Alcom-FT Technical Report Series
ALCOMFT-TR-01-30

On Multicriteria Online Problems*

Michele Flammini
Dipartimento di Matematica Pura ed Applicata, University of L'Aquila,
Via Vetoio loc. Coppito, I-67100 L'Aquila, Italy.
E-mail: `flammini@univaq.it`

Gaia Nicosia
Dipartimento di Informatica ed Automazione, University of Roma Tre,
Via della Vasca Navale 79, I-00146 Roma, Italy.
E-mail: `nicosia@dia.uniroma3.it`

Abstract

In this paper we consider multicriteria formulations of classical online problems in which an algorithm must simultaneously perform well with respect to two different cost measures. Every strategy for serving a sequence of requests is characterized by a pair of costs, and therefore there can be many different minimal or optimal incomparable solutions. The performance of the algorithm is compared with that of an adversary that serves the sequence selecting one of such optimal offline strategies according to a given selection function. We consider a parametric family of functions which includes all the possible selections. Then, starting from a simple general method that combines any multicriteria instance into a single-criterion one, we provide a universal multicriteria algorithm that can be applied to different online problems. In the multicriteria k-server formulation with two different edge weightings, for each function class, such a universal algorithm achieves competitive ratios that are only an $O(\log W)$ multiplicative factor away from the corresponding determined lower bounds, where W is the maximum ratio between the two weights associated to each edge. We then extend our results to more specific functions, for which optimal or nearly optimal competitive algorithms are obtained by exploiting more knowledge of the selection properties. Finally, we show how to apply our framework to other multicriteria online problems sharing similar properties.

1 Introduction

In this paper we consider instances of the k-server problem where each edge of the graph G is assigned a time and a length. k servers must move among the nodes in order to fulfill the various requests and to minimize both the total traveling time and length.

The above problem belongs to the class of the so called *multicriteria* or *multiobjective* problems, where the same solution has to be evaluated with respect to different cost measures. Such problems have been widely investigated in recent years with special attention to the determination of good approximate solutions [10, 11, 13, 17, 19, 15]. For instance, one can ask for the

*Partially supported by the IST Programme of the EU under contract number IST-1999-14186 (ALCOM-FT).

determination of a spanning tree of a graph whose global cost is low with respect to two different weightings of the edges [14, 16], or such that the diameter is low with respect to first weighting and has low global cost with respect to the second one [14].

In this paper we extend the multicriteria setting to online problems where the algorithm must satisfy a series of requests arriving along the time without knowledge of the future requests (see [2] for a survey). Assuming the existence of two cost measures for serving requests, the algorithm must compare favorably for both measures with respect to any algorithm knowing all the sequence in advance. This concept is not completely new in the area. For instance, in [6] the authors propose algorithms for routing and bandwidth allocation which simultaneously approximate fair and max throughput solutions.

A multicriteria algorithm can fight against two different types of adversaries. The first one is more powerful than the algorithm, as it can serve any sequence of requests differently for the two cost measures. This translates in finding, among all possible algorithms that are c_1-competitive with respect to the first cost measure, one that has the lowest competitive ratio c_2 with respect to the second one. Basically, it requires the use of standard competitive techniques with the additional constraint that, while minimizing the competitive ratio of one cost function, the ratio for the other measure must be kept below a certain limit.

In this paper we consider a more realistic type of adversary, called *fair*, that has the same constraints of the algorithm, that is, it is compelled to serve the sequence of requests in the same way for both the two cost measures. In fact, this gives a more realistic estimate of how the algorithm is paying (in terms of competitive ratio) its missing knowledge of the future and seems to correspond to a more appropriate view of the multicriteria setting. In this case, since any offline strategy for a sequence of requests σ is characterized by a pair of costs, there is not a unique optimal strategy for the adversary, but a set $\mathcal{O}(\sigma)$ of nondominated or incomparable minimal strategies. Thus, we assume the existence of a selection function f that associates to each σ a solution in $\mathcal{O}(\sigma)$ the performance of the algorithm is compared with.

We consider a complete classification of the selection functions based on their monotonicity properties. A function is called *monotone* if both its two optimal offline costs do not decrease as the sequence of requests grows, while is called (Δ_1, Δ_2)-monotone, for two parameters $\Delta_1 \leq 1$ and $\Delta_2 \leq 1$, if the decrease for each supersequence is bounded by a multiplicative factor equal to Δ_1 with respect to the first cost measure and Δ_2 with respect to the second one.

