last epoch's chunk. Choosing a jump candidate inside an epoch is a placeholder which will be filled by a specific marking-based algorithm that we will analyze. In the next subsection, we prove that even if we base our algorithm on the marking information only, we can still ascertain that it moves to the neighborhood of the gravity centers.

We call a subsequence between two movements of such an algorithm a *phase*. Alternatively speaking, a phase is a sequence of consecutive chunks, in which the algorithm remains at one node. Recall that chunks, phases, and epochs are all subsequences of input sequence. Moreover, the whole input sequence is partitioned into epochs, each epoch into phases, and each phase into chunks. If the input sequence ends with an unfinished epoch, it ends possibly with an unfinished phase, too. We already know that the division into epochs and chunks is independent from any algorithm. Additionally, the run-up epoch consists of only one phase. The division of each regular epoch into phases depends directly on the choice of jump candidates inside epochs.

Regular chunks' length: $K = 2 \cdot \sqrt{D}$

Choosing a jump candidate v^* inside epoch:
v^* is any not yet marked node

Figure 3.8: Mark properties

3.2.1 Deterministic algorithm MARK

In this subsection we present a deterministic Mark. Although it was informally described before, we restate it as a marking-based algorithm.

It is necessary to provide only two pieces of information (see Figure 3.8): the chunk length K and the way of choosing jump candidates inside epoch. Let $K = 2 \cdot \sqrt{D}$, i.e., the chunks length is the same in all epochs (run-up and regular ones). This implies that $T = 1$, i.e., a regular epoch ends, if all the nodes are marked at least once. Additionally, for a jump candidate inside epoch Mark chooses any not yet marked node.

In the remaining part of this section we prove the following result.

Theorem 3.14. *The algorithm* Mark *is* $O(n \cdot \sqrt{D})$*-competitive in the adversarial model of the DPM.*

First, we prove a bound on the number of Mark movements in each phase. Each node that Mark visits during an epoch is marked and will not be visited again during the same epoch. This implies the following lemma.

Lemma 3.15. *The number of* Mark *phases in any epoch (even unfinished one) is at most* n*.*

Proof. The run-up epoch contains one phase of any marking-based algorithm, and thus the lemma trivially holds there.

Any other epoch begins with all nodes unmarked and ends with all nodes marked. Additionally, in each phase at least one node is marked. Thus, the number of phases cannot be higher than n. Obviously, the proof works for an unfinished epoch as well. ■

Generalized Jump Sets

First, we prove that if Mark moves after some chunk, then as a jump candidate it chooses a node, which is close to the gravity centers corresponding to this chunk. In order to do this we adapt the definition of a jump set from the algorithm Dist. We make the definition slightly more general than needed for the analysis of the Mark algorithm; we use it later for a randomized version of Mark. Below we concentrate on a single chunk I, and we number its steps from 1 to K.

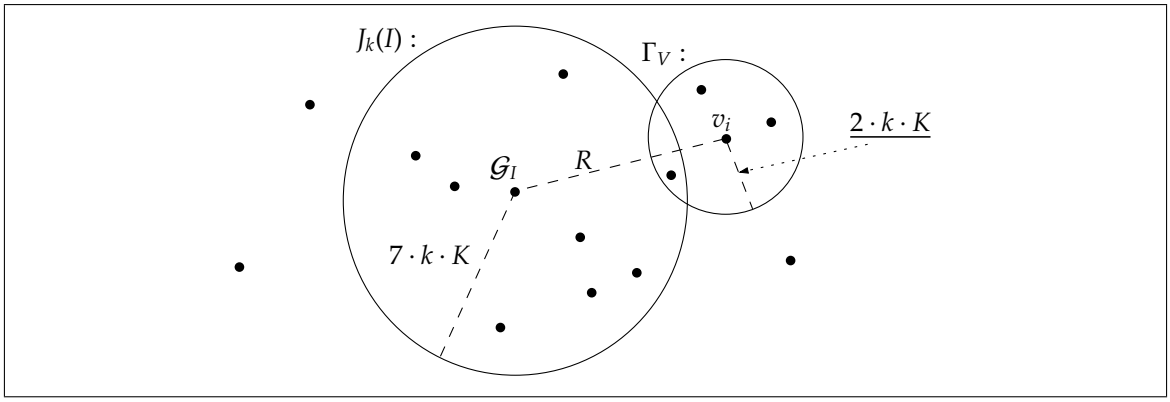

Figure 3.9: Jump Set $J_k(I)$

Definition 3.16. *For any chunk I and any integer $k \geq 1$, a* k-JumpSet*, which we denote by $J_k(I)$, is the set of all nodes whose distance to $\mathcal{G}_I$, measured in the last step of I, is at most $7 \cdot k \cdot K$, i.e., $J_k(I) = \{v \in V : d_K(v, \mathcal{G}_I) \leq 7 \cdot k \cdot K\}$.*

Intuitively, if an algorithm remains at a node which was far away from the gravity center or outside some jump sets, it had to pay much. This is formalized in the following lemma.

Lemma 3.17. *For any chunk I of K steps, any node $v_i \in V$, and any $k \geq 1$, if $v_i \notin J_k(I)$ at the end of I, then $A_i(I) \geq \frac{k}{4} \cdot K^2$.*

Proof. We look at the configuration of nodes in time step K. Let $R := d_K(\mathcal{G}_I, v_i)$. Since $v_i \notin J_k(I)$, $R > 7 \cdot k \cdot K$. By Γ we denote a set of time steps t from chunk I, such that the K-th step distance between σ_t and v_i is at most $2 \cdot k \cdot K$. Formally,

$$\Gamma := \{\, t \in I : d_K(\sigma_t, v_i) \leq 2 \cdot k \cdot K \,\} \ . \tag{3.9}$$

The situation in time step K is depicted in Figure 3.9. Γ_V, shown there, is a multi set of nodes induced by Γ, i.e.,

$$\Gamma_V = \{\sigma_t : d_K(\sigma_t, v_i) \leq 2 \cdot k \cdot K\} \ . \tag{3.10}$$

Intuitively, Γ_V is the set of nodes, which issued requests in I and are now close to v_i.

The idea of the proof is simple. We show that at least a constant fraction of nodes accessing the page are outside Γ_V. Then we argue that these nodes were far away from v_i at the moment when requests were issued at them, and therefore they must have incurred a high cost on the algorithm remaining at v_i for the whole chunk I.

First, we prove that $|\Gamma| \leq \frac{3}{4} \cdot K$. Assume the contrary, i.e., $|\Gamma| > \frac{3}{4} \cdot K$. Using the triangle inequality we obtain

$$\begin{aligned}
\sum_{t=1}^{K} d_K(v_i, \sigma_t) &= \sum_{t \in \Gamma} d_K(v_i, \sigma_t) + \sum_{t \notin \Gamma} d_K(v_i, \sigma_t) \\
&\leq |\Gamma| \cdot 2 \cdot k \cdot K + \sum_{t \notin \Gamma} (d_K(v_i, \mathcal{G}_I) + d_K(\mathcal{G}_I, \sigma_t)) \\
&< 2 \cdot k \cdot K^2 + \frac{1}{4} \cdot K \cdot R + \sum_{t \notin \Gamma} d_K(\mathcal{G}_I, \sigma_t) \\
&< \frac{3}{4} \cdot K \cdot (R - 2 \cdot k \cdot K) + \sum_{t \notin \Gamma} d_K(\mathcal{G}_I, \sigma_t) \ ,
\end{aligned}$$

where the last inequality follows from $R > 7 \cdot k \cdot K$. Since $\frac{3}{4} \cdot K < |\Gamma|$ and in the last step of I, the distance between $\mathcal{G}_I$ and any node from Γ is at least $R - 2 \cdot k \cdot K$, we get

$$\begin{aligned}
\sum_{t=1}^{K} d_K(v_i, \sigma_t) &< \sum_{t \in \Gamma} d_K(\mathcal{G}_I, \sigma_t) + \sum_{t \notin \Gamma} d_K(\mathcal{G}_I, \sigma_t) \\
&= \sum_{t=1}^{K} d_K(\mathcal{G}_I, \sigma_t) \ .
\end{aligned}$$

This contradicts that $\mathcal{G}_I$ is a gravity center.

Since $|\Gamma| \leq \frac{3}{4} \cdot K$, at least $\frac{1}{4} \cdot K$ of the requests were issued "far away" from v_i. Precisely, since during K steps each distance can be changed at most by an additive term of K, each of these requests was issued at the distance of at least $2 \cdot k \cdot K - K \geq k \cdot K$ from v_i. Therefore, $A_i(I) \geq (K - |\Gamma|) \cdot k \cdot K = \frac{k}{4} \cdot K^2$ and the lemma follows. ■

By the definition of the marking scheme we immediately conclude the following.

Corollary 3.18. *For any chunk I and a node v_i, if v_i is outside $J_k(I)$ at the end of I, then v_i received at least k marks in I.*

This corollary claims that by choosing nodes which have small number of marks we choose nodes which are close to the gravity center. After any chunk I, for a jump candidate v^* the algorithm Mark either chooses the gravity center of I, or a node which is not yet marked. But in the latter case by Corollary 3.18 such a node has to belong to the 1-JumpSet of I.

Corollary 3.19. *If* Mark *moves its page after I, then for a jump candidate v^* it always chooses a node belonging to $J_1(I)$, the 1-JumpSet of I.*

Amortized analysis

In this subsection we complete the proof of Theorem 3.14 using Lemma 3.12 and Corollary 3.19. In the proof we use potential function analysis. However, unlike in the DIST case, we cannot show that the amortized cost in each step is bounded by a term proportional to C_{OPT}. Instead, we prove a bound for amortized cost of MARK in any (finished or unfinished) epoch.

Let L denote the distance between the nodes holding the pages of MARK and OPT, respectively. Then we define a potential as

$$\Phi = f \cdot D \cdot L \ , \ \text{where} \ \ f = 2 \ . \tag{3.11}$$

Clearly, at the beginning of an input sequence $\Phi = 0$ and is always non-negative. For any subsequence $\mathcal{S}$, by $\Delta\Phi(\mathcal{S})$ we denote the difference between the potential after $\mathcal{S}$ (after both OPT and MARK moved their pages), and before $\mathcal{S}$ (at the very end of the step preceding $\mathcal{S}$; if $\mathcal{S}$ starts at the beginning of input, then the starting potential equals 0, since at that point of time $L = 0$). By an *amortized cost* of an action (e.g., serving requests or moving the page) we understand the actual cost of this action plus the change in the potential this action induced.

In fact, we can extend the definitions above to any marking-based algorithm. Most of the lemmas below holds for any such algorithm. In particular Φ may be the potential function for any marking-based algorithm MB, in which case it is equal to $f \cdot D$ times the distance between the pages of MB and OPT.

First, we bound the cost of MARK is one finished phase P. Let P consist of ℓ chunks, numbered from 1 to ℓ, i.e., $P = (I_1, I_2, \ldots, I_\ell)$. From a definition of a phase, we get that MARK remains at one node in the whole P. In the last step of the phase it moves to a jump candidate v^*.

Consider the following thought experiment. If MARK first moved to $\mathcal{G}_{I_\ell}$, and then to v^*, then its total amortized cost could only increase. Thus, in order to upper bound the amortized cost of MARK in P, we divide its cost into two parts which we bound separately.

1. The amortized cost of serving all requests in P and moving to $\mathcal{G}_{I_\ell}$. We denote this cost by $C^{\text{A}}_{\text{MARK}}(P)$.

2. The amortized cost of moving from $\mathcal{G}_{I_\ell}$ to v^*. We denote this cost by $C^{\text{B}}_{\text{MARK}}(P)$.

Note that the second part of the cost is non-existent for the last phase in the epoch, as for such phases $v^* \equiv \mathcal{G}_{I_\ell}$. In particular, it does not occur in the only phase of epoch $\mathcal{E}_0$. We can bound these two parts as follows.

Lemma 3.20 (Phase Lemma). *Let* MB *be any marking-based algorithm and* $P = (I_1, \ldots, I_\ell)$ *be one of its finished phases. Let* K *be the length of* I_j *chunks and* Φ *is the potential function of* MB*. Assume that at the end of* P*,* MB *moves to* $\mathcal{G}_{I_\ell}$*. Then*

$$C_{\text{MB}}(P) + \Delta\Phi(P) \leq O(D/K) \cdot C_{\text{OPT}}(P) + O(D \cdot K) \ .$$

The proof of the Phase Lemma was moved to Section 3.2.3 for clarity reasons. Obviously, since Mark is defined as a marking-based algorithm, we may utilize the lemma above for any finished phase P to get that $C^{\text{A}}_{\text{MARK}}(P) \leq O(D/K) \cdot C_{\text{OPT}}(P) + O(D \cdot K)$. The bound on $C^{\text{B}}_{\text{MARK}}$ can be derived easily, as an analogous bound on $C^{\text{B}}_{\text{DIST}}$.

Lemma 3.21. *For any finished phase* P *of* Mark*, it holds that*

$$C^{\text{B}}_{\text{MARK}}(P) \leq O(D \cdot K) \ .$$

Proof. By Corollary 3.19, a jump candidate v^* lies inside 1-JumpSet of I_ℓ. Thus, the distance between $\mathcal{G}_{I_\ell}$ and v^* is at most $7 \cdot K$. The (non-amortized) cost of moving the page between $\mathcal{G}_{I_\ell}$ and v^* is, therefore, at most $D \cdot (7 \cdot K + 1) = O(D \cdot K)$. An increase in the potential induced by this movement is at most $f \cdot D \cdot 7 \cdot K = O(D \cdot K)$. Thus, the amortized cost, $C^{\text{B}}_{\text{MARK}}(P) = O(D \cdot K)$. ■

For an unfinished phase of any marking-based algorithm we prove a counterpart of the Phase Lemma.

Lemma 3.22 (Auxiliary Phase Lemma). *Let* MB *be any marking-based algorithm and* P *be its unfinished (possibly empty) phase, containing chunks of length* K*. Let* Φ_{B} *be the potential at the beginning of* P*. Then*

$$C_{\text{MB}}(P) \leq \Phi_{\text{B}} + O(D/K) \cdot C_{\text{OPT}}(P) + O(D \cdot K) \ .$$

We postpone the proof to Section 3.2.3, as it is closely related to the proof of the Phase Lemma. Instead, we show how to use these lemmas to prove competitiveness of Mark. As the first step we show a lower bound on Opt's cost, and as the second one an upper bound for Mark in any sequence of phases.

Lemma 3.23. *For any input sequence* $\mathcal{I}$ *consisting of* $k + 1$ *epochs, with the last one possibly unfinished, and any marking-based algorithm* MB*, either*

(i) $C_{\text{OPT}}(\mathcal{I}) = C_{\text{MB}}(\mathcal{I})$, *or*

(ii) $C_{\text{OPT}}(\mathcal{I}) = \Omega((k+1) \cdot D)$.

Proof. Let $\mathcal{I} = (\mathcal{E}_0, \mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_k)$ be the division of input into epochs. We proceed with the case analysis.

1. $k \geq 1$. By Lemma 3.12, for each $i < k$, $C_{\text{OPT}}(\mathcal{E}_i) \geq D$. Thus,

$$C_{\text{OPT}}(\mathcal{I}) \;\geq\; k \cdot D \;=\; \Omega((k+1) \cdot D)$$

2. $k = 0$. In this case $\mathcal{I}$ consists of only one (run-up) epoch $\mathcal{E}_0$. If $\mathcal{E}_0$ is finished then again by Lemma 3.12, $C_{\text{OPT}}(\mathcal{I}) \geq D$. If OPT moves within $\mathcal{E}_0$, then its cost is at least D. Otherwise, OPT remains in v_1 for the whole unfinished epoch $\mathcal{E}_0$. MB also starts in v_1. Its first phase is not finished, and thus it remains in v_1, paying $C_{\text{MB}}(\mathcal{I}) = A_1(\mathcal{E}_0) = C_{\text{OPT}}(\mathcal{I})$.

Hence, in either case the lemma follows. ■

Note that the lemma above would not hold, if the run-up epoch was not present and $K = o(\sqrt{D})$. In that case it could be possible that the input sequence ends right after the first marking chunk for v_1 (and v_1 is marked exactly once in that chunk). Then the cost of the algorithm for moving to another node would be at least D and the cost of OPT would be lower-bounded only by the value of counter A_1, i.e., by $\Theta(K^2) = o(D)$. This justifies the existence of run-up chunks and the run-up epoch.

Lemma 3.24. *Assume that input $\mathcal{I}$ is divided by* MARK *into p finished phases and possibly one not finished. Then*

$$C_{\text{MARK}}(\mathcal{I}) \leq O(D/K) \cdot C_{\text{OPT}}(\mathcal{I}) + (p+1) \cdot O(D \cdot K) \;.$$

Proof. We number finished phases from P_1 to P_p. Potentially, there exists also an unfinished phase P_{p+1} at the end. Let Φ_{B} denote the potential at the beginning of P_{p+1}. Then,

$$\begin{aligned} C_{\text{MARK}}(\mathcal{I}) \;&=\; \sum_{j=1}^{p} C_{\text{MARK}}(P_j) + C_{\text{MARK}}(P_{p+1}) \\ &=\; \sum_{j=1}^{p} \Big[C_{\text{MARK}}(P_j) + \Delta\Phi(P_j) \Big] + \Big(-\Phi_{\text{B}} + C_{\text{MARK}}(P_{p+1}) \Big) \;. \end{aligned}$$

We bound the first summand using Phase Lemma and Lemma 3.21, and the second one using Auxiliary Phase Lemma, getting

$$\begin{aligned} C_{\text{MARK}}(\mathcal{I}) \;&\leq\; \sum_{j=1}^{p} \Big[O\Big(\frac{D}{K}\Big) \cdot C_{\text{OPT}}(P_j) + O(D \cdot K) \Big] + O\Big(\frac{D}{K}\Big) \cdot C_{\text{OPT}}(P_{p+1}) + O(D \cdot K) \\ &\leq\; O(D/K) \cdot C_{\text{OPT}}(\mathcal{I}) + (p+1) \cdot O(D \cdot K) \;. \end{aligned}$$

■

We may combine the two lemmas above, finishing the proof of Mark's $O(n \cdot \sqrt{D})$-competitiveness.

Proof of Theorem 3.14. Let $\mathcal{I}$ be any input sequence. Assume that it consists of $k+1$ epochs, where the last one is possibly unfinished, i.e., $\mathcal{I} = (\mathcal{E}_0, \mathcal{E}_1, \mathcal{E}_2 \ldots, \mathcal{E}_k)$. By Lemma 3.23, either $C_{\text{OPT}}(\mathcal{I}) = C_{\text{MARK}}(\mathcal{I})$, in which case the competitiveness of Mark follows trivially, or $C_{\text{OPT}}(\mathcal{I}) = \Omega((k+1) \cdot D)$. We consider the latter case.

The number of finished phases in $\mathcal{I}$ is, by Lemma 3.15, at most $n \cdot (k+1)$. By Lemma 3.24, we get

$$\begin{aligned} C_{\text{MARK}}(\mathcal{I}) &\leq O(D/K) \cdot C_{\text{OPT}}(\mathcal{I}) + (n \cdot (k+1) + 1) \cdot O(D \cdot K) \\ &= O(D/K) \cdot C_{\text{OPT}}(\mathcal{I}) + O(n \cdot K) \cdot C_{\text{OPT}}(\mathcal{I}) \\ &= O(n \cdot \sqrt{D}) \cdot C_{\text{OPT}}(\mathcal{I}) \ , \end{aligned}$$

which finishes the proof. ■

Combining MARK with other algorithms

Similarly to the case of algorithm Dist, it is possible to combine Mark with the $O(D)$-competitive algorithm Jump and $O(\lambda)$-competitive algorithm Dyn-M, getting a deterministic upper bound of $O(\min\{n \cdot \sqrt{D}, D, \lambda\})$. We show that it is possible to switch between these algorithms on-line, without prior knowledge of the maximum network extent λ. In fact, we present a general scheme that allows to transform any marking-based $\mathcal{R}$-competitive algorithm MB into an $O(\min\{\mathcal{R}, D, \lambda\})$-competitive strategy. We assume that MB uses a potential function $\Phi = 2 \cdot D \cdot L$, where L is the distance between the pages of MB and Opt.

The strategy is the same as in the case of the algorithm Dist. It uses the algorithm Dyn-M, till the maximum distance in the network exceeds D or $\mathcal{R}$, whichever is smaller, and then it switches appropriately either to the algorithm Jump or to MB. However, if it switches to MB, then we modify slightly the definition of MB. Let v_S denote the node holding the algorithm's page at the moment of switch. MB's run-up epoch ends after the chunk which is marking for v_S (and not after the marking chunk for v_1).

Lemma 3.25. *If we combine* Dyn-M *and* Jump *in this way, then the resulting deterministic strategy is strictly* $O(\min\{D, \lambda\})$*-competitive.*

Proof. The proof is identical to the proof of Lemma 3.7 (see page 43) concerning switching between Dyn-M and Jump. In this case, we may omit expected values, as both algorithms are deterministic. ■

Lemma 3.26. *If we combine* Dyn-M *and* MB *in this way, then the resulting deterministic strategy is strictly* $O(\min\{\mathcal{R}, \lambda\})$*-competitive.*

Proof. First, analogously to the proof of Lemma 3.8 (see page 44), we prove that the change in the potential at the moment of switch is negative. If the algorithm has its page at the same node as OPT, then $\Phi_{\text{DYN-M}}(t_1) \geq 0 = \Phi_{\text{MB}}(t_1)$. Otherwise, by Corollary 2.15 (see page 29), $\Phi_{\text{DYN-M}}(t_1) \geq k \cdot (\mathcal{R}+1) \cdot D$ for any chosen constant k. If we pick $k = 2$, then $\Phi_{\text{DYN-M}}(t_1) \geq 2 \cdot D \cdot \mathcal{R} \geq \Phi_{\text{MB}}(t_1)$.

However, in the proof of competitiveness of a marking-based algorithm MB, our argument required more than just bounding the change in the potential. Specifically, we heavily relied on the fact that the cost of OPT in any epoch is either at least $\Omega(D)$ or is equal to the cost of the algorithm (see Lemma 3.23 and Lemma 3.12). If at the moment of switch OPT is at the same node as the algorithm, then all these bounds hold, as the situation in indistinguishable from the normal starting situation. Below we argue that even if OPT and MB are at the different nodes at the switching point, then by choosing appropriately large constant k we can also prove the desired competitiveness.

Assume that after switching to MB, the remaining part of the input sequence (which we denote by $\mathcal{I}_2$) is divided into $\ell + 1$ epochs. In finished regular epochs we can still guarantee that the cost of OPT is at least D, as this bound does not depend on the starting positions of OPT in such an epoch. Thus, since there are at least $\ell - 1$ finished regular epochs, the cost of OPT is at least $(\ell - 1) \cdot D$, which is $\Omega((\ell + 1) \cdot D)$ for $\ell \geq 2$.

If $\ell < 2$, then we have at most 2 epochs. In proof of MB's competitiveness we used the guarantee of Lemma 3.23, claiming that $C_{\text{OPT}}(\mathcal{I}_2) \geq \Omega(2 \cdot D)$, and thus we were able to bound a term of at most $O(\mathcal{R} \cdot 2 \cdot D)$ against the cost of the optimal algorithm. Although we cannot guarantee such a lower bound on C_{OPT}, we may can use the potential gathered by DYN-M instead. It is sufficient that the k is chosen, so that $\Phi_{\text{DYN-M}}(t_1) \geq k \cdot (\mathcal{R}+1) \cdot D$ is greater than the term $O(\mathcal{R} \cdot 2 \cdot D)$. ■

A straightforward corollary follows from the two lemmas above.

Corollary 3.27. *There exists a deterministic strategy, which upon knowing n and D (or at least their constant approximations), is $O(\min\{n \cdot \sqrt{D}, D, \lambda\})$-competitive in the adversarial scenario of the DPM.*

3.2.2 Randomization against oblivious adversary

In this section we show how to use randomization with marking scheme to improve the competitive ratio achieved by algorithm MARK to $O(\sqrt{D \cdot \log n})$ against an oblivious adversary. Before we construct this algorithm, we show that a straightforward randomization of MARK already yields an algorithm, which is $O(\sqrt{D} \cdot \log n)$-competitive against such an adversary.

Regular chunks' length: $K = 2 \cdot \sqrt{D}$

Choosing a jump candidate v^* inside epoch:
v^* is a randomly (uniformly) chosen not yet marked node

Figure 3.10: R-Mark properties

Easy randomization: algorithm R-MARK

Let R-Mark be an algorithm described in Figure 3.10. R-Mark is defined almost exactly as Mark; the only difference is that inside an epoch, for a jump candidate v^*, R-Mark chooses from the set of not yet marked nodes not *any* node but a *random* one.

Theorem 3.28. *The algorithm* R-Mark *is* $O(\sqrt{D} \cdot \log n)$*-competitive against an oblivious adversary in the adversarial model of DPM.*

We note that all the phase lemmas hold, since R-Mark is a marking-based algorithm. In particular, we could divide the cost of R-Mark in a phase P into two parts, $C^{\mathrm{A}}_{\text{R–MARK}}(P)$ and $C^{\mathrm{B}}_{\text{R–MARK}}(P)$, as it was done in the analysis of Mark. Then the worst-case bounds on these costs presented in the Phase Lemma, the Auxiliary Phase Lemma, and the bound on $C^{\mathrm{B}}_{\text{R–MARK}}(P)$ guaranteed by Lemma 3.21 still apply. In effect, Lemma 3.24 holds too, i.e., if an input $\mathcal{I}$ is divided by R-Mark into p finished phases and possibly one not yet finished, then

$$C_{\text{R–MARK}}(\mathcal{I}) \leq O(D/K) \cdot C_{\text{OPT}}(\mathcal{I}) + (p+1) \cdot O(D \cdot K) \ . \tag{3.12}$$

However, for R-Mark we are able to derive a better bound on the (expected) number of phases in one epoch. Recall that by Lemma 3.15 the number of Mark phases in one epoch is not greater than n.

Lemma 3.29. *The expected number of* R-Mark *phases in any epoch (finished or unfinished) is* $O(\log n)$*. The expectation is taken over all random choices made by* R-Mark*.*

Proof. A run-up epoch contains a single phase, and thus the lemma holds trivially.

For a regular epoch, let $\{b_i\}_{i=1}^{n}$ be the nodes in the order they get the first mark in epoch $\mathcal{E}$. Ties are broken arbitrarily. Assume that in a phase P (not the last one) the algorithm is in a node b_k. Then at the end of phase P, it chooses a new node v^* uniformly at random from $n-k$ nodes, i.e., from the set $B_k^+ := \{b_i : k+1 \leq i \leq n\}$. Actually, it might happen that some of the nodes from B_k^+ were also marked at the end of phase P, in which case the algorithm has even fewer than $n-k$ nodes to choose randomly from. Consider

the moment at the end of any phase P. Let T_k be the expected number of remaining phases, provided that k nodes are still unmarked. We have a recursive formula

$$\begin{aligned} T_0 &= 0 \ , \\ T_k &= 1 + \sum_{i=0}^{k-1} \frac{1}{k} \cdot T_i \ , \end{aligned} \tag{3.13}$$

where the second equality follows from the linearity of the expected value. By a few technical transformations (see Section 3.5) we get the following claim.

Claim 3.30. *For any $k \geq 0$, it holds that $T_{k+1} = T_k + \frac{1}{k+1}$.*

Thus, $T_k = H_k = \sum_{i=1}^{k} \frac{1}{i}$. In fact, the expected number of phases can be even smaller than T_k, as some nodes might get marked concurrently, in which case R-MARK has fewer nodes to choose from.

At the beginning of an epoch, R-MARK is in a fixed node. After the first phase finishes, it chooses a jump candidate from a proper subset of V. Therefore, the expected number of phases in any epoch is at most $1 + T_{n-1} \leq 1 + H_{n-1} = O(\log n)$.

Clearly, the argument above works also for an unfinished epoch. ∎

We use the bounds on the expected number of phases to conclude with the competitiveness of R-MARK.

Proof of Theorem 3.28. Take any input sequence $\mathcal{I}$, consisting of $k+1$ epochs, where the last one is possibly unfinished, i.e., $\mathcal{I} = (\mathcal{E}_0, \mathcal{E}_1, \mathcal{E}_2 \ldots, \mathcal{E}_k)$. By Lemma 3.23, either $C_{\text{OPT}}(\mathcal{I}) = C_{\text{R–MARK}}(\mathcal{I})$, in which case the competitiveness of R-MARK follows trivially, or $C_{\text{OPT}}(\mathcal{I}) = \Omega((k+1) \cdot D)$. We consider the latter case. Let p be the number of finished phases in $\mathcal{I}$. By (3.12) we get

$$C_{\text{R–MARK}}(\mathcal{I}) \ \leq \ O(D/K) \cdot C_{\text{OPT}}(\mathcal{I}) + (p+1) \cdot O(D \cdot K) \ .$$

By Lemma 3.29, $\mathbf{E}[p] = O((k+1) \cdot \log n)$, where the expected value is taken over the random choices of the algorithm. Applying this, and using the lower bound on OPT, we get

$$\begin{aligned} \mathbf{E}[C_{\text{R–MARK}}(\mathcal{I})] &\leq O(D/K) \cdot C_{\text{OPT}}(\mathcal{I}) + ((\log n) \cdot (k+1) + 1) \cdot O(D \cdot K) \\ &= O(D/K) \cdot C_{\text{OPT}}(\mathcal{I}) + O(K \cdot \log n) \cdot C_{\text{OPT}}(\mathcal{I}) \\ &= O\left(\sqrt{D} \cdot \log n\right) \cdot C_{\text{OPT}}(\mathcal{I}) \ . \end{aligned} \tag{3.14}$$

This finishes the proof of R-MARK competitiveness. ∎

Balancing algorithm EBM

If we take a closer look at the proof of Theorem 3.28, we note that at the end we have two summands in (3.14), describing the upper bound on $\mathbf{E}[C_{\text{R-MARK}}(\mathcal{I})]$. These are $O(D/K) \cdot C_{\text{OPT}}(\mathcal{I})$ and $O(K \cdot \log n) \cdot C_{\text{OPT}}(\mathcal{I})$, respectively. They would become balanced, if K was reduced to $\Theta(\sqrt{D/\log n})$. However, for the analysis to hold, we should be able to guarantee that the cost in each phase can be bounded as before, and that each epoch consists (in expectation) of at most $O(\log n)$ phases.

At first glance, it is not clear whether the latter is possible at all. Assume for a while that in the marking-based algorithm, we may choose any node for a jump candidate, and we can guarantee that the cost of such jump and serving requests can be amortized against the optimal cost, as it was the case for Mark and R-Mark. This means that we only want to minimize the number of phases in one regular epoch. This simplified task we call *chunk traversing*.

Let T be the number of times each node has to be marked (recall that for Mark and R-Mark, $T = 1$, and if $K = 2 \cdot \sqrt{D/\log n}$, then $T = \log n$) in order for an epoch to finish. Fix any epoch $\mathcal{E}$ consisting of m chunks $(I_1, I_2, \ldots, I_m)$.

Obviously, each node is marked at least T times in $\mathcal{E}$. Moreover, there exists one node, which has less than T marks at the end of I_{m-1}, because epoch $\mathcal{E}$ would have lasted shorter otherwise. In the first phase, any marking-based algorithm is in some fixed node. After that, we have a process consisting of several *iterations*. One iteration involves choosing a jump candidate v^*, and remaining at v^* either till the end of the next marking chunk for v^*, or till the end of an epoch, whichever comes first. Note that the number of phases in epoch is equal to the number of iterations plus one.

For this informal introduction we may assume that during the very first phase of epoch $\mathcal{E}$ exactly one node is marked. Note that this is the wost case for the algorithm. If $T = 1$, then a trivial deterministic algorithm is able to traverse the remaining part of $\mathcal{E}$ in $n - 1$ iterations, by choosing not yet marked nodes for jump candidates. This is exactly what the algorithm Mark does. On the other hand, it is straightforward that for any deterministic algorithm it is possible to construct a sequence of nodes' marking, which requires $n - 1$ iterations.

As we have seen before, chunk traversing can be speeded-up to $O(\log n)$ iterations, if the algorithm is permitted to use randomization. It is not difficult to prove that this result is asymptotically optimal.

If $T = \log n$, a trivial approach may repeat $\log n$ times the approach of R-Mark. In other words, at the beginning it chooses the nodes which are not marked as jump candidates. Then if there are not any, it chooses nodes which are marked at most once. This process continues, till all nodes are marked T times. Unfortunately, one can construct a sequence of node marking, which requires in expectation $\Omega(\log^2 n)$ iterations. An example of such

Regular chunks' length: $K = 2 \cdot \sqrt{D / \log n}$

Choosing a jump candidate v^* inside epoch:
v_i is chosen with probability $\frac{2^{-M_i}}{\sum_{k \in [n]} 2^{-M_k}}$

Figure 3.11: EBM properties

a construction is an epoch consisting of $n \cdot \log n$ chunks; in each chunk exactly one node is marked. The sequence of marked nodes is equal to $(v_1, v_2, \ldots, v_{n-1}, v_n)$, repeated $\log n$ times.

Why did this concept fail to get the number of iterations below $\Omega(\log^2 n)$? If there are only a few nodes with a certain number of marks, then this approach concentrates too much on these nodes. Hence, for reducing the number of rounds to $o(\log^2 n)$, we have to resort to assigning different probabilities to different jump candidates, so that the nodes with low number of marks are preferred, but nodes with high number of marks are also taken into consideration. In particular, we consider an algorithm called Exponential Balancing Marking (EBM), which chooses a node v_i for a jump candidate with a probability inversely proportional to 2^{M_i}, where M_i is the number of marks v_i has.

The marking-based algorithm EBM is formally described on Figure 3.11. We introduce an additional notation for the marking scheme. If $\mathcal{S}$ is any subsequence, then by $M_i(\mathcal{S})$ and $M_i'(\mathcal{S})$ we denote the number of marks v_i has before $\mathcal{S}$ and after $\mathcal{S}$, respectively. We also define $\Delta M_i(\mathcal{S}) = M_i'(\mathcal{S}) - M_i(\mathcal{S})$. It appears that we can reasonably bound the number of EBM's jumps within one epoch.

Lemma 3.31. *The expected number of* EBM *phases in one epoch (also unfinished one) is* $O(\log n)$. *The expectation is taken over all random choices made by* EBM.

Proof. The lemma follows trivially for run-up epochs.

For regular epochs, we define a *value of a node* after any chunk I as $n \cdot 2^{-M_i'(I)}$. The *total value* after I is defined as the sum of nodes' values, i.e., $\mathcal{W}_I := \sum_{i \in [n]} n \cdot 2^{-M_i'(I)}$. We make two key observations. First, $\mathcal{W}_I$ is monotonically non-increasing within $\mathcal{E}$. Second, $\mathcal{W}_I \leq n^2$ for any chunk $I \in \mathcal{E}$, and $\mathcal{W}_{I_{m-1}} \geq 1$ (because, as mentioned earlier, after I_{m-1} there is at least one node having less than $\log n$ marks). We show that, with probability at least $1/2$, one iteration reduces the total value by a constant factor or ends the whole epoch. We call such an iteration *successful*.

After I, the choice of the jump candidate v^* determines where the next phase ends: either at the end of the first marking chunk for v^*, or at the end of I_m, if v^* is not marked in the remaining part of $\mathcal{E}$. This chunk we call *stopping* for v^*. We sort the nodes in the order induced by their stopping chunks, obtaining a sorted sequence $v_{i_1}, v_{i_2}, \ldots, v_{i_n}$. Let

$p_{i_1}, p_{i_2}, \ldots, p_{i_n}$ be the probabilities of choosing these nodes as jump candidates. Let j be the smallest index for which $\sum_{k=1}^{j} p_{i_k} \geq 1/2$, and I' be the stopping chunk for v_{i_j}. Since j is the smallest index with this property, it follows immediately that, with probability $\sum_{k=j}^{n} p_{i_k} \geq 1/2$, Ebm chooses one of $v_{i_j}, v_{i_{j+1}}, \ldots, v_{i_n}$ for a jump candidate. Any such choice guarantees that the phase lasts at least to the end of I'. If $I' = I_m$ then this iteration ends epoch $\mathcal{E}$, and the proof follows. Otherwise, note that between the end of I and the end of I', nodes $v_{i_1}, v_{i_2}, \ldots, v_{i_j}$ are marked at least once. Since probabilities p_{i_k} are directly proportional to the corresponding values of nodes and $\sum_{k=1}^{j} p_{i_k} \geq 1/2$, these nodes' values constituted at least one half of the total value $\mathcal{W}_I$. By marking them once, one half of their values (and thus at least 1/4 of the total value) was removed. Thus, $\mathcal{W}_{I'} \leq \frac{3}{4} \cdot \mathcal{W}_I$.

We need at most $\log_{4/3} n^2$ successful iterations to end the epoch or reduce the total value from n^2 to 1. Therefore, in expectation at most $2 \cdot \log_{4/3} n^2 = O(\log n)$ iterations suffice to either finish the epoch, or to end after the chunk I_{m-1}. In the latter case we have at most one additional phase containing only chunk I_m. Since the number of phases is equal to the number of iterations plus one, the lemma follows. Clearly, for an unfinished epoch the same bound holds. ■

Analysis of EBM phase

Since Ebm is a marking-based algorithm, the scheme of choosing jump candidates is coherent with the gravity center based approach in the sense guaranteed by Corollary 3.18 In particular, it implies the following.

Corollary 3.32. *For any chunk I and a node v_i, v_i belongs to the $(\Delta M_i(I) + 1)$-JumpSet at the end of I.*

We note that Ebm may choose nodes that already have $\log n$ or more marks. Thus, we cannot bound the cost of transporting the page in the worst case, as we did for bounding the $C^{\mathrm{B}}_{\mathrm{MARK}}$ or $C^{\mathrm{B}}_{\mathrm{R-MARK}}$. Instead, we show that even if sometimes Ebm moves to the nodes which are far away from the gravity centers, it moves there only occasionally.

Similarly to the proof of competitiveness of Mark or R-Mark, we divide the cost in any phase P, consisting of ℓ chunks $(I_1, I_2, \ldots, I_\ell)$, into three parts.

1. The amortized cost of serving all requests in P and moving to $\mathcal{G}_{I_\ell}$. We denote this cost by $C^{\mathrm{A}}_{\mathrm{EBM}}(P)$.
2. The amortized cost of moving from $\mathcal{G}_{I_\ell}$ to the boundary of the 1-JumpSet. We denote this cost by $C^{\mathrm{B}}_{\mathrm{EBM}}(P)$.
3. The amortized cost of moving from the boundary of the 1-JumpSet to a randomly chosen jump candidate v^*. We denote this cost by $C^{\mathrm{C}}_{\mathrm{EBM}}(P)$.

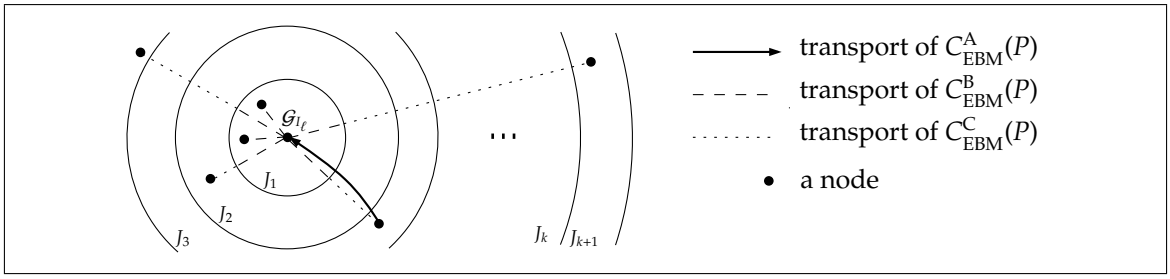

Figure 3.12: Transports at the end of phase $P = (I_1, \ldots, I_\ell)$

Note that for the last phase in a finished epoch, parts $C^{B}_{EBM}(P)$ and $C^{C}_{EBM}(P)$ do not exist, as in that case EBM moves only to the gravity center. This conceptually divides the movement of the page to the jump candidate v^* into three parts, called transports. These transports are schematically depicted in Figure 3.12.

For each epoch $\mathcal{E}$ consisting of p phases, we separately bound the expected values of these three parts. We define the same potential function Φ as for MARK and R-MARK. The bound on C^{A}_{EBM} follows directly from the Phase Lemma.

Corollary 3.33. *For any finished phase P, it holds that*

$$C^{A}_{EBM}(P) \leq O(D/K) \cdot C_{OPT}(P) + O(D \cdot K) \ .$$

On the other hand, since C^{B}_{EBM} describes a transport within the first jump set, the bound for this part is identical as the analogous bound on MARK presented by Lemma 3.21.

Corollary 3.34. *For any phase P, it holds that*

$$C^{B}_{EBM}(P) \leq O(D \cdot K) \ .$$

We note that in the two corollaries above we bound the random variables $C^{A}_{EBM}(P)$, $C^{B}_{EBM}(P)$ in the worst case, not only their expected values. On the other hand, we cannot hope for a reasonable worst case bound on $C^{C}_{EBM}(P)$, as EBM may jump very far away from the gravity center. Moreover, even if we bound the expected value of $C^{C}_{EBM}(P)$ for any single phase P, we may not combine it with the logarithmic bound on the expected number of phases in one epoch, as both bounds hold only on expectation and may depend on each other.

Therefore, we strive for constructing a bound for $\mathbf{E}[C^{C}_{EBM}(P)]$ that depends on the number of marks at the beginning and at the end of phase P. We show how, for any epoch $\mathcal{E}$, this yields a bound on $\mathbf{E}[C^{C}_{EBM}(\mathcal{E})]$ independently of the number of phases epoch $\mathcal{E}$ consists of.

Lemma 3.35. *For any phase P, it holds that*

$$\mathbf{E}[C^{\mathrm{C}}_{\mathrm{EBM}}(P)] \;\le\; 21 \cdot D \cdot K \cdot \log\left(\frac{\sum_{i\in[n]} 2^{-M_i(P)}}{\sum_{i\in[n]} 2^{-M'_i(P)}}\right) \;.$$

Proof. We denote the last chunk of P by I_ℓ. By Corollary 3.32, at the end of I_ℓ, each node v_i lies inside $(\Delta M_i(I_\ell)+1)$-JumpSet, and thus inside $(\Delta M_i(P)+1)$-JumpSet. Intuitively, the marking system is coherent with the approach of choosing nodes close to the gravity centers — if a node is far away from the gravity center, it has many marks and the probability that Ebm moves to such node is exponentially small.

