

# 1 Introduction

We have a fast memory, a cache, which can contain up to  $k$  different pages and we receive a sequence of requests to pages in an online manner. For instance,

$$P := ACFBG \tag{1}$$

$$\sigma = A, A, C, D, F, F, A, B \tag{2}$$

where  $\sigma$  is the sequence of page requests and  $k = 4$ .  $P$  is the cache content.

Whenever a page request is already included in the cache content, then it can be accessed at no cost, otherwise we suffer a *page fault*. Here one page must be evicted to make room for the request. This page to be evicted must be chosen in an online manner without knowledge of future requests. Therefore, we want to minimize the total number of page faults.

The online nature of caching and Metric Task Systems in general forces an algorithm to be suboptimal. This is why predictions are introduced. Nevertheless, the use of predictions does not guarantee an optimal behaviour of an algorithm. Therefore, we evaluate it based on three elements:

- **Consistency:** performance with perfect predictions should be optimal;
- **Robustness:** performance with bad predictions should not be worse than what is achievable by known algorithms that do not utilize predictions;
- **Smoothness:** deterioration of performance with prediction errors should be smooth between consistency and robustness.

Given the computationally intensive nature of producing predictions, it is interesting to produce algorithms that are parsimonious in the number of predictions used. The following algorithms work with **action predictions** where each prediction is the state of an optimal algorithm at a given time step. We study how the algorithms would behave when we have a bound on the total number of predictions and when queries to a predictor can be accessed only after some defined time steps.

## 1.1 Results

The first contribution is an algorithm for caching that receives predictions emulating states of an optimal offline algorithm, Belady's algorithm.

**Theorem 1.1.** *Let  $f$  be an increasing convex function such that  $f(0) = 0$  and  $f(i) \leq 2^i - 1$  for each  $i \geq 0$ . There is an algorithm for caching requiring at most  $f(\log k)OPT$  predictions which achieves consistency 1, robustness  $O(\log k)$ , and smoothness  $O(f^{-1}(\rho/OPT))$ , where  $\rho$  is the prediction error with respect to Belady and  $OPT$  is the number of page faults of Belady.*

**Theorem 1.2.** *Any  $f^{-1}(\rho)$ -smooth algorithm for caching with action predictions has to use a number of predictions bounded between  $f(\ln k)OPT \leq x \leq f(\log k)OPT$ .*

In terms of an MTS problem, the number of predictions cannot be bounded by a function of  $OPT$ , since the system can be scaled down to make  $OPT$  arbitrarily small. Given the small number of predictions that can be used, the algorithm proposed queries the predictor once in every time step  $a$ , thus making a total number of queries of  $\frac{\text{len}(\sigma)}{a}$ .

**Theorem 1.3.** *There is a deterministic algorithm for any MTS which receives a prediction only once per each  $a$  time steps and its cost is at most  $O(a) \cdot (OFF + 2\rho)$ , where  $OFF$  denotes the cost of an arbitrary offline algorithm.*

This means that the algorithm for any MTS would have a competitive smoothness of

$$\frac{O(a) \cdot (OFF + 2\rho)}{OFF} = O(a) \cdot \left(1 + \frac{2\rho}{OFF}\right) \quad (3)$$

and with perfect predictions, hence  $\rho = 0$ , consistency of  $O(a)$ . The same concept can be applied to caching algorithms thus having predictor queries separated by  $a$  time steps, without losing its stated properties in consistency, robustness and smoothness.