We prove that no algorithm for the class of the (Δ_1, Δ_2)-monotone functions with $\Delta_1\Delta_2 \geq \frac{1}{W^2}$ can be (c_1, c_2)-competitive with $c_1\Delta_1 + c_2\Delta_2 < k$. If $c_1 \leq \frac{k}{2\Delta_1}$, this implies $c_2 \geq \frac{k}{2\Delta_2}$. A similar lower bound is also shown for $\Delta_1\Delta_2 < \frac{1}{W^2}$.

Then, we give a simple algorithm for the multicriteria k-server problem that is based on the linear combination of multicriteria instances into single-criterion ones and realizes trade-offs between the two competitive ratios. Namely, denoted as W the maximum ratio between the two weights associated to each edge and given a fixed parameter λ such that $1/W \leq \lambda \leq W$, the two ratios are $O(kW\lambda)$ and $O(kW/\lambda)$. The above lower bound proves that for $\lambda = \frac{1}{W\Delta_1}$ such an algorithm is asymptotically optimal when $\Delta_1\Delta_2 \leq \frac{1}{W^2}$.

Clearly, the higher is the product $\Delta_1\Delta_2$, the worse is the performance of the simple combination algorithm. However, we show how it can be exploited to get a universal algorithm with competitive ratios $O(\frac{k \log W}{\Delta_1})$ and $O(\frac{k \log W}{\Delta_2})$ versus any (Δ_1, Δ_2)-monotone function. Hence, our algorithm in every case looses only an $O(\log W)$ multiplicative factor with respect to the lower bound.

Finally, we consider some specific selection functions, and by exploiting their structural properties, we propose competitive algorithms whose ratios compare nicely with respect to the specific lower bounds that we determine for each function.

We conclude by showing how our results can be extended to other multicriteria online problems dealing with homogeneous optimization criteria and sharing similar properties.

The paper is organized as follows. The next section is introductory and gives all the necessary background and definitions. In Section 3 we present the above mentioned combination algorithm and a simplified version of the universal one that works for strictly monotone selection functions. In Section 4 we extend it to the (Δ_1, Δ_2)-monotone functions for every $\Delta_1, \Delta_2 \leq 1$ and we prove the lower bound for the class. Section 5 gives some basic facts and properties related of the selection functions that will be useful in Section 6 to prove results related to specific selection functions. In Section 7 we extend our framework to other online problems and finally, in Section 8, we give some conclusive remarks and discuss some open problems.

2 Preliminaries

An algorithm A for an online problem is faced with a sequence $\sigma = \langle r_1, \ldots, r_m \rangle$ of requests and has to satisfy each incoming request r_i without any knowledge of the successive ones. The way A serves r_i causes a certain cost $cost(A, r_i)$ and affects future costs.

The performance of an online algorithm is usually compared with the one of an optimal offline algorithm that knows the sequence σ in advance.

Definition 2.1 [8, 18] *A is said c-competitive if, for all possible sequences σ, $cost(A, \sigma) = \sum_i cost(A, r_i) \leq c \cdot opt(\sigma) + \alpha$, where $opt(\sigma)$ is the cost paid to serve σ by an optimal offline algorithm that knows all the sequence σ in advance and α is a suitable constant independent from the length m of σ.*

This worst-case analysis is usually performed by assuming that the sequence of requests is supplied by an adversary that at each step provides the request so as to maximize the ratio between the overall cost paid by A and its own (optimal offline) cost. Hence, Definition 2.1 can be rephrased by saying that $cost(A, \sigma) \leq c \cdot opt(\sigma) + \alpha$, where σ is any sequence generated by the adversary.

In the bicriteria formulation of an online problem, it is assumed that the cost of serving each request r_i is evaluated simultaneously with respect to two different cost functions $cost_1$ and $cost_2$. Hence, the execution of any algorithm A over σ is characterized by a pair of costs $(cost_1(A, \sigma), cost_2(A, \sigma))$. As a consequence, in general there is not a unique optimal solution, but there can be many different minimal or optimal incomparable offline solutions.