Formally, if we transport the page to v_i, the $C^{\mathrm{C}}_{\mathrm{EBM}}(P)$ part of the cost reflects only the cost of moving the page from the boundary of 1-JumpSet to a node within $(\Delta M_i(P)+1)$-JumpSet, i.e., the cost at most $D\cdot(7\cdot K\cdot\Delta M_i(P))$. We do not consider the constant overhead for the communication, since it was already taken into account in the $C^{\mathrm{B}}_{\mathrm{EBM}}(P)$ part of the cost. As the corresponding change in the potential is at most twice this cost, the amortized cost of such a movement is at most $21\cdot K\cdot D\cdot\Delta M_i(P)$. Thus, the expected amortized cost of moving the page to v^* (taken over all possible random choices of v^*) is

$$\mathbf{E}[C^{\mathrm{C}}_{\mathrm{EBM}}(P)] \;\le\; \sum_{i\in[n]} \frac{2^{-M'_i(P)}}{\sum_{k\in[n]} 2^{-M'_k(P)}} \cdot \Delta M_i(P)\cdot 21\cdot D\cdot K \;.$$

To bound this, we use the following technical claim, which follows from the Jensen's Inequality [HLP88] (see Appendix A.2), and is proven at the end of this chapter.

Claim 3.36. *Fix any sequences $\{a_i\}_{i=1}^n$, $\{b_i\}_{i=1}^n$, such that $1\le a_i\le b_i$ for all i. Then*

$$\frac{\sum_i 2^{-b_i}\cdot(b_i-a_i)}{\sum_i 2^{-b_i}} \le \log\frac{\sum_i 2^{-a_i}}{\sum_i 2^{-b_i}} \;.$$

By applying the claim with $b_i = M'_i(P)$ and $a_i = M_i(P)$, we immediately get the lemma. ■

Lemma 3.37. *For any epoch $\mathcal{E} = (P_1, P_2 \ldots P_p)$, it holds that*

$$\mathbf{E}\left[\sum_{P_j\in\mathcal{E}} C^{\mathrm{C}}_{\mathrm{EBM}}(P_j)\right] = O(D\cdot K\cdot\log n) \;.$$

Proof. Note that $C^{\mathrm{C}}_{\mathrm{EBM}}(P_p) = 0$. Thus, it is sufficient to prove that $\sum_{j=1}^{p-1}\mathbf{E}[C^{\mathrm{C}}_{\mathrm{EBM}}(P_j)] = O(K\cdot D\cdot\log n)$. Utilizing Lemma 3.35,

$$\begin{aligned}\frac{\sum_{j=1}^{p-1}\mathbf{E}[C^{\mathrm{C}}_{\mathrm{EBM}}(P_j)]}{21\cdot K\cdot D} &\le \log\left(\prod_{j=1}^{p-1}\frac{\sum_{i\in[n]}2^{-M_i(P_j)}}{\sum_{i\in[n]}2^{-M'_i(P_j)}}\right)\\ &= \log\left(\frac{\sum_{i\in[n]}2^{-M_i(P_1)}}{\sum_{i\in[n]}2^{-M'_i(P_{p-1})}}\right) \;.\end{aligned}$$

Since $M_i(P_1) = 0$ for all i, the numerator in the last term above is equal to n. There exists a node v_i, which has less that $\log n$ marks at the end of P_{p-1}, otherwise epoch $\mathcal{E}$ would be finished earlier. Thus, the corresponding denominator is at least $1/n$, and we get

$$\sum_{j=1}^{p-1} \mathbf{E}[C^{\mathrm{C}}_{\mathrm{EBM}}(P_j)] \;\leq\; 21 \cdot D \cdot K \cdot \log \frac{n}{1/n} = O(D \cdot K \cdot \log n) \;.$$

This finishes the proof. ∎

Competitiveness of EBM

Finally, we combine the lemmas above to obtain the following bound on the amortized cost in any epoch.

Lemma 3.38. *For a finished epoch $\mathcal{E}$, it holds that*

$$\mathbf{E}[C_{\mathrm{EBM}}(\mathcal{E}) + \Delta\Phi(\mathcal{E})] \leq O(D/K) \cdot C_{\mathrm{OPT}}(\mathcal{E}) + O(D \cdot K \cdot \log n) \;.$$

Proof. Any finished run-up epoch consist only of one phase P, and cost of EBM in P consists only of $C^{\mathrm{A}}_{\mathrm{EBM}}(P)$ part, as the algorithm moves to the gravity center at the end of P. Thus, by Lemma 3.33, we get $C_{\mathrm{EBM}}(\mathcal{E}) \leq O(D/K) \cdot C_{\mathrm{OPT}}(\mathcal{E}) + O(D \cdot K)$.

For a finished regular epoch $\mathcal{E}$, let $\mathcal{E}$ consists of p phases, $(P_1, P_2, \ldots, P_p)$. We have

$$\begin{aligned}
\mathbf{E}[C_{\mathrm{EBM}}(\mathcal{E}) + \Delta\Phi(\mathcal{E})] \;&=\; \mathbf{E}\left[\sum_{j=1}^{p} \left(C_{\mathrm{EBM}}(P_j) + \Delta\Phi(P_j)\right)\right] \\
&\leq\; \mathbf{E}\left[\sum_{j=1}^{p} \left(C^{\mathrm{A}}_{\mathrm{EBM}}(P_j) + C^{\mathrm{B}}_{\mathrm{EBM}}(P_j)\right)\right] + \mathbf{E}\left[\sum_{j=1}^{p} C^{\mathrm{C}}_{\mathrm{EBM}}(P_j)\right] \;.
\end{aligned}$$

Applying Lemma 3.33, Lemma 3.34, and Lemma 3.37, we get

$$\begin{aligned}
\mathbf{E}[C_{\mathrm{EBM}}(\mathcal{E}) + \Delta\Phi(\mathcal{E})] \;&\leq\; \mathbf{E}\left[\sum_{j=1}^{p} \left(O(D/K) \cdot C_{\mathrm{OPT}}(P_j) + O(D \cdot K)\right)\right] \;+\; O(D \cdot K \cdot \log n) \\
&\leq\; O(D/K) \cdot C_{\mathrm{OPT}}(\mathcal{E}) + \mathbf{E}\left[p \cdot O(D \cdot K)\right] \;+\; O(D \cdot K \cdot \log n) \;.
\end{aligned}$$

Note that by Lemma 3.31 the expected number of phases is $O(\log n)$, and the term $O(D \cdot K)$ occurring in the expected value is a constant (not a random variable). Hence, we finally get a bound

$$\mathbf{E}[C_{\mathrm{EBM}}(\mathcal{E}) + \Delta\Phi(\mathcal{E})] \;\leq\; O(D/K) \cdot C_{\mathrm{OPT}}(\mathcal{E}) + O(D \cdot K \cdot \log n) \;.$$

This finishes the proof for the regular epochs. ∎

Lemma 3.39. *For an unfinished epoch $\mathcal{E}$, if Φ_B is the potential at the beginning of $\mathcal{E}$, then*

$$\mathbf{E}[C_{\text{EBM}}(\mathcal{E}) - \Phi_B] \leq O(D/K) \cdot C_{\text{OPT}}(\mathcal{E}) + O(D \cdot K \cdot \log n) \ .$$

Proof. For any unfinished run-up epoch, we may apply the Auxiliary Phase Lemma to its only (unfinished) phase to get $C_{\text{EBM}}(\mathcal{E}) \leq O(D/K) \cdot C_{\text{OPT}}(\mathcal{E}) + O(D \cdot K)$.

For an unfinished regular epoch $\mathcal{E}$, let $\mathcal{E}$ consist of $p+1$ phases $(P_1, P_2, \ldots, P_{p+1})$, where the last one is unfinished (P_{p+1} might be also empty). Let $\Phi_{B(p)}$ be the potential at the beginning of P_{p+1}. Then,

$$\mathbf{E}[C_{\text{EBM}}(\mathcal{E}) - \Phi_B] \ = \ \mathbf{E}\left[\sum_{j=1}^{p} \left(C_{\text{EBM}}(P_j) + \Delta\Phi(P_j)\right)\right] + \mathbf{E}\left[C_{\text{EBM}}(P_{p+1}) - \Phi_{B(p)}\right]$$

The first summand can be bounded exactly as in the previous lemma, as the bound on p holds also for unfinished phases. The second summand may be bounded again by the Auxiliary Phase Lemma, by $O(D/K) \cdot C_{\text{OPT}}(P_{p+1}) + O(D \cdot K)$. Hence, in total, $\mathbf{E}[C_{\text{EBM}}(\mathcal{E}) - \Phi_B] \leq O(D/K) \cdot C_{\text{OPT}}(\mathcal{E}) + O(D \cdot K \cdot \log n)$ ■

By combining the two lemmas above we get the proof of Ebm competitiveness.

Theorem 3.40. *The algorithm* Ebm *is $O(\sqrt{D \cdot \log n})$-competitive against an oblivious adversary in the adversarial model of the DPM.*

Proof. Consider any input $\mathcal{I}$ consisting of $m+1$ epochs $(\mathcal{E}_0, \mathcal{E}_1, \ldots, \mathcal{E}_m)$ (the last one is possibly unfinished). Summing the amortized cost over all epochs, by Lemma 3.38 and Lemma 3.39,

$$\mathbf{E}[C_{\text{EBM}}(\mathcal{I})] \leq O(D/K) \cdot C_{\text{OPT}}(\mathcal{I}) + (m+1) \cdot O(D \cdot K \cdot \log n) \ .$$

By Lemma 3.23, either $C_{\text{OPT}}(\mathcal{I}) = C_{\text{EBM}}(\mathcal{I})$, in which case competitiveness follows trivially, or $C_{\text{OPT}}(\mathcal{I}) = \Omega((m+1) \cdot D)$. In the latter case,

$$\begin{aligned} \mathbf{E}[C_{\text{EBM}}(\mathcal{I})] \ &\leq \ O(D/K + K \cdot \log n) \cdot C_{\text{OPT}}(\mathcal{I}) \\ &= \ O\left(\sqrt{D \cdot \log n}\right) \cdot C_{\text{OPT}}(\mathcal{I}) \ . \end{aligned}$$

Thus, Ebm is $O(\sqrt{D \cdot \log n})$-competitive. ■

Combining EBM with other algorithms

It is again possible to combine Ebm with Jump and Dyn-M. Since the proofs we presented in Lemma 3.25 and Lemma 3.26 work for any marking-based algorithm, we get the following corollary.

Corollary 3.41. *There exists a randomized strategy, which upon knowing n and D (or at least their constant approximations), is $O(\min\{\sqrt{D \cdot \log n}, D, \lambda\})$-competitive against an oblivious adversary in the adversarial scenario of the DPM.*

3.2.3 Proofs of Phase Lemmas

In this section we prove Lemma 3.20 (the Phase Lemma) and Lemma 3.22 (the Auxiliary Phase Lemma). Let MB be any marking-based algorithm working in chunks of length K. Let P be any (finished or unfinished) phase of MB. We assume that P consists of ℓ intervals $I_1, I_2, \ldots, I_\ell$. Let v_P denote the node in which algorithm MB has its page in whole phase P. We assume that at the end of P, MB moves to $\mathcal{G}_{I_\ell}$. We note that v_P is marked not earlier than in I_ℓ, although if P is the last phase, it might be not marked at all within P.

We divide the cost of MB in P into two parts: the cost incurred by $(I_1, I_2, \ldots, I_{\ell-1})$ and the cost incurred by I_ℓ. If P is a finished phase, then the latter is not only the cost of serving requests, but includes the cost of movement to $\mathcal{G}_{I_\ell}$. On the other hand, if P is an unfinished phase, then I_ℓ possibly has less then K time steps.

A bound for all chunks but the last one

Lemma 3.42. *Let P be any phase consisting of ℓ chunks $(I_1, I_2, \ldots, I_\ell)$ and let P' be the first $\ell - 1$ chunks of P. Then*

$$C_{\mathrm{MB}}(P') + \Delta\Phi(P') \leq O(D/K) \cdot C_{\mathrm{OPT}}(P') + O(D \cdot K) \ .$$

Proof. First, we note that the cost of serving requests within P', $C_{\mathrm{MB}}(P') = A_P(P') < \frac{1}{4} \cdot K^2 = O(D \cdot K)$, because otherwise v_P would be marked within P', and the phase would last shorter. Thus, it remains to bound the change in potential, $\Delta\Phi(P')$.

Let $s = |P'|$; we number the time steps within P' from 1 to s. Intuitively, since the cost of serving requests is small, we know that the total sum of distances between v_P and requests is even smaller, i.e., $\sum_{t=1}^{s} d_t(v_P, \sigma_t) \leq \frac{1}{4} \cdot K^2$. We call a request *close*, if it was issued at the distance at most $K/2$ from v_P. Otherwise, we call a request *far*. Clearly, at most $K/2$ requests from P' are far.

We denote the distance between v_P and P_{OPT}, the node holding OPT's page in step t, by L_t. The intuition behind the proof is described below. If the potential at the end of P' is large, then OPT's page is far away from MB's page. We show that even considering the possible adversarial changes of the network topology and possible movements of OPT's page, there are sufficiently many steps t in which the distance $L_t \geq K$. In such steps the close requests are at the distance of at least $K - K/2 = K/2$ from P_{OPT}, and they incur a high cost on the optimal offline algorithm. Therefore, we can amortize the change in the potential against the increase of OPT's cost.

Formally, the distance between Mark's and Opt's pages can increase only due to the adversarial changes to the network, in which case $L_{t+1} - L_t \leq 1$ or due to the jump of Opt. Let J denote the total distance across which the optimal offline algorithm moved[4] within P'. Then in any k steps of P' the increase in the distance between the node holding Opt's page and v_P is at most $k + J$. This observation we call a *bounded change condition*.

Consider L_s, the distance between v_P and P_{OPT} at the end of P'. We can derive two bounds on $\Delta\Phi(P')$. $\Delta\Phi(P') \leq f \cdot D \cdot L_s$, since the increase in the potential is at most its final value, and $\Delta\Phi(P') \leq f \cdot D \cdot (s + J)$, which follows from the bounded change condition. Therefore, if L_s is small, i.e., $L_s \leq J + \frac{3}{2} \cdot K$, then we immediately get

$$\begin{aligned} \Delta\Phi(P') &\leq f \cdot D \cdot L_s \\ &\leq f \cdot D \cdot J + \frac{3}{2} \cdot D \cdot K \\ &\leq O(1) \cdot C_{OPT}(P') + O(D \cdot K) \ . \end{aligned}$$

Otherwise, $L_s \geq J + \frac{3}{2} \cdot K$, and we consider two cases.

1. $J + K \leq L_s \leq s + J + K$.

   Consider any of the last $L_s - J - K$ steps of P', and denote it by t_0. It follows from the bounded change condition that the increase in the potential between step t_0 and step s is at most $(s - t_0) + J \leq (L_s - J - K) + J = L_s - K$. Thus, $L_{t_0} \geq K$.

2. $L_s \geq s + J + K$.

   In the same way we can prove that for each step t_0 in P', $L_{t_0} \geq K$.

In either case there are at least $\min\{s, L_s - J - K\}$ steps in which $L_t \geq K$. At most $K/2$ of these steps contain a far requests, and thus at least $\min\{s, L_s - J - K\} - K/2 \geq \min\{s + J + K, L_s\} - J - \frac{3}{2} \cdot K \geq 0$ of these steps contain a close request. Since in each such step (with a close request) the distance between v_P and P_{OPT} is at least K, and the distance between v_P and the requesting node is at most $K/2$, it follows from the triangle inequality that the distance between P_{OPT} and the request is at least $K/2$. Therefore, summing over all such steps we get

$$C_{OPT}(P') \geq \frac{K}{2} \cdot \left(\min\{s + J + K, L_s\} - J - \frac{3}{2} \cdot K \right) \ .$$

[4] We count here only the jumps of the algorithm, not movements incurred by the adversarial changes in the network.

Thus,

$$\begin{aligned}\Delta\Phi(P') &\leq f \cdot D \cdot \min\{s + J, L_s\} \\ &\leq f \cdot D \cdot \min\{s + J + K, L_s\} \\ &= f \cdot D \cdot \left(\min\{s + J + K, L_s\} - J - \frac{3}{2} \cdot K\right) + f \cdot D \cdot J + \frac{3}{2} \cdot f \cdot D \cdot K \\ &= O(D/K) \cdot C_{\text{OPT}}(P') + O(1) \cdot C_{\text{OPT}}(P') + O(D \cdot K) \ .\end{aligned}$$

Finally, we get that $C_{\text{MB}}(P') + \Delta\Phi(P') = O(D/K) \cdot C_{\text{OPT}}(P') + O(D \cdot K)$. ■

A bound for the last chunk

The remaining part of this subsection, concerning bounding the cost in I_ℓ, is inspired by the proof of 7-competitiveness of MOVE-TO-MIN algorithm [ABF93a]. However, in our proof the chunk lengths are shorter than D, and additionally we have to take into account the movement of the nodes, which makes the proof more entangled.

Before we bound the amortized cost of MB in I_ℓ, we construct a lower bound on OPT's cost in this chunk. Let K_0 be the length of I_ℓ. As $D \geq 4$, $K_0 \leq K \leq 2 \cdot \sqrt{D} \leq D$. We number all time steps within I_ℓ from 1 to K_0.

By a_{t-1} and a_t we denote the position of OPT, respectively at the beginning and at the end of the t-th step. In particular $a_0 = P_{\text{OPT}}(0)$ and $a_{K_0} = P'_{\text{OPT}}(K_0)$. In step $t \in [1, \ldots, K_0]$, OPT pays $c_t(a_{t-1}, \sigma_t)$ for serving a request and $D \cdot c_t(a_{t-1}, a_t)$ for moving the page. Thus,

$$C_{\text{OPT}}(I_\ell) = \sum_{t=1}^{K_0} [c_t(a_{t-1}, \sigma_t) + D \cdot c_t(a_{t-1}, a_t)] \ . \tag{3.15}$$

An example of OPT's behavior in several steps is given in Figure 3.13.

Lemma 3.43. *For the chunk I_ℓ of length $K_0 \leq D$ and any time step $T \in [1, \ldots, K_0]$, it holds that*

$$C_{\text{OPT}}(I_\ell) + O(K_0^2) \geq \sum_{t=1}^{K_0} d_t(a_T, \sigma_t) \ .$$

Before we prove the lemma above, note that if we replace term $\sum_{t=1}^{K_0} d_t(a_T, \sigma_t)$ by $\sum_{t=1}^{K_0} c_t(a_T, \sigma_t)$ (in fact these terms differ by at most K_0) and we neglect the constant $O(K_0^2)$, we can infer that by remaining at one node a_i throughout the whole chunk, OPT could lower its cost in a single chunk. This may even lead to a false conclusion: "Why does not OPT remain at one node, if this incurs a lower cost?". But OPT has to serve the whole input sequence optimally, not just minimize its cost in a single chunk. In particular, if the sets of the requesting nodes in two consecutive chunks are disjoint, and OPT remains at one requesting node in both chunks, then between the chunks it would have to move.

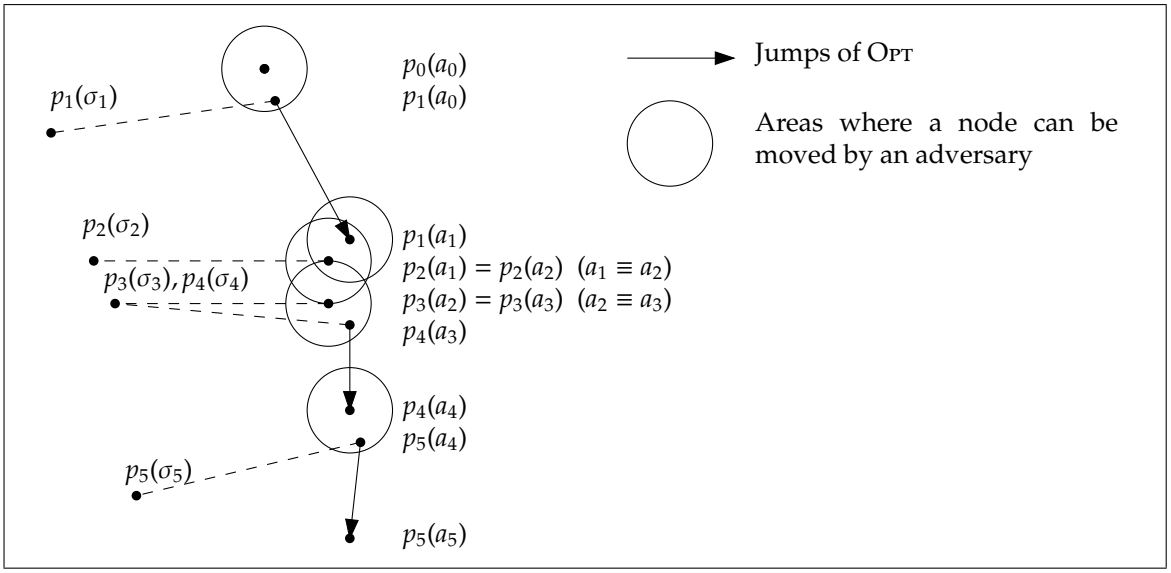

Figure 3.13: Illustration of Opt's cost

Proof of Lemma 3.43. It follows from the triangle inequality that

$$\sum_{t=1}^{K_0} d_t(a_T, \sigma_t) \leq \sum_{t=1}^{K_0} d_t(a_{t-1}, \sigma_t) + \sum_{t=1}^{K_0} d_t(a_T, a_{t-1}) \ . \tag{3.16}$$

The first summand of (3.16) is at most $\sum_{t=1}^{K_0} c_t(a_{t-1}, \sigma_t)$, the term which appears in (3.15), the definition of Opt's cost for serving requests in I_ℓ. Thus, we can concentrate on the second summand. Assume that we could prove that for any step $t \in [1, \ldots, K_0]$ and any $0 \leq i \leq j \leq T$ it holds that

$$d_t(a_i, a_j) \leq \sum_{k=1}^{K_0} d_k(a_{k-1}, a_k) + O(K_0) \ . \tag{3.17}$$

Then it would follow from (3.16) that

$$\begin{aligned}
\sum_{t=1}^{K_0} d_t(a_T, \sigma_t) &\leq \sum_{t=1}^{K_0} d_t(a_{t-1}, \sigma_t) + \sum_{t=1}^{K_0} \left[\sum_{k=1}^{K_0} d_k(a_{k-1}, a_k) + O(K_0) \right] \\
&\leq \sum_{t=1}^{K_0} c_t(a_{t-1}, \sigma_t) + K_0 \cdot \sum_{k=1}^{K_0} d_k(a_{k-1}, a_k) + O(K_0^2) \\
&\leq \textsc{Opt}(I_\ell) + O(K_0^2) \ ,
\end{aligned}$$

where in the last inequality we used $K_0 \leq D$. Therefore, it remains to prove (3.17).

Fix any $t \in [1, \ldots, K_0]$ and any $0 \le i \le j \le T$. Then using the fact that the adversary is $\frac{1}{2}$-restricted we get

$$\begin{aligned} d_t(a_i, a_j) &= d(p_t(a_i), p_t(a_j)) \\ &\le 2 \cdot (K_0/2) + d(p_i(a_i), p_j(a_j)) \ . \end{aligned}$$

The situation is exemplified on Figure 3.13. Using the triangle inequality we get

$$\begin{aligned} d_t(a_i, a_j) &\le K_0 + \sum_{k=i}^{j-1} d(p_k(a_k), p_{k+1}(a_{k+1})) \\ &\le K_0 + \sum_{k=i}^{j-1} (d(p_k(a_k), p_{k+1}(a_k)) + d(p_{k+1}(a_k), p_{k+1}(a_{k+1}))) \\ &\le K_0 + \sum_{k=i}^{j-1} \frac{1}{2} + \sum_{k=i}^{j-1} d_{k+1}(a_k, a_{k+1}) \\ &\le O(K_0) + \sum_{k=0}^{K_0-1} d_{k+1}(a_k, a_{k+1}) \ . \end{aligned}$$

This proves (3.17), and thus completes the proof of Lemma 3.43. ■

Now using the lower bound on OPT presented above, we can bound the cost of serving the requests in the chunk, i.e., $A_P(I_\ell)$ can be either paid from the potential at the beginning of the chunk or amortized against the cost of the optimal algorithm.

Lemma 3.44. *Let Φ_B denote the cost at the beginning of the chunk I_ℓ. Then*

$$A_P(I_\ell) - \Phi_B/2 \le C_{OPT}(I_\ell) + O(D)$$

Proof. We have $\Phi_B = f \cdot D \cdot d_0(v_P, a_0)$. Utilizing the triangle inequality, Lemma 3.43, and $K_0 \le 2 \cdot \sqrt{D} \le D$, we get

$$\begin{aligned} A_P(I_\ell) - \Phi_B/2 &= \sum_{t=1}^{K_0} c_t(v_P, \sigma_t) - D \cdot d_0(v_P, a_0) \\ &\le \sum_{t=1}^{K_0} (1 + d_t(v_P, a_0) + d_t(a_0, \sigma_t)) - D \cdot d_0(v_P, a_0) \\ &\le K_0 + \sum_{t=1}^{K_0} (K_0 + d_0(v_P, a_0)) + \sum_{t=1}^{K_0} d_t(a_0, \sigma_t) - D \cdot d_0(v_P, a_0) \\ &\le K_0 + K_0^2 + \sum_{t=1}^{K_0} d_0(v_P, a_0) + \left(C_{OPT}(I_\ell) + O(K_0^2)\right) - D \cdot d_0(v_P, a_0) \\ &\le C_{OPT}(I_\ell) + O(D) \ , \end{aligned}$$

which finishes the proof. ■

We can use the lemma above to prove that for finished phases the amortized cost of MB in the last chunk is also bounded.

Lemma 3.45. *If I_ℓ is a last chunk of a finished phase of* MB*, then*

$$C_{\mathrm{MB}}(I_\ell) + \Delta\Phi(I_\ell) \leq O(D/K) \cdot C_{\mathrm{OPT}}(I_\ell) + O(D \cdot K) \ .$$

Proof. In this proof we use the triangle inequality, and the inequality $c_t(x, y) \leq 1 + d_t(x, y)$. Since the chunk is finished, its length is equal to K. The adversary is $\frac{1}{2}$-restricted, and therefore the distance between two nodes can change by at most K within I_ℓ. Let $\Phi_{\mathrm{E}} = f \cdot D \cdot d_K(\mathcal{G}_{I_\ell}, a_K)$ denote the potential at the end of the chunk I_ℓ, after MB jumps to $\mathcal{G}_{I_\ell}$. The amortized cost of MB in I_ℓ is equal to

$$C_{\mathrm{MB}}(I_\ell) + \Delta\Phi(I_\ell) \ = \ A_{\mathrm{P}}(I_\ell) + D \cdot c_K(v_{\mathrm{P}}, \mathcal{G}_{I_\ell}) + \Phi_{\mathrm{E}} - \Phi_{\mathrm{B}} \ .$$

By Lemma 3.44 we get

$$\begin{aligned} C_{\mathrm{MB}}(I_\ell) + \Delta\Phi(I_\ell) &\leq C_{\mathrm{OPT}}(I_\ell) + O(D) + D \cdot d_K(v_{\mathrm{P}}, \mathcal{G}_{I_\ell}) + D + \Phi_{\mathrm{E}} - \Phi_{\mathrm{B}}/2 \\ &\leq C_{\mathrm{OPT}}(I_\ell) + D \cdot d_K(v_{\mathrm{P}}, \mathcal{G}_{I_\ell}) + 2 \cdot D \cdot d_K(\mathcal{G}_{I_\ell}, a_K) - D \cdot d_0(v_{\mathrm{P}}, a_0) + O(D) \\ &\leq C_{\mathrm{OPT}}(I_\ell) + D \cdot d_K(v_{\mathrm{P}}, \mathcal{G}_{I_\ell}) + 2 \cdot D \cdot d_K(\mathcal{G}_{I_\ell}, a_K) - D \cdot d_K(v_{\mathrm{P}}, a_0) + O(D \cdot K) \\ &\leq C_{\mathrm{OPT}}(I_\ell) + D \cdot d_K(\mathcal{G}_{I_\ell}, a_0) + 2 \cdot D \cdot d_K(\mathcal{G}_{I_\ell}, a_K) + O(D \cdot K) \ . \end{aligned}$$

Thus, it is sufficient to prove that for any $0 \leq T \leq K$ it holds that

$$D \cdot d_K(a_T, \mathcal{G}_{I_\ell}) \ \leq \ O(D/K) \cdot C_{\mathrm{OPT}}(I_\ell) + O(D \cdot K) \ . \tag{3.18}$$

To prove the inequality above, note that by the triangle inequality we have

$$D \cdot d_K(a_T, \mathcal{G}_{I_\ell}) \ \leq \ \frac{D}{K} \cdot \sum_{t=1}^{K} \left(d_K(a_T, \sigma_t) + d_K(\sigma_t, \mathcal{G}_{I_\ell}) \right) \ .$$

Since $\mathcal{G}_{I_\ell}$ is a gravity center of I_ℓ, $\sum_{t=1}^{K} d_K(\mathcal{G}_{I_\ell}, \sigma_t) \leq \sum_{t=1}^{K} d_K(a_T, \sigma_t)$, and thus

$$\begin{aligned} D \cdot d_K(a_T, \mathcal{G}_{I_\ell}) \ &\leq \ 2 \cdot \frac{D}{K} \cdot \sum_{t=1}^{K} d_K(a_T, \sigma_t) \\ &\leq \ 2 \cdot \frac{D}{K} \cdot \sum_{t=1}^{K} (K + d_t(a_T, \sigma_t)) \ . \end{aligned}$$

By Lemma 3.43, we finally get

$$D \cdot d_K(a_T, \mathcal{G}_{I_\ell}) \ \leq \ O(D/K) \cdot C_{\mathrm{OPT}}(I_\ell) + O(D \cdot K) \ ,$$

which proves (3.18), and finishes the proof. ■

Phase Lemmas

The proofs of phase lemmas are straightforward consequences of the lemmas above.

Proof of Lemma 3.20 (Crucial Phase Lemma). We want to bound amortized MB's cost in a finished phase $P = (I_1, \ldots, I_\ell)$. By Lemma 3.42,

$$C_{\text{MB}}((I_1, \ldots, I_{\ell-1})) + \Delta\Phi((I_1, \ldots, I_{\ell-1})) \;\leq\; O(D/K) \cdot C_{\text{OPT}}((I_1, \ldots, I_{\ell-1})) + O(D \cdot K) \;,$$

and by Lemma 3.45,

$$C_{\text{MB}}(I_\ell) + \Delta\Phi(I_\ell) \;\leq\; O(D/K) \cdot C_{\text{OPT}}(I_\ell) + O(D \cdot K) \;.$$

Summing up these inequalities, we get the proof of the Crucial Phase Lemma. ■

Proof of Lemma 3.22 (Auxiliary Phase Lemma). We want to bound amortized MB's cost in an unfinished phase $P = (I_1, \ldots, I_\ell)$. By Φ_{B} and $\Phi_{\text{B(I)}}$, we denote the potential at the beginning of P and at the beginning of I_ℓ, respectively. By Lemma 3.42,

$$C_{\text{MB}}((I_1, \ldots, I_{\ell-1})) + \Delta\Phi((I_1, \ldots, I_{\ell-1})) \;\leq\; O(D/K) \cdot C_{\text{OPT}}((I_1, \ldots, I_{\ell-1})) + O(D \cdot K) \;,$$

and by Lemma 3.44

$$C_{\text{MB}}(I_\ell) - \Phi_{\text{B(I)}} \;\leq\; C_{\text{OPT}}(I_\ell) + O(D) \;.$$

Summing up these inequalities, we get

$$C_{\text{MB}}(P) - \Phi_{\text{B}} \;\leq\; C_{\text{OPT}}(P) + O(D \cdot K) \;.$$

■

3.3 Lower bounds

In this section we prove a matching lower bound of $\Omega(\min\{n \cdot \sqrt{D}, D, \lambda\})$ for any randomized algorithm against an adaptive-online adversary. This implies optimality of the randomized memoryless strategy against an adaptive-online adversary and the optimality of the deterministic strategy presented in this chapter. For oblivious adversaries, we prove a lower bound of $\Omega(\min\{\sqrt{D \cdot \log n}, D^{2/3}, \lambda\})$, which is up to a constant factor tight for $D \geq \log^3 n$.

3.3.1 Lower bound against adaptive-online adversary

Let λ be the maximum possible extent of the network. In this subsection we prove the following theorem.

Theorem 3.46. *Consider any randomized, c-competitive algorithm for the adversarial scenario of the DPM problem playing against an adaptive-online adversary. Then* $c = \Omega(\min\{n \cdot \sqrt{D}, D, \lambda\})$.

We assume that $D \geq 16$, otherwise the theorem trivially holds. We may also safely assume that $\sqrt{D}$ is an integer, otherwise we could use $\lfloor \sqrt{D} \rfloor$ instead and lose only a constant factor in the analysis. Furthermore, we assume that $n \geq 3$ and $\lambda \geq \sqrt{D}$. For the case where $n = 2$ or $\lambda < \sqrt{D}$ we can use Theorem 2.9 concerning oblivious adversaries (see Section 2.3.2 on page 25), since an $\Omega(\min\{\sqrt{D}, \lambda\})$ lower bound guaranteed by this theorem holds also in our stronger, adaptive case.

Let $\mathcal{R} = \frac{1}{14} \cdot \min\{n \cdot \sqrt{D}, D, \lambda\}$. Under the assumptions above, it is sufficient to show that for any randomized algorithm Alg, and for any ℓ, there exists an adaptive-online adversary Adv, which adaptively creates an input sequence $\mathcal{I}$ of length at least ℓ, such that

$$\mathbf{E}[C_{\text{ALG}}(\mathcal{I}) - \mathcal{R} \cdot C_{\text{ADV}}(\mathcal{I})] \geq 0 \ , \tag{3.19}$$

where C_{ADV} is the cost of the answering part of adversary Adv. We also show that such a sequence incurs a cost of at least $\Omega(\ell \cdot D)$ on Alg.

The core of this proof is to show that there exists a class $\mathcal{A}_\ell$ of adaptive-online adversaries, such that for any online *deterministic* algorithm Det, most of the adversaries from this class incur a high competitive ratio on Det. Then we use a standard argument to show (non-constructively) that for any online *randomized* algorithm Alg, there exists an adaptive-online adversary which incurs a high competitive ratio on Alg.

Construction of class $\mathcal{A}_\ell$

Before we precisely define class $\mathcal{A}_\ell$, we try to give an informal description of a reasonable adaptive adversary for the DPM problem.

The adversary will follow the general construction of the proof of Theorem 2.9, i.e., it will try to move P_{ALG}, the node holding algorithm's page, away from the requests. Since the adversary is adaptive, it knows exactly which node to move away from the others. The other nodes will be grouped in the same place, and the requests will be given at an arbitrary node from this group. Again, since the adversary is adaptive, it does not have to stop the expanding part (the one, in which P_{ALG} is moved apart) at some fixed point of the time, but it may continue to increase this distance, till P_{ALG} moves to any other node. After the jump, nodes are contracted and a new phase (consisting of expanding and contracting part) begins. For this informal description we assume that $\lambda = \infty$.

Once again we explore the paradigm of OPT catching game. Even if the positions of the ADV's page are determined afterwards, it has to reside at some node. Thus, if ALG tries to jump, consecutively, to all the nodes, then at some point it catches ADV. Then ADV has to either move to another node, or its page will be moved apart, together with P_{ALG}, from the requests. If the adversary is adaptive-offline, it may decide afterwards, which one of these two options costs less. This would extremely simplify the analysis. However, since we are primarily interested in developing a lower bound against an adaptive-online adversary, we have to resort to an attempt to fool the algorithm.

Assume that the adversary moves its page at the beginning to a random position. (In the formal proof, we get rid of the randomization and replace it by the average argument). Assume that a randomized algorithm ALG moves at some point to the node holding ADV page. Note that this occurs, in expectation, after $n/2$ jumps. If ADV moves immediately to another node, then a simple algorithm ALG, which moves in each step to another node, might incur one jump of ADV for each n steps, trivially achieving competitive ratio of $O(n)$. On the other hand, if ADV waits in this node till ALG moves its page, then the expanding part may last for D steps. The costs of both ALG and ADV in this phase are $\Theta(D^2)$ then, and again such an event occurs once for n phases.

To balance these two heuristics, if $P_{\text{ALG}} \equiv P_{\text{ADV}}$ at the beginning of a phase, ADV may allow the expanding part to last $\sqrt{D}$ steps. If ALG jumps after $X < \sqrt{D}$ steps, then ADV does nothing, as the cost incurred on it is $\Theta(X^2)$, and the ALG's cost in this phase is $\Omega(D \cdot X)$. If ALG does not move after $\sqrt{D}$ steps, then ADV initiates the contracting part itself, and after this phase it jumps to a randomly chosen node. Note that in this case ADV may also explicitly "tell" the algorithm, "Yes, you have just caught me". There are two corollaries. First, if ALG wants to force ADV to move, it has to wait at least $\sqrt{D}$ steps in the expanding part, before moving the page. Then the cost incurred on ALG is at least $D \cdot \sqrt{D}$. Second, the event of catching ADV happens in expectation once for $n/2$ steps. Thus, the ADV's cost between its two jumps is approximately $O(D)$, and the cost of ALG is in expectation $\Omega(n \cdot D \cdot \sqrt{D})$, which yields a lower bound of $\Omega(n \cdot \sqrt{D})$ on the competitive ratio. These intuitions are formalized below.

The class $\mathcal{A}_\ell$ is constructed as follows. It consists of n^ℓ adversaries, each identified by a finite sequence r, consisting of ℓ integers from the set $[n]$, i.e., $r = (r_1, r_2, \ldots, r_\ell)$ and $r_i \in [n]$. We define an adaptive-online adversary ADV_r as follows. At the beginning all nodes are in the same point of $\mathcal{X}$, and the page of the adversary, as well as the page of the algorithm, is at the node v_1. In the first step the adversary issues a request in v_1 and moves to v_{r_0}. Next steps are divided into phases. Each phase consists of two parts: an *expanding part* and a *contracting part*.

Let P_{ALG} and P_{ADV_r} be the nodes at which the algorithm and the adversary, respectively, have their pages. During the whole request sequence, the requests are issued at v_1, unless

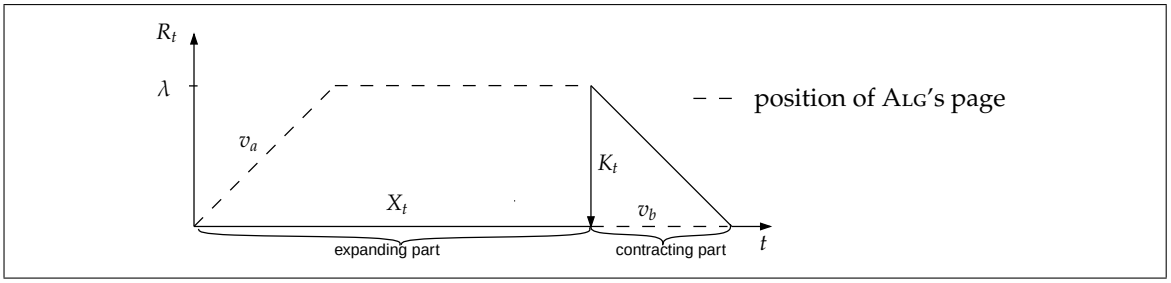

Figure 3.14: k-th (long) phase

$P_{\text{ALG}} \equiv v_1$, in which case the request are issued at v_2.

In each step of the expanding part the adversary increases the distance between P_{ALG} and the rest of the nodes, i.e., in the t-th step of the expanding part we have $d(P_{\text{ALG}}, v_i) = \min\{t-1, \lambda\}$ for any vertex $v_i \not\equiv P_{\text{ALG}}$. The distances between any other pairs of nodes remain equal to 0.

If $P_{\text{ALG}} \not\equiv P_{\text{ADV}_r}$ at the beginning of the phase, then the expanding part continues till the algorithm decides to move its page to a new vertex, say from v_a to v_b.[5] Let X_k be the duration of the expanding part in the k-th phase. Then $K_k := \min\{X_k - 1, \lambda\}$ is the distance to which algorithm's page at the node v_a was moved away from the rest of the nodes. Then a contracting part comes, in which the node v_a is moved closer to the other nodes. Formally, the contracting part of phase k takes $K_k + 1$ time steps and in the t-th time step of this part $d_t(v_a, v_i) = K_k - t + 1$, for all $v_i \not\equiv v_a$. The distances between any other pairs of nodes remain equal to 0. Illustration of such a phase is given in Figure 3.14, where R_t denotes the distance between v_a and the rest of the nodes. Note that in terms of the notation from the proof of Theorem 2.9, the expanding part contain these time steps, which we previously called the expanding and the main part.

However, if at the beginning of the phase $P_{\text{ALG}} \equiv P_{\text{ADV}_r}$ then the expanding part lasts at most $\sqrt{D}$ steps. If the algorithm did not move its page in $\sqrt{D}$ steps, then the adversary starts the contracting part by itself. If it happens, the algorithm can be sure that the adversary is at the same node as the algorithm, and we say that the algorithm has *detected the adversary's position*. In either case, the contracting part takes the same number of time steps as the expanding part did. After the contracting part, if the adversary's position has been detected, the adversary moves its page to the next place from the sequence r.

Additionally we call a phase *long*, if its expanding part is at least $\sqrt{D}$, and *short* otherwise. The adversary maintains a counter of the long phases since it has moved its page last time, and if this counter exceeds $\lfloor n/2 \rfloor$, the adversary moves its page, in the last

[5] Note that if the algorithm never jumps, the expanding part would last forever. After some point of time the algorithm would pay at least $\lambda + 1$ for step, while the adversary cost would still be 1. This would immediately result in $C_{\text{ALG}} \geq \Omega(\lambda) \cdot C_{\text{ADV}_r}$.

step of the $\lfloor n/2 \rfloor$-th long phase, to the next place from the sequence r. This additional ending condition will simplify the analysis later.

Moving a page by the adversary at the end of some phases partitions the sequence of phases into *epochs*. Formally, an epoch begins with a jump of ADV and contains all the phases till the next jump of ADV. Therefore, in epoch i the position of the adversary is fixed and equal to r_i.[6] The game between the adversary ADV_r and the algorithm ends after ℓ epochs, i.e., after a phase after which the adversary would normally move its page to the next position from the sequence r, but it has already used all elements of r.

Note, that for a given r the behavior of the adversary ADV_r is completely deterministic and depends only on the behavior of the algorithm ALG.

It is possible to prove that if we choose r uniformly at random (e.g., by choosing each element of r sequence uniformly at random from the set $[n]$), then for any deterministic algorithm ALG, the expected cost of the adversary is smaller by a factor of $\mathcal{R}$ than the cost of this algorithm.

Lemma 3.47. *Let ℓ be any positive integer. Fix any deterministic algorithm* DET. *If we choose r uniformly at random from $[n]^\ell$, then*

$$\mathbf{E}_r[C_{\text{DET}} - \mathcal{R} \cdot C_{\text{ADV}_r}] \geq 0 \ ,$$

where ADV_r are the adversaries from the class $\mathcal{A}_\ell$. The expected value is taken over the possible choices of r.

Before we prove this lemma, we argue how the lower bound for any randomized algorithm follows from it.

Proof of Theorem 3.46. Let ALG be any but fixed online randomized algorithm. Since we do not impose any memory restrictions on ALG, ALG is equivalent to some probability distribution $\{p_k\}_k$ over the set of all possible deterministic algorithms $\{\text{DET}_k\}_k$. See [BE98, chapter 6] for a discussion on the equivalence of behavioral and mixed strategies of the randomized online algorithms.

Fix any ℓ. We construct a matrix M, with rows indexed with all possible deterministic algorithms DET_k, and columns indexed with all possible ℓ-element sequences of integers from the set $[n]$, i.e., columns are elements of $[n]^\ell$. We set the values in this matrix to

$$M_{k,r} := C_{\text{DET}_k} - \mathcal{R} \cdot C_{\text{ADV}_r} \ . \tag{3.20}$$

[6] In the very first step ADV and the request are at v_1. This incurs no cost on ADV. At the end of the first step ADV jumps and the first epoch begins.

It follows from Lemma 3.47 that the average over each row is at least 0, i.e., for any k, $\sum_r \frac{1}{n^\ell} \cdot M_{k,r} \geq 0$. Thus, it is also the case for the sum over all deterministic algorithms, weighted with p_k.

$$\sum_k \sum_r p_k \cdot \frac{1}{n^\ell} \cdot M_{k,r} \geq 0 \tag{3.21}$$

Then there exists a column r in which the average (weighted with p_k) is non-negative. Otherwise, if for each column r holds $\sum_k p_k \cdot M_{k,r} < 0$, then this inequality summed over all possible $r \in [n]^\ell$ would contradict (3.21). In other words, there exists a deterministic adversary $\textsc{Adv}_r$ such that Inequality 3.19 is fulfilled, i.e.,

$$\mathbf{E}[C_{\text{ALG}} - \mathcal{R} \cdot C_{\text{ADV}_r}] = \sum_k p_k \cdot M_{k,r} \geq 0 \ .$$

The expected value is taken over all possible random choices of the algorithm.

Since we proved it for any sequence of length ℓ, we may construct input sequences that are arbitrarily long. Note that an algorithm may only catch $\textsc{Adv}_r$ (and force $\textsc{Adv}_r$ to move to the next element from sequence r) in a long phase. In such a phase $\textsc{Alg}$ pays at least $\Omega(D)$ for serving the requests. Thus, we may construct a sequence which incurs an arbitrarily high cost on $\textsc{Alg}$, which implies that the cost of $\textsc{Alg}$ cannot be hidden in a constant A occurring in the definition (1.1) of the competitive ratio. Hence, the competitive ratio of any randomized algorithm against adaptive-online adversary is at least $\mathcal{R} = \Omega(\min\{n \cdot \sqrt{D}, D, \lambda\})$, which finishes the proof. ■

Proof for random adversary

Fix any deterministic algorithm $\textsc{Det}$. In this part we prove that if choose a random adversary $\textsc{Adv}_r \in \mathcal{A}_\ell$, then in expectation the $\textsc{Det}$'s cost is high in comparison to the $\textsc{Adv}_r$'s cost. This will lead to a proof of Lemma 3.47.

We consider the i-th epoch. Note that the behavior of the algorithm can depend only on the past events, such as detecting the adversary's position in previous epochs. In particular, till the last phase (in which it may potentially detect the adversary's position), it does not depend on the adversary's position r_i in this epoch.

Assume for a while that the algorithm never detects the position of the adversary. Then the adversary ends the epoch after the $\lfloor n/2 \rfloor$-th long phase. We number all phases (within the current epoch) starting from 1. Let P_j be a node in which the algorithm has its page during the expanding part of the j-th phase, and let X_j be the duration of the expanding part of phase j. Then the sequence of $(P_j, X_j)_j$ completely characterizes the algorithm's behavior in this epoch (whether it detects the adversary's position or not), and we call it a *canonic sequence* in epoch i.

Note that the canonic sequence can be computed independently of the information, where the adversary has its page. When this sequence is fixed, the performance of the algorithm depends only on r_i, the position of the adversary in epoch i. In fact, the behavior of the algorithm follows the canonic sequence, either till the phase in which the algorithm detects the position of the adversary (i.e., till the end of the first long phase j such that $P_j = r_i$), or till the end of the whole canonic sequence if the algorithm does not detect ADV's position.

To prove Lemma 3.47, it is sufficient to show that for any epoch i, any choice of the algorithm's canonic sequence leads in expectation to $\mathbf{E}_{r_i}[C_{\mathrm{DET}}(r_i)] \geq \mathcal{R} \cdot \mathbf{E}_{r_i}[C_{\mathrm{ADV}}(r_i)]$, where r_i is the position of the adversary's page. By $C_{\mathrm{DET}}(r_i)$ and $C_{\mathrm{ADV}}(r_i)$ we denote the costs of the algorithm and the adversary, respectively, in the current epoch i, under the assumption that during the epoch the adversary is in node r_i. Since r_i are picked uniformly from the set $[n]$, it is equivalent to proving

$$\sum_{r_i \in [n]} C_{\mathrm{DET}}(r_i) \geq \mathcal{R} \cdot \sum_{r_i \in [n]} C_{\mathrm{ADV}}(r_i) \ . \tag{3.22}$$

To prove this inequality, we introduce a couple of definitions first. We define $\mathcal{S}$ as the set of indices of all short phases from the canonic sequence, and $\mathcal{L}$ as the set of indices of long phases. From the definition of canonic sequence follows that $|\mathcal{L}| = \lfloor n/2 \rfloor$. Note that for some choices of r_i, if the canonic sequence contains a long phase with $P_j \equiv r_i$ (detection of the adversary), the epoch ends earlier and does not contain all phases from $\mathcal{S}$ or $\mathcal{L}$. However, there exists at least $\lceil n/2 \rceil \geq n/2$ choices of r_i, such that the epoch does contain all the phases from canonic sequence.

In the following we divide the cost $\sum_{r_i \in [n]} C_{\mathrm{DET}}(r_i)$ into $\sum_{r_i \in [n]} C^{\mathcal{S}}_{\mathrm{DET}}(r_i)$ and $\sum_{r_i \in [n]} C^{\mathcal{L}}_{\mathrm{DET}}(r_i)$, the cost incurred in short and long phases, respectively. We divide $\sum_{r_i \in [n]} C_{\mathrm{ADV}}(r_i)$ in the same way. Moreover, the adversary moves exactly once in the epoch, and for the analysis we assume that this cost was incurred in a long phase. In this proof by *total cost* we mean the sum of costs for epoch i for all possible choices of r_i, e.g., the total cost of ADV is $\sum_{r_i \in [n]} C_{\mathrm{ADV}}(r_i)$. We now show how to relate the total costs of ADV and DET.

Lemma 3.48. *For any epoch i, and any corresponding canonic sequence $(P_j, X_j)_j$ of* DET*, it holds that*

$$\sum_{r_i \in [n]} C^{\mathcal{S}}_{\mathrm{DET}}(r_i) \geq \mathcal{R} \cdot \sum_{r_i \in [n]} C^{\mathcal{S}}_{\mathrm{ADV}}(r_i) \ .$$

Proof. In short phases ADV does not move, and its cost of serving requests might be bounded as follows. Fix any short phase j. Since $\lambda \geq \sqrt{D}$, P_{ALG} is moved away during

the whole expanding part, and thus $K_j = X_j - 1$. Therefore, the adversary pays

$$\begin{aligned} 2 \cdot \sum_{k=0}^{K_j} (k+1) \;&=\; X_j^2 + X_j && \text{if } r_i \equiv P_j \;, \\ 2 \cdot X_j && \text{if } r_i \not\equiv P_j \;. \end{aligned}$$

Thus, the total cost in short phases is

$$\begin{aligned} \sum_{r_i \in [n]} C^{\mathcal{S}}_{\mathrm{ADV}}(r_i) \;&\le\; \sum_{j \in \mathcal{S}} \left(X_j^2 + X_j + (n-1) \cdot 2 \cdot X_j \right) \\ &\le\; \sum_{j \in \mathcal{S}} \left(\sqrt{D} \cdot X_j + 2 \cdot n \cdot X_j \right) \;, \end{aligned} \tag{3.23}$$

where the last inequality follows from $X_j \le \sqrt{D}$ for any short phase

Since $\lambda \ge \sqrt{D}$, the cost of moving the Det's page after a short phase j is equal to $D \cdot (K_j + 1) = D \cdot X_j$. We forgive Det the cost of serving requests in short phases. As mentioned before, there are at least $n/2$ choices of r_i, for which the epoch contains all the phases from the canonic sequence. Thus, the total cost of Det is at least

$$\sum_{r_i \in [n]} C^{\mathcal{S}}_{\mathrm{DET}}(r_i) \;\ge\; \frac{n}{2} \cdot \sum_{j \in \mathcal{S}} D \cdot X_j \;. \tag{3.24}$$

Combining bounds (3.23) and (3.24), we obtain

$$\begin{aligned} \mathcal{R} \cdot \sum_{r_i \in [n]} C^{\mathcal{S}}_{\mathrm{ADV}}(r_i) \;&\le\; \mathcal{R} \cdot \sum_{j \in \mathcal{S}} \left(\sqrt{D} \cdot X_j + 2 \cdot n \cdot X_j \right) \\ &\le\; \sum_{j \in \mathcal{S}} \left(\frac{n}{14} \cdot \sqrt{D} \cdot \sqrt{D} \cdot X_j + \frac{n}{7} \cdot D \cdot X_j \right) \\ &\le\; \sum_{r_i \in [n]} C^{\mathcal{S}}_{\mathrm{DET}}(r_i) \;, \end{aligned}$$

which proves the lemma. ∎

Lemma 3.49. *For any epoch i, and any corresponding canonic sequence $(P_j, X_j)_j$ of* Det*, it holds that*

$$\sum_{r_i \in [n]} C^{\mathcal{L}}_{\mathrm{DET}}(r_i) \ge \mathcal{R} \cdot \sum_{r_i \in [n]} C^{\mathcal{L}}_{\mathrm{ADV}}(r_i) \;.$$

Proof. The easiest part is to bound the total cost of Adv's movement. Since each of n adversaries move the page exactly once, along distance 0, it incurs a total cost of $n \cdot D$ Since $3 \cdot |\mathcal{L}| = 3 \cdot \lfloor n/2 \rfloor \ge n$, the total cost of movement is at most $\sum_{j \in \mathcal{L}} 3 \cdot D$.

On the other hand, if we fix any long phase j then the cost of serving requests in this phase is equal to

$$\begin{cases} 2 \cdot \sum_{k=0}^{\sqrt{D}} (k+1) \;=\; D + 3\sqrt{D} + 2 \;<\; 2 \cdot D & \text{if } r_i \equiv P_j \;, \\ 2 \cdot X_j & \text{if } r_i \not\equiv P_j \;. \end{cases}$$

The first formula follows, since ADV begins a contracting part after at most $\sqrt{D}$ steps. Therefore, the total cost of serving requests in such phases is bounded by $\sum_{j \in \mathcal{L}} (2 \cdot D + n \cdot 2 \cdot X_j)$. Adding the cost of moving the page, we get

$$\sum_{r_i \in [n]} C^{\mathcal{L}}_{\mathrm{ADV}}(r_i) \;\leq\; \sum_{j \in \mathcal{L}} \Big(2 \cdot n \cdot X_j + 5 \cdot D \Big) \;. \tag{3.25}$$

To bound DET's cost, we consider any long phase j. If the length of its expanding part X_j is not greater than λ, then the cost of moving the page at the end of the expanding part is $D \cdot X_j$ (and again we forgive DET the cost of serving requests). Otherwise, the cost of serving requests in the expanding part is $\sum_{k=0}^{\lambda}(k+1) + (X_j - \lambda - 1) \cdot (\lambda + 1) \geq X_j \cdot \lambda - \frac{\lambda^2}{2} \geq \frac{1}{2} \cdot \lambda \cdot X_j$, and the cost of moving the page afterwards is $(\lambda + 1) \cdot D \geq D \cdot \sqrt{D}$. As mentioned above there are $\lceil n/2 \rceil \geq n/2$ choices of r_i, which cause all the long phases from the canonic sequence to really occur.

$$\sum_{r_i \in [n]} C^{\mathcal{L}}_{\mathrm{DET}}(r_i) \;\geq\; \frac{n}{2} \cdot \left(\sum_{j \in \mathcal{L} \text{ and } X_j \leq \lambda} D \cdot X_j + \sum_{j \in \mathcal{L} \text{ and } X_j > \lambda} \left(\frac{1}{2} \cdot \lambda \cdot X_j + D \cdot \sqrt{D} \right) \right) \;. \tag{3.26}$$

By combining bounds (3.25) and (3.26), we get

$$\begin{aligned} \mathcal{R} \cdot \sum_{r_i \in [n]} C^{\mathcal{L}}_{\mathrm{ADV}}(r_i) \;&\leq\; \mathcal{R} \cdot \sum_{\substack{j \in \mathcal{L} \\ X_j \leq \lambda}} \Big(2 \cdot n \cdot X_j + 5 \cdot D \Big) + \mathcal{R} \cdot \sum_{\substack{j \in \mathcal{L} \\ X_j > \lambda}} \Big(2 \cdot n \cdot X_j + 5 \cdot D \Big) \\ &\leq\; \sum_{\substack{j \in \mathcal{L} \\ X_j \leq \lambda}} \left(\frac{n}{7} \cdot D \cdot X_j + \frac{5 \cdot n}{14} \cdot D \cdot \sqrt{D} \right) + \sum_{\substack{j \in \mathcal{L} \\ X_j > \lambda}} \left(\frac{n}{7} \cdot \lambda \cdot X_j + \frac{5 \cdot n}{14} \cdot D \cdot \sqrt{D} \right) \end{aligned}$$

As j ranges over long phases only, in the first summand we may bound $\frac{5 \cdot n}{14} \cdot D \cdot \sqrt{D}$ by $\frac{5 \cdot n}{14} \cdot D \cdot X_j$. Thus,

$$\begin{aligned} \mathcal{R} \cdot \sum_{r_i \in [n]} C^{\mathcal{L}}_{\mathrm{ADV}}(r_i) \;&\leq\; \sum_{\substack{j \in \mathcal{L} \\ X_j \leq \lambda}} \frac{n}{2} \cdot D \cdot X_j + \sum_{\substack{j \in \mathcal{L} \\ X_j > \lambda}} \left(\frac{n}{7} \cdot \lambda \cdot X_j + \frac{5 \cdot n}{14} \cdot D \cdot \sqrt{D} \right) \\ &\leq\; \sum_{r_i \in [n]} C^{\mathcal{L}}_{\mathrm{DET}}(r_i) \;, \end{aligned}$$

which finishes the proof of the lemma. ∎

We may combine these two lemmas to prove Lemma 3.47.

Proof of Lemma 3.47. By adding the guarantees of Lemma 3.48 and Lemma 3.49, we get that for any epoch i, Inequality 3.22 holds, i.e., $\sum_{r_i \in [n]} C_{\text{DET}}(r_i) \geq \mathcal{R} \cdot \sum_{r_i \in [n]} C_{\text{ADV}}(r_i)$. By dividing both sides by n, we get

$$\mathbf{E}_{r_i}[C_{\text{DET}}(r_i)] \geq \mathcal{R} \cdot \mathbf{E}_{r_i}[C_{\text{ADV}}(r_i)] \ .$$

By summing it over all ℓ epochs in the input sequence and using linearity of expectation, we get the lemma. ■

3.3.2 Lower bound against oblivious adversary

In this section we show that no randomized online algorithm can achieve a competitive ratio better than $\Omega(\min\{\sqrt{D \cdot \log n}, D^{2/3}, \lambda\})$ even against an oblivious adversary. We present a generic scheme, based on the proof of $\Omega(\min\{\sqrt{D}, \lambda\})$ lower bound (see Theorem 2.9 in Section 2.3.2). This scheme will be parameterized by four variables p, B_{main}, B_{exp}, and B_{fix}. Later, we show how to choose these variables to achieve the lower bound of $\Omega(\min\{\sqrt{D \cdot \log n}, \lambda\})$ and a lower bound of $\Omega(\min\{D^{2/3}, \lambda\})$, which, combined, yield the result.

We construct a probability distribution over inputs and prove that each deterministic algorithm (even knowing this distribution) has a high competitive ratio. Then we apply the Yao min-max principle (see Section 2.3.2) to show that the same lower bound holds for any randomized algorithm against an oblivious adversary.

We assume that n is a power of 2. If it is not the case, then the adversary can give requests only in the first $2^{\lfloor \log n \rfloor} = \Theta(n)$ nodes, placing the other nodes exactly at the same point of space $\mathcal{X}$ as v_1. Then, for any algorithm Alg that uses these additional nodes, there exists an algorithm Alg′ which uses v_1 instead and has a cost at most as large as Alg. We may also assume that $\lambda \geq \sqrt{D}$. Otherwise, we could use the Theorem 2.9, which would yield the lower bound of $\Omega(\lambda)$.

In the following we number the nodes from 0 to $n-1$. Hence, we may assume that the binary representation of each node's number has length $\log n$. We call this the *binary representation of a node*. For all $1 \leq i \leq \log n$, let V_i^0 be the set of all the nodes, whose binary representation has the i-th bit set to 0. V_i^1 is the set of all remaining nodes, i.e., the ones, whose binary representation has the i-th bit set to 1.

We divide time into epochs, each epoch containing $p \leq \log n$ phases, each of length $B_{\text{main}} + 2 \cdot \min\{B_{\text{exp}}, \lambda\}$. In the i-th phase we divide the set V into two sets V_i^0 and V_i^1. In this phase the positions of all nodes from set V_i^0 are the same; the same holds for nodes from V_i^1. Therefore, in step t of phase i, the distance between the sets V_i^0 and V_i^1 is well defined and we call it $R_i(t)$.

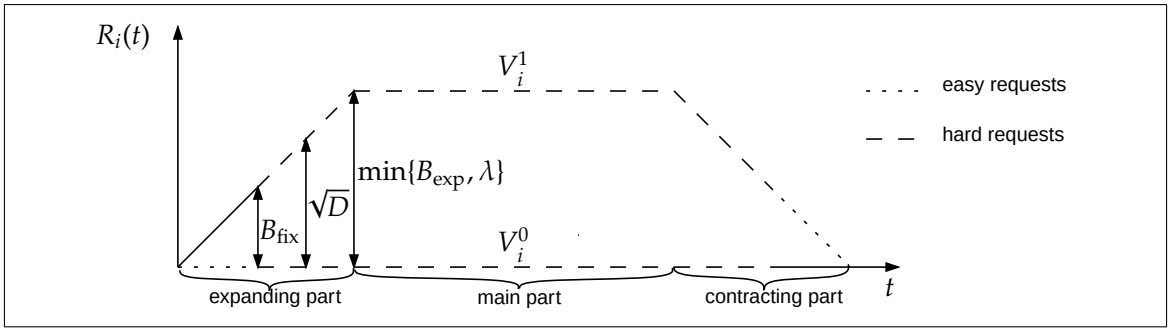

Figure 3.15: Lower bound: The i-th phase of an epoch

First, we describe the movement of nodes in the i-th phase. The situation is depicted in Figure 3.15. Each phase consists of an *expanding part*, which lasts for $\min\{B_{\text{exp}}, \lambda\}$ steps, in which the sets V_i^0 and V_i^1 are moved apart, a *main part*, lasting for B_{main} steps, and a *contracting part* also of length B_{exp}, in which the sets V_i^0 and V_i^1 are brought closer. In the t-th step of the expanding part the distance R_i is set to $t - 1$. In all the steps of the main part the distance between sets V_i^0 and V_i^1 is exactly $\min\{B_{\text{exp}}, \lambda\}$. Finally, in the j-th step of the contracting part, R_i is equal to $B_{\text{exp}} - t$. Hence, the nodes' movement is fixed deterministically.

The first $B_{\text{fix}} \leq \sqrt{D}$ requests in the expanding part of a phase are always given at v_0. The last B_{fix} requests in the contracting part are always given at $v_{2^{i-1}}$. We call these requests *easy*. The requests in the remaining steps of a phase are given in one node — with probability 1/2 all are given at v_0 (belonging to V_i^0), and with probability 1/2 all are given at node $v_{2^{i-1}}$ (belonging to V_i^1). We call these requests *hard*, and we call the set, whose node issued these requests, a *requesting set*. Note that the main part is a (usually proper) subset of a sequence containing all the hard requests.

Before we continue with the analysis, we note that if we set $B_{\text{exp}} = B_{\text{fix}} = \sqrt{D}$, $B_{\text{main}} = D$, and $p = 1$, then we get the construction of the $\Omega(\min\{\sqrt{D}, \lambda\})$ lower bound, presented in Section 2.3.2 on page 22.

Lemma 3.50. *Fix any T. Let $\mathcal{I}$ be a randomly generated input sequence of T epochs, each constructed as described above. Then for any deterministic algorithm* DET *for the DPM problem,*

$$\mathbf{E}_{\mathcal{I}}[C_{\text{DET}}(\mathcal{I})] \geq \Omega\left(\frac{\min\{D \cdot B_{\text{fix}},\ B_{\text{exp}} \cdot B_{\text{main}},\ \lambda \cdot B_{\text{main}}\}}{B_{\text{main}} + B_{\text{exp}} + B_{\text{fix}}^2 + D/p}\right) \cdot C_{\text{OPT}}(\mathcal{I}) \ .$$

Proof. Consider any randomly generated p phases of epoch $\mathcal{E}$.

Since $p \leq \log n$, there exists a node v^+, which is in the requesting set for all the phases. As its distance to hard requests in each phase is 0, the cost incurred by hard requests on an algorithm, which remains for the whole epoch in v^+, is at most 1 per request, which sums up to $p \cdot (B_{\text{main}} + 2 \cdot \min\{B_{\text{exp}}, \lambda\}) = O(p \cdot (B_{\text{main}} + B_{\text{exp}}))$ in total. The optimal

algorithm could move to v^+ at the beginning of the epoch, paying at most D. Finally, the cost incurred by easy requests on any algorithm is at most $2 \cdot \sum_{j=1}^{B_{\text{fix}}} j = O(B_{\text{fix}}^2)$ in each phase, which accounts for $O(p \cdot B_{\text{fix}}^2)$ for all requests in epoch $\mathcal{E}$. Thus,

$$C_{\text{OPT}}(\mathcal{E}) = O\left(p \cdot \left(B_{\text{main}} + B_{\text{exp}} + B_{\text{fix}}^2\right) + D\right) \ .$$

On the other hand, for any i, when hard requests start in the i-th phase, any deterministic algorithm Det has its page in the "wrong" set with probability 1/2. At this point, and for all hard requests, $R_i(t) = B_{\text{fix}}$. Hence, if Det moves its page to the other set within the hard requests, it pays at least $D \cdot B_{\text{fix}}$. Otherwise, it has to pay $\min\{B_{\text{exp}}, \lambda\}$ for serving each of B_{main} hard requests in the main part of a phase. Therefore, the expected cost of Det in epoch $\mathcal{E}$ is

$$\mathbf{E}[C_{\text{DET}}(\mathcal{E})] \geq \frac{1}{2} \cdot p \cdot \min\{D \cdot B_{\text{fix}},\ B_{\text{exp}} \cdot B_{\text{main}},\ \lambda \cdot B_{\text{main}}\} \ .$$

By summing these bounds on Det's and Opt's cost over all T epochs, and using linearity of expectation we get the lemma. ■

In the following we choose $B_{\text{main}} = B_{\text{exp}} = D/p$ and $B_{\text{fix}} = \sqrt{D/p}$. This way the denominator of the ratio guaranteed by Lemma 3.50 is equal to $\Theta(D/p)$. Thus, by the lemma above, for a randomly generated input $\mathcal{I}$

$$\mathbf{E}_{\mathcal{I}}[C_{\text{DET}}(\mathcal{I})] \geq \Omega\left(\min\left\{\sqrt{D \cdot p},\ D/p,\ \lambda\right\}\right) \cdot C_{\text{OPT}}(\mathcal{I}) \ .$$

If $\log n \leq \sqrt[3]{D}$, then by setting $p = \log n$, $\mathbf{E}_{\mathcal{I}}[C_{\text{DET}}(\mathcal{I})] = \Omega(\min\{\sqrt{D \cdot \log n}, \lambda\})$. If $\log n \geq \sqrt[3]{D}$, then by setting $p = \sqrt[3]{D}$, $\mathbf{E}_{\mathcal{I}}[C_{\text{DET}}(\mathcal{I})] = \Omega(\min\{D^{2/3}, \lambda\})$. Thus, in either case we may choose p, such that

$$\mathbf{E}_{\mathcal{I}}[C_{\text{DET}}(\mathcal{I})] \geq \Omega\left(\min\{\sqrt{D \cdot \log n},\ D^{2/3},\ \lambda\}\right) \cdot C_{\text{OPT}}(\mathcal{I}) \ . \tag{3.27}$$

We may apply this inequality to prove the desired lower bound.

Theorem 3.51. *Consider any randomized algorithm* Alg *which is c-competitive against an oblivious adversary for DPM problem in a network with maximal extent λ. Then*

$$c = \Omega\left(\min\left\{\sqrt{D \cdot \log n},\ D^{2/3},\ \lambda\right\}\right) \ .$$

Proof. The cost of Opt in any phase is at least $\Omega(\min\{D, B_{\text{fix}}\})$, as Opt has either to pay at least 1 for half of the easy requests, or move the page in the meantime. Thus, it is possible to create a sequence with an arbitrarily high cost of the optimal schedule. Hence, applying the Yao min-max principle to Inequality 3.27 yields the theorem. ■

Algorithm	Lower bound	Upper bound
Deterministic	$\Omega\left(\min\left\{n\cdot\sqrt{D}, D, \lambda\right\}\right)$	$O\left(\min\left\{n\cdot\sqrt{D}, D, \lambda\right\}\right)$
Randomized against adaptive-online adversary	$\Omega\left(\min\left\{n\cdot\sqrt{D}, D, \lambda\right\}\right)$	$O\left(\min\left\{n\cdot\sqrt{D}, D, \lambda\right\}\right)$
Randomized against oblivious adversary	$\Omega\left(\min\left\{\sqrt{D\cdot\log n}, D^{2/3}, \lambda\right\}\right)$	$O\left(\min\left\{\sqrt{D\cdot\log n}, D, \lambda\right\}\right)$

Table 3.1: Competitive ratios in the adversarial scenario

3.4 Concluding remarks

The results of this chapter, dedicated to the adversarial scenario of the DPM problem, are gathered in Table 3.1. We note that both for deterministic case as well as for the case of adaptive adversaries, we created strategies which are up to a constant factor optimal (in terms of competitive ratios).

An interesting task would be to close the gap for randomized algorithms playing against oblivious adversaries. The following argument might seem vague, but it tries to explain why it is unlikely for a lower bound to be easily improved. Assume that n is very large, i.e., $\log n = \omega(\sqrt[3]{D})$ and we still want to get a lower bound of $\Omega(\sqrt{D\cdot\log n})$. If we use the same construction as in the proof of Theorem 3.51, we have to set B_{exp} to at least $\Omega(\sqrt{D\cdot\log n})$, i.e., use distances of order $\sqrt{D\cdot\log n} = \omega(D^{2/3})$. Otherwise, a trivial $O(\lambda)$-competitive algorithm would be $O(\sqrt{D\cdot\log n})$-competitive.

On the other hand, to balance (for the optimal algorithm) the cost of moving the page with the cost of serving requests, the length of an epoch should be $\Theta(D)$, as OPT has to pay at least 1 for each request. Formally, in each time step, OPT does not have to pay 1, but all the A_i counters but one increase by at least 1. Even if OPT chooses to remain at the only node for which A_i counter is low, the algorithm might quickly detect it and catch OPT.

This reasoning leaves the adversary only $o(\sqrt[3]{D}) = o(\log n)$ phases to use. Therefore, we suppose that for a large n, the adversary simply does not have enough time steps to really use all the nodes to deceive the algorithm. On the other hand, in the construction or the analysis of the algorithms presented in this chapter, we have not exploited the fact, that OPT pays 1 for most of the steps. Thus, we conjecture the following.

Conjecture 3.52. *There exists a randomized algorithm, which achieves a competitive ratio of $O(D^{2/3})$ against an oblivious adversary in the adversarial scenario of the DPM. Moreover, it is possible to combine this algorithm with* EBM *and* DYN-M, *getting an asymptotically optimal $O(\min\{\sqrt{D\cdot\log n}, D^{2/3}, \lambda\})$-competitive strategy.*

3.5 Proofs of technical claims

Proof of Claim 3.30. From the definition of T_k we have

$$T_k = 1 + \sum_{i=0}^{k-1} \frac{1}{k} \cdot T_i \ .$$

Adding $\sum_{i=0}^{k-1} T_i$ to both sides,

$$\sum_{i=1}^{k} T_i = 1 + \sum_{i=0}^{k-1} \frac{k+1}{k} \cdot T_i \ .$$

By dividing both sides by $k+1$, we get

$$\frac{1}{k+1} \cdot \sum_{i=1}^{k} T_i = \frac{1}{k+1} + \frac{1}{k} \cdot \sum_{i=0}^{k-1} T_i \ .$$

Again applying the definitions of T_k and T_{k+1} to both sides, respectively, we get

$$T_{k+1} - 1 = \frac{1}{k+1} + T_k - 1 \ ,$$

which directly implies the lemma. ■

Proof of Claim 3.36. Let $f(x) := x \cdot \log \frac{1}{x}$. It is easy to check that f is continuous and concave up for all $x > 0$. Therefore, we can apply Jensen's Inequality [HLP88] (see Appendix A.2) to get

$$f\left(\sum_i p_i \cdot x_i\right) \geq \sum_i p_i \cdot f(x_i) \ ,$$

for any $x_i > 0$ and for $0 \leq p_i \leq 1$, such that $\sum_i p_i = 1$. Let $p_i = 2^{-a_i} / \sum_k 2^{-a_k}$ and $x_i = 2^{-b_i+a_i}$. Then we have $\sum_i p_i \cdot x_i = \sum_i 2^{-b_i} / \sum_i 2^{-a_i}$, and therefore

$$\begin{aligned} \frac{\sum_i 2^{-b_i}}{\sum_i 2^{-a_i}} \cdot \log \frac{\sum_i 2^{-a_i}}{\sum_i 2^{-b_i}} &\geq \sum_i \frac{2^{-a_i}}{\sum_k 2^{-a_k}} \cdot 2^{-b_i+a_i} \cdot \log\left(\frac{1}{2^{-b_i+a_i}}\right) \\ &= \frac{\sum_i 2^{-b_i}}{\sum_k 2^{-a_k}} \cdot (b_i - a_i) \ . \end{aligned}$$

Multiplying both sides by $\sum_i 2^{-a_i} / \sum_i 2^{-b_i}$, we get

$$\log \frac{\sum_i 2^{-a_i}}{\sum_i 2^{-b_i}} \geq \frac{\sum_i 2^{-b_i} \cdot (b_i - a_i)}{\sum_i 2^{-b_i}} \ ,$$

which finishes the proof. ■

Chapter 4

Brownian Motion Scenario

In this chapter we consider a scenario in which the network adversary is replaced by a random process. We concentrate on the case, in which the space $\mathcal{X}$ is a one-dimensional discrete torus (or, alternatively speaking, a discrete ring) of diameter B. By diameter we understand the number of points on this ring. We assume that each node performs a simple random walk described below. Later, in Section 4.4, we argue that our results partially extend to the multi-dimensional case, or to the case where torus is replaced by a discrete mesh of the same diameter.

In Section 4.1 we construct a deterministic algorithm Maj for this scenario. Roughly speaking, Maj computes frequencies of requests at individual nodes and moves to the one which issued a majority of requests in the recent steps. We prove that Maj achieves a competitive ratio of $O(\max\{1, \min\{\sqrt{B}, \sqrt{D/B}, n\}\} \cdot \text{polylog}(D, B, n))$ on rings with diameters $B \geq \sqrt[3]{D}$. The ratio is achieved both with high probability and in expectation. For the case of smaller ring sizes, we may use the $O(\lambda) = O(B)$-competitive algorithm Dyn-M presented in Section 2.4.2.

The algorithm Maj would perform well also in static uniform networks, which is caused by the fact that it uses information about node positions in a very limited manner. The proof of its competitiveness assumes the largest possible cost of communication for the Maj algorithm, and shows that in this scenario the optimal offline algorithm is rarely able to exploit the nodes' movement, or use short distances occurring sometimes between pairs of nodes. We give the roadmap of the proof in the same section, and formally prove the bounds in Section 4.2 and Section 4.3.

We conclude with some extensions and open questions.

Scenario definition

First, we formally define how the input sequence $\mathcal{I}$ is generated. At the beginning, the request adversary picks the request sequence $(\sigma_t)_t$ and the network adversary chooses

the initial configuration C_0 of n points. They may cooperate while creating these parts of input. The rest of the configuration sequence $(C_t)_t$ is generated randomly, i.e., for each t, C_{t+1} is generated from C_t in the following way. For each node v, we compute its new position. Let $p_t(v)$ denote its current position (coordinate) at step t. For each node v we define a random variable $Z_t(v)$.

$$Z_t(v) = \begin{cases} -1 & \text{with probability } 1/3 \\ 0 & \text{with probability } 1/3 \\ 1 & \text{with probability } 1/3 \end{cases} \tag{4.1}$$

Then, the position of v in step $t+1$ is defined as

$$p_{t+1}(v) = p_t(t) + Z_v(t) \ . \tag{4.2}$$

These two formulas above are further referred to as the *movement rule*.

In time step $t \geq 1$, the positions of the nodes are set according to C_t, and then a request is issued at the node σ_t. In the remaining part of time step t, the algorithm serves requests and (optionally) moves its page.

We emphasize that the sequence $(\sigma_t)_t$ is created without knowledge of the actual outcome of the random walk of nodes, i.e., without knowledge of the configuration sequence $(C_t)_t$.

Performance metric

As already mentioned in Chapter 1, in order to analyze the performance of an online algorithm in the Brownian motion scenario, we adapt classical competitive analysis [ST85, BE98] to the model, where the input sequence is created both by the adversary and the stochastic process. We say that an algorithm ALG achieves a competitive ratio $\mathcal{R}$ (or is $\mathcal{R}$-competitive) with probability p, if there exists a constant A, such that for all request sequences (σ_t) it holds that

$$\mathbf{Pr}_{(C_t)_t}\Big[\, C_{\mathrm{ALG}}((C_t,\sigma_t)_t) \leq \mathcal{R} \cdot C_{\mathrm{OPT}}((C_t,\sigma_t)_t) + A \,\Big] \geq p \ , \tag{4.3}$$

where $C_{\mathrm{ALG}}((C_t,\sigma_t)_t)$ and $C_{\mathrm{OPT}}((C_t,\sigma_t)_t)$ are costs of ALG and the optimal offline algorithm, respectively. The probability is taken over all possible configuration sequences generated by the random movement (4.1). We say that ALG is $\mathcal{R}$-competitive in expectation, if there exists a constant A, such that for all request sequences $(\sigma_t)_t$ it holds that

$$\mathbf{E}_{(C_t)_t}\left[\frac{C_{\mathrm{ALG}}((C_t,\sigma_t)_t) - A}{C_{\mathrm{OPT}}((C_t,\sigma_t)_t)}\right] \leq \mathcal{R} \ . \tag{4.4}$$

Results

We present a solution which works for diameters B greater than $\sqrt[3]{D}$. Namely, we construct two deterministic online algorithms: MajL and MajS. While they differ in details, they share the same framework, which we call Maj. MajL was designed to work well when the ring diameter B is long, i.e., $B \geq \sqrt{D}$, while MajS is for short ring diameters, i.e., for $B \in [\sqrt[3]{D}, \sqrt{D}]$. We prove that both algorithms achieve a competitive ratio of $O(\mathcal{R})$, where

$$\mathcal{R} \;=\; \log^2(B \cdot n) \cdot \max\left\{1, \min\left\{\sqrt{B}, \sqrt{D/B}, n\right\}\right\} \quad . \tag{4.5}$$

The competitive ratio is achieved both with high probability and in expectation. Note that these results imply that the competitive ratio is $\tilde{O}(\min\{\sqrt[4]{D}, n\})$. This also immediately yields a competitive ratio of $O(\log^2(n \cdot B))$ for the network consisting of constant number of nodes, as well as for the case of large diameters, where $B = \Omega(D)$.

4.1 Majority algorithms

The underlying intuition and rationale for our algorithms is described below. If nodes perform independent random walks on the ring, then the distance between any two fixed nodes is usually $\Omega(B)$. Since the adversary has to choose the request sequence $(\sigma_t)_t$ beforehand, i.e., before the actual outcome of the random walk of nodes, we claim that it leaves the adversary, essentially, two options.

- The adversary gives requests at one node for a long period of time. In this case the algorithm may detect such a situation and move to the requesting node. It is hard for the adversary to synchronize the moment in which it changes requesting node to another one with the point of time where these two nodes are close to each other. Hence, usually, any algorithm, even the optimal one, has to pay a certain amount for the movement. On the other hand, when the decision about a potential movement is already made, the algorithm may want to wait for its source and destination nodes to come closer, as the cost paid for the requests in this waiting period might be smaller than the gain of moving the page at shorter distances. We show that for short diameters B, waiting for such a good opportunity with moving the page is a sound tactic.

- The adversary changes the accessing nodes frequently. But in this case, since the adversary has no control over the configuration sequence $(C_t)_t$, the requests will be scattered on the whole ring. Thus, any algorithm would pay a lot in this case, and our algorithm does not have to move to achieve a good competitive ratio.

This informal argument motivates us to construct an algorithm, which would perform well in the static uniform network and then show that it also performs well when nodes move. Certainly, the algorithms presented in the previous chapter are not well suited, as their competitive ratios in static networks are $\Omega(\sqrt{D})$. On the other hand, there exist simple *majority* algorithms, which achieve constant competitive ratio in static networks.

A majority algorithm works in phases of some fixed length K. It keeps counters of how many requests in the current phase were issued at individual nodes. Formally, for any time interval I and a node v, *weight* of v in I, is defined as the number of requests issued by v during I. We denote this weight by $w_I(v)$. After the end of each phase, the algorithm decides to move to a node v^*, which issued a strict majority of requests in this phase, i.e., whose weight in this phase is greater than $K/2$. As mentioned above, for short diameters the algorithm waits some additional number of steps, hoping that the chosen jump candidate v^* gets closer to the node currently holding the page. If all nodes' weights are at most $K/2$, then the algorithm remains at the same node.

Algorithm MAJ

In this section we formally present two algorithms MAJL and MAJS. Although they differ in details, their framework is essentially the same. For the sake of this presentation, we denote the algorithm by MAJ; the two algorithms will be just refinements of the MAJ framework.

MAJ is an example of a majority algorithm. It works in phases of fixed length K, where $K = D$ for MAJL and $K = B^2 \cdot \log B$ for MAJS. In a phase P, MAJ remains in one node denoted $P_{\text{MAJ}}(P)$.

If there exists a node $v^* \neq P_{\text{MAJ}}(P)$ such that $w_P(v^*) \geq K/2$, then MAJ decides to move to v^*, otherwise it remains in the same node. The decision is made in the last step of P (where MAJ has the full knowledge of the past requests), however the actual movement might be postponed a little bit to some time step in the future. If the algorithm decides to move, we distinguish between two cases depending on the length of the diameter B.

- For long diameters, MAJ moves immediately in the last step of P.

- For short diameters, the next $6 \cdot B^2 \cdot \log B$ steps after such a phase are called *migration sequence*. If at some step of the migration sequence the distance between $P_{\text{MAJ}}(P)$ and v^* becomes at most 1, MAJ moves to v^* in this step. If these nodes are always further than 1 within the migration sequence, then MAJ moves at the end of it. Note that migration sequence always lasts $6 \cdot B^2 \cdot \log B$ steps, independently of when MAJ migrates its phase.

The next phase begins right after the end of migration sequence (or right after the current phase, if there is no migration sequence). Note that phases and migration sequences do not overlap. Thus, for long diameters the input consists entirely of phases, and for short diameters the input can be partitioned into phases and migration sequences.

In this chapter we prove that Maj is $O(\mathcal{R})$ competitive with high probability and in expectation. While we concentrate on proving the former, we get the latter as a byproduct.

Theorem 4.1. Maj *is $O(\mathcal{R})$-competitive in the Brownian motion scenario of the DPM, with high probability.*

We show that there exists a constant A (depending on B, D, and n, but independent of the input sequence), such that for any γ, any input sequence $(\sigma_t)_t$ and any starting configuration C_0, if $(C_t)_t$ is generated according to (4.1), then it holds that

$$\mathbf{Pr}_{(C_t)_t}\left[C_{\mathrm{MAJ}}((C_t,\sigma_t)_t) \le O(\mathcal{R})\cdot C_{\mathrm{OPT}}((C_t,\sigma_t)_t) + \gamma\cdot A\right] \;\ge\; 1 - 6\cdot(B\cdot D)^{-\gamma}\;. \tag{4.6}$$

In the next section we show some basic properties: for the analysis we group phases into epochs and show a global bound on Maj's cost in one epoch. Then in Section 4.1.2 we start with a roadmap of the proof of Maj's competitiveness. We define a useful notion of auxiliary weight and propose two lemmas, which relate the costs of Maj and Opt in individual epochs, to the auxiliary weight. In the same section we combine these bounds to prove Theorem 4.1. The proposed lemmas are the crucial part of our analysis, but as their proofs are rather lengthy and tedious, they are presented later in Section 4.2 and Section 4.3.