Definition 2.2 *Given any sequence of requests σ, a solution for σ with costs $(cost_1(B, \sigma), cost_2(B, \sigma))$ provided by an offline algorithm B is a* nondominated solution *if and only if no algorithm C exists such that $cost_1(C, \sigma) < cost_1(B, \sigma)$ and $cost_2(C, \sigma) \leq cost_2(B, \sigma)$, or $cost_1(C, \sigma) \leq cost_1(B, \sigma)$ and $cost_2(C, \sigma) < cost_2(B, \sigma)$.*

The set $\mathcal{O}(\sigma)$ of nondominated solutions associated to σ thus forms the frontier of the optimal offline strategies, sometimes called *Pareto frontier* or *efficient frontier*. In order to evaluate the performance of an online algorithm A, we assume the existence of a selection criteria f known by A that associates to each σ a nondominated strategy in $\mathcal{O}(\sigma)$ with costs $(opt_1(f,\sigma), opt_2(f,\sigma))$.

Definition 2.3 *Given a selection function f, an algorithm A is said (c_1, c_2)-competitive versus f if, for all possible sequences σ, $cost_1(A,\sigma) \leq c_1 \cdot opt_1(f,\sigma) + \alpha_1$ and $cost_2(A,\sigma) \leq c_2 \cdot opt_2(f,\sigma) + \alpha_2$, where $(opt_1(f,\sigma), opt_2(f,\sigma))$ is the pair of costs paid by the optimal offline strategy in $\mathcal{O}(\sigma)$ selected by f and α_1, α_2 are suitable constants.*

Again, the analysis can be performed by assuming that the sequence of requests is supplied by an adversary that knowing f generates σ in order to maximize the two competitive ratios. Consequently, in Definition 2.3 it is possible to say that σ is any sequence generated by the adversary.

A reasonable assumption over the selection function f is that both $opt_1(f,\sigma)$ and $opt_2(f,\sigma)$ do not decrease as σ grows, or their decrease can be suitably bounded.

Definition 2.4 *A selection function f is monotone if and only if, for any sequence of requests σ and for every prefix σ' of σ, $opt_1(f,\sigma) + \alpha_1 \geq opt_1(f,\sigma')$ and $opt_2(f,\sigma) + \alpha_2 \geq opt_2(f,\sigma')$, where α_1 and α_2 are suitable constants.*

Definition 2.5 *Given any $\Delta_1 \leq 1$ and $\Delta_2 \leq 1$, a selection function f is (Δ_1, Δ_2)-monotone if and only if, for any sequence of requests σ and for every prefix σ' of σ, $opt_1(f,\sigma) + \alpha_1 \geq \Delta_1 \cdot opt_1(f,\sigma')$ and $opt_2(f,\sigma) + \alpha_2 \geq \Delta_2 \cdot opt_2(f,\sigma')$, where α_1 and α_2 are suitable constants.*

The constants α_1 and α_2 in Definitions 2.4 and 2.5 may depend on the problem characteristics, but not on the sequence σ.

For any sequence of requests σ and selection f, let f^σ be the function defined only on the prefixes σ' of σ and such that $f^\sigma(\sigma')$ coincides with the restriction of $f(\sigma)$ on σ'. Notice that $f^\sigma(\sigma')$ in general might not belong to $\mathcal{O}(\sigma)$, i.e., it can be dominated. The pair of costs associated to a dominated strategy $g(\sigma)$ will always be denoted as $cost_1(g,\sigma)$ and $cost_2(g,\sigma)$.

Clearly, if $f(\sigma') = f^\sigma(\sigma')$ for every σ and prefix σ' of σ, then f is monotone. On the other hand, a (Δ_1, Δ_2)-monotononiticy with low values of Δ_1 and Δ_2 is in general related to the fact that $f(\sigma')$ and $f^\sigma(\sigma')$ can be extremely different. In this case an online algorithm pays both the fact that it does not know the future and that it does not know the offline nondominated strategy it is comparing with.

In this paper we will mainly concentrate on the k-server problem (see [4, 5, 12]) in which, given a weighted graph $G = (V, E)$ with strictly positive edge weights and k mobile servers in G, a request generated at a particular node $u \in V$ involves the movement of one server from its current position $v \in V$ to u, incurring a cost equal to the distance traveled in G. The cost $cost(A,\sigma)$ paid by an algorithm A over σ is the total distance traveled by all the servers.

In the bicriteria formulation of the k-server problem each edge $e \in E$ has a pair of weights $(w_1(e), w_2(e))$ and $(cost_1(A,\sigma), cost_2(A,\sigma))$ is the pair of costs paid by A with respect to the two different edge weightings. We denote as $W = \max_{e \in E} \left\{ \frac{w_1(e)}{w_2(e)}, \frac{w_2(e)}{w_1(e)} \right\}$ the *maximum edge ratio* and as $w_{max} = \max_{e \in E}\{w_1(e), w_2(e)\}$ the *maximum edge weight* of G.