4.1.1 Epochs

We group the phases (and the corresponding migration sequences) into epochs. For MajL an epoch consists of $\lceil B^2/D\rceil$ phases; for MajS an epoch consists of just one phase, optionally with its migration sequence. This grouping might seem a little bit artificial, it will however become clear in Section 4.3, where we explicitly require that each epoch length is at least B^2. It can be easily checked that our definition of an epoch fulfills this property. The described parameters of epochs and phases are gathered in Table 4.1.

An important property of such division into phases, corresponding migration sequences, and epochs is the independence of the configuration sequence or the algorithm, i.e., the division can be determined entirely on the basis of the request sequence $(\sigma_t)_t$.

Diameter	Long	Short
Algorithm	MAJL	MAJS
B	$B \geq \sqrt{D}$	$\sqrt[3]{D} \leq B \leq \sqrt{D}$
Migration sequence	none	optional
Epoch	$\lceil B^2/D \rceil$ phases	1 phase + migration sequence
Phase length	D	$B^2 \cdot \log B$

Table 4.1: MAJ parameters for different diameter lengths

Cost in a single epoch

It is possible to prove that in any single epoch the cost of MAJ is bounded. Let

$$L_p = 2 \cdot B^2 \cdot \log B \ , \tag{4.7}$$
$$C_p = 4 \cdot B^3 \cdot \log B + D \cdot B \ . \tag{4.8}$$

Lemma 4.2. *For any epoch $\mathcal{E}$ (also for an unfinished one at the end of the input sequence) its length is at most L_p and $C_{MAJ}(\mathcal{E}) \leq C_p$.*

Proof. Using information from Table 4.1, we may compute that each epoch's length is at most

$$\begin{aligned} \max\{\lceil B^2/D \rceil \cdot D,\, B^2 \cdot \log B\} &\leq 2 \cdot B^2 + B^2 \cdot \log B \\ &\leq 2 \cdot B^2 \cdot \log B \\ &= L_p \end{aligned}$$

We note that the cost of communication on the ring is bounded by $\lceil B/2 \rceil + 1 \leq B$ (as $B \geq 2$), and thus the cost of moving the page is at most $D \cdot B$. MAJ moves the page at most once for each phase. Thus, for long diameters, it moves the page at most $\lceil B^2/D \rceil$ times in one epoch, paying at most $D \cdot B \cdot \lceil B^2/D \rceil \leq 2 \cdot B^3$, whereas for short diameters MAJ moves the page at most once, paying at most $D \cdot B$.

Since the total cost of serving requests is at most $L_p \cdot B$, the total cost of MAJ in any epoch $\mathcal{E}$ is bounded by

$$\begin{aligned} C_{MAJ}(\mathcal{E}) &\leq L_p \cdot B + 2 \cdot B^3 + D \cdot B \\ &\leq 4 \cdot B^3 \cdot \log B + D \cdot B \\ &= C_p \ , \end{aligned}$$

which finishes the proof. ■

4.1.2 Competitiveness of MAJ

We fix any input sequence $\mathcal{I}$ and divide it into epochs $\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3 \ldots$. By Lemma 4.2, the cost in the last unfinished epoch is at most C_p; we show that we may neglect it. Thus, we may restrict our considerations to the input sequences consisting of whole epochs only.

We want to relate the expected cost of Maj to the expected cost of the optimal offline algorithm Opt in each epoch. Later, we are going to apply concentration bounds to get that this relation holds also with high probability. However, to do this, it would be necessary that this relation is independent for each epoch, and we cannot guarantee such a strong condition. Nevertheless, it turns out that it is possible to construct *randomized bounds* on Maj's and Opt's costs in each epoch $\mathcal{E}$, show the relation between these bounds and claim that these bounds depend only on random walks of nodes in epoch $\mathcal{E}$, and two epochs preceding $\mathcal{E}$. Hence, by considering these bounds for each third epoch, we get that these bounds are independent. This coarse-grained description as well as the notion of a randomized bound will be precised and explained below.

In fact, we do not establish a direct relation between Maj and Opt, but we propose a notion of *auxiliary weight*, which depends only on the request sequence $(\sigma_t)_t$ and tries to characterize how costly the epoch is for any algorithm. Further, we relate auxiliary weight to the bounds on costs of Maj and Opt.

Auxiliary weight

The notion of *auxiliary weight* is defined for any time interval I. It describes the discrepancy of the requests within this interval and will play a central role in our considerations.

Definition 4.3 (Auxiliary weight). *Fix any time interval I. Let $v_{\max}$ be the node which has the maximal weight in interval I, with ties broken arbitrarily. We define* auxiliary weight *of I as $W_A(I) = |I| - w_I(v_{\max})$.*

We note that $W_A(I)$ depends only on the subsequence of $(\sigma_t)_t$ in I and is, in a sense, a measure of how expensive the requests in I can be. If it is low, then there exists a node $v_{\max}$, such that the algorithm which remains in this node within I pays relatively few. Unfortunately, if the discrepancy is high, it does not directly imply that the cost of any algorithm is high. In particular, it may happen that all nodes are very close to each other throughout the whole interval I, which means that the cost of some algorithm in I could be very low. However, we show that such a configuration sequence is very unlikely to occur.

The auxiliary weight can be also computed for non-contiguous periods of time, e.g., for a union of two disjoint intervals. Note that this may substantially differ (i.e., be larger) from the sum of auxiliary weights of these two intervals. For example, if we have two intervals I_1 and I_2, both of length ℓ, and in I_1 all requests are given at v_1, and in I_2 all requests are given at v_2, then $W_A(I_1) + W_A(I_2) = 0$, but $W_A(I_1 \uplus I_2) = \ell$.

Randomized bounds

In this part, we present two lemmas relating the costs of MAJ and OPT in individual epochs to the auxiliary weight of some intervals. In the following we sometimes treat epochs as sets of phases, thus by "for all $P \in \mathcal{E}_j$" we understand "for all the phases from epoch $\mathcal{E}_j$". Furthermore, if P is a phase, then by P_{prev} we denote the phase preceding P, and by P_{migr}, we denote the migration sequence right after P. If P does not have a migration sequence, then $P_{\text{migr}} = \emptyset$.[1]

We define the weight of an epoch $\mathcal{E}_j$ as

$$W_j = B \cdot \sum_{P \in \mathcal{E}_j} W_A(P_{\text{prev}} \uplus P) \tag{4.9}$$

We observe that this notion depends only on the request sequence $(\sigma_t)_t$ in epochs $\mathcal{E}_{j-1}$ and $\mathcal{E}_j$ (in fact, only the last phase of epoch $\mathcal{E}_{j-1}$ is involved).

We relate the costs incurred on MAJ in $\mathcal{E}_j$ and on OPT in epochs $(\mathcal{E}_{j-1} \uplus \mathcal{E}_j)$ to W_j. However, as mentioned before, if we just relate the random variables (or rather their expected values) $\mathbf{E}[C_{\text{MAJ}}(\mathcal{E}_j)]$ and $\mathbf{E}[C_{\text{OPT}}(\mathcal{E}_{j-1} \uplus \mathcal{E}_j)]$ to W_j, then later we will not be able to apply the concentration bounds to relate $C_{\text{MAJ}}(\mathcal{I})$ and $C_{\text{OPT}}(\mathcal{I})$ to $\sum_{\mathcal{E}_j \in \mathcal{I}} W_j$, with high probability. The reason is, naturally, that for two different epochs $\mathcal{E}_{j_1}$ and $\mathcal{E}_{j_2}$ $C_{\text{OPT}}(\mathcal{E}_{j_1-1} \uplus \mathcal{E}_{j_1})$ and $C_{\text{OPT}}(\mathcal{E}_{j_2-1} \uplus \mathcal{E}_{j_2})$ may depend on each other.

However, it is possible to show that this dependence is rather limited. Alternatively speaking, it is possible to construct a *randomized upper bound* on $C_{\text{MAJ}}(\mathcal{E}_j)$ and a *randomized lower bound* on $C_{\text{OPT}}(\mathcal{E}_{j-1} \uplus \mathcal{E}_j)$. These bounds are random variables, which we call $C^{\text{U}}_{\text{MAJ}}(\mathcal{E}_j)$ and $C^{\text{L}}_{\text{OPT}}(\mathcal{E}_{j-1} \uplus \mathcal{E}_j)$, respectively or shortly $C^{\text{U}}_{\text{MAJ}}(j)$ and $C^{\text{L}}_{\text{OPT}}(j)$. To specify their properties we introduce two definitions.

Definition 4.4. *A random variable X we call* two-valued *if*

$$X = \begin{cases} a & \text{with probability } p \\ b & \text{with probability } 1 - p \end{cases}$$

for some real values a,b, and $0 \le p \le 1$. We purposely do not specify the probability space here, as we will assume that different two-valued random variables are always independent.

[1] As in the previous chapter we treat phases and time intervals either as sequences or sets of time steps.

One may think of two-valued variable as about an outcome of the biased coin tossing. All bounds $C^{\mathrm{U}}_{\mathrm{MAJ}}(j)$ and $C^{\mathrm{L}}_{\mathrm{OPT}}(j)$ will be two-valued random variables. Now, we clarify what we understand by a randomized bound.

Definition 4.5 (Stochastic dominance). *Let X and Y be two real-valued random variables. We say that X dominates Y, if for all real x it holds that*

$$\mathbf{Pr}[X \geq x] \geq \mathbf{Pr}[Y \geq x] \ .$$

However, if the variable $C^{\mathrm{U}}_{\mathrm{MAJ}}(j)$ stochastically dominates $C_{\mathrm{MAJ}}(\mathcal{E}_j)$, then this would not be sufficient to claim that *the sum of* $C^{\mathrm{U}}_{\mathrm{MAJ}}(j)$ stochastically dominates *the sum of* $C_{\mathrm{MAJ}}(\mathcal{E}_j)$. Since we are going to bound the latter using a bound on former, we need a notion stronger than the stochastic dominance. We say that $C^{\mathrm{U}}_{\mathrm{MAJ}}(j)$ stochastically dominates $C_{\mathrm{MAJ}}(\mathcal{E}_j)$ *with respect to a time interval I*, if $C^{\mathrm{U}}_{\mathrm{MAJ}}(j)$ stochastically dominates $C_{\mathrm{MAJ}}(\mathcal{E}_j)$ for any fixed outcomes of random walks *outside I*. Analogously, we will construct $C^{\mathrm{L}}_{\mathrm{OPT}}(j)$, which is stochastically dominated by $C_{\mathrm{OPT}}(\mathcal{E}_{j-1} \uplus \mathcal{E}_j)$ with respect to some time interval I. Note that for any two intervals $I_1 \subseteq I_2$, stochastic dominance with respect to I_1 implies stochastic dominance with respect to I_2. In practice, we will prove stochastic dominance for $C^{\mathrm{U}}_{\mathrm{MAJ}}(j)$ and $C^{\mathrm{L}}_{\mathrm{OPT}}(j)$ with respect to interval $(\mathcal{E}_{j-2} \uplus \mathcal{E}_{j-1} \uplus \mathcal{E}_j)$. Later, we will show how the stochastic dominance with respect to disjoint intervals implies the stochastic dominance for sums of variables.

Below we propose two lemmas specifying properties of variables $C^{\mathrm{U}}_{\mathrm{MAJ}}(j)$ and $C^{\mathrm{L}}_{\mathrm{OPT}}(j)$.

Lemma 4.6 (Upper bound on MAJ). *There exist constants c_1 and c_2, such that for any $j \geq 3$, there exists a two-valued random variable $C^{\mathrm{U}}_{\mathrm{MAJ}}(j)$, such that*

(i) $C^{\mathrm{U}}_{\mathrm{MAJ}}(j)$ *stochastically dominates* $C_{\mathrm{MAJ}}(\mathcal{E}_j)$ *with respect to* $(\mathcal{E}_{j-2} \uplus \mathcal{E}_{j-1} \uplus \mathcal{E}_j)$,

(ii) $c_1 \cdot W_j \ \leq \ C^{\mathrm{U}}_{\mathrm{MAJ}}(j) \ \leq \ c_2 \cdot W_j \cdot \sqrt{D}$,

(iii) $\mathbf{E}\left[C^{\mathrm{U}}_{\mathrm{MAJ}}(j)\right] \ = \ O(W_j)$.

Lemma 4.7 (Lower bound on OPT). *There exist constants c_3, c_4, and c_5, such that for any $j \geq 3$, there exists a two-valued random variable $C^{\mathrm{L}}_{\mathrm{OPT}}(j)$, such that*

(i) $C^{\mathrm{L}}_{\mathrm{OPT}}(j)$ *is stochastically dominated by* $C_{\mathrm{OPT}}(\mathcal{E}_{j-1} \uplus \mathcal{E}_j)$, *with respect to* $(\mathcal{E}_{j-2} \uplus \mathcal{E}_{j-1} \uplus \mathcal{E}_j)$,

(ii) $c_3 \cdot \frac{1}{R} \cdot \frac{1}{B} \cdot W_j \ \leq \ C^{\mathrm{L}}_{\mathrm{OPT}}(j) \ \leq \ c_4 \cdot \frac{1}{R} \cdot W_j$,

(iii) $\mathbf{E}\left[C^{\mathrm{L}}_{\mathrm{OPT}}(j)\right] \ \geq \ c_5 \cdot \frac{1}{R} \cdot W_j$.

For clarity reasons, the proofs of these crucial lemmas are presented in Section 4.2 and Section 4.3, respectively. In the remaining part of this section we show how to use these bounds to prove the competitiveness of MAJ. First, we formulate a lemma, which shows that if we consider a set of each third epoch from the input sequence, e.g., a set $\mathcal{M} = \{\mathcal{E}_3, \mathcal{E}_6, \mathcal{E}_9, \mathcal{E}_{12}, \ldots\}$, then the variable $\sum_{\mathcal{E}_j \in \mathcal{M}} C^{\mathrm{U}}_{\mathrm{MAJ}}(j)$ stochastically dominates $\sum_{\mathcal{E}_j \in \mathcal{M}} C_{\mathrm{MAJ}}(\mathcal{E}_j)$.

Lemma 4.8 (Stochastic dominance for sums). *Consider a product probability space $\Omega = \Omega_1 \times \Omega_2 \times \ldots \times \Omega_m$, where each Ω_i is a discrete event space. Let $\{X_i\}_{i=1}^n$ be random variables defined on Ω, and $\{Y_i\}_{i=1}^n$ be a sequence of n independent variables, such that for all i, Y_i stochastically dominates X_i with respect to Ω_i. Formally, this mean that for any fixed random experiments $(\omega'_1, \omega'_2, \ldots, \omega'_{i-1}, \omega'_{i+1}, \ldots, \omega'_m) \in \Omega_1 \times \Omega_2 \times \ldots \times \Omega_{i-1} \times \Omega_{i+1} \times \ldots \Omega_m$, and all real x, it holds that*

$$\mathbf{Pr}[Y_j \geq x] \geq \mathbf{Pr}_{\omega_i \in \Omega_i}[X(\omega'_1, \omega'_2, \ldots, \omega'_{i-1}, \omega_i, \omega'_{i+1}, \ldots, \omega'_m) \geq x] \ .$$

Then $\sum_{i=1}^n Y_i$ stochastically dominates $\sum_{i=1}^n X_i$.

The proof can be found at the end of this chapter. In our example we might apply the lemma with $X_i = C_{\mathrm{MAJ}}(\mathcal{E}_{3\cdot i})$, $Y_i = C^{\mathrm{U}}_{\mathrm{MAJ}}(3 \cdot i)$, and Ω_i containing all possible outcomes of random walk within $(\mathcal{E}_{3\cdot i-2} \uplus \mathcal{E}_{3\cdot i-1} \uplus \mathcal{E}_{3\cdot i})$. Equivalently, we may prove a version where Y_i are stochastically dominated by X_i, and infer that $\sum_{\mathcal{E}_j \in \mathcal{M}} C^{\mathrm{L}}_{\mathrm{OPT}}(j)$ is stochastically dominated by $\sum_{\mathcal{E}_j \in \mathcal{M}} C_{\mathrm{OPT}}(\mathcal{E}_{j-1} \uplus \mathcal{E}_j)$.

Relating global costs

We defer the proofs of the two lemmas above to the next sections and instead we show how Theorem 4.1 follows from these lemmas. To prove it we use one of the theorems concerning concentration of the measure, the Hoeffding bound [Hoe63]. We use its variant presented in [RPRR01] (see Appendix A.2 for an exact formulation).

Proof of Theorem 4.1. We show that there exists a constant A, such that for any constant γ, Inequality 4.6 holds. Let $A = 3 \cdot c_{\mathrm{A}} \cdot A_0 + 3 \cdot C_{\mathrm{p}}$, where

$$A_0 = 2 \cdot \max\left\{\frac{c_2^2}{c_1^2}, \frac{c_4^2}{c_5^2}\right\} \cdot D \cdot C_{\mathrm{p}} \cdot \ln(D \cdot B) \ , \tag{4.10}$$

$$c_{\mathrm{A}} = c_2 \cdot \sqrt{D} \ . \tag{4.11}$$

Constants c_1, c_2, c_3, c_4, and c_5 are the ones guaranteed by Lemma 4.6 and Lemma 4.7. We fix any input $\mathcal{I}$ and divide it into epochs $\mathcal{E}_1, \mathcal{E}_2, \ldots$. Clearly, the cost incurred on MAJ in the first two epochs together with the cost incurred in the last unfinished epoch is at most $3 \cdot C_{\mathrm{p}}$, and can be therefore hidden in the additive constant A.

We divide the remaining epochs into three disjoint sets $\mathcal{M}_0$, $\mathcal{M}_1$, and $\mathcal{M}_2$, where

$$\mathcal{M}_\chi := \{\mathcal{E}_j : j \equiv \chi \mod 3\} \ .$$

Assume that for any $\mathcal{M}_\chi$ we could prove that

$$\mathbf{Pr}\left[C_{\text{MAJ}}(\mathcal{M}_\chi) \le O(\mathcal{R}) \cdot C_{\text{OPT}}(\mathcal{I}) + \gamma \cdot c_{\text{A}} \cdot A_0\right] \ \ge \ 1 - 2 \cdot (B \cdot D)^{-\gamma} \ . \tag{4.12}$$

If we sum this guarantee over all three sets and add the cost of Maj in the first, the second and the last epoch, we would get

$$\mathbf{Pr}\left[C_{\text{MAJ}}(\mathcal{I}) \le O(3 \cdot \mathcal{R}) \cdot C_{\text{OPT}}(\mathcal{I}) + \left(\gamma \cdot 3 \cdot c_{\text{A}} \cdot A_0 + 3 \cdot C_{\text{p}}\right)\right] \ \ge \ 1 - 6 \cdot (B \cdot D)^{-\gamma} \ ,$$

which is all that we need to claim that Maj is $O(R)$-competitive with high probability. Thus, it remains to prove Inequality 4.12 for any set $\mathcal{M}_\chi$.

We consider the value of $\sum_{\mathcal{E} \in \mathcal{M}_\chi} W_j$. If it is smaller than $\gamma \cdot A_0$, then by Lemma 4.6, $C_{\text{MAJ}}(\mathcal{M}_\chi)$ is at most $c_2 \cdot (\gamma \cdot A_0) \cdot \sqrt{D} \le \gamma \cdot c_{\text{A}} \cdot A_0$, and (4.12) trivially follows.

Otherwise, we get $\sum_{\mathcal{E} \in \mathcal{M}_\chi} W_j \ge \gamma \cdot A_0$. Since all $C^{\text{U}}_{\text{MAJ}}(j)$ are independent for $\mathcal{E}_i \in \mathcal{M}_\chi$, we use Hoeffding inequality to bound the probability that the cost of Maj in epochs from $\mathcal{M}_\chi$ deviates much from its expected value. By Lemma 4.6, $c_1 \cdot W_j \le C^{\text{U}}_{\text{MAJ}}(j) \le c_2 \cdot \sqrt{D} \cdot W_j$, and therefore we get

$$\mathbf{Pr}\left[\sum_{\mathcal{E}_j \in \mathcal{M}_\chi} C^{\text{U}}_{\text{MAJ}}(j) > \frac{3}{2} \cdot \sum_{\mathcal{E}_j \in \mathcal{M}_\chi} \mathbf{E}\left[C^{\text{U}}_{\text{MAJ}}(j)\right]\right] \ \le \ \exp\left(-\frac{2 \cdot \left(\frac{1}{2} \cdot \sum_{\mathcal{E}_j \in \mathcal{M}_\chi} \mathbf{E}\left[C^{\text{U}}_{\text{MAJ}}(j)\right]\right)^2}{\sum_{\mathcal{E}_j \in \mathcal{M}_\chi} \left(c_2 \cdot \sqrt{D} \cdot W_j\right)^2}\right) \ .$$

We denote the exponent by $(-M)$. By (4.9), the definition of W_j, we get $W_j \le 2 \cdot C_{\text{p}}$. Using this and $C^{\text{U}}_{\text{MAJ}}(j) \ge c_1 \cdot W_j$, we obtain

$$\begin{aligned}
M \ &\ge \frac{1}{2} \cdot \frac{c_1^2 \cdot \left(\sum_{\mathcal{E}_j \in \mathcal{M}_\chi} W_j\right)^2}{c_2^2 \cdot D \cdot \sum_{\mathcal{E}_j \in \mathcal{M}_\chi} W_j^2} \\
&\ge \frac{c_1^2}{2 \cdot c_2^2} \cdot \frac{\left(\sum_{\mathcal{E}_j \in \mathcal{M}_\chi} W_j\right)^2}{D \cdot \left(\sum_{\mathcal{E}_j \in \mathcal{M}_\chi} W_j\right) \cdot \max_{\mathcal{E}_j \in \mathcal{M}_\chi}\{W_j\}} \\
&\ge \frac{c_1^2}{2 \cdot c_2^2} \cdot \frac{\gamma \cdot A_0}{2 \cdot D \cdot C_{\text{p}}} \\
&= \gamma \cdot \ln(D \cdot B) \ .
\end{aligned}$$

Analogously, we can get the bound on $C^{\text{L}}_{\text{OPT}}$. In fact, for variables $C^{\text{L}}_{\text{OPT}}$ we do not even have the factor D in the denominator, thus achieving high probability is easier.

$$\mathbf{Pr}\left[\sum_{\mathcal{E}_j \in \mathcal{M}_\chi} C^{\text{L}}_{\text{OPT}}(j) < \frac{1}{2} \cdot \sum_{\mathcal{E}_j \in \mathcal{M}_\chi} \mathbf{E}\left[C^{\text{L}}_{\text{OPT}}(j)\right]\right] \ \le \ (D \cdot B)^{-\gamma}$$

These bounds combined with the results of the Lemma 4.6 and Lemma 4.7 imply that

$$\sum_{\mathcal{E}_j \in \mathcal{M}_\chi} C^{\mathrm{U}}_{\mathrm{MAJ}}(j) = O(1) \cdot \sum_{\mathcal{E}_j \in \mathcal{M}_\chi} W_j \ , \qquad \text{with probability } 1 - (D \cdot B)^{-\gamma}$$

$$\sum_{\mathcal{E}_j \in \mathcal{M}_\chi} C^{\mathrm{L}}_{\mathrm{OPT}}(j) = \Omega(1) \cdot (1/\mathcal{R}) \cdot \sum_{\mathcal{E}_j \in \mathcal{M}_\chi} W_j \ , \qquad \text{with probability } 1 - (D \cdot B)^{-\gamma}$$

By stochastic dominance (see Lemma 4.8), the same bounds apply also for random variables $\sum_{\mathcal{E}_j \in \mathcal{M}_\chi} C_{\mathrm{MAJ}}(\mathcal{E}_j) = C_{\mathrm{MAJ}}(\mathcal{M}_\chi)$ and $\sum_{\mathcal{E}_j \in \mathcal{M}_\chi} C_{\mathrm{OPT}}(\mathcal{E}_{j-1} \uplus \mathcal{E}_j) \leq C_{\mathrm{OPT}}(\mathcal{I})$, respectively. Therefore, with probability at least $1 - 2 \cdot (D \cdot B)^{-\gamma}$, $C_{\mathrm{MAJ}}(\mathcal{M}_\chi) = O(\mathcal{R}) \cdot C_{\mathrm{OPT}}(\mathcal{I})$, and thus Inequality 4.12 follows. This finishes the proof. ■

As a corollary we get that the expected competitive ratio is $O(\mathcal{R})$, as well.

Corollary 4.9. *Algorithm* MAJ *achieves an expected competitive ratio of* $O(\mathcal{R})$ *in the Brownian motion scenario of the DPM.*

Proof. Fix any request sequence $(\sigma_t)_t$, and an initial configuration C_0. By Lemma 4.6 and Lemma 4.7, for all randomly generated configuration sequences $(C_t)_t$, it holds that

$$C^{\mathrm{U}}_{\mathrm{MAJ}}(j) = O\left(B \cdot \sqrt{D} \cdot \mathcal{R}\right) \cdot C^{\mathrm{L}}_{\mathrm{OPT}}(j) \ ,$$

and thus for $\mathcal{I} = (C_t, \sigma_t)_t$

$$C_{\mathrm{MAJ}}(\mathcal{I}) = O\left(B \cdot \sqrt{D} \cdot \mathcal{R}\right) \cdot C_{\mathrm{OPT}}(\mathcal{I}) \ .$$

The bound above is the worst-case bound, i.e., it holds always. Now, we fix a constant A as in the proof of Theorem 4.1. Then the expected value of the quotient $\left(\frac{C_{\mathrm{ALG}}(\mathcal{I}) - A}{C_{\mathrm{OPT}}(\mathcal{I})}\right)$ can be bounded as

$$\begin{aligned} \mathbf{E}_{(C_t)_t}\left[\frac{C_{\mathrm{ALG}}(\mathcal{I}) - A}{C_{\mathrm{OPT}}(\mathcal{I})}\right] &\leq O(\mathcal{R}) \cdot \left(1 - \frac{6}{D \cdot B}\right) + O\left(\sqrt{D} \cdot B \cdot \mathcal{R}\right) \cdot \frac{6}{D \cdot B} \\ &= O(\mathcal{R}) \ . \end{aligned}$$

This proves the expected competitive ratio of MAJ. ■

4.2 Bounding cost of MAJ

In this section we show that the performance of the MAJ algorithm in $\mathcal{E}_j$ depends directly on the sum of auxiliary weights of phases from $\mathcal{E}_{j-1}$ and $\mathcal{E}_j$. We charge MAJ maximum cost, B, for each request issued at a node different from the node holding its page. Thus, $C^{\mathrm{U}}_{\mathrm{MAJ}}$, the bound on MAJ's cost, does not depend the configuration sequence $(C_t)_t$,

except for the migration sequences, in which $(C_t)_t$ decides whether a good opportunity occurs. We show that in expectation the cost of Maj's in the migration sequences can be neglected when compared to the cost of Maj in the corresponding phase. Finally, we use these results to prove Lemma 4.6.

While we finally construct a random variable $C^{\mathrm{U}}_{\mathrm{MAJ}}(\mathcal{E}_j)$, in the meanwhile we introduce several randomized bounds on the Maj's cost in different parts of $\mathcal{E}_j$. We also call them $C^{\mathrm{U}}_{\mathrm{MAJ}}(\cdot)$.

MAJ's cost in one phase

First, we present a useful characterization of the Maj's phases. Fix any phase P of length K. We distinguish between three cases.

1. *Wait phase* occurs, if $w_P(P_{\mathrm{MAJ}}(P)) > K/2$.
2. *Mixed phase* occurs, if for all nodes v, $w_P(v) \leq K/2$.
3. *Change phase* occurs, if there exists a node $v^* \not\equiv P_{\mathrm{MAJ}}(P)$ such that $w_P(v^*) > K/2$.

Clearly, the migration sequence occurs (for short diameters), if and only if the corresponding phase is a change phase. We begin with creating a bound on the Maj's cost in any single phase P. Recall that by P_{prev} we denote the phase preceding P. We choose

$$C^{\mathrm{U}}_{\mathrm{MAJ}}(P) := 4 \cdot B \cdot W_{\mathrm{A}}(P_{\mathrm{prev}} \uplus P) \ . \tag{4.13}$$

Lemma 4.10. *For any phase P, it holds that $C_{\mathrm{MAJ}}(P) \leq C^{\mathrm{U}}_{\mathrm{MAJ}}(P)$.*

Proof. Note that in this lemma we are comparing two real numbers, and not random variables. The cost of paying for any request is at most B. We proceed with case analysis.

1. If P is a wait phase, then Maj pays for $K - w_P(P_{\mathrm{MAJ}}(P)) \leq W_{\mathrm{A}}(P)$ requests. Thus, $C_{\mathrm{MAJ}}(P) \leq B \cdot W_{\mathrm{A}}(P)$.
2. If P is a mixed phase, then $w_P(v) \leq K/2$ for all nodes v, and hence $W_{\mathrm{A}}(P) \geq K/2$. Maj pays for at most K requests, and thus $C_{\mathrm{MAJ}}(P) \leq K \cdot B \leq 2 \cdot B \cdot W_{\mathrm{A}}(P)$.
3. If P is a change phase, then Maj pays for at most K requests. Additionally, if the diameter is long, then Maj moves at the end of P, paying at most $D \cdot B = K \cdot B$. Thus, $C_{\mathrm{MAJ}}(P) \leq 2 \cdot B \cdot K$.

   On the other hand, there exists a node $v^* \not\equiv P_{\mathrm{MAJ}}(P)$ (to which Maj moves at the end of P, or in the corresponding migration sequence). However, $w_{P_{\mathrm{prev}}}(v^*) \leq K/2$, because otherwise Maj would have moved to v^* after phase P_{prev},[2] and would remain in v^* in the whole phase P. Therefore, $w_{P_{\mathrm{prev}} \uplus P}(v^*) \leq \frac{3}{2} \cdot K$.

[2] By "after phase P_{prev}" we mean "in the last step of P_{prev} or during the migration sequence of P_{prev}".

This inequality holds also for any v_i. Indeed, since $w_P(v^*) > K/2$, for any node $v_i \not\equiv v^*$ it holds that $w_P(v_i) < K/2$, and hence $w_{P_{\text{prev}} \uplus P}(v_i) < \frac{3}{2} \cdot K$.

Therefore, $W_{\text{A}}(P_{\text{prev}} \uplus P) \geq K/2$, and thus $C_{\text{MAJ}}(P) = 4 \cdot B \cdot W_{\text{A}}(P_{\text{prev}} \uplus P)$.

For the first two cases, we note that W_{A} is monotonic, and thus $W_{\text{A}}(P) \leq W_{\text{A}}(P_{\text{prev}} \uplus P)$. Hence, for any phase P, it holds that $C_{\text{MAJ}}(P) \leq 4 \cdot B \cdot W_{\text{A}}(P_{\text{prev}} \uplus P) = C^{\text{U}}_{\text{MAJ}}(P)$. ■

MAJ's cost in migration sequence

Additionally, we may prove that the expected cost incurred in the migration sequence is asymptotically not greater than the cost incurred in the corresponding phase. First, we compute the probability that a good opportunity occurs in a migration phase.

Lemma 4.11. *Consider any two nodes v_a and v_b. If both move according to the movement rule, then, with probability at least $1 - 1/B$, there exists a time step t within the next $6 \cdot B^2 \cdot \log B$ steps, such that $d_t(v_a, v_b) \leq 1$.*

Proof. We show that if we start from any positions of v_a and v_b, then with probability $\frac{1}{2}$ they will meet (will be at the distance of at most 1) at some time step within next $6 \cdot B^2$ steps. These steps constitute one round. The lemma will follow immediately, as the event that nodes fail to meet in any of $\log B$ rounds, happens with probability at most $(1/2)^{\log B} = \frac{1}{B}$.

Let X_t be the distance, counted clockwise, between v_a and v_b in step t. In each step t, $Z_t(v_a) - Z_t(v_b)$ is the change in this distance. By E we denote the event that after $6 \cdot B^2$ steps the total change in this distance, $\sum_{t=1}^{6B^2}[Z_t(v_a) - Z_t(v_b)]$ belongs to $(-B, B)$. Clearly, if E does not occur, then there must have been a step where the points v_a and v_b met. Thus, it suffices to prove that $\mathbf{Pr}[E] \leq 1/2$.

We have $\mathbf{Pr}[E] = \mathbf{Pr}[-B < \sum_{i=1}^{12B^2} A_i < B]$, where A_i are independent random variables with identical distributions as $Z_t(v)$ variables. Let $S_0 = |\{i : A_i = 0\}|$. We have

$$\mathbf{Pr}[E] \leq \mathbf{Pr}[E \mid S_0 \leq 6B^2] + \mathbf{Pr}[S_0 > 6B^2] \ . \tag{4.14}$$

Thus, it remains to show that both summands above are small.

For bounding the second summand of (4.14) we use the Hoeffding bound [RPRR01] (see Appendix A.2). Since for any i, $A_i = 0$ with probability $1/3$, it holds that $\mathbf{E}[S_0] = 4 \cdot B^2$. Thus, using $B \geq 2$, we get

$$\begin{aligned} \mathbf{Pr}[S_0 > 6 \cdot B^2] &\leq \exp\left(-\frac{2 \cdot (2 \cdot B^2)^2}{12 \cdot B^2}\right) \\ &\leq \mathrm{e}^{-2 \cdot B^2/3} \\ &\leq \mathrm{e}^{-8/3} \ . \end{aligned} \tag{4.15}$$

For bounding the first summand of (4.14), we assume that $S_0 \leq 6 \cdot B^2$. Then the set of variables A_i which are different from zero has $k := 12 \cdot B^2 - S_0 \geq 6 \cdot B^2$ elements. Each of these k variables equals 1 with probability 1/2, and -1 also with probability 1/2. Therefore, for any $x \in [-k, k]$ we get

$$\mathbf{Pr}\left[\sum_{i=1}^{12B^2} A_i = x \;\middle|\; S_0 = 12 \cdot B^2 - k\right] = \begin{cases} \frac{1}{2^k} \cdot \binom{k}{(k+x)/2} \leq \frac{1}{2^k} \cdot \binom{k}{\lceil k/2 \rceil} & \text{if } x + k \text{ is even,} \\ 0 & \text{otherwise.} \end{cases}$$

Summing up over $x \in (-B, B)$, we obtain

$$\mathbf{Pr}\left[\sum_{i=1}^{12B^2} A_i \in (-B, B) \;\middle|\; S_0 = 12 \cdot B^2 - k\right] \leq B \cdot \frac{1}{2^k} \cdot \binom{k}{\lceil k/2 \rceil} \ .$$

Note that $\frac{1}{2^k} \cdot \binom{k}{\lceil k/2 \rceil}$ is a monotonically decreasing function of k. It follows from the Stirling formula [Fel68] (see Appendix A.2), that for even k, it holds that $\frac{1}{2^k} \cdot \binom{k}{k/2} \leq \sqrt{\frac{2}{\pi}} \cdot \frac{1}{\sqrt{k}}$. Thus,

$$\mathbf{Pr}\left[\sum_{i=1}^{12B^2} A_i \in (-B, B) \;\middle|\; S_0 \geq 6 \cdot B^2\right] \leq B \cdot \sqrt{\frac{2}{\pi}} \cdot \frac{1}{\sqrt{6 \cdot B^2}} = \sqrt{\frac{1}{3 \cdot \pi}} \ . \tag{4.16}$$

Finally, if we apply the bounds on the summands of (4.14), we get $\mathbf{Pr}[E] \leq \sqrt{1/3\pi} + \mathrm{e}^{-8/3}$, and therefore it can be numerically checked that $\mathbf{Pr}[E] < 1/2$. ■

Now we can construct a bound $C^{\mathrm{U}}_{\mathrm{MAJ}}(P_{\mathrm{migr}})$ for $C_{\mathrm{MAJ}}(P_{\mathrm{migr}})$. If the diameter is long, then let $C^{\mathrm{U}}_{\mathrm{MAJ}}(P_{\mathrm{migr}}) = 0$. Otherwise, let $C^{\mathrm{U}}_{\mathrm{MAJ}}(P_{\mathrm{migr}})$ be a random two-valued variable equal to $6 \cdot B^3 \cdot \log B + X$, where X is equal to $2 \cdot D$ with probability $1 - 1/B$, and $B \cdot D$ with probability $1/B$. Then we may prove the following.

Lemma 4.12. *$C^{\mathrm{U}}_{\mathrm{MAJ}}(P_{\mathrm{migr}})$ stochastically dominates $C_{\mathrm{MAJ}}(P_{\mathrm{migr}})$ with respect to P_{migr}, and it holds that*

$$\mathbf{E}[C^{\mathrm{U}}_{\mathrm{MAJ}}(P_{\mathrm{migr}})] = O(1) \cdot C^{\mathrm{U}}_{\mathrm{MAJ}}(P) \ .$$

Proof. The cost incurred on Maj within P_{migr} consists of the cost of serving requests and the cost of moving the page. The former can be easily bounded by $|P_{\mathrm{migr}}| \cdot B \leq 6 \cdot B^3 \cdot \log B$. For the latter note, that by Lemma 4.11 for any starting configuration C_t at the beginning of P_{migr} a good opportunity occurs with probability at least $1 - 1/B$, in which case Maj pays at most $(1+1) \cdot D$. Otherwise, Maj pays at most $B \cdot D$. Thus, $C^{\mathrm{U}}_{\mathrm{MAJ}}(P_{\mathrm{migr}})$ stochastically dominates $C_{\mathrm{MAJ}}(P_{\mathrm{migr}})$ with respect to P_{migr}.

The expected value of $C^{\mathrm{U}}_{\mathrm{MAJ}}(P_{\mathrm{migr}})$ can be bounded by

$$\begin{aligned} \mathbf{E}[C^{\mathrm{U}}_{\mathrm{MAJ}}(P_{\mathrm{migr}})] &\leq 6 \cdot B^3 \cdot \log B + 2 \cdot D \cdot (1 - 1/B) + B \cdot D \cdot 1/B \\ &\leq 6 \cdot B^3 \cdot \log B + 3 \cdot D \ . \end{aligned}$$

We note that the migration sequence occurs only after a change phase and only if the diameter is short. Since $B \geq \sqrt[3]{D}$, the cost accounted for a change phase is

$$\begin{aligned} C^{\mathrm{U}}_{\mathrm{MAJ}}(P) &\geq \frac{1}{2} \cdot K \cdot B \\ &= \frac{1}{2} \cdot B^3 \cdot \log B \\ &= \Omega\left(\mathbf{E}[C^{\mathrm{U}}_{\mathrm{MAJ}}(P_{\mathrm{migr}})]\right) \ . \end{aligned}$$

This finishes the proof. ∎

MAJ's cost in one epoch

We may combine the bounds on the cost in P and P_{migr} to prove Lemma 4.6. Let

$$C^{\mathrm{U}}_{\mathrm{MAJ}}(\mathcal{E}_j) = \sum_{P \in \mathcal{E}_j} \left(C^{\mathrm{U}}_{\mathrm{MAJ}}(P) + C^{\mathrm{U}}_{\mathrm{MAJ}}(P_{\mathrm{migr}})\right) \ .$$

Proof of Lemma 4.6 (Upper bound on MAJ). We fix any epoch $\mathcal{E}_j$. First, we prove the stochastic dominance. It the ring diameter is long, then $C^{\mathrm{U}}_{\mathrm{MAJ}}(\mathcal{E}_j)$ (denoted also by $C^{\mathrm{U}}_{\mathrm{MAJ}}(j)$) is a constant and summing the guarantee of Lemma 4.10 over all the phases, we get that $C^{\mathrm{U}}_{\mathrm{MAJ}}(j)$ is always greater than $C_{\mathrm{MAJ}}(\mathcal{E}_j)$. Otherwise, in the case of short diameters, $\mathcal{E}_j$ consists of at most one migration sequence P_{migr}. Then by Lemma 4.10 and Lemma 4.12, $C^{\mathrm{U}}_{\mathrm{MAJ}}(j)$ stochastically dominates $C_{\mathrm{MAJ}}(\mathcal{E}_j)$ with respect to P_{migr}, and thus also with respect to $(\mathcal{E}_{j-2} \uplus \mathcal{E}_{j-1} \uplus \mathcal{E}_j)$.

Second, we prove the bound on the expected value of $C^{\mathrm{U}}_{\mathrm{MAJ}}(j)$. By Lemma 4.10 and Lemma 4.12, we get that for any phase $P \in \mathcal{E}_j$, it holds that

$$\mathbf{E}[C^{\mathrm{U}}_{\mathrm{MAJ}}(P) + C^{\mathrm{U}}_{\mathrm{MAJ}}(P_{\mathrm{migr}})] = O(1) \cdot B \cdot W_{\mathrm{A}}(P_{\mathrm{prev}} \uplus P) \ .$$

By summing this inequality over all the phases within epoch, we get that

$$\begin{aligned} \mathbf{E}[C^{\mathrm{U}}_{\mathrm{MAJ}}(\mathcal{E}_j)] &= O(1) \cdot B \cdot \sum_{P \in \mathcal{E}_j} W_{\mathrm{A}}(P_{\mathrm{prev}} \uplus P) \\ &= O(1) \cdot W_j \ . \end{aligned}$$

Third, we show the bounds on the variable $C^{\mathrm{U}}_{\mathrm{MAJ}}(j)$. For a lower bound, by definition (4.13), we obtain $C^{\mathrm{U}}_{\mathrm{MAJ}}(\mathcal{E}_j) \geq \sum_{P \in \mathcal{E}_j} C^{\mathrm{U}}_{\mathrm{MAJ}}(P) \geq 4 \cdot W_j$. For an upper bound, we note that for long diameters $C^{\mathrm{U}}_{\mathrm{MAJ}}(j)$ is a constant, and then $C^{\mathrm{U}}_{\mathrm{MAJ}}(j) = \mathbf{E}[C^{\mathrm{U}}_{\mathrm{MAJ}}(j)] = O(1) \cdot W_j$. For short diameters, the variable $C^{\mathrm{U}}_{\mathrm{MAJ}}(j)$ can differ at most by a factor of $B = O(\sqrt{D})$ from its expected value (we showed that on average the cost of movement is $O(D)$, whereas in the worst case it can be $B \cdot D$). Thus, for these diameters

$$C^{\mathrm{U}}_{\mathrm{MAJ}}(j) = O\left(\sqrt{D}\right) \cdot \mathbf{E}[C^{\mathrm{U}}_{\mathrm{MAJ}}(j)] = O(1) \cdot \sqrt{D} \cdot W_j \ .$$

Therefore, all the conditions of the lemma are met. ∎

4.3 Bounding cost of OPT

In the previous subsection we assume that the cost of serving request by Maj is B. However, we cannot hope for the same kind of argument for the optimal algorithm. In other words, we cannot assure that Opt pays $\Omega(B)$ for each request issued at the node different from the node holding the Opt's page. This makes the construction of a lower bound for Opt much more complicated.

In this section we present the roadmap of the proof, formulate several lemmas and show how they lead to the proof of Lemma 4.7. For clarity, the proofs of these lemmas are divided between the next subsections.

For any fixed request sequence $(\sigma_t)_t$ within some time interval, there are many possible choices of positions for nodes, such that the requests are issued in the different parts of the ring. If positions of the nodes satisfy this property, then we call the nodes *scattered*. Intuitively, if we had a static network with scattered nodes, then a high discrepancy of the requests would cause a high cost on any algorithm. However, in our situation we might encounter two serious problems. First, even if nodes are scattered at some point, their random walk may change their positions and destroy their distribution property. Second, Opt may use the nodes' movement to quickly move the page between nodes. Since Opt knows the directions, in which the nodes will move in the future, it might move to a distant node, paying much less than $D \cdot B$. Luckily, there is one remedy to both these problems. If the time interval that we consider is short, then we may prove that with a constant probability nodes are moving slowly, preserving the distribution property and not allowing Opt to move at a very low cost. On the other hand, shorter interval means less requests, and, in consequence, lesser cost. Thus, these two objectives have to be balanced.

A few notations

We introduce three functions of B, D, and n, which are used throughout this proof.

$$\beta = \sqrt{2 \cdot \ln(n \cdot B^4)} \ , \tag{4.17}$$

$$Q_0 = \min\left\{\sqrt{B}, \sqrt{D/B}, n\right\} \ , \tag{4.18}$$

$$Q = \max\{1, Q_0\} \ . \tag{4.19}$$

We note that by (4.5), the definition of $\mathcal{R}$,

$$\mathcal{R} = \Theta(\beta^4 \cdot Q) \ . \tag{4.20}$$

Narrow sets

As mentioned above, the core of constructing lower bound for OPT within interval $(\mathcal{E}_{j-1} \uplus \mathcal{E}_j)$ is finding a set $\mathcal{A}_j$ of disjoint short intervals from $(\mathcal{E}_{j-1} \uplus \mathcal{E}_j)$, such that the sum of auxiliary weights of these intervals is relatively high compared to W_j. Moreover, we require that these intervals are all contained in a short time period. Formally, for a set $\mathcal{S}$ of intervals, we define $\text{span}(\mathcal{S})$ as the shortest time interval containing all intervals from $\mathcal{S}$. Then our requirement is that $|\text{span}(\mathcal{A}_j)|$ should be small.

We formally enumerate the desired properties of such a short set

Definition 4.13. *Let*

$$\mathcal{N}_{\mathcal{A}} := \frac{B^2}{(32 \cdot \beta)^2} \quad \text{and} \quad \mathcal{N}_{\mathrm{I}} := \frac{\min\{D, B^2\}}{Q \cdot (16 \cdot \beta)^2}$$

A set of intervals $\mathcal{S}$ we call narrow, *if the following conditions hold*

(i) $|\text{span}(\mathcal{S})| \leq \mathcal{N}_{\mathcal{A}}$

(ii) For any interval $I \in \mathcal{S}$, it holds that $|I| \leq \mathcal{N}_{\mathrm{I}}$.

It appears that by using a variant of an average argument, it is possible to extract a narrow set from $(\mathcal{E}_{j-1} \uplus \mathcal{E}_j)$, such that the sum of auxiliary weights of intervals from this set is relatively high. The proof of the following lemma is presented in Section 4.3.1.

Lemma 4.14. *For any epoch $j \geq 2$, it is possible to choose a narrow set $\mathcal{A}_j$ contained in $(\mathcal{E}_{j-1} \uplus \mathcal{E}_j)$, such that*

$$\sum_{I \in \mathcal{A}_j} B \cdot W_{\mathrm{A}}(I) \;\geq\; b_1 \cdot \frac{1}{\mathcal{R}} \cdot W_j \;,$$

where b_1 is a constant.

Since in the previous subsection the expected cost of MAJ in $\mathcal{E}_j$ was proven to be proportional to W_j, for proving $O(\mathcal{R})$-competitiveness of MAJ it would be sufficient to show that with a constant probability the cost of OPT on *any* interval from $\mathcal{A}_j$ is equal to $\Omega(B \cdot W_{\mathrm{A}}(I))$. This probability should depend only on the random walk inside of $(\mathcal{E}_{j-2} \uplus \mathcal{E}_{j-1} \uplus \mathcal{E}_j)$ and hold for any configuration C_t at the beginning of $\mathcal{E}_{j-2}$.[3] We cannot guarantee that, but we can relax the conditions and prove a very similar result, which also yields $O(\mathcal{R})$-competitiveness. Namely, we show that with a constant probability there exists a *subset* $\mathcal{A}'_j$ of $\mathcal{A}_j$ such that

- on each interval $I \in \mathcal{A}'_j$ it holds that $C_{\mathrm{OPT}}(I) \geq B \cdot W_{\mathrm{A}}(I)$, and
- the auxiliary weights of intervals from $\mathcal{A}'_j$ constitute a constant fraction of the auxiliary weights of intervals from $\mathcal{A}_j$.

Obviously a subset of a narrow set is narrow, too.

[3] This is what we need to claim the stochastic dominance with respect to $(\mathcal{E}_{j-2} \uplus \mathcal{E}_{j-1} \uplus \mathcal{E}_j)$.

Preconditions

Below we describe two preconditions, which hold with a constant probability. If these preconditions are met, we may assure that it is possible to lower-bound the cost of Opt in a narrow set $\mathcal{A}_j$ by $\Omega(\sum_{I\in\mathcal{A}_j} B \cdot W_A(I))$.

Precondition A: Scattered nodes

First, we concentrate on a single interval I. As mentioned above, we may hope for a high cost of Opt (or any other algorithm) on this interval, if the nodes which issue requests in I are scattered on the ring. Below we formalize this notion.

Definition 4.15. *For any subset $R \subseteq \mathcal{X}$, and any node v, by "$v \in R$", we understand that the position of node v is in set R.*

Definition 4.16. *Fix any nodes configuration C_t in any time step t and an interval I. We say that the nodes are* I-scattered *in time t, or that C_t* is scattered according to *I, if it is possible to partition the ring into 4 disjoint contiguous segments R_1, R_2, R_3, R_4, each containing $B/4$ points from $\mathcal{X}$, as presented in Figure 4.1, such that both $w_I(R_1)$ and $w_I(R_3)$ are at least $\frac{1}{16} \cdot W_A(I)$. By $w_I(R_i)$ we denote the total weight accumulated in the segment R_i, i.e., $w_I(R_i) = \sum_{v\in R_i} w_I(v)$.*

An example of such partition into four segments R_1,R_2,R_3, and R_4 is presented in Figure 4.1. We note that this definition makes sense for any, even completely unrelated, interval I and time step t.

For sets of intervals the definition of being scattered is similar. However we require only that nodes are scattered according to a *sufficiently large fraction* of this set.

Definition 4.17. *Fix any nodes configuration C_t at any time step t, a set of intervals $\mathcal{S}$, and a constant $c \leq 1$. We say that the nodes are* $(\mathcal{S}, c)$-scattered *at time t, if there exists a subset $\mathcal{S}' \subseteq \mathcal{S}$ with the following properties.*

(i) It holds that

$$\sum_{I\in\mathcal{S}'} W_A(I) \geq c \cdot \sum_{I\in\mathcal{S}} W_A(I) \ .$$

(ii) For all intervals $I \in \mathcal{S}'$ nodes are I-distributed.

Precondition B: Smoothness

Even if the nodes are scattered at the beginning of some narrow set, their random walk may quickly destroy this property. To describe the desired behavior of the nodes, we introduce a following notion.

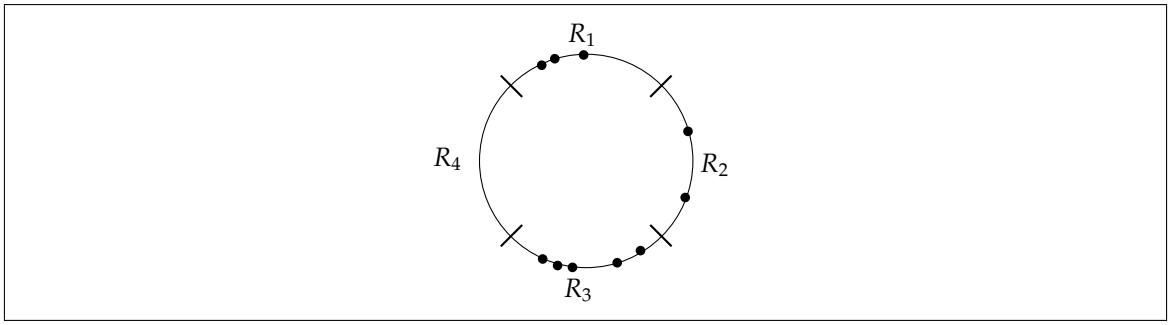

Figure 4.1: Ring partitioning

Definition 4.18. *We call a configuration sequence of length ℓ* smooth*, if for any $1 \leq i < j \leq \ell$ and any node v, the positions of node v in time step i and j differ by at most $\beta \cdot \sqrt{j-i}$. For any time interval I, by* smooth(I) *we denote the event that a configuration sequence in I is smooth.*

Intuitively, the nodes during a smooth configuration sequence do not behave too wildly, i.e., they never run away quickly from their starting positions.

Preconditions are likely

Before we state the result, we informally argue that the probability that both preconditions hold simultaneously is constant. We consider any three consecutive epochs $\mathcal{E}_{j-2}$, $\mathcal{E}_{j-1}$, and $\mathcal{E}_j$, as presented in Figure 4.2. The smoothness is the easier precondition; we may use elementary combinatorics to prove the following bound (see Section 4.3.2).

Lemma 4.19. *For any narrow set $\mathcal{A}_j$, it holds that*

$$\mathbf{Pr}[\text{smooth}(\text{span}(\mathcal{A}_j))] \geq 1/2 \ .$$

Moreover, this probability depends only on the random walks of nodes inside span($\mathcal{A}_j$).

On the other hand, consider any single interval $I \in \mathcal{A}_j$. Using standard probability tools (Markov inequality, Hoeffding bound), it is not difficult to prove that if we could choose the positions of the nodes uniformly at random at the beginning of span($\mathcal{A}_j$), then with a constant probability they would be $(\mathcal{A}_j, \Omega(1))$-scattered. We have no chance for having truly uniform distribution at that point of time; nevertheless, it is possible to assure an approximately uniform distribution. Namely, assume any (worst possible) starting configuration C_t at the beginning of $\mathcal{E}_{j-2}$. We call the steps between this point of time and the beginning of span($\mathcal{A}_j$) a *mixing sequence*. Such a sequence contains at least the whole epoch $\mathcal{E}_{j-2}$, and thus has length at least B^2. In the whole mixing sequence the nodes perform independently their random walks, and hence we may infer that their positions at the beginning of span($\mathcal{A}_j$) are independent random variables with almost

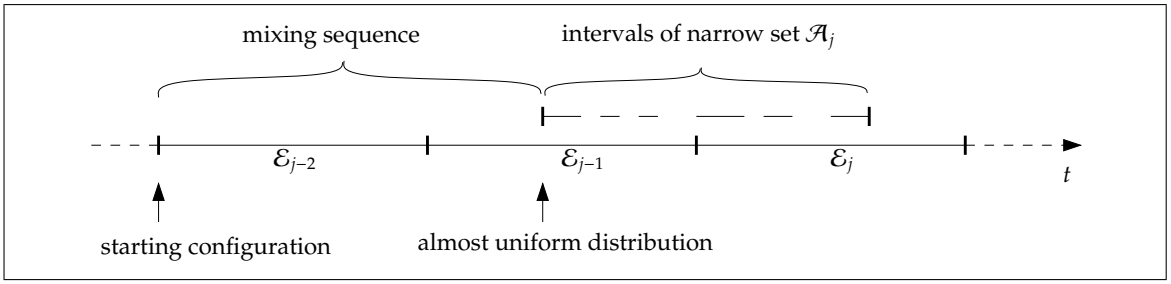

Figure 4.2: Epochs $\mathcal{E}_{j-2}$, $\mathcal{E}_{j-1}$, and $\mathcal{E}_j$

uniform distribution (on the ring). The argument uses the fact that the Markov chain (see, e.g., [Fel68, Sen81, Ros95] for a gentle introduction to Markov chains) induced by a random walk of nodes converges quickly to the uniform distribution. The following lemma puts together our informal arguments and is proven in Section 4.3.3.

Lemma 4.20. *For any configuration at the beginning $\mathcal{E}_{j-2}$ and any narrow set $\mathcal{A}_j$ belonging to $\mathcal{E}_{j-1} \uplus \mathcal{E}_j$, with a constant probability nodes are $(\mathcal{A}_j, \Omega(1))$-scattered at the beginning of* span$(\mathcal{A}_j)$*. Moreover, this probability depends only on the random walk of nodes in the mixing sequence.*

Since two lemmas above (Lemma 4.19 and Lemma 4.20) depend on different sources of randomness, i.e., configuration sequences in the mixing sequence and in span$(\mathcal{A}_j)$, respectively, their guarantees are independent.

Corollary 4.21. *There exists a constant b_2, such that for any configuration at the beginning of $\mathcal{E}_{j-2}$ and any narrow set $\mathcal{A}_j$ belonging to $(\mathcal{E}_{j-1} \uplus \mathcal{E}_j)$, with a constant probability p_{precond}, it holds that*

(i) nodes are $(\mathcal{A}_j, b_2)$-scattered at the beginning of span$(\mathcal{A}_j)$*, and*

(ii) the configuration sequence within span$(\mathcal{A}_j)$ *is smooth.*

Moreover, this probability depends only on the random walks of nodes inside $(\mathcal{E}_{j-2} \uplus \mathcal{E}_{j-1} \uplus \mathcal{E}_j)$.

Lower bound on OPT

Now, we formally state a Crucial Property, using the two preconditions described above (scattered nodes and smoothness). This property will be directly used to lower-bound the cost of Opt.

Lemma 4.22 (Crucial Property). *Fix any algorithm* Alg*, any narrow set $\mathcal{A}_j$, and a constant c. If at the beginning of* span$(\mathcal{A}_j)$ *the nodes are $(\mathcal{A}_j, c)$-scattered, and the configuration sequence within* span$(\mathcal{A}_j)$ *is smooth, then*

$$C_{\text{ALG}}(\mathcal{A}_j) \geq b_3 \cdot c \cdot \sum_{I \in \mathcal{A}_j} B \cdot W_{\text{A}}(I) \ ,$$

where b_3 is a constant.

We defer the proof of this lemma to Section 4.3.4. Here, we show how this Crucial Property can be used to construct the lower bound on OPT, specified by Lemma 4.7.

Proof of Lemma 4.7 (Lower bound on OPT). By Lemma 4.14, it is possible to find a narrow set $\mathcal{A}_j$ belonging to $(\mathcal{E}_{j-1} \uplus \mathcal{E}_j)$, whose intervals have high auxiliary weight. Let b_1, b_2, and b_3, be the constants defined in Lemma 4.14, Corollary 4.21, and Lemma 4.22. Let p_{precond} be the probability occurring in Corollary 4.21.

First, we prove that for any configuration sequence it holds that

$$C_{\text{OPT}}(\mathcal{A}_j) \geq \sum_{I \in \mathcal{A}_j} W_{\text{A}}(I) \ . \tag{4.21}$$

Note that this bound is $\Theta(B)$ times weaker than the one we would like to have, but it holds always and not only with some probability. Fix any $I \in \mathcal{A}_j$. The length of I is at most $\mathcal{N}_{\text{I}}$. If OPT remains in one node P_{OPT} within I, then it pays at least 1 for each request issued at a node different from P_{OPT}, which in total accounts for at least $W_{\text{A}}(I)$. If it moves within I, then it pays at least $D \geq \mathcal{N}_{\text{I}} \geq W_{\text{A}}(I)$. Since all intervals from $\mathcal{A}_j$ are disjoint, by summing over all of them, we get that (4.21) holds. Hence, by Lemma 4.14, $C_{\text{OPT}}(\mathcal{A}_j) \geq b_1 \cdot (1/B) \cdot (1/\mathcal{R}) \cdot W_j$.

For showing a better (randomized) lower bound, we note that by Corollary 4.21, with probability p_{precond} the nodes are $(\mathcal{A}_j, b_2)$-scattered at the beginning of span$(\mathcal{A}_j)$ *and* smooth(span(A_j)) holds. Then by the Crucial Property, we get $C_{\text{OPT}}(\mathcal{A}_j) \geq b_2 \cdot b_3 \cdot \sum_{I \in \mathcal{A}_j} B \cdot W_{\text{A}}(I)$. Finally, the auxiliary weights of intervals from the set $\mathcal{A}_j$ can be bounded by Lemma 4.14, which implies that $C_{\text{OPT}}(\mathcal{A}_j) \geq b_1 \cdot b_2 \cdot b_3 \cdot \frac{1}{\mathcal{R}} \cdot W_j$ (with probability at least p_{precond}). Thus, if we define $C^{\text{L}}_{\text{OPT}}(j)$ as

$$C^{\text{L}}_{\text{OPT}}(j) = \begin{cases} b_1 \cdot b_2 \cdot b_3 \cdot \frac{1}{\mathcal{R}} \cdot W_j & \text{with probability } p_{\text{precond}} \ , \\ b_1 \cdot \frac{1}{B} \cdot \frac{1}{\mathcal{R}} \cdot W_j & \text{otherwise} \ , \end{cases}$$

then $C^{\text{L}}_{\text{OPT}}(j)$ is stochastically dominated by $C_{\text{OPT}}(\mathcal{A}_j) \leq C_{\text{OPT}}(\mathcal{E}_{j-1} \uplus \mathcal{E}_j)$, with respect to $(\mathcal{E}_{j-2} \uplus \mathcal{E}_{j-1} \uplus \mathcal{E}_j)$. Since p_{precond} is a constant, $\mathbf{E}[C^{\text{L}}_{\text{OPT}}(j)] = \Omega(\frac{1}{\mathcal{R}} \cdot W_j)$. ■

In the next four subsections, we formally prove the lemmas stated in this section (narrow sets, probabilities for both preconditions, and the Crucial Property).

4.3.1 Narrow sets

In this section we show how to choose narrow sets from $(\mathcal{E}_{j-1} \uplus \mathcal{E}_j)$, so that the intervals in set chosen have high auxiliary weight. We consider three cases.

1. Short diameters, i.e., $B \leq \sqrt{D}$
2. Long diameters, such that $\sqrt{D} \leq B \leq 64 \cdot \beta \cdot \sqrt{D}$.
3. Long diameters, such that $B \geq 64 \cdot \beta \cdot \sqrt{D}$.

We present three proofs of Lemma 4.14, one for each of the cases listed above. For the needs of this section, the diameters described by case 2 above are called *semi-long*.

Construction for short diameters

Proof of Lemma 4.14 for short diameters. For short diameters, an epoch $\mathcal{E}_j$ consists of only one phase, which we denote by P. Therefore, the singleton set $\{(\mathcal{E}_{j-1} \uplus \mathcal{E}_j)\}$ would be a good candidate for a narrow set, as $W_{\mathrm{A}}(\mathcal{E}_{j-1} \uplus \mathcal{E}_j) \geq W_{\mathrm{A}}(P_{\mathrm{prev}} \uplus P) = \frac{1}{B} \cdot W_j$.

However, this interval is too long, i.e., it has length $\Theta(B^2 \cdot \log B)$. The desired length of this interval would be $\min\{\mathcal{N}_{\mathcal{A}}, \mathcal{N}_{\mathrm{I}}\} \geq \frac{\min\{D, B^2\}}{(32\beta)^2 \cdot Q}$. As $B \leq \sqrt{D}$, this term is equal to $\frac{B^2}{(32\beta)^2 \cdot Q}$. We may find a subinterval of $(\mathcal{E}_{j-1} \uplus \mathcal{E}_j)$ of this length losing at most a factor of $O(\beta^2 \cdot Q \cdot \log B) = O(\mathcal{R})$ in the auxiliary weight. To do so we apply the following technical claim to the interval $(\mathcal{E}_{j-1} \uplus \mathcal{E}_j)$.

Claim 4.23 *(Average argument for W_{A}). For any interval I and any length $4 \leq \ell \leq |I|$, there exists an interval $J \subseteq I$ of length ℓ, such that $W_{\mathrm{A}}(J) \geq \Omega(1) \cdot \frac{\ell}{|I|} \cdot W_{\mathrm{A}}(I)$.*

The claim is proven at the end of this chapter. We apply this claim with ℓ set to $\min\{\mathcal{N}_{\mathcal{A}}, \mathcal{N}_{\mathrm{I}}\} = \frac{B^2}{(32 \cdot \beta) \cdot Q}$. We may safely assume that $\ell \geq 4$ (otherwise $B = O(\beta^2)$ and we may use, for example, the algorithm Dyn-M constructed in Section 2.4.2 to achieve the competitive ratio of $O(B) = O(\beta^2)$). For $\mathcal{A}_j$ we choose a singleton set containing the interval found by Claim 4.23. Clearly, $\mathcal{A}_j$ is narrow, and it holds that $\sum_{I \in \mathcal{A}_j} W_{\mathrm{A}}(I) = \Omega\left(\frac{1}{\mathcal{R}}\right) \cdot \frac{1}{B} \cdot W_j$. ∎

Construction for long diameters

In case of long diameters, finding a narrow set is more complicated, because each epoch consists of multiple phases, and we cannot apply Claim 4.23 directly. In the following we concentrate on an epoch $\mathcal{E}_j$ and assume that it consists of $\kappa := \left\lceil \frac{B^2}{D} \right\rceil$ phases $P_1, P_2, \ldots, P_\kappa$, each of length D. Let P_0 be the last phase of $\mathcal{E}_{j-1}$.

Proof of Lemma 4.14 for semi-long diameters. For such a choice of B, an epoch consists of at most $64 \cdot \beta^2$ phases. Not losing much, we may pick one phase with high auxiliary weight and then use Claim 4.23 to shorten it to the desired length.

Formally, we choose a phase $P_i \in \mathcal{E}_j$, for which the value of $W_A(P_{i-1} \uplus P_i)$ is maximized. It follows from the average argument that there exists at least one phase, say P^*, for which this value is at least $\frac{1}{\kappa} \cdot \sum_{P \in \mathcal{E}_j} W_A(P_{\text{prev}} \uplus P) = \Omega(1/\beta^2) \cdot \sum_{P \in \mathcal{E}_j} W_A(P_{\text{prev}} \uplus P)$. Later, again by Claim 4.23, we may choose a subinterval I of phase P^* of length at most $\min\{\mathcal{N}_{\mathcal{A}}, \mathcal{N}_I\} \geq \frac{D}{(32\cdot\beta)^2\cdot Q}$ losing at most a factor of $O(\beta^2 \cdot Q)$. Thus, for this interval it holds that

$$\begin{aligned} W_A(I) &= \Omega\left(\frac{1}{\beta^4 \cdot Q}\right) \cdot \sum_{P \in \mathcal{E}_j} W_A(P_{\text{prev}} \uplus P) \\ &= \Omega\left(\frac{1}{\mathcal{R}}\right) \cdot \frac{1}{B} \cdot W_j \end{aligned}$$

Moreover the singleton set $\mathcal{A}_j := \{I\}$ is narrow, which finishes the proof. ■

Proof of Lemma 4.14 for long diameters. In this case the number of phases is at least $\kappa \geq (64 \cdot \beta)^2$. We proceed in three steps. In the first one we choose a contiguous subset of phases $\{P_0, P_1, \ldots, P_\kappa\}$, such that the total length of this subsequence is smaller than $\mathcal{N}_{\mathcal{A}} = B^2/(32\beta)^2$. In the second step we choose disjoint intervals from this subset, each consisting of exactly two phases. In the third, final step, using Claim 4.23, we shorten each of these intervals from length $2\cdot D$ to the length $\mathcal{N}_I = \frac{\min\{D,B^2\}}{(16\cdot\beta)^2\cdot Q} = \frac{D}{(16\cdot\beta)^2\cdot Q}$. For each step we carefully count the loss in the auxiliary weight, and prove that in total it diminishes at most by a factor $O(R)$.

First, we find a contiguous sequence of phases, such that the sum of auxiliary weights of these phases is large. Divide phases of $\mathcal{E}_j$ into disjoint chunks, each containing $\lfloor \mathcal{N}_{\mathcal{A}}/D \rfloor - 1$ phases; the last chunk is possibly shorter. There are $m = \lceil \frac{\kappa}{\lfloor \mathcal{N}_{\mathcal{A}}/D \rfloor - 1} \rceil$ chunks. Since $\mathcal{N}_{\mathcal{A}}$ is quite large, i.e., $\mathcal{N}_{\mathcal{A}} \geq 4 \cdot D$, the rounding incurred by ceiling and floor functions does not change the asymptotic value of $m = \Theta\left(\frac{1}{\beta^2}\right)$. It follows from the average argument that there exists one chunk C, for which it holds that

$$\sum_{P_i \in C} W_A(P_{i-1} \uplus P_i) \;\geq\; \Omega\left(\frac{1}{\beta^2}\right) \cdot \sum_{P_i \in \mathcal{E}_j} W_A(P_{i-1} \uplus P_i) \;. \tag{4.22}$$

Let C' be a subset of C created by taking each second phase from C, in such a way that

$$\sum_{P_i \in C'} W_A(P_{i-1} \uplus P_i) \;\geq\; \frac{1}{2} \cdot \sum_{P_i \in C} W_A(P_{i-1} \uplus P_i) \;. \tag{4.23}$$

If C contains only one phase, then we choose $C' = C$. As a consequence, each two phases from C' are separated by at least one phase not belonging to C'. Let

$$\mathcal{A}_j^0 \;:=\; \{(P_{\text{prev}} \uplus P) : P \in C'\} \;, \tag{4.24}$$

and thus $\mathcal{A}_j^0$ is the set of disjoint intervals, each consisting of two phases. The union of intervals from $\mathcal{A}_j^0$ contains the whole chunk C (and possibly a phase preceding the chunk C). On the other hand, $\mathcal{A}_j^0$ is contained in the $\lfloor \mathcal{N}_{\mathcal{A}}/D \rfloor$ consecutive phases, and hence $|\mathrm{span}(\mathcal{A}_j^0)| \leq \mathcal{N}_{\mathcal{A}}$. By the construction of $\mathcal{A}_j^0$, it follows immediately from (4.22), (4.23), and (4.24) that

$$\begin{aligned}\sum_{I \in \mathcal{A}_j^0} W_{\mathrm{A}}(I) &= \Omega\left(\frac{1}{\beta^2}\right) \cdot \sum_{P_i \in \mathcal{E}_j} W_{\mathrm{A}}(P_{i-1} \uplus P_i) \\ &= \Omega\left(\frac{1}{\beta^2}\right) \cdot \frac{1}{B} \cdot W_j \ .\end{aligned}$$

Since each interval from $\mathcal{A}_j^0$ has length at most $2 \cdot D$, we can use Claim 4.23 to shorten each $I \in \mathcal{A}_j^0$ to length $\mathcal{N}_{\mathrm{I}} = \frac{\min\{D, B^2\}}{(16\beta)^2 \cdot Q} = \frac{D}{(16\beta)^2 \cdot Q}$ losing additional factor of $O(\beta^2 \cdot Q)$. Let $\mathcal{A}_j$ be the set of shortened intervals from $\mathcal{A}_j^0$. Then $\mathcal{A}_j$ is narrow and

$$\begin{aligned}\sum_{I \in \mathcal{A}_j} W_{\mathrm{A}}(I) &= \Omega\left(\frac{1}{\beta^2 \cdot Q}\right) \cdot \sum_{I \in \mathcal{A}_j^0} W_{\mathrm{A}}(I) \\ &= \Omega\left(\frac{1}{\mathcal{R}}\right) \cdot \frac{1}{B} \cdot W_j \ ,\end{aligned}$$

which finishes the construction. ■

4.3.2 Precondition: smooth movement

In this section we prove that with a constant probability within a narrow set a randomly generated configuration sequence is smooth. In fact, the only property we use is that the length of an considered interval is at most B^2.

Proof of Lemma 4.19. Let $\ell = |\mathrm{span}(\mathcal{A}_j)| \leq B^2$. For any node v and any steps i, j within $\mathrm{span}(\mathcal{A}_j)$, we define

$$H_v(i, j) := \sum_{t=i+1}^{j} Z_t(v) \ .$$

Then $H_v(i, j)$ is an upper bound on the distance between the positions of v in steps i and j. $H_v(i, j)$ may be strictly greater than the actual distance, as it might exceed the half of the ring diameter. Since $\mathbf{E}[Z_v(t)] = 0$, $\mathbf{E}[H_v(i, j)] = 0$ as well. Moreover, since $Z_v(t)$ are all independent, we can apply the Hoeffding bound [RPRR01] (see Appendix A.2) to

obtain

$$\begin{aligned}\mathbf{Pr}\left[H_v(i,j) > \beta\cdot\sqrt{j-i}\right] &\leq \exp\left(-\frac{2\cdot\left(\beta\cdot\sqrt{j-i}\right)^2}{\sum_{t=i+1}^{j} 2^2}\right)\\ &\leq \mathrm{e}^{-\beta^2/2}\ .\end{aligned}$$

We aim to bound the probability of the event that $H_v(i,j) \leq \beta\cdot\sqrt{j-i}$ holds for all nodes v and all $1 \leq i < j \leq \ell$. The probability of the complementary event can be bounded as follows.

$$\begin{aligned}\mathbf{Pr}\left[\text{ not smooth}(\text{span}(\mathcal{A}_j))\right] &\leq \sum_{v\in V}\sum_{1\leq i<j\leq\ell}\mathbf{Pr}\left[H_v(i,j) > \beta\cdot\sqrt{j-i}\right]\\ &\leq n\cdot\frac{1}{2}\ell^2\cdot e^{-\beta^2/2}\\ &\leq \frac{1}{2}\cdot n\cdot\ell^2\cdot\frac{1}{n\cdot B^4}\\ &\leq \frac{1}{2}\ ,\end{aligned}$$

which finishes the proof. ■

4.3.3 Precondition: scattered nodes

Let $\mathcal{A}_j$ be any narrow set lying inside $(\mathcal{E}_{j-1} \uplus \mathcal{E}_j)$ and let the configuration at the beginning of $\mathcal{E}_{j-2}$ be any but fixed. The proof that nodes are $(\mathcal{A}_j, \Omega(1))$-scattered at the beginning of $\text{span}(\mathcal{A}_j)$ consists of several stages. First, we specify a property of *almost uniform distribution* of the nodes on the ring. Second, we show that for any fixed starting configuration at the beginning of $\mathcal{E}_{j-2}$, nodes at the beginning of $\text{span}(\mathcal{A}_j)$ are almost uniformly distributed. Third, we show that if nodes are almost uniformly distributed, then for a fixed interval $I \in \text{span}(\mathcal{A}_j)$ nodes are I-scattered with a constant probability. Finally, we show that it is (also with a constant probability) possible to extract a subset (with intervals having high auxiliary weights), such that the nodes are scattered according to *all intervals* from this subset. This will ultimately produce a proof of Lemma 4.20.

Convergence

First, we show that for any node configuration C_t at the beginning of $\mathcal{E}_{j-2}$, at the beginning of $\mathcal{A}_j$ nodes are distributed almost uniformly on the ring. To define and prove it formally, we introduce several definitions. Recall that $\mathcal{X}$ denotes our space, i.e., the discrete ring of diameter B.

Definition 4.24. *By μ we denote a uniform probability distribution over all points from $\mathcal{X}$.*

Definition 4.25. *A variation distance between two probability distributions ν_1 and ν_2 on $\mathcal{X}$, is defined as $\|\nu_1 - \nu_2\| := \max_{A \subseteq \mathcal{X}} |\nu_1(A) - \nu_2(A)|$.*

Definition 4.26. *Let $\mathcal{D}$ be the set of all probability distributions over the ring, whose variation distance to the uniform distribution is at most $1/64$. Any $\nu \in \mathcal{D}$ we call an* almost uniform distribution.

The technical lemma below follows from the convergence rate of Markov chain induced by the random walk (4.1).

Lemma 4.27. *If a node v starts from any position $x_t(v) \in \mathcal{X}$ at some step t, then its position after $k \geq B^2$ steps is a random variable with an almost uniform probability distribution.*

Proof. Fix a node v and its position $x_t(v)$ in step t. By ν_{t+k} we denote the probability distribution over all possible positions of v in step $t + k$. The random walk performed by v is isomorphic to a random walk on finite abelian group $\mathbb{Z}/(B)$, i.e., the integers modulo B, with transition probabilities given by the following matrix P:

$$P(x, y) = \begin{cases} 1/3 & \text{if } y \in \{x - 1, x, x + 1\}, \\ 0 & \text{otherwise.} \end{cases} \tag{4.25}$$

This random walk converges to the uniform distribution on $\mathbb{Z}/(B)$. Moreover, one may construct an explicit bound on its convergence rate (see for example Rosenthal [Ros95] or Appendix A.2). It holds that

$$\|\nu_{t+k} - \mu\|^2 \leq \frac{\exp\left(-\frac{4 \cdot \pi^2}{3 \cdot B^2} \cdot k\right)}{1 - \exp\left(-\frac{4 \cdot \pi^2}{3 \cdot B^2} \cdot k\right)} \ . \tag{4.26}$$

For $k \geq B^2$, the numerator is at most $e^{-\frac{4}{3}\pi^2} < 2^{-13}$, and thus the denominator is greater than $1/2$. It implies that the whole fraction on the right hand side of the inequality is at most 2^{-12}, and thus the variation distance $\|\nu_{t+k} - \mu\| \leq 1/64$. ■

As defined previously, the steps between the beginning of $\mathcal{E}_{j-2}$ and the beginning of $\text{span}(\mathcal{A}_j)$ a called *mixing sequence* (see Figure 4.2). Since the random walk of each node is independent and the mixing sequence has length at least B^2, we get the following.

Corollary 4.28. *For any configuration of nodes at the beginning of $\mathcal{E}_{j-2}$, the position of each node at the beginning of* $\text{span}(\mathcal{A}_j)$ *is a random variable with an almost uniform distribution. Moreover, all these random variables are independent. We denote such a situation by $V \sim \mathcal{D}$.*

Scattered configurations

In this part we consider any single interval I from the narrow set $\mathcal{A}_j$. We show that if the distribution of nodes on the ring is *almost uniform*, then with a constant probability the configuration of nodes is I-scattered.

Lemma 4.29. *Fix any interval I. If at some time step t holds $V \sim \mathcal{D}$, then the probability that nodes are I-scattered is at least $1/7$.*

Proof. Since we focus on one interval, we omit I in the subscripts of weight functions, i.e., we use $w(\cdot)$ and W_A in place of $w_I(\cdot)$ and $W_A(I)$, respectively.

In the following we construct a partition of ring into four segments, and prove that with probability at least $1/7$, both $w(R_1)$ and $w(R_3)$ are greater than $\frac{1}{16} \cdot W_A$.

Let $v_{\max}$ be the node with the maximal weight in I. We fix the partition of the ring into segments R_i in any manner, provided that $v_{\max} \in R_1$. Now, we can view the process of randomly distributing the nodes as a process of throwing the remaining nodes, independently, into four bins R_i. For any node v_j, bin R_i is chosen with the probability $\nu_j(R_i)$, where $\nu_j \in \mathcal{D}$. Thus, $\mu(R_i) - \frac{1}{64} \leq \nu_j(R_i) \leq \mu(R_i) + \frac{1}{64}$, where μ denotes the uniform probability distribution over $\mathcal{X}$. Let $w'(R_i)$ be the weight accumulated in R_i, not counting the weight of $v_{\max}$. Then

$$\frac{15}{64} \cdot W_A \leq \mathbf{E}[w'(R_i)] \leq \frac{17}{64} \cdot W_A \ . \tag{4.27}$$

We proceed with the case analysis, depending on how heavy $v_{\max}$ is.

1. $w(v_{\max}) \geq W/16$.

   As $v_{\max} \in R_1$, in this case we already have that $w(R_1) \geq W/16$. We have to assure only that there is sufficient weight accumulated in R_3. Achieving this with a constant probability is not difficult, as the expected value of this weight is already a constant fraction of W_A. Formally, $\mathbf{E}[w'(R_1)+w'(R_2)+w'(R_4)] \leq \frac{51}{64} \cdot W_A$, and thus by applying the Markov inequality [Fel68] (see Appendix A.2), it follows that

   $$\mathbf{Pr}\left[w'(R_1) + w'(R_2) + w'(R_4) \geq \frac{15}{16} \cdot W_A\right] \leq \frac{17}{20} \ .$$

   Therefore, $\mathbf{Pr}[w'(R_3) > W_A/16] \geq 3/20 > 1/7$.

2. $w(v_{\max}) < W/16$

In this case the weight is distributed among many nodes, as for each node v different from $v_{\max}$ it holds that $w(v) < W_A/15$. Since the nodes choose their ring segment independently, we can apply the Hoeffding bound [RPRR01] (see Appendix A.2), to show that for each segment R_i, it holds that

$$\begin{aligned}\mathbf{Pr}\left[w'(R_i) < \frac{4}{15}\cdot\mathbf{E}[w'(R_i)]\right] &< \exp\left(-\frac{2\cdot\left(\frac{11}{15}\cdot\mathbf{E}[w'(R_i)]\right)^2}{\sum_{v\in V\setminus\{v_{\max}\}}(w(v))^2}\right)\\ &\leq \exp\left(-\frac{2\cdot\left(\frac{11}{15}\cdot\frac{15}{64}\cdot W_A\right)^2}{\sum_{v\in V\setminus\{v_{\max}\}} w(v)\cdot\max_{v\in V\setminus\{v_{\max}\}} w(v)}\right)\\ &< \exp\left(-\frac{\frac{121}{2048}\cdot W_A^2}{W_A\cdot W_A/15}\right)\\ &< \exp\left(-\frac{1815}{2048}\right)\ .\end{aligned}$$

It can be numerically checked that $e^{-\frac{1815}{2048}} < 3/7$, and thus, by (4.27),

$$\mathbf{Pr}[w'(R_i) < W_A/16] < 3/7\ .$$

Therefore the probability that $w'(R_1) \geq W_A/16$ and $w'(R_3) \geq W_A/16$ is at least $1 - 3/7 - 3/7 = 1/7$.

Thus, in both cases the lemma holds. ■

Now we try to generalize the result to the whole sets of intervals. As for two (disjoint) intervals I_1, I_2, events that nodes are I_1-scattered and that they are I_2-scattered may be (and usually are) dependent, we cannot apply any standard concentration bounds. However, for our needs a reasoning based on Markov inequality is sufficient.

Lemma 4.30. *Fix any set S of k intervals $\{I_1, I_2, \ldots, I_k\}$. If at some time step t holds $V \sim \mathcal{D}$, then with probability $1/13$ nodes are $(S, \frac{1}{14})$-scattered.*

Proof. Choose randomly the positions of the nodes (each one independently according to a probability distribution from $\mathcal{D}$). For any interval $I_i \in S$, let X_i be the random variable denoting whether nodes are I_i-scattered. X_i is defined as

$$X_i = \begin{cases} W_A(I_i) & \text{if nodes are } I_i\text{-scattered}\\ 0 & \text{otherwise}\end{cases}$$

By Lemma 4.29, $\mathbf{E}[X_i] \geq \frac{1}{7}\cdot W_A(I_i)$. We want to prove that, with probability at least $1/13$, it holds that $\sum_{i=1}^k X_i \geq \frac{1}{14}\cdot\sum_{i=1}^k W_A(I_i)$. To achieve this, we define variables Y_i as

$$Y_i = W_A(I_i) - X_i\ .$$

Then $\mathbf{E}[Y_i] \leq \frac{6}{7} \cdot W_{\mathrm{A}}(I)$. By Markov inequality (see Appendix A.2) and linearity of expectation we obtain

$$\begin{aligned}\mathbf{Pr}\left[\sum_{i=1}^{k} X_i \leq \frac{1}{14} \cdot \sum_{i=1}^{k} W_{\mathrm{A}}(I_i)\right] &= \mathbf{Pr}\left[\sum_{i=1}^{k} Y_i \geq \frac{13}{14} \cdot \sum_{i=1}^{k} W_{\mathrm{A}}(I_i)\right] \\ &\leq \frac{\mathbf{E}\left[\sum_i Y_i\right]}{\frac{13}{14} \cdot \sum_i W_{\mathrm{A}}(I_i)} \\ &\leq \frac{12}{13}\ ,\end{aligned}$$

and therefore the lemma follows. ■

Finally, we may combine the lemmas above to prove Lemma 4.20

Proof of Lemma 4.20. Fix any configuration at the beginning of $\mathcal{E}_{j-2}$ and any narrow set $\mathcal{A}_j$ belonging to $(\mathcal{E}_{j-1} \uplus \mathcal{E}_j)$. By Corollary 4.28, we get that at the beginning of $\mathcal{A}_j$, it holds that $V \sim \mathcal{D}$. Then, by applying Lemma 4.30 with t equal to the first step of $\mathrm{span}(\mathcal{A}_j)$, the lemma immediately follows. ■

4.3.4 Proof of the Crucial Property

In this part show a lower bound for cost of any algorithm ALG in a narrow set $\mathcal{A}_j$, provided that the nodes are $(\mathcal{A}_j, \Omega(1))$-scattered and the movement of nodes within $\mathrm{span}(\mathcal{A}_j)$ is smooth. We note that bounds of this section hold in the worst case, not only in expectation. First, we concentrate on a single interval $I \in \mathcal{A}_j$. To lower-bound $C_{\mathrm{ALG}}(I)$ we observe that if $\mathrm{smooth}(\mathrm{span}(\mathcal{A}_j))$ holds, then the smoothness of movement impose a speed restriction, which creates a tradeoff: any algorithm either moves its page from one point of $\mathcal{X}$ to another slowly, or it has to pay much. To formalize this observation we need the following definition.