Often in the following we will assume that G is a multigraph with multiple edges between each pair of nodes. In fact, by means of a simple reduction substituting each edge e with a path of two nodes that globally preserves the weights of e, it can be shown that this problem is not more general than the one on simple graphs.

Since $w_1(e)/w_2(e) \leq W$ and $w_2(e)/w_1(e) \leq W$ for every $e \in E$, given any sequence of requests σ and selection function f, $opt_1(f,\sigma) \leq W opt_2(f,\sigma)$ and $opt_2(f,\sigma) \leq W opt_1(f,\sigma)$. Therefore, it is possible to prove the following fact.

Fact 2.6 *Every selection function f for the bicriteria k-server problem is $(\frac{1}{W^2}, \frac{1}{W^2})$-monotone.*

Proof. Let g be any selection function such that $g(\sigma) \in \mathcal{O}(\sigma)$ for every sequence of requests σ. Then, if $opt_1(g,\sigma) < opt_1(f,\sigma)$, $opt_1(g,\sigma) \geq opt_2(g,\sigma)/W \geq opt_2(f,\sigma)/W \geq opt_1(f,\sigma)/W^2$. Thus, in every case $opt_1(g,\sigma) \geq opt_1(f,\sigma)/W^2$.

The claim then holds by observing that for any σ and prefix σ' of σ, if g is such that $g(\sigma') = f^\sigma(\sigma')$ if $f^\sigma(\sigma') \in \mathcal{O}(\sigma)$, otherwise $g(\sigma')$ is a nondominated solution in $\mathcal{O}(\sigma)$ that dominates $f^\sigma(\sigma')$, $opt_1(f,\sigma) \geq cost_1(f^\sigma,\sigma') \geq opt_1(g,\sigma') \geq opt_1(f,\sigma')/W^2$

A symmetric argument shows that also $opt_2(g,\sigma) \geq opt_2(f,\sigma)/W^2$ and as a direct consequence $\Delta_1 \geq 1/W^2$ and $\Delta_2 \geq 1/W^2$. □

We conclude the section with some example of reasonable selection functions that will be considered in the sequel.

Example 2.7 *Let f_1, f_2, f_3, f_4 and f_5 be the selection functions for the bicriteria k-server problem defined as follows:*

1. *f_1 chooses a nondominated solution in $\mathcal{O}(\sigma)$ such that $opt_1(f_1,\sigma)$ is minimum;*

2. *given a positive real number β, f_2 chooses a nondominated solution in $\mathcal{O}(\sigma)$ minimizing $\max(opt_1(f_2,\sigma), \beta opt_2(f_2,\sigma))$;*

3. *given a positive real number β, f_3 chooses a nondominated solution in $\mathcal{O}(\sigma)$ such that $opt_1(f_3,\sigma) \approx \beta opt_2(f_3,\sigma)$, i.e., minimizing $|opt_1(f_3,\sigma) - \beta opt_2(f_3,\sigma)|$;*

4. *given a positive real number β, f_4 chooses a nondominated solution in $\mathcal{O}(\sigma)$ that minimizes $opt_1(f_4,\sigma) + \beta opt_2(f_4 \sigma)$;*

5. *given a positive real number β, f_5 chooses a nondominated solution in $\mathcal{O}(\sigma)$ such that $opt_1(f_5,\sigma)$ does not exceed a given budget $\beta \cdot m$ proportional to the length m of σ and such that $opt_2(f_5,\sigma)$ is minimum.*

When $k = 1$ it is possible to show that f_1, f_2 and f_3 are monotone, f_4 is $(\frac{\beta+W}{W(\beta W+1)}, \frac{\beta W+1}{W(\beta+W)})$ and not (Δ_1, Δ_2)-monotone for $\Delta_1 > \frac{\beta+W}{W(\beta W+1)}$ or $\Delta_2 > \frac{\beta W+1}{W(\beta+W)}$, and f_5 is $(1, \frac{1}{W^2})$-monotone and not $(1, \Delta_2)$-monotone for $\Delta_2 > \frac{1}{W^2}$. For $k > 1$ in general the monotonicity becomes worse and also f_1, f_2 and f_3 are not monotone. The above selection functions and their properties will be considered in more detail in Section 6.