Definition 4.31 (Trails). *Fix any algorithm* ALG *and any interval I. By a trail $\mathcal{T}_{\mathrm{ALG}}(I)$ we denote the sequence of points of $\mathcal{X}$, in which* ALG *had its page in interval I. Formally, the trail in one step t, $\mathcal{T}_{\mathrm{ALG}}(t)$, is defined as*

- *the position of the node $P_{\mathrm{ALG}}(t)$ if* ALG *does not move,*
- *the sequence of points on the shortest path between $P_{\mathrm{ALG}}(t)$ and $P'_{\mathrm{ALG}}(t)$ if* ALG *moves.*

The trail $\mathcal{T}_{\mathrm{ALG}}(I)$ is just a concatenation of $\mathcal{T}_{\mathrm{ALG}}(t)$ for all time steps t from I.

We present two lemmas, lower-bounding the costs of any algorithm which wants to move along a constant fraction of the ring withing one interval from a narrow set $\mathcal{A}_j$. These lemmas show a lower bound for long and short diameters, respectively.

Lemma 4.32. *Fix any time interval I of length $\ell \leq \mathcal{N}_{\mathrm{I}}$ and assume that $B \geq \sqrt{D}$. If* smooth(I) *and* Alg*'s trail $\mathcal{T}_{\mathrm{ALG}}(I)$ contains two points from $\mathcal{X}$ lying at the distance of at least $B/8$, then*

$$C_{\mathrm{ALG}}(I) = \Omega(1/Q) \cdot D \cdot B \ .$$

Proof. We prove that Alg has to pay at least $\Omega(1/Q) \cdot D \cdot B$ just for moving its page. We group time steps of I into a sequence of m stages. In the i-th stage, Alg remains in one node denoted by w_i, for some number (denoted by X_i) of steps an then jumps to node w_{i+1}, along a distance Y_i. In the last stage, Alg possibly does not jump. This defines three sequences $(w_i)_{i=1}^m$, $(X_i)_{i=1}^m$, and $(Y_i)_{i=1}^m$. Obviously, $\sum_{i=1}^m X_i = \ell$.

Since we charge Alg only for jumps, we may assume that all w_i are different, i.e., $m \leq n$. Alg's cost for moving the page in I is at least

$$\begin{aligned} C^{\mathrm{move}}_{\mathrm{ALG}}(I) \ &\geq \left(\sum_{i=1}^{m-1} D \cdot (Y_i + 1) \right) + D \cdot Y_m \\ &= D \cdot (m-1) + D \cdot \sum_{i=1}^{m} Y_i \ . \end{aligned}$$

As $B \geq \sqrt{D}$, we have $Q_0 = \min\{\sqrt{D/B}, n\}$ and thus $Q = \max\{1, \min\{\sqrt{D/B}, n\}\}$. We consider two cases.

1. Alg makes many jumps, i.e., $m \geq \frac{B \cdot \sqrt{B}}{\sqrt{D}}$. Then even if we charge it only D for a single jump, the cost of Alg would be high.

   Since $m \leq n$, $n \geq \frac{B \cdot \sqrt{B}}{\sqrt{D}} \geq \sqrt{D/B}$, and thus

   $$\begin{aligned} C^{\mathrm{move}}_{\mathrm{ALG}}(I) \ &\geq \ \Omega(D \cdot m) \\ &= \ \Omega\left(D \cdot B / \sqrt{D/B}\right) \\ &= \ \Omega\left(\max\left\{\frac{1}{\sqrt{D/B}}, \frac{1}{n}\right\}\right) \cdot D \cdot B \\ &= \ \Omega(1/Q_0) \cdot D \cdot B \ . \end{aligned}$$

2. Alg makes small number of jumps, i.e., $m < \frac{B \cdot \sqrt{B}}{\sqrt{D}}$. Then we prove that some of this jumps have to be far, as the possibilities of using nodes movement to move from one point of the ring to another are limited.

   From the definition of a smooth configuration sequence follows that a node w_i can cover only a distance of $\beta \cdot \sqrt{X_i}$. Therefore, the total distance covered by Alg is at most $\sum_{i=1}^m \left(\beta \cdot \sqrt{X_i}\right) + \sum_{i=1}^m Y_i$, which, from the lemma assumption, is at least $B/8$. We can bound the term $\sum_{i=1}^m \sqrt{X_i}$ using the following argument.

Since $\sum_{i=1}^{m} X_i = \ell$, we may apply the Cauchy-Schwarz inequality [HLP88] (see Appendix A.2) for the sequences $(\sqrt{X_i})_{i=1}^{m}$ and $(1)_{i=1}^{m}$ to get that $\sum_{i=1}^{m} \sqrt{X_i} \leq \sqrt{m \cdot \ell}$. This actually shows that it would be optimal for the algorithm, if all X_i had the same length. To bound $m \cdot \ell$ we use

$$\ell \leq \mathcal{N}_{\mathrm{I}} \leq \frac{\min\{D, B^2\}}{(16\beta)^2 \cdot \max\left\{1, \min\left\{\sqrt{D/B}, n\right\}\right\}}$$

and consider two cases.

- If $n \geq \sqrt{D/B}$, then

$$\begin{aligned} m \cdot \ell &\leq \frac{B \cdot \sqrt{B}}{\sqrt{D}} \cdot \frac{D}{(16 \cdot \beta)^2 \cdot \max\{1, \sqrt{D/B}\}} \\ &\leq \frac{B^2}{(16 \cdot \beta)^2} \ . \end{aligned}$$

- If $n < \sqrt{D/B}$, then

$$\begin{aligned} m \cdot \ell &\leq n \cdot \frac{B^2}{(16 \cdot \beta)^2 \cdot n} \\ &= \frac{B^2}{(16 \cdot \beta)^2} \ . \end{aligned}$$

Therefore in either case $m \cdot \ell \leq \frac{B^2}{(16 \cdot \beta)^2}$, which implies $\sum_{i=1}^{m} Y_i \geq \frac{B}{8} - \beta \cdot \sqrt{m \cdot \ell} = \frac{B}{16}$. Hence, we get $C_{\mathrm{ALG}}^{\mathrm{move}}(I) \geq D \cdot \sum_{i=1}^{m} Y_i = \Omega(D \cdot B)$.

Thus, in either case we obtain that $C_{\mathrm{ALG}}^{\mathrm{move}}(I) = \Omega(\min\{1/Q_0, 1\}) \cdot D \cdot B = \Omega(1/Q) \cdot D \cdot B$. ■

Lemma 4.33. *Fix any time interval I of length $\ell \leq \mathcal{N}_{\mathrm{I}}$ and assume that $B \leq \sqrt{D}$. If* smooth(I) *and* ALG*'s trail $\mathcal{T}_{\mathrm{ALG}}(I)$ contains two points from $\mathcal{X}$ lying at the distance of at least $B/8$, then*

$$C_{\mathrm{ALG}}(I) = \Omega(1/Q) \cdot D \cdot B \ .$$

Proof. In this lemma we again bound the cost incurred by page movement only. We take the division into m stages and the definition of sequences $(w_i)_{i=1}^{m}$, $(X_i)_{i=1}^{m}$ and $(Y_i)_{i=1}^{m}$ from the previous lemma. As $B \leq \sqrt{D}$, we get $Q_0 = \min\{\sqrt{B}, n\}$, and thus $Q = Q_0$. We apply the reasoning similar to the one from the previous proof, and consider two cases

1. $m \geq \sqrt{B}$. In this case, $C_{\mathrm{ALG}}^{\mathrm{move}}(I) = \Omega(D \cdot \sqrt{B}) = \Omega(1/Q) \cdot D \cdot B$.

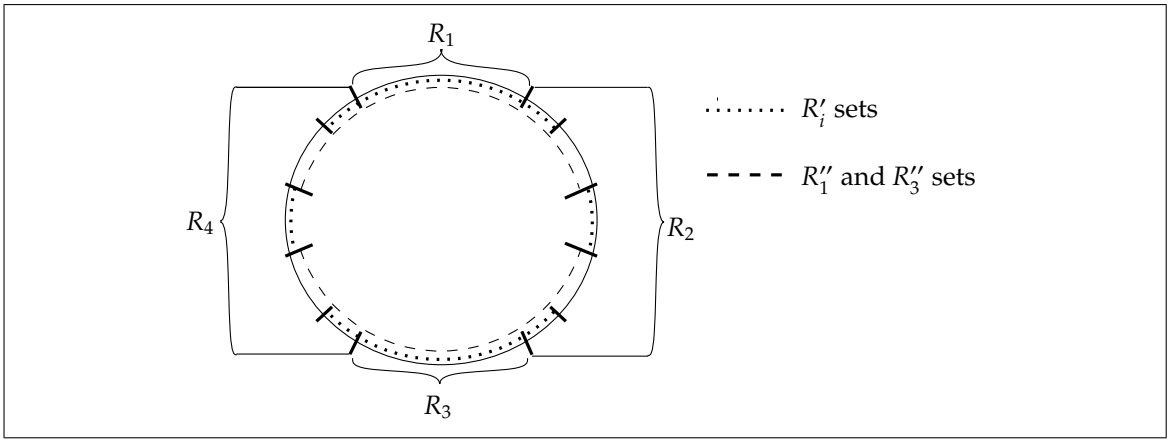

Figure 4.3: Changes in ring partitioning

2. $m \leq \sqrt{B}$. To get that the cost of Alg is $\Omega(D \cdot B)$, it is again sufficient that $\sqrt{m \cdot \ell} \leq \frac{B}{16 \cdot \beta}$. This follows easily, as

$$\begin{aligned} m \cdot \ell &\leq \min\left\{\sqrt{B}, n\right\} \cdot \frac{B^2}{(16 \cdot \beta)^2 \cdot Q} \\ &= \frac{B^2}{(16 \cdot \beta)^2} \ . \end{aligned}$$

Hence, in either case we obtain that $C^{\text{move}}_{\text{ALG}}(I) = \Omega(1/Q) \cdot D \cdot B$. ■

On the other hand, it can be proven that if nodes are I-distributed, then any algorithm either moves along long distances of it has to pay much for serving requests. As we observed in the two lemmas above, the former case incurs a high cost if the length of I is short. These observations create a basis for the next lemma.

Lemma 4.34. *Fix any algorithm* Alg *and any interval I from a narrow set $\mathcal{A}_j$. Assume that at the beginning of* span$(\mathcal{A}_j)$ *the nodes are I-scattered and the configuration sequence in* span$(\mathcal{A}_j)$ *is smooth. Then it holds that*

$$C_{\text{ALG}}(I) = \Omega(B \cdot W_{\text{A}}(I)) \ .$$

Proof. Since the nodes are I-scattered, it is possible to partition the ring into four segments R_1, R_2, R_3, R_4 (see Figure 4.3), such that $w_I(R_1), w_I(R_3) \geq W_{\text{A}}(I)/16$. In the picture, the segments R_i are drawn not proportionally, although they all contain $B/4$ points. Intuitively, since the configuration sequence in span$(\mathcal{A}_j)$ is smooth, this partition is approximately preserved within the whole span$(\mathcal{A}_j)$, and thus in I. Formally, we define segments R'_1, R'_2, R'_3, R'_4, R''_1, and R''_3 as shown in Figure 4.3 and described below.

- R'_1 (or respectively R'_3) has R_1 (or R_3) in its center and contains $B/4 + 2 \cdot B/32$ points.

- R''_1 (or respectively R''_3) has R_1 (or R_3) in its center and contains $B/4+2\cdot B/16$ points.

- R'_2 (or respectively R'_4) is located in the center of R_2 (or R_4) and contains $B/8$ points.

It follows from these definitions that each pair of points from different R'_i sets is separated by a distance at least $B/32$. Additionally, R''_1, R'_2, R''_3 and R'_4 create a partition of the whole ring.

Since the configuration sequence in I is smooth, each node initially placed in R_1 (or respectively in R_3) can move by at most $\beta\cdot\sqrt{|\mathrm{span}(\mathcal{A}_j)|}\leq B/32$, and thus remains within the set R'_1 (or R'_3) during the whole $\mathrm{span}(\mathcal{A}_j)$. This means that the number of requests issued in I at points from R'_1 (or R'_3) is at least $w_I(R_1)\geq W_{\mathrm{A}}(I)/16$.

Now, we claim that ALG has essentially four options. It either remains in $(R''_1\uplus R'_2\uplus R'_4)$, remains in $(R''_3\uplus R'_2\uplus R'_4)$, or its trail has to contain either all the points from R'_2 or all the points from R'_4. We consider two cases, the other two are symmetric.

1. ALG remains in $(R''_1\uplus R'_2\uplus R'_4)$ for the whole interval I. Then it has to pay at least $B/32$ for each of the requests issued at the points from R'_3, i.e., for at least $W_{\mathrm{A}}(I)/16$ requests. Thus, $C_{\mathrm{ALG}}(I)=\Omega(B\cdot W_{\mathrm{A}}(I))$.

2. The trail of ALG's page contains all the points from segment R'_2. Since $|I|\leq\mathcal{N}_{\mathrm{I}}$, we can apply Lemma 4.32 or Lemma 4.33 (depending on whether B is greater or smaller than $\sqrt{D}$) to get $C_{\mathrm{ALG}}(I)=\Omega(1/Q)\cdot D\cdot B$.

Summing up, $C_{\mathrm{ALG}}(I)=\Omega(B\cdot\min\{D/Q,W_{\mathrm{A}}(I)\})$. Since the length of I is at most $\mathcal{N}_{\mathrm{I}}<D/Q$, we finally get $C_{\mathrm{ALG}}(I)=\Omega(B\cdot W_{\mathrm{A}}(I))$. ■

A proof of the Crucial Property is an easy consequence of the lemma above.

Proof of Lemma 4.22 (Crucial Property). Since at the beginning of $\mathrm{span}(\mathcal{A}_j)$ nodes are $(\mathcal{A}_j,c)$-scattered, there exists a subset $\mathcal{A}'_j$, such that

$$\sum_{I\in\mathcal{A}'_j}W_{\mathrm{A}}(I)\geq c\cdot\sum_{I\in\mathcal{A}_j}W_{\mathrm{A}}(I)\ ,$$

and the nodes at the beginning of $\mathrm{span}(\mathcal{A}_j)$ are I-scattered for all $I\in\mathcal{A}'_j$. By Lemma 4.34, there exists a constant, which we denote by b_3, such that for any $I\in\mathcal{A}'_j$ it holds that

$C_{\mathrm{ALG}}(I) \geq b_3 \cdot B \cdot W_{\mathrm{A}}(I)$. Since all intervals from $\mathcal{A}'_j$ are disjoint, we obtain

$$\begin{aligned} C_{\mathrm{ALG}}(\mathcal{A}_j) &\geq C_{\mathrm{ALG}}(\mathcal{A}'_j) \\ &= \sum_{I \in \mathcal{A}'_j} C_{\mathrm{ALG}}(I) \\ &\geq \sum_{I \in \mathcal{A}'_j} b_3 \cdot B \cdot W_{\mathrm{A}}(I) \\ &\geq \sum_{I \in \mathcal{A}_j} c \cdot b_3 \cdot B \cdot W_{\mathrm{A}}(I) \ . \end{aligned}$$

This finishes the proof. ■

4.4 Extensions and conclusions

In this section we conclude with some possible extensions and open questions.

Extension to a multi-dimensional case

We may extend the definition of the scenario to the case, where our space $\mathcal{X}$ is not a ring but a linear array (this is equivalent to having one edge of the ring removed). Note that for the cost of serving requests by Maj, this transformation of the scenario does not change anything, since we already assumed that the cost of communication is the highest possible, i.e., B for each request. Also the probability that two nodes meet during the migration sequence changes at most by a constant. For bounding the cost of the optimal algorithm, we are still able to assure that in $\Theta(B^2)$ steps the random walk converges to almost uniform distribution on such an array. The division of the ring into four segments can be mapped into the division of an array (one segment is possibly split into two parts but this does not affect our argument). Since our lower bound on Opt does not depend on particular properties of the random walk, but only on the precondition of smooth configuration sequence, Maj is competitive also on a linear array.

We may also extend our scenario to the case where $\mathcal{X}$ is a d-dimensional torus or mesh, with $d > 1$. In that case, each node performs a random walk, which is a superposition of independent random walks in each dimension, each defined by (4.1). A two-dimensional example is presented in Figure 4.4; each of the 9 new possible node's positions has the same probability. The metric chosen for such a mesh could be either the Euclidean distance or the city metric (sum of distances in each dimension).

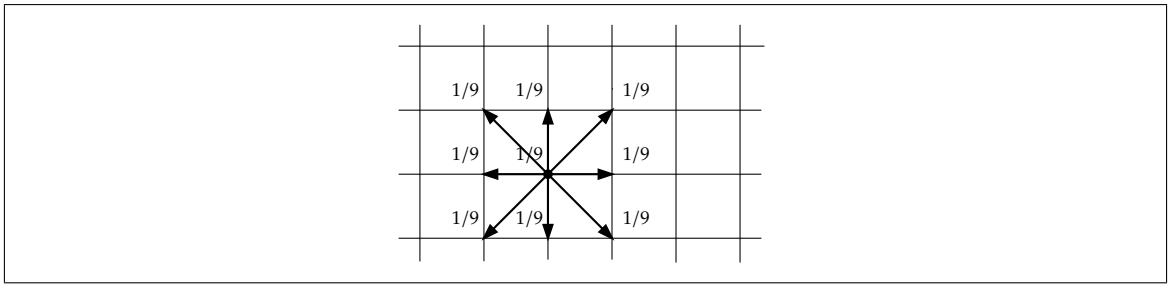

Figure 4.4: Random walk in two dimensions

It appears that for $B \geq \sqrt{D}$ we are able to guarantee a competitive ratio which is at most $O(d \cdot R)$. To prove this, it is sufficient to note that in our proofs for MAJ we always assumed the worst possible bound on the communication cost, B. This bound increases now to at most $d \cdot B$ (or $\sqrt{d} \cdot B$ for the Euclidean metric). On the other hand, for constructing a lower bound for OPT we may assume that communication along all the dimensions but the first one is free. This shows that the lower bound on cost of OPT derived for the one-dimensional case still applies.

Open problems

In this chapter we presented an $\tilde{O}(\min\{\sqrt[4]{D}, n\})$ algorithm for the case $B \geq \sqrt[3]{D}$. One of the possible research opportunities is to develop an algorithm which works also for the remaining, smallest diameters. We conjecture that it is possible to find an algorithm which in expectation performs better than the trivial $O(B)$-competitive algorithm DYN-M (see Section 2.4.2).

In our analysis, we lost a factor of $\Theta(Q)$ because the optimal algorithm was able to use its clairvoyance to exploit the nodes movement to transport its page from one point of the space to another paying less than $D \cdot B$ (or at least we were not able to show that it is not the case). If we assume that the height of the competitive ratio is not the deficiency of our analysis, it might be interesting to look for another models of mobility, where the algorithm would have more knowledge about the future movement of nodes. In particular, the *random waypoint model* (see [CBD02]), where each node picks randomly a destination and then goes there with a constant speed, seems to be a promising direction of the further research.

4.5 Proofs of technical claims

Proof of Claim 4.23. In the following we prove that for each $4 \leq \ell' \leq |I|/3$ we can find a subinterval $J \subseteq I$, such that $|J| = \ell'$ and $W_A(J) \geq \frac{1}{6} \cdot \frac{\ell'}{|I|} \cdot W_A(I)$. Therefore, if $\ell \leq |I|/3$,

then the lemma follows. Otherwise, we can choose any interval J' containing found interval J, such that $|J'| = \ell$. Then

$$W_{\mathrm{A}}(J') \geq W_{\mathrm{A}}(J) \geq \frac{1}{6} \cdot \frac{\ell'}{|I|} \cdot W_{\mathrm{A}}(I) \geq \frac{1}{18} \cdot \frac{\ell}{|I|} \cdot W_{\mathrm{A}}(I) \ ,$$

and the lemma follows.

Thus, in the following we assume that $\ell' \leq |I|/3$. Let $v_{\max}$ be the heaviest node in I. We may assume that $w_I(v_{\max}) < |I|$, otherwise the lemma follows trivially. We number time steps within I from 1 to $|I|$. For all $1 \leq i \leq |I| - \ell' + 1$, we define J_i as the subinterval of I, of length ℓ', starting at time step i. Let q and r be defined as a unique integer solution of $|I| = q \cdot \ell' + r$, where $0 \leq r < \ell'$. We know that $q \geq 3$.

Since any two values from the sequence $w_{J_1}(v), w_{J_2}(v), \ldots, w_{J_{|I|-\ell'+1}}(v)$ can differ by at most 1, the following fact, which we call a *continuity property*, follows. Fix any node v. If there exist J_a and J_b, such that $k_a \leq w_{J_a}(v)$, and $w_{J_b}(v) \leq k_b$, and there is at least one integer in the interval $[k_a, k_b]$, then there exists J_c, such that $w_{J_c}(v) \in [k_a, k_b]$.

We use the continuity property to prove the lemma. We consider two cases.

1. If $w_I(v_{\max}) \geq \frac{5}{8} \cdot |I|$, then for the sake of this case we treat all the nodes except $v_{\max}$ as one node $v^{\mathrm{c}}_{\max}$, and thus $w_I(v^{\mathrm{c}}_{\max}) \leq \frac{3}{8} \cdot |I|$. We prove that there exists an interval J_c of length ℓ', such that $w_{J_c}(v) \in \left[\frac{1}{2} \cdot \frac{\ell'}{|I|} \cdot w_I(v^{\mathrm{c}}_{\max}),\ \ell'/2\right]$. In such an interval $v_{\max}$ is the heaviest node, and thus $W_{\mathrm{A}}(J_c) = w_{J_c}(v) \geq \frac{1}{2} \cdot \frac{\ell'}{|I|} \cdot W_{\mathrm{A}}(I)$. To prove the existence of J_c we use the continuity property, and thus we have to assure that the following three conditions are met.

   - There exists an interval J_b, such that $w_{J_b}(v^{\mathrm{c}}_{\max}) \leq \ell'/2$. From the relaxed average argument (see Lemma 4.35 below) follows the existence of J_b, such that $w_{J_b}(v^{\mathrm{c}}_{\max}) \leq \frac{4}{3} \cdot \frac{\ell'}{|I|} \cdot w_I(v^{\mathrm{c}}_{\max}) \leq \ell'/2$.
   - There exists an interval J_a, such that $w_{J_a}(v^{\mathrm{c}}_{\max}) \geq \frac{1}{2} \cdot \frac{\ell'}{|I|} \cdot w_I(v^{\mathrm{c}}_{\max})$. We consider two cases. If $\frac{\ell'}{|I|} \cdot w_I(v^{\mathrm{c}}_{\max}) < 2$, then for J_a we may choose any interval in which there is one request not at $v_{\max}$. Then $w_{J_a}(v^{\mathrm{c}}_{\max}) \geq 1 \geq \frac{1}{2} \cdot \frac{\ell'}{|I|} \cdot w_I(v^{\mathrm{c}}_{\max})$. Otherwise, we have $\frac{\ell'}{|I|} \cdot w_I(v^{\mathrm{c}}_{\max}) \geq 2$, in which case by the relaxed average argument (see Lemma 4.35 below), $w_{J_a}(v^{\mathrm{c}}_{\max}) \geq \frac{3}{4} \cdot \frac{\ell'}{|I|} \cdot w_I(v^{\mathrm{c}}_{\max}) \geq \frac{1}{2} \cdot \frac{\ell'}{|I|} \cdot w_I(v^{\mathrm{c}}_{\max})$.
   - The interval $\left[\frac{1}{2} \cdot \frac{\ell'}{|I|} \cdot w_I(v^{\mathrm{c}}_{\max}),\ \ell'/2\right]$ contains at least one integer. This follows since $w_I(v^{\mathrm{c}}_{\max}) \leq \frac{3}{8} \cdot |I|$ and $\ell' \geq 4$.

2. If $w_I(v_{\max}) < \frac{5}{8} \cdot |I|$, then we prove that there exists an interval J, such that $W_{\mathrm{A}}(J) \geq \ell'/6$. This obviously implies that $W_{\mathrm{A}}(J) \geq \frac{1}{6} \cdot \frac{\ell'}{|I|} \cdot W_{\mathrm{A}}(I)$.

By $v_{\max}(i)$ we denote the heaviest node in J_i (with ties broken arbitrarily). If there exists an interval J_i such that $w_{J_i}(v_{\max}(i)) \leq \frac{5}{6} \cdot \ell'$, then $W_A(J_i) \geq \ell'/6$ and the lemma holds. Assume the contrary, i.e., $w_{J_i}(v_{\max}(i)) > \frac{5}{6} \cdot \ell'$ for all i. Note that all $v_{\max}(i)$ cannot be the same node, because the relaxed sum argument (see Lemma 4.35 below) would imply $w_I(v_{\max}) > \frac{3}{4} \cdot \frac{5}{6} \cdot |I| = \frac{5}{8} \cdot |I|$. Therefore, there exist two consecutive intervals J_i and J_{i+1}, such that $v_{\max}(i) \not\equiv v_{\max}(i+1)$. Consider their overlap $J_i \cap J_{i+1}$. The weights of both $v_{\max}(i)$ and $v_{\max}(i+1)$ in this interval are at least $\frac{5}{6}\ell' - 1$, and therefore, the total weight accumulated in $J_i \cap J_{i+1}$ is $2 \cdot (\frac{5}{6}\ell' - 1)$. But the length of $J_i \cap J_{i+1}$ is $\ell' - 1$, which implies $\ell' \leq \frac{3}{2}$, a contradiction.

Therefore, in either of the two cases above, we are able to find an interval J, which has the desired auxiliary weight of at least $\frac{1}{6} \cdot \frac{\ell'}{|I|} \cdot W_A(I)$. ■

Lemma 4.35 (Relaxed Sum and Average Argument). *Let f be any real valued function defined on a sequence $S = (s_1, s_2, \ldots, s_\ell)$ of length ℓ. We extend f to any subset $X \subseteq S$, i.e., $f(X) = \sum_{s_i \in X} f(s_i)$. By a subsequence of S we mean a* contiguous subsequence *of S. Let $\ell' \leq \ell$ be any integer and let $q := \lfloor \ell/\ell' \rfloor \geq 1$. For all $1 \leq i \leq \ell - \ell' + 1$, let $S_i := (s_i, s_{i+1}, \ldots, s_{i+\ell-1})$. S_i are all possible subsequences of length ℓ'. Then it holds that*

(i) (Sum) If all $f(S_i) > k$ for some k, then $f(S) > q \cdot k$.

(ii) (Lower Average) There exists a subsequence $S_{\text{low}} \subseteq S$ of length ℓ', such that

$$f(S_{\text{low}}) \leq \frac{q+1}{q} \cdot \frac{\ell'}{\ell} \cdot f(S) \ .$$

(iii) (Upper Average) There exists a subsequence $S_{\text{up}} \subseteq S$ of length ℓ', such that

$$f(S_{\text{up}}) \geq \frac{q}{q+1} \cdot \frac{\ell'}{\ell} \cdot f(S) \ .$$

Proof. We consider a set of subintervals $\{T_i\}_{i=0}^{q}$, where $T_i := S_{1+i\cdot\ell'}$ for $0 \leq i < q$, and $T_q := S_{\ell-\ell'+1}$.

- For proving the *relaxed sum argument* we note that T_i are disjoint for all $0 \leq i < q$, and thus $f(S) \geq \sum_{i=0}^{q-1} f(T_i) > q \cdot k$.
- For *relaxed average arguments*, assume that for all $0 \leq i \leq q$ it holds that $f(T_i) < \frac{q}{q+1} \cdot \frac{\ell'}{\ell} \cdot f(S)$. Since $\bigcup_{i=0}^{q} T_i$ covers the whole S, $f(S) \leq \sum_{i=0}^{q} f(T_i) < (q+1) \cdot \frac{q}{q+1} \cdot \frac{\ell'}{\ell} \cdot f(S) \leq f(S)$, which is a contradiction. Similarly, assume that for all S_i it holds that $f(S_i) > \frac{q+1}{q} \cdot \frac{\ell'}{\ell} \cdot f(S)$. Then from the relaxed sum argument we have $f(S) > (q+1) \cdot \frac{\ell'}{\ell} \cdot f(S) > f(S)$, which is also a contradiction.

Note that since $q \geq 1$, for any ℓ' we can find S_{low} and S_{up}, such that $f(S_{\text{low}}) \leq 2 \cdot \frac{\ell'}{\ell} \cdot f(S)$ and $f(S_{\text{up}}) \geq \frac{1}{2} \cdot \frac{\ell'}{\ell} \cdot f(S)$. ■

Proof of Lemma 4.8. For any i, we define Z_i as a random variable on Ω_i. For any $\omega_i \in \Omega_i$, let

$$Z_i(\omega_i) := \max_{(\omega'_1,\omega'_2,\ldots,\omega'_{i-1},\omega'_{i+1},\ldots,\omega'_n)\in\Omega_1\times\Omega_2\times\ldots\times\Omega_{i-1}\times\Omega_{i+1}\times\ldots\times\Omega_n} \{ X_i(\omega'_1, \omega'_2, \ldots, \omega'_{i-1}, \omega_i, \omega'_{i+1}, \ldots, \omega'_n) \} \ .$$

Since Y_i stochastically dominates X_i with respect to Ω_i, it also stochastically dominates variable Z_i. Moreover, all Z_i are independent, and thus $\sum_{i=1}^n Y_i$ stochastically dominates $\sum_{i=1}^n Z_i$. It remains to show that $\sum_{i=1}^n Z_i$ stochastically dominates $\sum_{i=1}^n X_i$. In fact, we may even show that $\sum_{i=1}^n Z_i$ is greater than $\sum_{i=1}^n X_i$ for any random choice of $\omega = (\omega_1, \omega_2, \ldots, \omega_n) \in \Omega_1 \times \Omega_2 \times \ldots \times \Omega_n$. For any such ω it holds that

$$\begin{aligned}
\left(\sum_{i=1}^n Z_i\right)(\omega) &= \sum_{i=1}^n Z_i(\omega) \\
&= \sum_{i=1}^n Z_i(\omega_i) \\
&\geq \sum_{i=1}^n X_i(\omega) \\
&= \left(\sum_{i=1}^n X_i\right)(\omega) \ .
\end{aligned}$$

This finishes the proof. ■

CHAPTER 5

Stochastic Requests Scenario

In this chapter we consider a scenario symmetric to the Brownian motion one. We assume that the network mobility is adversarial, but the requests are generated randomly by a stochastic process. In each step the processor issuing the request is chosen independently at random, according to some fixed probability distribution π. This is a natural situation, if we know that the mobile nodes' accesses to the page appear periodically with some given frequencies.

In this scenario we generalize the cost function to

$$c_t(v_a, v_b) = \begin{cases} [d_t(v_a, v_b)]^\alpha + 1 & \text{if } v_a \not\equiv v_b \\ 0 & \text{if } v_a \equiv v_b \end{cases} \tag{5.1}$$

for any integer exponent α. This make sense, if the nodes are mobile stations communicating by means of radio with adjustable transmission power, and we choose α to be the propagation exponent of the medium (for example equal to 2 for an ideally free space, see [Rap96]). In such a case, the communication cost between two nodes defined by (5.1) is proportional to the energy required to send a message. Then the DPM problem's objective becomes equivalent to minimizing the total energy consumption in the system, which consists of mobile stations moving with a constant speed.

Scenario definition

Formally, we construct the *stochastic requests scenario* of the DPM problem as follows. First, the whole configuration sequence $(C_t)_t$, including the initial configuration C_0, is created by the δ-restricted network adversary, for some constant δ. The request adversary chooses the probability distribution over all indices of the nodes $\pi : [n] \to (0, 1)$. (Later we show how to extend our argument to handle distributions $\pi : [n] \to [0, 1]$). In time step $t \geq 1$ the following happens.

1. The positions of the nodes in this time step are defined by C_t.

2. A request is generated randomly, i.e., σ_t is chosen randomly according to the distribution π and node v_{σ_t} issues a request. For clarity, we abuse the notation and write *node* σ_t instead of *node* v_{σ_t}.

In the remaining part of the time step t, we incorporate the casual model of the DPM, i.e., the algorithm has to serve the request, and it may optionally move the page.

Performance metric

As in the previous chapter, to analyze the performance in the stochastic requests scenario, we adapt classical competitive analysis [ST85, BE98] for the model where the input sequence is created both by the adversary and the stochastic process. We say that an algorithm ALG achieves competitive ratio $\mathcal{R}$ (or is $\mathcal{R}$-competitive) with probability p, if there exists an integer $T_{\min}$, such that for all configuration sequences $(C_t)_t$ of length $T \geq T_{\min}$ and all probability distributions π, it holds that

$$\mathbf{Pr}_{(\sigma_t)_t}\left[C_{\mathrm{ALG}}((C_t,\sigma_t)_t) \leq \mathcal{R} \cdot C_{\mathrm{OPT}}((C_t,\sigma_t)_t) \right] \geq p \ , \tag{5.2}$$

where $C_{\mathrm{ALG}}((C_t,\sigma_t)_t)$ and $C_{\mathrm{OPT}}((C_t,\sigma_t)_t)$ are costs of ALG and the optimal offline algorithm, respectively. The probability is taken over all possible random choices made for generating sequence (σ_t).

Similarly to the previous chapter, we define an additional notion of an expected competitive ratio. We say that an algorithm achieves an expected competitive ratio of $\mathcal{R}$ (or is $\mathcal{R}$-competitive in expectation), if there exists an integer $T_{\min}$, such that for all sequences $(C_t)_t$ of length $T \geq T_{\min}$ and all π, it holds that

$$\mathbf{E}_{(\sigma_t)_t}\left[\frac{C_{\mathrm{ALG}}((C_t,\sigma_t)_t)}{C_{\mathrm{OPT}}((C_t,\sigma_t)_t)} \right] \leq \mathcal{R} \ . \tag{5.3}$$

To be consistent with the previous definition, we assume that the ratio is 1, if both $C_{\mathrm{ALG}}((C_t,\sigma_t)_t)$ and $C_{\mathrm{OPT}}((C_t,\sigma_t)_t)$ are equal to 0. However, we do not permit situations where $C_{\mathrm{OPT}}((C_t,\sigma_t)_t) = 0 \neq C_{\mathrm{ALG}}((C_t,\sigma_t)_t)$.

In the Brownian motion scenario, we were able to always guarantee that in the worst case the cost of the algorithm was at most $O(B)$ times more than its expected cost. Note that in the stochastic scenario the maximum extent of the network might be unbounded. This means that the competitive ratio $\mathcal{R}$ achieved with high probability does not directly imply the expected competitive ratio $O(\mathcal{R})$, since the former notion does not explicitly exclude inputs, on which the competitive ratio is very large or even infinite.

Results

In Section 5.2 we design and analyze a deterministic algorithm Move-To-First-Request (Mtfr) for the DPM problem. We prove that in the stochastic requests scenario, its competitiveness is constant[1] with high probability, for any constant-restricted network adversary. In this context, high probability means that for any constant γ we can achieve probability $1 - O(D^{-\gamma})$ if the input sequence is sufficiently long. Additionally, we prove that in the worst case (occurring with negligible probability) the competitive ratio of Mtfr is finite. These results assure that the performance of Mtfr stabilizes with time, and additionally allows us to conclude that in expectation its competitive ratio is also constant. The achieved competitiveness is strict, i.e., the additive constant A in the definition of the algorithm's cost (see (1.8) on page 10) is zero.

To show that the model does not become trivial, when we generalize the cost function (i.e., for $\alpha > 1$), in the next section we present a lower bound of $\Omega(\min\{D^{\frac{\alpha}{\alpha+1}}, \lambda^{\alpha}\})$, which holds in the adversarial model even for two-node networks and oblivious adversaries.

5.1 Lower bound for the extended cost model

In this subsection we go back for a while to the adversarial scenario and we consider it with the generalized cost function. We prove that even against an oblivious adversary the lower bound on the competitive ratio of any algorithm is at least $\Omega(\min\{D^{\frac{\alpha}{\alpha+1}}, \lambda^{\alpha}\})$. The construction is a slight reformulation of the lower bound for two-node networks presented in Section 2.3.2.

Again, we resort to Yao min-max principle, and construct a probability distribution, such that any deterministic online algorithm has a high cost (compared to an optimal cost) on a randomly chosen input. The construction works for two-node networks. For larger networks, the adversary can give requests only at v_1 and v_2, and put the other nodes exactly in the same point of space $\mathcal{X}$ as v_1. Then, for any algorithm Alg that uses these additional nodes, there exists an algorithm Alg′ which uses v_1 instead, and has cost at most as large as Alg.

The construction divides time into phases, each phase consisting of an expanding part (B_{exp} steps), a main part (D steps), and a contracting part (B_{exp} steps), exactly as it was done in the construction of Section 2.3.2 and in Figure 2.3 on page 24. The construction of requests is also identical to that one, i.e., all requests in the expanding part are at v_1, in contracting part at v_2, and the requesting node for the whole main part is chosen uniformly at random from $\{v_1, v_2\}$. The only difference between the original construction is that we now choose $B_{\text{exp}} = \min\{D^{\frac{1}{\alpha+1}}, \lambda\}$.

[1] This constant depends on α, but we assume that α is constant for all practical applications.

In the similar manner as in the proof presented in Section 2.3.2, we may show the following lemma.

Lemma 5.1. *Let P be a phase, randomly generated as described above. Then for any deterministic algorithm* DET *for the DPM problem,*

$$\mathbf{E}_P[C_{\mathrm{DET}}(P)] \geq \Omega(B_{\mathrm{exp}}^{\alpha}) \cdot \mathbf{E}_P[C_{\mathrm{OPT}}(P)] \ ,$$

where the expectation is taken over possible random choices of P.

Proof. The optimal offline algorithm may move at the beginning of the phase (paying D) not to pay anything in the main part. Additionally, OPT has to pay $\sum_{t=0}^{B_{\mathrm{exp}}-1}(t^{\alpha}+1) = \Theta((B_{\mathrm{exp}})^{\alpha+1}) = \Theta(D)$ for serving all the requests either in the expanding part or in the contracting part. Therefore, in total, $C_{\mathrm{OPT}}(P) = \Theta(D)$.

On the other hand, a deterministic online algorithm DET has to decide in the last step of the expanding part whether to end this step at node v_1 or v_2. Independently of DET's choice, with probability $1/2$, all the next D requests in the main part are given at the opposite node. In that case, if DET moves the page within the main part, then it pays at least $D \cdot (B_{\mathrm{exp}}^{\alpha}+1)$ for moving the page. Otherwise, it pays $D \cdot (B_{\mathrm{exp}}^{\alpha}+1)$ for serving the requests during this part. Thus, the expected cost of DET in P is at least $\frac{1}{2} \cdot D \cdot B_{\mathrm{exp}}^{\alpha}$.

Comparing these two bound completes the proof. ■

By generating sufficiently many phases we may use the lemma above to prove a lower bound on any randomized algorithm in the extended cost model.

Theorem 5.2. *Consider any randomized algorithm* ALG *which is c-competitive against an oblivious adversary in the adversarial scenario of the DPM problem with generalized cost function. Then $c = \Omega(\min\{D^{\frac{\alpha}{\alpha+1}}, \lambda^{\alpha}\})$, where λ is the maximal extent of the network, and α is the integer exponent in the cost function.*

Proof. By constructing sufficiently many random phases, we may achieve an arbitrarily high cost on OPT. Thus, by using linearity of expectation and applying the Yao min-max principle to the result of the lemma above, the theorem follows. ■

5.2 Algorithm MTFR

In this section we present a deterministic algorithm MTFR, which, on sufficiently long input sequences, achieves a constant competitive ratio in expectation and with high probability.

Before we formally specify the algorithm, we try to give an informal description of its desired properties. Assume that the algorithm knows the probability distribution π beforehand. This assumption makes sense, as on the long run, the algorithm may learn π, or at least approximate this distribution with an arbitrarily high accuracy. Then the algorithm may choose a node v^* that has the highest chance of being picked as the requesting node, i.e., a node v_i, such that $\pi(i)$ is maximized. The algorithm moves to v^* at the very beginning, and remains there for the whole request sequence.

The adversary could easily construct a hard input instance, i.e., a distribution π and the configuration sequence $(C_t)_t$, such that if the request sequence $(\sigma_t)_t$ is generated randomly according to π, the algorithm above has a competitive ratio of at least $\Omega(n)$. Briefly, a node v_1 has probability of being chosen set to $\pi(1) = \frac{2}{n}$, and the remaining probability is equally distributed among other nodes. Moreover, the remaining nodes occupy one point of the space, while v_1 is placed at the distance s from them. Then a randomly generated request, in expectation, incurs cost $(1 - \pi(1)) \cdot s = \Omega(s)$ on our algorithm, while the expected cost of an algorithm which remains for the whole input sequence at v_2 would be at most $\pi(1) \cdot s + (1 - \pi(1)) \cdot 1 = \Omega(s/n + 1)$. By choosing a large s, the expected competitive ratio of the algorithm remaining at v_1 would be $\Omega(n)$.

While developing algorithms, which try to recognize clusters of nodes having large probabilities $\pi(i)$ of being chosen, and then moving to these clusters might be tempting, there exists a much simpler approach. Although we want our algorithm to be deterministic, we may use the random bits delivered with the request stream $(\sigma_t)_t$. This allows us to choose a node v^* randomly according to the probability distribution π and then move to v^*. However, if we do this only once for the whole input sequence, we may prove that the competitive ratio is low in expectation, but we have no chance to guarantee it with high probability. Thus, the solution would be to choose a new node v^* from time to time, move to v^* and remain there for some number of steps. We have to choose carefully the number of steps between two movements, because when the time interval between them is too short, the algorithm may suffer from high costs of frequent page movements.

Additionally, we note that we do not have to learn the distribution π to be able to choose nodes according to this distribution, because the request adversary does this job for us, choosing σ_t for each step independently, with the distribution π. This construction rationale is formalized below.

The algorithm

Let Mtfr be a deterministic algorithm which divides time steps into phases of length $\ell := D^{\alpha+1}$. Without loss of generality, we may assume that D is even, and so is ℓ. In the first step of each phase, after serving a request, Mtfr moves the page to the node which

issued a request. In this and the next two subsections we aim to prove the following theorem.

Theorem 5.3. MTFR *is* $O(1)$*-competitive, with high probability, in the stochastic requests scenario of the DPM.*

Let us introduce some notation. Let $\pi_{\min} = \min_i \{\pi(i)\}$. Let

$$T_\gamma = m_\gamma \cdot \ell \ , \tag{5.4}$$

for

$$m_\gamma = \left(c_T \cdot \gamma \cdot \frac{\ln D}{\pi_{\min}^2} \right)^{\alpha+1} \cdot \left(\frac{2 \cdot \ell}{\alpha} \right)^{\alpha}, \tag{5.5}$$

where c_T is a constant, which will be specified later. In the following, we prove that MTFR is $O(1)$-competitive with probability $1 - O\left(D^{-\gamma}\right)$ if run for $T \geq T_\gamma$ time steps. Later, we prove that on inputs of length $T \geq T_{(\alpha+1)^2}$, MTFR achieves expected competitive ratio of $O(1)$.

The brief idea of the proof of Theorem 5.3 is as follows. We consider an input sequence $\mathcal{I}$ of length T. By $\mathcal{P}$ we denote a set of all finished phases of $\mathcal{I}$, i.e., phases of length ℓ. At the end of $\mathcal{I}$ we might also have one unfinished phase; we denote it by p_{last}. This way $\mathcal{I} = \left(\biguplus_{p \in \mathcal{P}} p\right) \uplus p_{\text{last}}$.

For clarity, the proof was divided into subsections. In this subsection, we show some general relations concerning the algorithm MTFR, and we define $\mathcal{K}_p$ as the average cost of communication in phase p. In Section 5.2.1 we show an $\Omega(\sum_{p \in \mathcal{P}} \mathcal{K}_p)$ lower bound on the cost of OPT, which holds with probability $1 - O(D^{-\gamma})$ on input sequences of length at least T_γ. In Section 5.2.2 we show an analogous upper bound of $O(\sum_{p \in \mathcal{P}} \mathcal{K}_p)$ for the cost of MTFR, holding also with probability $1 - O(D^{-\gamma})$. Clearly, combining these two results yields a constant factor gap between costs of MTFR and OPT, which holds with high probability.

We prove a constant competitiveness against a $\frac{1}{2 \cdot \alpha}$-restricted network adversary. The proof for any constant-restricted network adversary follows from the Reduction Lemma presented in Section 2.2. Note that since the adversary is $\frac{1}{2 \cdot \alpha}$-restricted, the distance between any two nodes can change only by at most $1/\alpha \leq 1$ per round.

Skewed Triangle Inequality

A property that we heavily exploited in the analysis of the algorithms for the adversarial scenario was the triangle inequality. For the case $\alpha = 1$, it holds not only for the distances, but also for the costs of communication. It appears that up to a constant factor (depending on α), it holds also for $\alpha > 1$. In particular, we are able to prove the following lemma.

Lemma 5.4 (Skewed Triangle Inequality). *For any sequence of $k+1$ nodes $v_{i_1}, v_{i_2}, \ldots, v_{i_{k+1}}$ and any time step t it holds that*

$$\sum_{j=1}^{k} c_t(v_{i_j}, v_{i_{j+1}}) \geq \frac{1}{k^{\alpha-1}} \cdot c_t(v_{i_1}, v_{i_{k+1}}) \ .$$

Note that for $k = 1$ and $\alpha = 1$ this describes the casual triangle inequality on non-generalized cost function. To prove it, we first introduce a technical claim, based on Hölder's Inequality [HLP88] (see Appendix A.2) and proven at the end of this chapter.

Claim 5.5. *For any sequence of k non-negative numbers $a_1, a_2, \ldots, a_k$ and any integer $s \geq 1$, it holds that*

$$\left(\sum_i a_i\right)^s \leq \left(\sum_i a_i^s\right) \cdot k^{s-1} \ .$$

In the remaining part of this chapter, we mark the inequalities which follow from this claim with $\overset{(\mathrm{H})}{\leq}$.

Proof of Lemma 5.4. If $v_{i_1} \equiv v_{i_{k+1}}$, then $c_t(v_{i_1}, v_{i_{k+1}}) = 0$, and thus the inequality follows trivially. Otherwise, there exists $j \in \{1, \ldots, k\}$, such that $v_{i_j} \not\equiv v_{i_{j+1}}$, i.e., $c_t(v_{i_j}, v_{i_{j+1}}) = 1 + [d_t(v_{i_j}, v_{i_{j+1}})]^\alpha$. In this case,

$$\begin{aligned}
\sum_{j=1}^{k} c_t(v_{i_j}, v_{i_{j+1}}) &\geq 1 + \sum_{j=1}^{k} \left[d_t(v_{i_j}, v_{i_{j+1}})\right]^\alpha \\
&\overset{(\mathrm{H})}{\geq} 1 + \frac{1}{k^{\alpha-1}} \cdot \left(\sum_{j=1}^{k} d_t(v_{i_j}, v_{i_{j+1}})\right)^\alpha \\
&\geq 1 + \frac{1}{k^{\alpha-1}} \cdot [d_t(v_{i_1}, v_{i_{k+1}})]^\alpha \\
&\geq \frac{1}{k^{\alpha-1}} \cdot c_t(v_{i_1}, v_{i_{k+1}}) \ .
\end{aligned}$$

This concludes the proof. ∎

General relations

Let K_t be the *average cost* of sending a unit of data between two nodes in time step t, i.e.,

$$K_t := \sum_{i=1}^{n} \sum_{j=1}^{n} \pi(i) \cdot \pi(j) \cdot c_t(v_i, v_j) \ . \tag{5.6}$$

K_t is the cost of communication between two nodes chosen randomly with distribution π. We note that some of the terms contributing to the average cost K_t are always 0, because we counted also the nodes sending data to themselves. Therefore, we have

$$\begin{aligned} K_t &= \sum_i \sum_j \pi(i) \cdot \pi(j) \cdot c_t(v_i, v_j) \\ &= \sum_i \pi(i)^2 \cdot \underbrace{c_t(v_i, v_i)}_{=0} + \sum_i \sum_{j \neq i} \pi(i) \cdot \pi(j) \cdot (1 + d_t^\alpha(v_i, v_j)) \ . \end{aligned}$$

Let $\beta = \sum_i \sum_{j \neq i} \pi(i)\pi(j)$. It follows that $K_t \geq \beta$ for any time step t. For the further considerations we need to relate the K_t values in any two consecutive steps. Let $k_t = \sqrt[\alpha]{K_t/\beta}$, i.e.,

$$K_t = \beta \cdot k_t^\alpha \ . \tag{5.7}$$

In a sense, k_t behaves like an *average distance* occurring between the nodes.[2] In particular, the following claim can be proven by means of the Jensen's Inequality [HLP88] (see Section 5.4 for the proof).

Claim 5.6. *For all t, $k_t \geq 1$, and $|k_{t+1} - k_t| \leq 1/\alpha$.*

As an immediate consequence we get

$$\frac{K_{t+1}}{K_t} \ \leq \ \frac{\left(k_t + \frac{1}{\alpha}\right)^\alpha}{k_t^\alpha} \ \leq \ \left(1 + \frac{1}{\alpha}\right)^\alpha \ \leq \ \mathrm{e} \ . \tag{5.8}$$

The same holds for the quotient of costs of communication between any pair of nodes in two consecutive steps, i.e., for any v_a and v_b it holds that

$$\frac{c_{t+1}(v_a, v_b)}{c_t(v_a, v_b)} \ \leq \ \mathrm{e} \ . \tag{5.9}$$

Let $c_t^{\max}$ be the maximum cost of communication between two nodes in time step t. We may also establish a relation between the average and the worst-case cost of communication in any round t, in essence showing that they may differ by at most $O(\pi_{\min})$.

Lemma 5.7. *For any time step t, it holds that*

$$K_t \geq \Omega(1) \cdot \pi_{\min} \cdot c_t^{\max} \ .$$

[2] If we chose $K_t = \beta \cdot (1 + k_t^\alpha)$, then k_t would be α-power weighted mean of the distances occurring in the network. However for our considerations, the form presented in (5.7) is more convenient.

Proof. Fix any time step t. Without loss of generality we can assume that the largest communication cost occurs between the nodes v_1 and v_2, i.e., $c_t(v_1, v_2) = c_t^{\max}$. Then we have

$$\begin{aligned} K_t &= \sum_i \sum_j \pi(i) \cdot \pi(j) \cdot c_t(v_i, v_j) \\ &\geq \pi(1)\pi(2)c_t(v_1, v_2) + \pi(2)\pi(1)c_t(v_2, v_1) + \sum_{i=3}^{n} [\pi(i)\pi(1)c_t(v_1, v_i) + \pi(i)\pi(2)c_t(v_i, v_2)] \\ &\geq \pi(1) \cdot \pi_{\min} \cdot c_t(v_1, v_2) + \pi(2) \cdot \pi_{\min} \cdot c_t(v_1, v_2) + \sum_{i=3}^{n} \pi(i) \cdot \pi_{\min} \cdot [c_t(v_i, v_1) + c_t(v_i, v_2)] \end{aligned}$$

By Lemma 5.4, $c_t(v_1, v_i) + c_t(v_i, v_2) \geq \frac{1}{2^{\alpha-1}} \cdot c_t(v_1, v_2)$, and thus

$$\begin{aligned} K_t &\geq \frac{1}{2^{\alpha-1}} \cdot \pi_{\min} \cdot c_t(v_1, v_2) \cdot \sum_{i=1}^{n} \pi(i) \\ &= \frac{1}{2^{\alpha-1}} \cdot \pi_{\min} \cdot c_t^{\max} \ , \end{aligned}$$

which finishes the proof. ∎

Average phase cost

For any phase p we define the *average phase cost* $\mathcal{K}_p = \sum_{t \in p} K_t$. In the next two subsections we show how to relate it to the Mtfr's and Opt's cost. In particular, we prove the two following lemmas.

Lemma 5.8. *For any input sequence* $\mathcal{I} = (\biguplus_{p \in \mathcal{P}} p) \uplus p_{\text{last}}$ *of length at least* T_γ,

$$C_{\text{OPT}}(\mathcal{I}) = \Omega\left(\sum_{p \in \mathcal{P}} \mathcal{K}_p\right) \ ,$$

with probability $1 - O(D^{-\gamma})$.

Lemma 5.9. *For any input sequence* $\mathcal{I} = (\biguplus_{p \in \mathcal{P}} p) \uplus p_{\text{last}}$ *of length at least* T_γ,

$$C_{\text{MTFR}}(\mathcal{I}) = O\left(\sum_{p \in \mathcal{P}} \mathcal{K}_p\right) \ ,$$

with probability $1 - O(D^{-\gamma})$.

Proof of Theorem 5.3. The $O(1)$-competitiveness of Mtfr follows immediately by combining Lemma 5.8 with Lemma 5.9. ∎

5.2.1 Lower bound for OPT

In this subsection we show that for any phase $p \in \mathcal{P}$ the cost of OPT in this phase is bounded from below by L_p, which is a random variable with expectation $\Theta(\mathcal{K}_p)$. Additionally, $0 \leq L_p \leq \sum_{t \in p} c_t^{\max}$ and all L_p are independent for different phases $p \in \mathcal{P}$. Later, we show how to apply concentration bounds to $\sum_{p \in \mathcal{P}} L_p$, proving that this sum, and thus OPT's cost in all phases, is lower-bounded by $\Omega(\sum_{p \in \mathcal{P}} \mathcal{K}_p)$.

Lower bound in a single phase

Consider any phase $p \in \mathcal{P}$, and number all the time steps within this phase from 1 to ℓ. By $[\ell]_{\text{odd}}$ we denote a set of all odd numbers from $\{1, \dots \ell\}$. For all $t \in [\ell]_{\text{odd}}$, we define

$$s_t := \frac{1}{3^\alpha} \cdot c_t(\sigma_t, \sigma_{t+1}) \ . \tag{5.10}$$

We note that for odd t, s_t are independent random variables. This follows from the fact that the adversary first chooses cost functions c_t (by dictating the network mobility), and then a request sequence $(\sigma_t)_t$ is picked randomly and independently. Let

$$L_p := \sum_{t \in [\ell]_{\text{odd}}} s_t \tag{5.11}$$

We can show that this sum of s_t constitutes a lower bound for the OPT's cost in phase p.

Lemma 5.10. *For any phase $p \in \mathcal{P}$ it holds that*

$$C_{\text{OPT}}(p) \geq L_p \ .$$

The inequality holds for all random choices of sequence $(\sigma_t)_t$ in phase p.

Proof. We prove that for any algorithm ALG and any time step t the cost of serving requests σ_t and σ_{t+1} is at least s_t. This summed over all $t \in [\ell]_{\text{odd}}$ would yield the lemma.

The situation in time step t is depicted in Figure 5.1. $P_{\text{ALG}}(t)$ and $P_{\text{ALG}}(t+1)$ denote the nodes in which ALG has its page in step t and $t+1$, respectively. If $\sigma_t \equiv \sigma_{t+1}$, then $s_t = 0 \leq C_{\text{ALG}}(\sigma_t, \sigma_{t+1})$ and the lemma follows trivially. Otherwise, ALG has to pay at least $c_t(P_{\text{ALG}}(t), \sigma_t) + c_{t+1}(P_{\text{ALG}}(t+1), \sigma_{t+1})$ for the requests in steps t and $t+1$, and $D \cdot c_t(P_{\text{ALG}}(t), P_{\text{ALG}}(t+1))$ for moving its page at the end of step t. In total, the cost of ALG in these two steps is equal to

$$C_{\text{ALG}}(t, t+1) \ = \ c_t(P_{\text{ALG}}(t), \sigma_t) + D \cdot c_t(P_{\text{ALG}}(t), P_{\text{ALG}}(t+1)) + c_{t+1}(P_{\text{ALG}}(t+1), \sigma_{t+1}) \ .$$

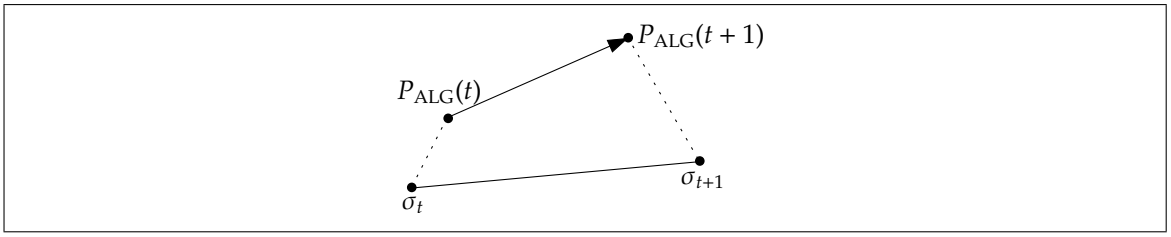

Figure 5.1: Illustration of Lemma 5.10

We apply Inequality 5.9 and the Skewed Triangle Inequality to get

$$\begin{aligned} C_{\text{ALG}}(t, t+1) &\geq c_t(\sigma_t, P_{\text{ALG}}(t)) + c_t(P_{\text{ALG}}(t), P_{\text{ALG}}(t+1)) + \frac{1}{\mathrm{e}} \cdot c_t(P_{\text{ALG}}(t+1), \sigma_{t+1}) \\ &\geq \frac{1}{\mathrm{e}} \cdot \frac{1}{3^{\alpha-1}} \cdot c_t(\sigma_t, \sigma_{t+1}) \\ &> s_t \ , \end{aligned}$$

which finishes the proof. ∎

Note that unlike the bounds on Opt and the algorithm presented in the previous chapter, this bound relates two random variables defined on the same probability space. Moreover, the inequality $C_{\text{OPT}}(p) \geq L_p$ holds for any fixed sequence $(\sigma_t)_t$. This notion is much stronger than stochastic dominance used previously, and renders results similar to Lemma 4.8 (see page 98) unnecessary here.

The expected value of L_p can be easily computed.

Lemma 5.11. *For any phase p, $\mathbf{E}[L_p] = \Omega(1) \cdot \mathcal{K}_p$.*

Proof. From the definition of s_t we have

$$\begin{aligned} \mathbf{E}[s_t] &= \frac{1}{3^\alpha} \cdot \sum_i \sum_j \pi(i) \cdot \pi(j) \cdot c_t(v_i, v_j) \\ &= \frac{1}{3^\alpha} \cdot K_t \ , \end{aligned}$$

and therefore $\mathbf{E}[L_p] = \mathbf{E}[\sum_{t \in [\ell]_{\text{odd}}} s_t] = \frac{1}{3^\alpha} \cdot \sum_{t \in [\ell]_{\text{odd}}} K_t$. It follows from Inequality 5.8 that any two consecutive K_t can differ at most by a multiplicative factor of e. Therefore,

$$\begin{aligned} \mathbf{E}[L_p] &\geq \frac{1}{3^\alpha} \sum_{t \in [\ell]_{\text{odd}}} \frac{1}{2} \cdot \left(K_t + \frac{K_{t+1}}{\mathrm{e}} \right) \\ &\geq \frac{1}{2 \cdot \mathrm{e} \cdot 3^\alpha} \sum_{t \in [\ell]_{\text{odd}}} (K_t + K_{t+1}) \\ &= \frac{1}{2 \cdot \mathrm{e} \cdot 3^\alpha} \cdot \mathcal{K}_p \ . \end{aligned}$$

∎

Hence, combining the two lemmas above, and using linearity of expectation we immediately get a lower bound, i.e., $\mathbf{E}[C_{\mathrm{OPT}}(p)] = \Omega(\mathcal{K}_p)$. Below we show how to get from this good bound holding in expectation to an asymptotically the same bound holding with high probability.

Concentration

Summing up, L_p are independent random variables fulfilling

(i) $0 \leq L_p = \sum_{t \in [\ell]_{\mathrm{odd}}} s_t \leq \sum_{t \in p} c_t^{\max}$,

(ii) $\mathbf{E}[L_p] = \Omega(1) \cdot \mathcal{K}_p$.

We want to prove that $\sum_{p \in \mathcal{P}} L_p$ is at least $\Omega(\sum_{p \in \mathcal{P}} \mathcal{K}_p)$, with high probability. We do not have any global upper bound on L_p variables, which renders classical Chernoff bound [Che52] unusable. However, since nodes can move with a constant speed only, we can relate $\mathcal{K}_p$ in any two consecutive phases and use the following concentration bound. The bound presented in lemma below utilizes Hoeffding bound [RPRR01] (see Appendix A.2) and is proven at the end of the chapter in Section 5.4.1.

Lemma 5.12 (Concentration bound). *Let $\{X_i\}_{i=1}^m$ be the sequence of m independent random variables. Assume that there exist y_1, y_2, δ, and a sequence $\{A_i\}_{i=1}^m$, such that for all i, it holds that*

(i) $0 \leq X_i \leq y_1 \cdot y_2 \cdot A_i^\alpha$,

(ii) $\mathbf{E}[X_i] = \Omega(1) \cdot y_2 \cdot A_i^\alpha$,

(iii) $|A_i - A_{i-1}| \leq \delta$,

(iv) $A_i \geq 1$.

Then there exists a constant c, such that for any γ, if $m \geq \left(c \cdot \gamma \cdot y_1^2 \cdot \ln D\right)^{\alpha+1} \cdot \delta^\alpha$, then it holds that

$$\sum_{t=1}^{m} X_t \;=\; \Omega\left(\sum_{t=1}^{m} y_2 \cdot A_i^\alpha\right) \;,$$

with probability $1 - D^{-\gamma}$. Moreover, if we replace condition (ii) by $\mathbf{E}[X_i] = O(1) \cdot y_2 \cdot A_i^\alpha$, then we get

$$\sum_{t=1}^{m} X_t \;=\; O\left(\sum_{t=1}^{m} y_2 \cdot A_i^\alpha\right) \;,$$

with probability $1 - D^{-\gamma}$.

To use the concentration bound presented above for the variables L_p, we have to represent $\mathcal{K}_p$ and $\sum_{t\in p} c_t^{\max}$ in the form to which the concentration bound above is applicable. If we set $A_p = \sqrt[\alpha]{\mathcal{K}_p/(\ell\cdot\beta)}$, or equivalently

$$\mathcal{K}_p = \ell\cdot\beta\cdot A_p^\alpha \ , \tag{5.12}$$

then we may relate two consecutive A_p. It follows from the definition of $\mathcal{K}_p$ that $\mathcal{K}_p = \sum_{t\in p} K_t = \sum_{t\in p}\beta\cdot k_t^\alpha$. Then we have an obvious relation

$$1 \le \min_{t\in p}\{k_t\} \le A_p \le \max_{t\in p}\{k_t\} \ , \tag{5.13}$$

and thus, by Claim 5.6, for any two consecutive phases p_{i-1} and p_i it holds that

$$\begin{aligned} A_{p_i} - A_{p_{i-1}} &\le \max_{t\in p_i}\{k_t\} - \min_{t\in p_{i-1}}\{k_t\} \\ &\le 2\cdot\ell\cdot(1/\alpha) \ . \end{aligned} \tag{5.14}$$

By symmetry, we have the same bound on the absolute difference $|A_{p_i} - A_{p_{i-1}}|$.

Proof of Lemma 5.8. We choose $y_2 = \ell\cdot\beta$. By Lemma 5.7 it is possible to choose $y_1 = \Theta(\frac{1}{\pi_{\min}})$, such that $K_t \ge \frac{1}{y_1}\cdot c_t^{\max}$ for any time step t. Then we have the following properties

(i) $0 \le L_p \le \sum_{t\in p} c_t^{\max} \le y_1\cdot\mathcal{K}_p = y_1\cdot y_2\cdot A_p^\alpha$,

(ii) $\mathbf{E}[L_p] = \Omega(\mathcal{K}_p) = \Omega(1)\cdot y_2\cdot A_p^\alpha$,

(iii) By (5.14), $|A_{p_i} - A_{p_{i-1}}| \le 2\cdot\ell/\alpha$,

(iv) By (5.13), $A_p \ge 1$.

Thus, we may apply Lemma 5.12 to the sequence of variables L_p. We get that there exists a constant c, such that for any constant γ, if the number of phases of the input sequence is at least

$$m \ge \left(c\cdot\gamma\cdot\frac{\ln D}{\pi_{\min}^2}\right)^{\alpha+1}\cdot\left(\frac{2\cdot\ell}{\alpha}\right)^\alpha \ ,$$

then $\sum_{p\in\mathcal{P}} L_p = \Omega(\sum_{p\in\mathcal{P}} y_2\cdot A_p^\alpha) = \Omega(\sum_{p\in\mathcal{P}}\mathcal{K}_p)$, with probability $1-D^{-\gamma}$. Thus, if we choose c_T occurring in (5.5) (definition of m_γ) to be greater or equal to c, then we guarantee that on input sequences consisting of at least m_γ phases (i.e., of length at least T_γ), it holds that

$$\sum_{p\in\mathcal{P}} C_{\text{OPT}}(p) = \Omega\left(\sum_{p\in\mathcal{P}}\mathcal{K}_p\right) \ ,$$

with probability $1 - D^{-\gamma}$. ■

5.2.2 Upper bound for MTFR

In this subsection we analyze the cost of MTFR on an input sequence $\mathcal{I}$ of length $T \geq T_\gamma$ (for some constant γ). First, we prove that the expected cost of MTFR in any phase p, but the first and the last one, is $O(\mathcal{K}_p)$. Moreover, if we consider only even or only odd phases, then the costs of MTFR in individual phases are independent random variables. We show that by applying concentration bound of Lemma 5.12 we may guarantee that both the total cost in even phases and the total cost in odd phases can be bounded be $O(\sum_{p\in\mathcal{P}} \mathcal{K}_p)$, with high probability. Finally, we show that on the request sequences longer than T_1, the cost in the first and the last phase is bounded by $O(\sum_{p\in\mathcal{P}} \mathcal{K}_p)$ even in the worst case.

Bounding expected cost in one phase

Fix any phase p different than the first or the last one. We number all time steps within p from 1 to ℓ. We denote the cost of serving requests by MTFR in step $t \geq 2$ by F_t. The cost in one step t depends on where the request is issued, i.e., at σ_t, and where the algorithm has currently its page, i.e., at σ_1. We have

$$\begin{aligned}\mathbf{E}[F_t] &= \sum_{i,j} \mathbf{Pr}\left[\sigma_1 = i \wedge \sigma_t = j\right] \cdot c_t(v_i, v_j) \\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{Pr}\left[\sigma_1 = i\right] \cdot \mathbf{Pr}\left[\sigma_t = j\right] \cdot c_t(v_i, v_j) \\ &= K_t \ .\end{aligned}$$

Summing it over all steps $t \geq 2$ from one phase, we get

$$\mathbf{E}\left[\sum_{t=2}^{\ell} F_t\right] \leq \mathcal{K}_p \tag{5.15}$$

Now, for bounding the total expected cost in phase p, it suffices to bound the cost incurred in the first step of p. Let p_{prev} denote the phase preceding p, and $(p_{\text{prev}})_1$ be its first step. At the beginning of p, the algorithm is in node $\sigma_{(p_{\text{prev}})_1}$, and thus the cost of serving the request in the first step of p and moving the page to σ_1 is equal to $(1 + D) \cdot c_1(\sigma_{(p_{\text{prev}})_1}, \sigma_1)$. The expected value of this cost is $(1 + D) \cdot K_1$.[3]

Since the phase is quite long, we may amortize the expected cost of moving the page in the first step against the expected cost of serving requests in the remaining part of the phase. To achieve this, we introduce the following lemma.

[3] Note that it is no longer true if we consider the first phase, since then the cost is equal to $(1+D)\cdot c_1(v_1, \sigma_1)$.

Lemma 5.13. *Fix any legal sequence $(K_t)_t$ of length s and choose any step t_0. Let $Q := K_{t_0}$. Then it holds that*

$$\sum_{t=1}^{s} K_t = \Omega(\beta \cdot s + s^{\frac{1}{\alpha+1}} \cdot Q) \ .$$

Proof. Recall that by (5.7), $K_t = \beta \cdot k_t^\alpha$. As all $k_t \geq 1$, we get $\sum_{t=1}^{s} K_t \geq \beta \cdot s$. We consider two cases.

- If $Q \leq \beta \cdot s^{\frac{\alpha}{\alpha+1}}$, then the term $\beta \cdot s$ majorizes the whole term on the right side of the equation above, and therefore the lemma follows.
- If $Q > \beta \cdot s^{\frac{\alpha}{\alpha+1}}$, then we have to prove that $K_t = \Omega(s^{\frac{1}{\alpha+1}} \cdot Q)$. In this case $k_{t_0} = (Q/\beta)^{\frac{1}{\alpha}} > s^{\frac{1}{\alpha+1}}$. As two consecutive values of k_t can differ by at most $1/\alpha \leq 1$, there exist at least $\frac{1}{2} \cdot s^{\frac{1}{\alpha+1}}$ time steps t, in which $k_t \geq k_{t_0} - \frac{1}{2} \cdot s^{\frac{1}{\alpha+1}} \geq \frac{1}{2} \cdot k_{t_0}$. In all these time steps the value of K_t is at least $\beta \cdot (\frac{1}{2} \cdot k_{t_0})^\alpha = \frac{1}{2^\alpha} \cdot Q$. Thus,

$$\begin{aligned} \sum_{t=1}^{s} K_t \ &\geq \ \frac{1}{2} \cdot s^{\frac{1}{\alpha+1}} \cdot \frac{1}{2^\alpha} \cdot Q \\ &= \ \Omega(s^{\frac{1}{\alpha+1}} \cdot Q) \ . \end{aligned}$$

Hence, in both cases the lemma holds. ∎

We apply the lemma to the sequence of $(K_t)_{t=2}^{\ell}$ with $t_0 = 2$. Then we get

$$K_2 \cdot \ell^{\frac{1}{\alpha+1}} = O\left(\sum_{t=2}^{\ell} K_t\right) \ .$$

Since by (5.8) any two consecutive K_t can differ by at most a multiplicative factor of e, the right side of the equation above is at most $\mathcal{K}_p$, and $\ell = D^{\alpha+1}$, we get

$$(1 + D) \cdot K_1 = O(\mathcal{K}_p) \ . \tag{5.16}$$

Thus, by (5.15) and (5.16), we have

$$\mathbf{E}[C_{\text{MTFR}}(p)] = O(\mathcal{K}_p) \ .$$

From expectation to high probability

For any p, $C_{\text{MTFR}}(p)$ is a random variable, which depends only on the randomly generated requests in p and in the first step of the phase preceding p. Thus, if we consider each second phase, the corresponding variables $C_{\text{MTFR}}(p)$ are independent and we may apply our concentration bound.

Lemma 5.14. *Let $\mathcal{I}$ be any input sequence of length at least T_γ, and let $\mathcal{I}_1$ be the set of its all even phases (without the first and the last phase). Then*

$$C_{\text{MTFR}}(\mathcal{I}_1) = O\left(\sum_{p\in\mathcal{P}} \mathcal{K}_p\right) ,$$

with probability $1 - D^{-\gamma}$. The same holds for the set of all odd phases.

Proof. We prove it only for even phases, as the proof for odd phases is identical. As mentioned above variables $C_{\text{MTFR}}(p)$ are independent for different $p \in \mathcal{I}_1$. Similarly to the proof of Lemma 5.8, we choose $y_1 = \Theta(\frac{1}{\pi_{\min}})$ and $y_2 = \ell \cdot \beta$. The only difference between this proof and the proof of Lemma 5.8 is that we get $\mathbf{E}[C_{\text{MTFR}}(p)] = O(\mathcal{K}_p) = O(1)\cdot y_2 \cdot A_p^\alpha$.

Then the concentration bound (Lemma 5.12), applied to random variables $C_{\text{MTFR}}(p)$, guarantees that there exists a constant c, such that for any γ, if the number of *even* phases of the input sequence is at least

$$m \geq \left(c \cdot \gamma \cdot \frac{\ln D}{\pi_{\min}^2}\right)^{\alpha+1} \cdot \left(\frac{2\cdot\ell}{\alpha}\right)^{\alpha} ,$$

then

$$\begin{aligned} C_{\text{MTFR}}(\mathcal{I}_1) &= O\left(\sum_{p\in\mathcal{I}_1} y_2 \cdot A_p^\alpha\right) \\ &= O\left(\sum_{p\in\mathcal{I}_1} \mathcal{K}_p\right) , \end{aligned}$$

with probability $1 - D^{-\alpha}$. Thus, it is sufficient that the constant c_T occurring in the definition of m_γ is at least $2 \cdot c$, and then we get high probability for input sequences consisting of at least m_γ phases. ■

Bounding cost in the first and the last phase

In this part we prove that on the input sequences longer than T_1, the worst-case cost in the first or in the last phase (possibly unfinished one) can be bounded by $O(\sum_{p\in\mathcal{P}} \mathcal{K}_p)$. To prove it for the first phase p_1, let t^* be the first step of p_1, and let $Q^* = K_{t^*}$. We will relate both the worst-case bound on $C_{\text{MTFR}}(p_1)$ and the value of $O(\sum_{p\in\mathcal{P}} \mathcal{K}_p)$ to Q^*. The bound on the cost in the last phase p_{last} follows almost identically if we consider the value of K_t in the first step of p_{last}.

We number all the steps within phase p_1 from 1 to ℓ. By relation between K_t and $c_t^{\max}$, we get

$$\begin{aligned} C_{\text{MTFR}}(p_1) &\leq D \cdot c_1^{\max} + \sum_{t=1}^{\ell} c_t^{\max} \\ &\leq O(1) \cdot \frac{1}{\pi_{\min}} \cdot \left(D \cdot K_1 + \sum_{t=1}^{\ell} K_t \right). \end{aligned} \tag{5.17}$$

To bound the latter summand, we use the following lemma, which is complementary to Lemma 5.13.

Lemma 5.15. *Fix any legal sequence $(K_t)_t$ of length s and choose any step t_0. Let $Q := K_{t_0}$. Then it holds that*

$$\sum_{t=1}^{s} K_t = O(\beta \cdot s^{\alpha+1} + s \cdot Q) \ .$$

Proof. We have $Q = \beta \cdot k_{t_0}^{\alpha}$. We proceed with case analysis.

1. If $k_{t_0} \leq s$, then since two consecutive k_t can differ by at most $1/\alpha \leq 1$, for all t it holds that $k_t \leq k_{t_0} + s = 2 \cdot s$, and thus $\sum_{t=1}^{s} K_t = s \cdot (\beta \cdot (2 \cdot s)^{\alpha}) = O(\beta \cdot s^{\alpha+1})$.
2. If $k_{t_0} > s$, then $Q > \beta \cdot s^{\alpha}$ and then for any t, $k_t \leq (Q/\beta)^{\frac{1}{\alpha}} + s \leq 2 \cdot (Q/\beta)^{\frac{1}{\alpha}}$. Hence, all values of K_t are at most $2^{\alpha} \cdot Q$, and thus $\sum_{t=1}^{s} K_t = O(s \cdot Q)$.

Thus, in either case the bound follows. ■

Therefore, by the lemma above and Inequality 5.17, we get

$$C_{\text{MTFR}}(p_1) = O(1) \cdot \frac{1}{\pi_{\min}} \cdot (\beta \cdot \ell^{\alpha+1} + \ell \cdot Q^*) \ . \tag{5.18}$$

We can now compare the bound above to $O(\sum_{p \in \mathcal{P}} \mathcal{K}_p)$.

Lemma 5.16. *Fix any input sequence of length $T \geq T_1$ and let p_1 be its first phase. Then it holds that*

$$C_{\text{MTFR}}(p_1) = O\left(\sum_{p \in \mathcal{P}} \mathcal{K}_p \right) \ .$$

The same holds for the last phase p_{last}.

Proof. Let $\mathcal{I}$ be any input sequence of length $T \geq T_1$, Let s be the number of steps in finished phases, i.e., $s = |\mathcal{P}| \cdot \ell$. By applying Lemma 5.13 to the sequence of all K_t from the set of finished phases $\mathcal{P}$, we get that

$$\sum_{p \in \mathcal{P}} \sum_{t \in p} K_t = \ \Omega(\beta \cdot s + s^{\frac{1}{\alpha+1}} \cdot Q^*)$$

Since $s \geq T_1$, $s = \Omega((\frac{\ell}{\pi_{\min}})^{\alpha+1})$, and we get

$$\sum_{p \in \mathcal{P}} \mathcal{K}_p = \Omega\left(\frac{1}{\pi_{\min}} \cdot (\beta \cdot \ell^{\alpha+1} + \ell \cdot Q^*)\right) .$$

Combining this with Inequality 5.18, we get the lemma. ■

Lemma 5.9 is an easy consequence of the argument presented in this paper.

Proof of Lemma 5.9. We take any input sequence $\mathcal{I}$ of length $T \geq T_\gamma$ and divide it into phases, i.e., $\mathcal{I} = (\biguplus_{p \in \mathcal{P}} p) \uplus p_{\text{last}}$. Then these phases may be grouped in three parts.

1. $\mathcal{I}_1$, set of all even phases, but the first and the last phase,
2. $\mathcal{I}_2$, set of all odd phases, but the first and the last phase,
3. $\mathcal{I}_3$, the first and the last phase.

Let $\mathcal{K} = \sum_{p \in \mathcal{P}} \mathcal{K}_p$. Then by Lemma 5.14, we get $C_{\text{MTFR}}(\mathcal{I}_1) = O(\mathcal{K})$ with probability at least $1 - D^{-\gamma}$, and $C_{\text{MTFR}}(\mathcal{I}_1) = O(\mathcal{K})$ also with probability at least $1 - D^{-\gamma}$. Moreover, by Lemma 5.16, the cost induced in $\mathcal{I}_3$ is at most $O(\mathcal{K})$. Then with probability at least $1 - 2 \cdot D^{-\gamma}$, it holds that

$$C_{\text{MTFR}}(\mathcal{I}) = O(\mathcal{K}) ,$$

which finishes the proof. ■

5.2.3 Expected competitive ratio

Theorem 5.3, combining the results of two previous subsections, shows that on a sequence $\mathcal{I}$ of length $T \geq T_\gamma$, the competitive ratio of MTFR is constant, with probability $1 - O(D^{-\gamma})$. In this subsection we prove that it is also constant in expectation.

To show it, we first prove that even if both configuration and request sequences are generated by an adversary, then MTFR is competitive.

Lemma 5.17. *For any input sequence (even when both request and configuration sequences are chosen by the adversary) the competitive ratio of* MTFR *is at most* $O(\ell^{\alpha+1})$.

For the proof we fix any input sequence $\mathcal{I}$ of length T. We divide this input into phases. In the following we concentrate on any phase p of length $\ell_0 \leq \ell = D^{\alpha+1}$. We number all time steps within p from 1 to ℓ_0. We assume that the adversary is $\frac{1}{2}$-restricted, and thus the distances between any pair can change by at most 1 per time step.

Let L_c be the cost of communication between P_{MTFR} and P_{OPT}, the nodes holding the pages of MTFR and OPT, respectively. We define a potential function

$$\Phi = f \cdot D \cdot L_c \ , \quad \text{where} \quad f = 2 \cdot 3^\alpha \ . \tag{5.19}$$

By a potential at the beginning of phase p, $\Phi_B(p)$ we understand a potential at the very beginning of the first time step of p, i.e., *before* the network adversary moves the nodes. Respectively, a potential at the end of p, $\Phi_F(p)$ is a potential measured in the last step of p, after MTFR and OPT move their pages.

At the beginning of the first phase p_1, $\Phi_B(p_1) = 0$, because MTFR and OPT have their pages at the same node, v_1. For any two consecutive phases p_i and p_{i+1} it holds that $\Phi_F(p_i) = \Phi_B(p_{i+1})$. Thus, for proving that MTFR is $O(\ell^{\alpha+1})$-competitive, it is sufficient to show that for any phase p it holds that

$$C_{\text{MTFR}}(p) + \Phi_F(p) - \Phi_B(p) \leq O(\ell^{\alpha+1}) \cdot C_{\text{OPT}}(p) \ . \tag{5.20}$$

We consider two cases, captured by two lemmas below.

Lemma 5.18. *Fix any phase p. If $C_{\text{OPT}}(p) = 0$, then Inequality 5.20 holds.*

Proof. In this case OPT has to remain at $P_{\text{OPT}}(1)$ for the whole phase p and all requests have to be issued at this node, too. On the other hand, in the first step MTFR has to pay for serving the request and moving the page to $\sigma_1 \equiv P_{\text{OPT}}(1)$, i.e.,

$$\begin{aligned} C_{\text{MTFR}}(1) &= (1+D) \cdot c_1(P_{\text{MTFR}}(1), P_{\text{OPT}}(1)) \\ &\leq 2 \cdot D \cdot \mathrm{e} \cdot c_0(P_{\text{MTFR}}(1), P_{\text{OPT}}(1)) \\ &\leq \Phi_B(p) \ . \end{aligned}$$

After this step, the algorithm remains at the same node as OPT, paying 0. Therefore, at the end of the phase $P_{\text{MTFR}} \equiv P_{\text{OPT}}$, which implies $\Phi_F(p) = 0$. Summing up, the amortized cost of MTFR in p is non-positive and the lemma follows. ■

Lemma 5.19. *Fix any phase p. If $C_{\text{OPT}}(p) \geq 1$, then Inequality 5.20 holds.*

Proof. Let $F := d_0(P_{\text{MTFR}}(1), P_{\text{OPT}}(1))$ be the distance between the pages of MTFR and OPT at the very beginning of the phase (before the adversary moves the nodes). Clearly,

$$\Phi_B(p) \geq f \cdot D \cdot F^\alpha \ .$$

Let X_t be the distance between σ_t and $P_{\text{OPT}}(t)$ in step t. Finally, for any time step $1 \leq t \leq \ell_0$, let Y_t be the distance across which OPT moves its page in time step t.

The cost of OPT in step t is at least $C_{\text{OPT}}(t) \geq X_t^\alpha + D \cdot Y_t^\alpha$. In the first step, the distance between MTFR and OPT is at most $F + 1$, and therefore the distance between MTFR and OPT is at most $F + 1 + X_1$. Thus, MTFR's cost in the first step is at most

$$\begin{aligned} C_{\text{MTFR}}(1) &\leq (1+D) \cdot [F + 1 + X_1]^\alpha \\ &\overset{(\mathbf{H})}{\leq} 2D \cdot 3^{\alpha-1} \cdot (F^\alpha + 1 + X_1^\alpha) \\ &\leq O(D) + \Phi_B(p) + O(D) \cdot X_1^\alpha \\ &\leq O(D) \cdot C_{\text{OPT}}(p) + \Phi_B(p) \ . \end{aligned} \tag{5.21}$$

At the end of the first step, MTFR moves to σ_1 and the distance between $P_{\text{MTFR}}(1)$ and $P_{\text{OPT}}(1)$ becomes at most $X_1 + Y_1$. In each step $t > 1$, the distance between $P_{\text{MTFR}}(t) \equiv P_{\text{MTFR}}(1)$ and $P_{\text{OPT}}(t)$ can increase by at most 1 due to the changes in the network made by the network adversary, and additionally by at most Y_t due to the movement of the OPT's page. Thus, for any time step $t \geq 1$, $d_t(P_{\text{MTFR}}(t), P_{\text{OPT}}(t)) \leq X_1 + \ell_0 + \sum_{t=1}^{\ell_0} Y_t$. We denote this bound by U. We have

$$\begin{aligned} U^\alpha &\overset{(\mathbf{H})}{=} O(1) \cdot \left(X_1^\alpha + \ell_0^\alpha + \left(\sum_{t=1}^{\ell_0} Y_t \right)^\alpha \right) \\ &\overset{(\mathbf{H})}{=} O(1) \cdot \left(X_1^\alpha + \ell_0^\alpha + \ell_0^{\alpha-1} \cdot \sum_{t=1}^{\ell_0} Y_t^\alpha \right). \end{aligned}$$

Since the distance to the request in step t is at most $U + X_t$, for all steps $1 < t \leq \ell_0$ it holds that $C_{\text{MTFR}}(t) \leq 1 + (U + X_t)^\alpha$. Additionally, $\Phi_{\text{F}}(p) \leq f \cdot D \cdot [1 + U^\alpha]$. Summing the MTFR's cost over all time steps (but the first one) from the phase, we get

$$\begin{aligned} \left(\sum_{t=2}^{\ell_0} C_{\text{MTFR}}(t) \right) + \Phi_{\text{F}}(p) &\leq \sum_{t=2}^{\ell_0} (U + X_t)^\alpha + O(D) \cdot U^\alpha + O(D) \\ &\overset{(\mathbf{H})}{\leq} O(1) \cdot \sum_{t=2}^{\ell_0} X_t^\alpha + O(\ell) \cdot U^\alpha + O(D) \\ &\leq O(1) \cdot \sum_{t=2}^{\ell_0} X_t^\alpha + O(\ell) \cdot \left(X_1^\alpha + \ell_0^\alpha + \ell_0^{\alpha-1} \cdot \sum_{t=1}^{\ell_0} Y_t^\alpha \right) + O(D) \\ &\leq O(\ell^{\alpha+1}) \cdot C_{\text{OPT}}(p) \ . \end{aligned} \tag{5.22}$$

By summing (5.21) with (5.22), we get the lemma. ■

Proof of Lemma 5.17. Inequality 5.20 follows from the two lemmas above. When we sum this inequality over all the phases in the input sequence, we immediately get that MTFR is $O(\ell^{\alpha+1})$-competitive. ■

Now we can apply this bound on the competitive ratio to compute the expected competitive ratio of MTFR on any input sequence of length $T \geq T_{(\alpha+1)^2}$.

Theorem 5.20. *For any input $\mathcal{I}$ of length $T \geq T_{(\alpha+1)^2}$, MTFR achieves expected constant competitive ratio of $O(1)$.*

Proof of Theorem 5.20. Fix any configuration sequence $(C_t)_t$ and a probability distribution π. Then by Theorem 5.3 and Lemma 5.17 it follows that

$$\begin{aligned} \mathbf{E}_{(\sigma_t)} \left[\frac{C_{\text{ALG}}((C_t, \sigma_t)_t)}{C_{\text{OPT}}((C_t, \sigma_t)_t)} \right] &= \left(1 - O\left(\frac{1}{D^{(\alpha+1)^2}} \right) \right) \cdot O(1) + O\left(\frac{1}{D^{(\alpha+1)^2}} \right) \cdot O(\ell^{\alpha+1}) \\ &= O(1) \ , \end{aligned}$$

which proves that the competitive ratio of Mtfr is constant in expectation. ■

5.3 Extensions and conclusions

The results of this chapter show that the considered stochastic scenario is much more favorable than the one where both configuration and request sequences are created by an adversarial entity. This supports the claim that in fact the competitive ratio in adversarial scenario is so high, because the network and sequence adversaries may combine and *synchronize* their efforts. The constant competitive ratio of the stochastic scenario presented here follows partially from the fact that it is extremely unlikely that the random request sequence contains constructions similar to the construction of lower bound presented in Section 5.1.

Extending probability function

As stated at the beginning of this chapter, we may extend our arguments to the case where π can additionally take values 0 and 1, i.e., $\pi : [n] \to [0, 1]$. If for some v_i the probability π is equal to 1, then on the long run our algorithm is the best possible, as all the requests are given at v_i, and Mtfr moves there at the very beginning of the input sequence.

The case where some nodes have zero probabilities is a little more involved, since in our proof we used the term $\frac{1}{\min_{i\in[n]} \pi(i)}$, which obviously does not make sense here. However, we may treat these zero-probability nodes as non-existent, reducing the situation to the nodes with positive probabilities. In this case, we define $c_t^{\max}$ as the maximum cost of communication between nodes v_i and v_j such that $\pi_i, \pi_j > 0$. Since our algorithm would ignore zero-probability nodes, the only place in our proof that may raise concern is the lower bound on Opt. Opt may potentially use the nodes which have zero probabilities. However, the proof of Lemma 5.10 does not impose any restrictions on the nodes, holding Opt's page in the individual steps. Thus, $C_{\text{OPT}}(p)$ majorizes L_p for any phase p, even if Opt wants to store its page at the zero-probability nodes. Hence, Mtfr is $O(1)$-competitive also for a generalized function π.

Open Problems

On the other hand, it might be interesting to consider another scenario where the distribution π is not equal for each time step, but depends, for example, on the node which issued a request in the previous step. Such a model captures the locality of accesses (e.g., if a processor v_a accesses the page, then either processor v_b or v_c will access it in the next step). The competitive ratio of such a *Markovian scenario* remains unknown. However,

we conjecture that it is possible to construct an algorithm, which achieves a reasonably low competitive ratio.

Another open question is whether it is possible to construct an algorithm for the stochastic scenario, which will be better than $\Omega\left(D^{(\alpha+1)^2}\right)$-competitive in the worst case, still assuring constant ratio in expectation. Combining MTFR with the algorithms for the adversarial scenario presented in Chapter 3, might be challenging and yield interesting results.

It might be also interesting to investigate the extended cost model (i.e., the case of $\alpha > 1$) in other scenarios, especially in the adversarial one. We conjecture that the lower bound presented in Section 5.1 can be generalized to exploit the number of nodes greater than 2, similarly to the bounds presented in Section 3.3. On the other hand, extending algorithms in the adversarial scenarios is not a trivial task since in the generalized cost model the triangle inequality for the costs of communication (which was extensively used in Chapter 3) is no longer fulfilled.

5.4 Proofs of technical claims

Proof of Claim 5.5. For $s = 1$ the lemma follows trivially. Hölder's Inequality [HLP88] (see Appendix A.2) states that for any $p, q > 1$ such that $\frac{1}{p} + \frac{1}{q} = 1$ and for any non-negative sequences $(a_i)_{i=1}^k$ and $(b_i)_{i=1}^k$, it holds that

$$\sum_{i=1}^{k} (a_i \cdot b_i) \leq \left(\sum_{i=1}^{k} a_i^p\right)^{1/p} \cdot \left(\sum_{i=1}^{k} b_i^q\right)^{1/q} .$$

By setting $p = s$, $q = \frac{s}{s-1}$ and $b_i = 1$ for all i we obtain

$$\sum_{i=1}^{k} a_i \leq \left(\sum_{i=1}^{k} a_i^s\right)^{1/s} \cdot k^{\frac{s-1}{s}} .$$

By raising both sides to the s-th power we get the lemma. ∎

Proof of Claim 5.6. We define k'_t as $\sqrt[\alpha]{K_t/\beta - 1}$. Equivalently $K_t = \beta \cdot \left(1 + (k'_t)^\alpha\right)$ or

$$k_t^\alpha = 1 + (k'_t)^\alpha . \tag{5.23}$$

Let $\delta = 1/\alpha$. As an intermediate step we will prove that $k'_{t+1} - k'_t \leq \delta$; later we will use it to prove $k_{t+1} - k_t \leq \delta$.

From the definition of k'_t we have $\beta \cdot (k'_t)^\alpha = \sum_i \sum_{j \neq i} \pi(i) \cdot \pi(j) \cdot d_t^\alpha(v_i, v_j)$. For succinctness of the proof we denote $\frac{1}{\beta} \cdot \pi(i) \cdot \pi(j)$ by $p_{i,j}$. Then

$$(k'_t)^\alpha = \sum_{i \neq j} p_{i,j} \cdot (d_t(v_i, v_j))^\alpha ,$$

and $\sum_{i \neq j} p_{i,j} = 1$. We will prove that $k'_{t+1} \leq k'_t + \delta$.

We fix any constant non-negative integer $s < \alpha$, and consider variables $y_t(i,j) := (d_t(v_i, v_j))^{\alpha - s}$ for all $i \neq j$ where $i, j \in [n]$. Since the function $f(y) = y^{\frac{\alpha}{\alpha - s}}$ is convex and $\sum_{i \neq j} p_{i,j} = 1$, we can apply Jensen's Inequality [HLP88] (see Appendix A.2) for the $y_t(i,j)$ variables to get that

$$\left[\sum_{i \neq j} p_{i,j} \cdot y_t(i,j) \right]^{\frac{\alpha}{\alpha - s}} \leq \sum_{i \neq j} p_{i,j} \cdot [\, y_t(i,j) \,]^{\frac{\alpha}{\alpha - s}} \quad .$$

Thus, by raising both sides to the power $\frac{\alpha - s}{\alpha}$, we obtain

$$\begin{aligned} \sum_{i \neq j} p_{i,j} \cdot [\, d_t(v_i, v_j) \,]^{\alpha - s} &= \left[\sum_{i \neq j} p_{i,j} \cdot [\, d_t(v_i, v_j) \,]^{\alpha} \right]^{\frac{\alpha - s}{\alpha}} \\ &= (k'_t)^{\alpha - s} \quad . \end{aligned}$$

Note that the inequality above holds trivially also for the case of $s = \alpha$. Hence, we can multiply both sides of the above inequality by $\binom{\alpha}{s} \cdot \delta^s$ and sum them over all $s \in \{0, \ldots, \alpha\}$.

$$\sum_{s=0}^{\alpha} \sum_{i \neq j} p_{i,j} \cdot \binom{\alpha}{s} \cdot [\, d_t(v_i, v_j) \,]^{\alpha - s} \cdot \delta^s \leq \sum_{s=0}^{\alpha} \binom{\alpha}{s} \cdot (k'_t)^{\alpha - s} \cdot \delta^s$$

By folding the binomial formula, we get

$$\sum_{i \neq j} p_{i,j} \cdot [\, d_t(v_i, v_j) + \delta \,]^{\alpha} \leq (k'_t + \delta)^{\alpha} \quad . \tag{5.24}$$

But from the definition of k'_{t+1},

$$(k'_{t+1})^{\alpha} = \sum_{i \neq j} p_{i,j} \cdot [\, d_{t+1}(v_i, v_j) \,]^{\alpha} \leq \sum_{i \neq j} p_{i,j} \cdot [\, d_t(v_i, v_j) + \delta \,]^{\alpha} \quad . \tag{5.25}$$

Combining (5.24) with (5.25), and taking α-th root from both sides, we finally get

$$k'_{t+1} \leq k'_t + \delta$$

Now we show how the relation between two consecutive values k'_t and k'_{t+1} implies the bounded difference between two consecutive values k_t and k_{t+1}. Using definition (5.23), we obtain

$$\begin{aligned} k_{t+1} - k_t &= \sqrt[\alpha]{(k'_{t+1})^{\alpha} + 1} - \sqrt[\alpha]{(k'_t)^{\alpha} + 1} \\ &\leq \sqrt[\alpha]{(k'_t + \delta)^{\alpha} + 1} - \sqrt[\alpha]{(k'_t)^{\alpha} + 1} \quad . \end{aligned}$$

The last term is a function of k'_t; we denote it by $g(k'_t)$. Function g is monotonically increasing, as its first derivative is greater than 0 for $k'_t, \delta \geq 0$. Additionally, $\lim_{k'_t \to \infty} g(k'_t) = \delta$, and therefore $k_{t+1} - k_t \leq g(k'_t) < \delta$ for all k'_t.

The proof of $k_t - k_{t+1} \leq \delta$ is analogous, and thus the claim follows. ■

5.4.1 Proof of the concentration bound

Before we prove the concentration bound claimed by Lemma 5.12, we show a combinatorial lemma, which will be essential to the further proof.

Lemma 5.21. *Let $(A_i)_{i=1}^m$ be the sequence of $m \geq 16$ real numbers, such that for any i, $A_i \geq 1$, and there exists δ, such that for any $i < m$, $|A_{i+1} - A_i| \leq \delta$. If $\delta \geq 1$ and $m \geq \delta^\alpha$, then*

$$\frac{\left(\sum_{i=1}^m A_i^\alpha\right)^2}{\sum_{i=1}^m (A_i^\alpha)^2} \geq b_1 \cdot \left(\frac{m}{\delta^\alpha}\right)^{\frac{1}{\alpha+1}} ,$$

where b_1 is a constant.

Proof. First, we scale down all elements of sequence $(A_i)_i$ by dividing them by $\min_i\{A_i\}$. Note that the considered ratio

$$R := \frac{\left(\sum_{i=1}^m A_i^\alpha\right)^2}{\sum_{i=1}^m (A_i^\alpha)^2}$$

remains invariant, the property $|A_{i+1} - A_i| \leq \delta$ still holds, and after scaling we have $\min_i\{A_i\} = 1$. Let $k = \max_i\{A_i\}$. Let S be the smallest possible (in terms of the number of elements) subset of $\{A_i\}$ with the following properties.

(i) $1, k \in S$.

(ii) Let $(S_i)_i$ be the sequence of all elements from set S, sorted in non-descending order. Then for a pair of consecutive elements S_i and S_{i+1} it holds that $|S_{i+1} - S_i| \leq \delta$.

The existence of such a set is assured, since $\{A_i\}$ fulfills these properties itself.

For any integer j, let I_j denote the interval $(\delta(j-1), \delta j]$, and let $\kappa := \lceil k/\delta \rceil$. It is straightforward that all the elements of the sequence $\{S_i\}$ belong to the $\biguplus_{j=1}^{\kappa} I_j$. Furthermore, each interval I_j from this union contains at least one element of $\{S_i\}$ (from the second property of set S) and at most two elements of $\{S_i\}$ (from the minimality of S).

In the trivial case, $\kappa = 1$, all elements A_i are between 1 and δ. Therefore,

$$\begin{aligned} R &= \frac{\left(\sum_{i=1}^{m} A_i^{\alpha}\right)^2}{\sum_{i=1}^{m}(A_i^{\alpha})^2} \\ &\geq \frac{\left(\sum_{i=1}^{m} A_i^{\alpha}\right)^2}{\max_i\{A_i^{\alpha}\} \cdot \sum_{i=1}^{m} A_i^{\alpha}} \\ &\geq \frac{m}{\delta^{\alpha}} \\ &\geq \left(\frac{m}{\delta^{\alpha}}\right)^{\frac{1}{\alpha+1}} . \end{aligned}$$

In the general case, $\kappa > 1$, and we have

$$\sum_{i=1}^{|S|} S_i^{\alpha} \geq \sum_{j=1}^{\kappa} [\delta(j-1)]^{\alpha} \geq a_1 \cdot \delta^{\alpha} \cdot \kappa^{\alpha+1} , \tag{5.26}$$

for some constant a_1, which depends only on α. On the other hand,

$$\sum_{i=1}^{|S|} S_i^{2\alpha} \leq \sum_{j=1}^{\kappa} 2\left(\delta \cdot j\right)^{2\alpha} \leq a_2 \cdot \delta^{2\alpha} \cdot \kappa^{2\alpha+1} , \tag{5.27}$$

for some constant a_2. Thus, combining (5.26) and (5.27),

$$\begin{aligned} R &= \frac{\left(\sum_{i\in S} A_i^{\alpha} + \sum_{i\notin S} A_i^{\alpha}\right)^2}{\sum_{i\in S} A_i^{2\alpha} + \sum_{i\notin S} A_i^{2\alpha}} \\ &\geq \frac{\left(a_1 \cdot \delta^{\alpha} \cdot \kappa^{\alpha+1} + \sum_{i\notin S} A_i^{\alpha}\right)^2}{a_2 \cdot \delta^{2\alpha} \cdot \kappa^{2\alpha+1} + \sum_{i\notin S} A_i^{2\alpha}} . \end{aligned}$$

Since $\sum_{i\notin S} A_i^{2\alpha} \leq \max_i\{A_i^{\alpha}\} \cdot \sum_{i\notin S} A_i^{\alpha} \leq k^{\alpha} \cdot \sum_{i\notin S} A_i^{\alpha} \leq (2\delta \cdot \kappa)^{\alpha} \cdot \sum_{i\notin S} A_i^{\alpha}$, we obtain

$$R \geq \frac{a_1^2 \cdot \delta^{2\alpha} \cdot \kappa^{2\alpha+2} + 2 \cdot a_1 \cdot \delta^{\alpha} \cdot \kappa^{\alpha+1} \cdot \sum_{i\notin S} A_i^{\alpha} + \left(\sum_{i\notin S} A_i^{\alpha}\right)^2}{a_2 \cdot \delta^{2\alpha} \cdot \kappa^{2\alpha+1} + 2^{\alpha} \cdot \delta^{\alpha} \cdot \kappa^{\alpha} \cdot \sum_{i\notin S} A_i^{\alpha}} .$$

By omitting either the first or the third term from the numerator above, we get $R \geq a_3 \cdot \frac{\sum_{i\notin S} A_i^{\alpha}}{\delta^{\alpha}\cdot\kappa^{\alpha}}$ and $R \geq a_4 \cdot \kappa$, for some constants a_3, a_4. Thus, if $\kappa \geq (m/\delta^{\alpha})^{\frac{1}{\alpha+1}}$, then the lemma follows immediately. Otherwise, S contains at most $2 \cdot \kappa \leq 2 \cdot m^{\frac{1}{\alpha+1}}$ elements, and thus $\sum_{i\notin S} A_i^{\alpha} \geq (m - 2 \cdot m^{\frac{1}{\alpha+1}}) \cdot 1 \geq \frac{m}{2}$. Therefore, in this case

$$\begin{aligned} R &\geq \frac{a_3}{2} \cdot \frac{m}{\delta^{\alpha} \cdot \kappa^{\alpha}} \\ &\geq \frac{a_3}{2} \cdot (m/\delta^{\alpha})^{\frac{1}{\alpha+1}} , \end{aligned}$$

and the lemma holds. ■

Proof of Lemma 5.12 (Concentration bound). We have that for all X_i, $\mathbf{E}[X_i] \geq b_2 \cdot y_2 \cdot A_i^\alpha$ for some constant b_2. Since all variables X_i are independent, we may apply Hoeffding bound (see Appendix A.2 for an exact formulation) to get

$$\begin{aligned}\mathbf{Pr}\left[\sum_{i=1}^{m} X_i < \frac{1}{2} \cdot \sum_{i=1}^{m} b_2 \cdot y_2 \cdot A_i^\alpha\right] &\leq \exp\left(-\frac{2 \cdot \left(\frac{1}{2} \cdot \sum_{i=1}^{m} b_2 \cdot y_2 \cdot A_i^\alpha\right)^2}{\sum_{i=1}^{m} (y_1 \cdot y_2 \cdot A_i^\alpha)^2}\right) \\ &\leq \exp\left(-\frac{b_2^2}{2 \cdot y_1^2} \cdot \frac{\left(\sum_{i=1}^{m} A_i^\alpha\right)^2}{\sum_{i=1}^{m} (A_i^\alpha)^2}\right)\end{aligned} \tag{5.28}$$

Let b_1 be the constant in the formulation of Lemma 5.21. We choose

$$m \;:=\; \left(\frac{2}{b_1 \cdot b_2^2} \cdot \gamma \cdot y_1^2 \cdot \ln D\right)^{\alpha+1} \cdot \delta^\alpha \;.$$

The precondition of Lemma 5.21, $m \geq \delta^\alpha$, is fulfilled, and thus by applying this lemma we get that

$$\frac{\left(\sum_{i=1}^{m} A_i^\alpha\right)^2}{\sum_{i=1}^{m} (A_i^\alpha)^2} \;\geq\; \frac{2 \cdot y_1^2}{b_2^2} \cdot \gamma \cdot \ln D \;.$$

Substituting this bound into (5.28), we get that

$$\mathbf{Pr}\left[\sum_{i=1}^{m} X_i < \frac{1}{2} \cdot \sum_{i=1}^{m} b_2 \cdot y_2 \cdot A_i^\alpha\right] \;\leq\; D^{-\alpha}$$

It is straightforward that the same bound would hold for $\mathbf{Pr}[\sum_{i=1}^{m} X_i > \frac{3}{2} \cdot \sum_{i=1}^{m} b_2 \cdot y_2 \cdot A_i^\alpha]$ if we had $\mathbf{E}[X_i] \leq b_2 \cdot y_2 \cdot A_i^\alpha$. ■

Chapter 6

Summary and Outlook

This thesis aims to bring the dynamic behavior to the world of data management problems in networks. We consider the most basic of these problems, called the Page Migration. By dynamics we mean that the network is subject to small continuous changes, like changes in bandwidth capacity, or the changes in the topology induced by node mobility. These network alterations induce the changes in the costs of communication between pairs of nodes. This thesis is based on the first papers concerning the *analytic* treatment of this problem. While our model might seem rather simple, it covers quite a lot of common cases.

Our algorithms exploit topological localities of requests, i.e., they are trying to adapt to the changing patterns of the accesses to the shared object by moving the object "near" the requesting nodes. Our main concern was to construct algorithms which are robust to the network changes. We considered several scenarios, which differed in the way of how the input sequence was created, and rigorously analyzed each of them, using the competitive analysis or its variants.

As the exact list of our results can be found in Section 1.2.2 and the open questions concerning particular results are presented at the ends of the individual chapters, we refrain from repeating them here. Instead, we try to provide the reader with a broader view on the new highlights brought by this work, in particular by the modelling used.

One of the most interesting contribution of this thesis is modelling the problem using two adversaries. Surprisingly, we were not able to locate any prior work, which considers two independent sources of online events (in our case, network dynamics and accesses to the memory page). If the corresponding two adversaries are allowed to cooperate, which is the case presented in Chapter 3, then this modelling is equivalent to having a single adversary and does not lead us beyond the pure competitive analysis [ST85]. However, we were able to prove that in such adversarial scenario the competitive ratios are inherently high (see Table 3.1 on page 86).

The over-pessimistic estimates of algorithms by the generic online analysis are one of the reasons why it was criticized in the last years. To compensate this effect, several methods and refinements were proposed, either giving extra resources to an online algorithm (lookahead properties, resource augmentation [KP00a]), restricting the adversary (ordered inputs [Mey01], access graph models [BIRS95], diffuse adversaries [KP00b], or smoothed competitive analysis [BLM+03]), or changing the performance metric (comparative analysis [KP00b] or again smoothed competitive analysis).

In our case, hardly any of these simplifications are applicable. Essentially, we wanted to have a notion which forbids the cooperation between our two adversaries. However, as it was not semantically clear how to define non-cooperativeness, we have added another piece to the list of refinements above, considering the case where one of the adversaries is replaced by a stochastic process. This leads to the case where the input consists of two interleaving sequences, one of which is the worst-possible and the second one is generated randomly. We have presented a complementary notion of competitive ratio attained with high probability and in expectation.[1] Although similar, this modelling substantially differs from the smoothed analysis, where the random distortion is added to the whole sequence, whereas in our approach only part of the input is randomized. In Chapter 4 and Chapter 5 we show that our modelling may significantly reduce the optimal algorithm's advantage of being clairvoyant. Although our algorithms presented for this model were specially designed for the DPM problem, we hope that the toolbox and ideas we have created might be reused for other problems.

There are, however, not many known problems, which allow for a natural division of the input sequence between two independent online resources. Examples are data management or scheduling in dynamic networks. The former is an area which we pioneered with this thesis. Extending our work to file allocation or distributed paging problems in such dynamic networks might be a challenging task. For the latter, a noble example is the paper by Leonardi, Marchetti-Spaccamela, and Meyer auf der Heide [LMM04]. Although they presented a solution to an *offline* scheduling problem in a network where free time slots of the network processors appear online, the problem itself can be easily reformulated to the one in which we have two independent online input streams (tasks to schedule and free slots of the processors).

[1] In fact the notion of expected competitive ratio was considered previously for completely random input sequences in [SSS02].

Bibliography

[AA92] Noga Alon and Yossi Azar. On-line steiner trees in the euclidean plane. In *Proc. of the 8th ACM Symp. on Computational Geometry (SoCG)*, pages 337–343, 1992.

[ABF93a] Baruch Awerbuch, Yair Bartal, and Amos Fiat. Competitive distributed file allocation. In *Proc. of the 25th ACM Symp. on Theory of Computing (STOC)*, pages 164–173, 1993.

[ABF93b] Baruch Awerbuch, Yair Bartal, and Amos Fiat. Heat & dump: Competitive distributed paging. In *Proc. of the 34th IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 22–31, 1993.

[ABF98] Baruch Awerbuch, Yair Bartal, and Amos Fiat. Distributed paging for general networks. *Journal of Algorithms*, 28(1):67–104, 1998. Also appeared in *Proc. of the 7th SODA*, pages 574–583, 1996.

[ABS03] Baruch Awerbuch, André Brinkmann, and Christian Scheideler. Anycasting in adversarial systems: routing and admission control. In *Proc. of the 30th Int. Colloq. on Automata, Languages and Programming (ICALP)*, pages 1153–1168, 2003.

[ACN00] Dimitris Achlioptas, Marek Chrobak, and John Noga. Competitive analysis of randomized paging algorithms. *Theoretical Computer Science*, 234(1–2):203–218, 2000.

[AK95] Susanne Albers and Hisashi Koga. Page migration with limited local memory capacity. In *Proc. of the 4th Int. Workshop on Algorithms and Data Structures (WADS)*, pages 147–158, 1995.

[AK98] Susanne Albers and Hisashi Koga. New on-line algorithms for the page replication problem. *Journal of Algorithms*, 27(1):75–96, 1998. Also appeared in *Proc. of the 4th SWAT*, pages 25–36, 1994.

[Bar95] Yair Bartal. *Competitive Analysis of Distributed On-line Problems — Distributed Paging*. PhD thesis, Tel-Aviv University, 1995.

[Bar96a] Yair Bartal. Distributed paging. In *Dagstul Workshop on On-line Algorithms*, pages 97–117, 1996.

[Bar96b] Yair Bartal. Probabilistic approximations of metric spaces and its algorithmic applications. In *Proc. of the 37th IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 184–193, 1996.

[Bar98] Yair Bartal. On approximating arbitrary metrics by tree metrics. In *Proc. of the 30th ACM Symp. on Theory of Computing (STOC)*, pages 161–168, 1998.

[BB05] Marcin Bienkowski and Jarosław Byrka. Bucket game with applications to set multicover and dynamic page migration. In *Proc. of the 13th European Symp. on Algorithms (ESA)*, 2005. To appear.

[BBK+90] Shai Ben-David, Allan Borodin, Richard M. Karp, Gabor Tardos, and Avi Wigderson. On the power of randomization in online algorithms. In *Proc. of the 22nd ACM Symp. on Theory of Computing (STOC)*, pages 379–386, 1990.

[BCI01] Yair Bartal, Moses Charikar, and Piotr Indyk. On page migration and other relaxed task systems. *Theoretical Computer Science*, 268(1):43–66, 2001. Also appeared in *Proc. of the 8th SODA*, pages 43–52, 1997.

[BDK05] Marcin Bienkowski, Miroslaw Dynia, and Miroslaw Korzeniowski. Improved algorithms for dynamic page migration. In *Proc. of the 22nd Symp. on Theoretical Aspects of Computer Science (STACS)*, pages 365–376, 2005.

[BE98] Allan Borodin and Ran El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998.

[BFR95] Yair Bartal, Amos Fiat, and Yuval Rabani. Competitive algorithms for distributed data management. *Journal of Computer and System Sciences*, 51(3):341–358, 1995. Also appeared in *Proc. of the 24nd STOC*, pages 39–50, 1992.

[Bie05] Marcin Bienkowski. Dynamic page migration with stochastic requests. In *Proc. of the 17th ACM Symp. on Parallelism in Algorithms and Architectures (SPAA)*, pages 270–278, 2005.

[BIRS95] Allan Borodin, Sandy Irani, Prabhakar Raghavan, and Baruch Schieber. Competitive paging with locality of reference. *Journal of Computer and System Sciences*, 50(2):244–258, 1995. Also appeared in *Proc. of the 23rd STOC*, pages 249–259, 1991.

[BK05] Marcin Bienkowski and Miroslaw Korzeniowski. Dynamic page migration under brownian motion. In *Proc. of the European Conf. in Parallel Processing (Euro-Par)*, 2005. To appear.

[BKM04] Marcin Bienkowski, Miroslaw Korzeniowski, and Friedhelm Meyer auf der Heide. Fighting against two adversaries: Page migration in dynamic networks. In *Proc. of the 16th ACM Symp. on Parallelism in Algorithms and Architectures (SPAA)*, pages 64–73, 2004.

[BKR03] Marcin Bienkowski, Miroslaw Korzeniowski, and Harald Räcke. A practical algorithm for constructing oblivious routing schemes. In *Proc. of the 15th ACM Symp. on Parallelism in Algorithms and Architectures (SPAA)*, pages 24–33, 2003.

[BLM+03] Luca Becchetti, Stefano Leonardi, Alberto Marchetti-Spaccamela, Guido Schäfer, and Tjark Vredeveld. Average case and smoothed competitive analysis of the multi-level feedback algorithm. In *Proc. of the 44th IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 462–471, 2003.

[BM05] Marcin Bienkowski and Friedhelm Meyer auf der Heide. Page migration in dynamic networks. In *Proc. of the 30th Int. Symp. on Mathematical Foundations of Computer Science (MFCS)*, 2005. Invited paper. To appear.

[BS89] David L. Black and Daniel D. Sleator. Competitive algorithms for replication and migration problems. Technical Report CMU-CS-89-201, Department of Computer Science, Carnegie-Mellon University, 1989.

[CBD02] Tracy Camp, Jeff Boleng, and Vanessa Davies. A survey of mobility models for ad hoc network research. *Wireless Communications & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, 2(5):483–502, 2002.

[Che52] Herman Chernoff. A measure of assymptotic efficiency for tests of a hypothesis based on the sum of observations. *Annals of Mathematical Statistics*, 23:493–509, 1952.

[CLLR97] Marek Chrobak, Lawrence L. Larmore, Carsten Lund, and Nick Reingold. A better lower bound on the competitive ratio of the randomized 2-server problem. *Information Processing Letters*, 63(2):79–83, 1997.

[CLR97] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. The MIT Press, eighteenth edition, 1997.

[CLRW93] Marek Chrobak, Lawrence L. Larmore, Nick Reingold, and Jeffery Westbrook. Page migration algorithms using work functions. In *Proc. of the 4th Int. Symp. on Algorithms and Computation (ISAAC)*, pages 406–415, 1993.

[Fel68] William Feller. *An Introduction to Probability Theory and Its Applications*, volume I. John Wiley & Sons, Inc., third edition, 1968.

[FGS04] Rudolf Fleischer, Włodzimierz Głazek, and Steve S. Seiden. New results for online page replication. *Theoretical Computer Science*, 324(2–3):219–251, 2004.

[FKL+91] Amos Fiat, Richard M. Karp, Michael Luby, Lyle A. McGeoch, Daniel D. Sleator, and Neal E. Young. Competitive paging algorithms. *Journal of Algorithms*, 12(4):685–699, 1991.

[FS00] Rudolf Fleischer and Steven S. Seiden. New results for online page replication. In *Proc. of the 3rd Int. Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX)*, pages 144–154, 2000.

[Gła99] Włodzimierz Głazek. Lower and upper bounds for the problem of page replication in ring networks. In *Proc. of the 24th Int. Symp. on Mathematical Foundations of Computer Science (MFCS)*, pages 273–283, 1999.

[Gła01] Włodzimierz Głazek. Online algorithms for page replication in rings. *Theoretical Computer Science*, 268(1):107–117, 2001.

[HHR03] Chris Harrelson, Kirsten Hildrum, and Satish Rao. A polynomial-time tree decomposition to minimize congestion. In *Proc. of the 15th ACM Symp. on Parallelism in Algorithms and Architectures (SPAA)*, pages 34–43, 2003.

[HLP88] Godfrey H. Hardy, John E. Littlewood, and George Pólya. *Inequalities*. Cambridge University Press, 2nd edition, 1988.

[Hoe63] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistics Association*, 58(301):13–30, 1963.

[IW91] Makoto Imase and Bernard M. Waxman. Dynamic steiner tree problem. *Journal on Discrete Mathematics*, 4(3):369–384, 1991.

[KMR+02] Christof Krick, Friedhelm Meyer auf der Heide, Harald Räcke, Berthold Vöcking, and Matthias Westermann. Data management in networks: Experimental evaluation of a provably good strategy. *Theory of Computing Systems*, 2:217–245, 2002. Also appeared in *Proc. of the 11nd SPAA*, pages 165–174, 1999.

[KMRS88] Anna R. Karlin, Mark S. Manasse, Larry Rudolph, and Daniel D. Sleator. Competitive snoopy caching. *Algorithmica*, 3:77–119, 1988. Also appeared in *Proc. of the 27th FOCS*, pages 244-254, 1986.

[Kog93] Hisashi Koga. Randomized on-line algorithms for the page replication problem. In *Proc. of the 4th Int. Symp. on Algorithms and Computation (ISAAC)*, pages 436–445, 1993.

[KP00a] Bala Kalyanasundaram and Kirk Pruhs. Speed is as powerful as clairvoyance. *Journal of the ACM*, 47(4):617–643, 2000. Also appeared in *Proc. of the 36nd FOCS*, pages 214–221, 1995.

[KP00b] Elias Koutsoupias and Christos H. Papadimitriou. Beyond competitive analysis. *SIAM Journal on Computing*, 30(1):300–317, 2000. Also appeared in *Proc. of the 35th FOCS*, pages 394–400, 1994.

[LMM04] Stefano Leonardi, Alberto Marchetti-Spaccamela, and Friedhelm Meyer auf der Heide. Scheduling against an adversarial network. In *Proc. of the 16th ACM Symp. on Parallelism in Algorithms and Architectures (SPAA)*, pages 151–159, 2004.

[LRWY99] Carsten Lund, Nick Reingold, Jeffery Westbrook, and Dicky C. K. Yan. Competitive on-line algorithms for distributed data management. *SIAM Journal on Computing*, 28(3):1086–1111, 1999. Also appeared as On-Line Distributed Data Management in *Proc. of the 2nd ESA*, pages 202–214, 1994.

[Mey01] Adam Meyerson. Online facility location. In *Proc. of the 42nd IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 426–431, 2001.

[MMVW97] Bruce M. Maggs, Friedhelm Meyer auf der Heide, Berthold Vöcking, and Matthias Westermann. Exploiting locality for data management in systems of limited bandwidth. In *Proc. of the 38th IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 284–293, 1997.

[MR95] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[MS91] Lyle A. McGeoch and Daniel D. Sleator. A strongly competitive randomized paging algorithm. *Algorithmica*, 6(6):816–825, 1991.

[MVW99] Friedhelm Meyer auf der Heide, Berthold Vöcking, and Matthias Westermann. Provably good and practical strategies for non-uniform data management in networks. In *Proc. of the 7th European Symp. on Algorithms (ESA)*, pages 89–100, 1999.

[MVW00] Friedhelm Meyer auf der Heide, Berthold Vöcking, and Matthias Westermann. Caching in networks. In *Proc. of the 11th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 430–439, 2000.

[NM44] John von Neumann and Oskar Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1st edition, 1944.

[Räc02] Harald Räcke. Minimizing congestion in general networks. In *Proc. of the 43rd IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 43–52, 2002.

[Räc03] Harald Räcke. *Data management and routing in general networks*. PhD thesis, Universität Paderborn, 2003.

[Raj02] Rajmohan Rajaraman. Topology control and routing in ad hoc networks: a survey. *SIGACT News*, 33(2):60–73, 2002.

[Rap96] Theodore S. Rappaport. *Wireless Communications: Principles and Practices*. Prentice Hall, 1996.

[Ros95] Jeffrey S. Rosenthal. Convergence rates for Markov chains. *SIAM Review*, 37(3):387–405, 1995.

[RPRR01] Sanguthevar Rajesekaran, Panos M. Pardalos, John H. Reif, and Jose Rolim. *Handbook of Randomized Computing*, volume I and II. Kluwer Academic Publishers, 2001.

[Sch02] Christian Scheideler. Models and techniques for communication in dynamic networks. In *Proc. of the 19th Symp. on Theoretical Aspects of Computer Science (STACS)*, pages 27–49, 2002.

[Sen81] Eugene Seneta. *Non-negative Matrices and Markov Chains*. Springer-Verlag, New York, 2nd edition, 1981.

[SLRV03] Christian Schindelhauer, Tamás Lukovszki, Stefan Rührup, and Klaus Volbert. Worst case mobility in ad hoc networks. In *Proc. of the 15th ACM Symp. on Parallelism in Algorithms and Architectures (SPAA)*, pages 230–239, 2003.

[SS04] Guido Schäfer and Naveen Sivadasan. Topology matters: Smoothed competitiveness of metrical task systems. In *Proc. of the 21st Symp. on Theoretical Aspects of Computer Science (STACS)*, pages 489–500, 2004.

[SSS02] Mark Scharbrodt, Thomas Schickinger, and Angelika Steger. A new average case analysis for completion time scheduling. In *Proc. of the 34th ACM Symp. on Theory of Computing (STOC)*, pages 170–178, 2002.

[ST85] Daniel D. Sleator and Robert E. Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28(2):202–208, 1985.

[Tar85] Robert E. Tarjan. Amortized computational complexity. *SIAM Journal on Algebraic and Discrete Methods*, 6(2):306–318, 1985.

[Wes92] Jeffery Westbrook. Randomized algorithms for multiprocessor page migration. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 7:135–150, 1992.

[Wes00] Matthias Westermann. *Caching in Networks: Non-Uniform Algorithms and Memory Capacity Constraints*. PhD thesis, Universität Paderborn, 2000.

[Yao77] Andrew Chi-Chih Yao. Probabilistic computation: towards a uniform measure of complexity. In *Proc. of the 18th IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 222–227, 1977.

Appendix

A.1 Notations

The following list summarizes some basic notations used in the thesis.

- $\mathbb{N} = \{0, 1, 2, \ldots\}$, the set of non-negative integers
- $\mathbb{Z}$ is the set of all integers
- $\mathbb{Z}/(n)$ is the group of integers modulo n
- $\mathbb{R}$ is the set of all real numbers
- For $n \in \mathbb{N}$, $[n] := \{1, \ldots, n\}$
- Let $f(n)$ by any function. Following [CLR97, chapter 2], we define the following sets of functions
  - $O(f(n)) = \{g(n) \mid \exists c \in \mathbb{R}^+ \text{ and } n_0 \in \mathbb{N} : \forall n \geq n_0 \; f(n) \leq c \cdot g(n)\}$
  - $\Omega(f(n)) = \{g(n) \mid \exists c \in \mathbb{R}^+ \text{ and } n_0 \in \mathbb{N} : \forall n \geq n_0 \; f(n) \geq c \cdot g(n)\}$
  - $\Theta(f(n)) = \{g(n) \mid g(n) = O(f(n)) \text{ and } g(n) = \Omega(f(n))\}$
  - $o(f(n)) \;\; = \{g(n) \mid \forall c \in \mathbb{R}^+ \; \exists n_0 \in \mathbb{N} : \forall n \geq n_0 \; f(n) \leq c \cdot g(n)\}$
  - $\omega(f(n)) = \{g(n) \mid \forall c \in \mathbb{R}^+ \; \exists n_0 \in \mathbb{N} : \forall n \geq n_0 \; f(n) \geq c \cdot g(n)\}$

  For any two functions $f(n)$, $g(n)$, by $f(n) = O(g(n))$ we mean $f(n) \in O(f(n))$. The same holds for the other classes of functions defined above.
- As O, Ω, and Θ notation neglect any constants in the function definitions, the analogous definitions of classes $\tilde{O}$, $\tilde{\Omega}$, and $\tilde{\Theta}$ neglect the polylogarithmic terms.
- For any two probability distributions ν_1 and ν_2 defined on some discrete space $\mathcal{X}$, their variation distance is defined as $\|\nu_1 - \nu_2\| := \max_{A \subseteq \mathcal{X}} |\nu_1(A) - \nu_2(A)|$.

A.2 Mathematical tools

In this part of the appendix we present some standard mathematical tools, inequalities, and bounds used throughout this thesis.

Probability theory

Lemma A.1 (Markov inequality [Fel68]). *Let X be a random variable taking non-negative values. Then for all $t \in \mathbb{R}^+$ it holds that,*

$$\mathbf{Pr}[X \geq t] \leq \frac{\mathbf{E}[X]}{t} \ .$$

Lemma A.2 (Hoeffding bound [Hoe63, RPRR01]). *Let $X_1, \ldots, X_n$ be independent random variables such that, for all $i \in [n]$, there are values a_i and b_i such that $a_i \leq X_i \leq b_i$. Let $X = \sum_{i=1}^{n} X_i$ and $\mu = \mathbf{E}[X]$. Then for any $t \geq 0$*

$$\mathbf{Pr}[X - \mu > t] \ \leq \ \exp\left(-\frac{2 \cdot t^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\right) \ \text{and}$$
$$\mathbf{Pr}[X - \mu < -t] \ \leq \ \exp\left(-\frac{2 \cdot t^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\right) \ .$$

Lemma A.3 (Random walk on abelian group [Ros95]). *Consider any additive group $\mathbb{Z}/(k)$ (i.e., integers modulo k with addition). Fix any starting probability distribution π_0 over the elements of $\mathbb{Z}/(k)$. Consider a random walk on this group with transition probabilities given by the following stochastic matrix P:*

$$P(x, y) = \begin{cases} 1/3 & \text{if } y \in \{x-1, x, x+1\}, \\ 0 & \text{otherwise.} \end{cases}$$

After k steps the probability distribution on $\mathbb{Z}/(k)$ is equal to $\pi_k = P^k \cdot \pi_0$. Then the variation distance between π_k and a uniform distribution μ fulfills,

$$\|\pi_k - \mu\| \leq \sqrt{\frac{\exp\left(-\frac{4 \cdot \pi^2}{3 \cdot B^2} \cdot k\right)}{1 - \exp\left(-\frac{4 \cdot \pi^2}{3 \cdot B^2} \cdot k\right)}} \ .$$

Common inequalities

Lemma A.4 (Cauchy-Schwarz inequality [HLP88]). *For any sequences $(a_i)_{i=1}^n$ and $(b_i)_{i=1}^n$, it holds that*

$$\left(\sum_{i=1}^{n} a_i \cdot b_i\right)^2 \leq \left(\sum_{i=1}^{n} a_i^2\right) \cdot \left(\sum_{i=1}^{n} b_i^2\right) \ .$$

Lemma A.5 (Hölder's Inequality [HLP88]). *For any $p, q > 1$ such that $\frac{1}{p} + \frac{1}{q} = 1$ and for any non-negative sequences $(a_i)_{i=1}^k$ and $(b_i)_{i=1}^k$, it holds that*

$$\sum_{i=1}^{k} (a_i \cdot b_i) \;\leq\; \left(\sum_{i=1}^{k} a_i^p\right)^{1/p} \cdot \left(\sum_{i=1}^{k} b_i^q\right)^{1/q} \quad .$$

Lemma A.6 (Jensen's inequality [HLP88]). *If f is a convex and continuous function, then for any sequence of numbers $(x_i)_{i=1}^n$ and any $(p_i)_{i=1}^n$, such that $0 \leq p_i \leq 1$ and $\sum_{i=1}^n p_i = 1$, it holds that*

$$f\left(\sum_{i=1}^{n} p_i \cdot x_i\right) \;\leq\; \sum_{i=1}^{n} p_i \cdot f(x_i) \quad .$$

Miscellaneous

Lemma A.7 (Stirling formula [Fel68]). *For any natural n, let $A_n = \sqrt{2\pi} \cdot n^{n+1/2} \cdot \mathrm{e}^{-n}$. Then*

$$A_n \cdot \mathrm{e}^{1/(12n+1)} < n! < A_n \cdot \mathrm{e}^{1/(12n)} \quad .$$