3 A universal algorithm for monotone selections

In this section we present a universal algorithm for the bicriteria k-server problem versus monotone selection functions. As we will see in the sequel, it can be extended to all (Δ_1, Δ_2)-monotone functions with $\Delta_1, \Delta_2 \leq 1$ and to deal with other bicriteria problems.

The basic idea downstanding our algorithm relies on the algorithm in Figure 1. Informally, given a c-competitive algorithm A for the single-criterion k-server problem and a parameter λ such that $1/W \leq \lambda \leq W$, $COMB(A, \lambda)$ consists in moving the servers in G as A in the instance of the (single-criterion) k-server problem with the graph G_λ having edge weights $w(e) = w_1(e) + \lambda w_2(e)$ for each $e \in E$.

`INPUT: algorithm` A `for the` k`-server problem, real number` λ `with` $1/W \leq \lambda \leq W$ `and graph` $G = (V, E)$ `with edge weights` $(w_1(e), w_2(e))$ $\forall e \in E$

- `Let` G_λ `be such that` $w(e) = w_1(e) + \lambda w_2(e)$ $\forall e \in E$
- `Upon the arrival of a new request` r
  - `serve` r `moving the servers in` G `as algorithm` A `in` G_λ

Figure 1: The algorithm $COMB(A, \lambda)$.

Lemma 3.1 *Given any c-competitive algorithm A for the (single-criterion) k-server problem and any real λ such that $1/W \leq \lambda \leq W$, $COMB(A, \lambda)$ is $(c(W\lambda + 1), c(\frac{W}{\lambda} + 1))$-competitive versus any selection function f.*

Proof. Given any sequence of requests σ, let $opt(\lambda, \sigma)$ be the optimal offline cost to serve σ on G_λ. Clearly, for any selection function f, $opt(\lambda, \sigma) \leq opt_1(f, \sigma) + \lambda opt_2(f, \sigma)$. The cost incurred by A on G_λ for σ can be written as

$$cost(A, \sigma) = cost_1(COMB(A, \lambda), \sigma) + \lambda cost_2(COMB(A, \lambda), \sigma) \leq$$

$$c \cdot opt(\lambda, \sigma) + \alpha \leq c(opt_1(f, \sigma) + \lambda opt_2(f, \sigma)) + \alpha,$$

for a suitable constant α.

Since $opt_1(f, \sigma) \leq W opt_2(f, \sigma)$ and $opt_2(f, \sigma) \leq W opt_1(f, \sigma)$,

$$cost_1(COMB(A, \lambda), \sigma) \leq c(opt_1(f, \sigma) + \lambda opt_2(f, \sigma)) + \alpha \leq$$

$$c(W\lambda + 1) opt_1(f, \sigma) + \alpha$$

and

$$cost_2(COMB(A, \lambda), \sigma) \leq c\left(\frac{opt_1(f, \sigma)}{\lambda} + opt_2(f, \sigma)\right) + \frac{\alpha}{\lambda} \leq$$

$$c\left(\frac{W}{\lambda}+1\right) opt_2(f,\sigma) + W\alpha,$$

hence the thesis. □

Let $U(A,f)$ be the algorithm obtained by executing $COMB(A,\lambda)$ with the value of λ such that $opt_1(f,\sigma) = \lambda opt_2(f,\sigma)$. Then, analogously to the proof of the above lemma, it results $cost_1(U(A,f),\sigma) \le c(opt_1(f,\sigma) + \lambda opt_2(f,\sigma)) + \alpha = 2c \cdot opt_1(f,\sigma) + \alpha$ and $cost_2(U(A,f),\sigma) \le c(\frac{opt_1(f,\sigma)}{\lambda} + opt_2(f,\sigma)) + \frac{\alpha}{\lambda} \le 2c \cdot opt_2(f,\sigma) + W\alpha$, since $\lambda \ge 1/W$. Unfortunately, such a λ cannot be determined in advance and therefore the above algorithm $U(A,f)$ is not online. Our method to get an online version of $U(A,f)$ is based on the observation that working with an approximation of the real λ gives competitive ratios close to the ones yielded by λ and that λ can be learned online during the processing of the sequence σ. Our algorithm at each step maintains the invariant that the used λ is a 2-approximation of the correct value of the currently processed prefix σ' of σ, that is $\frac{1}{2}\frac{opt_1(f,\sigma')}{opt_2(f,\sigma')} \le \lambda \le 2\frac{opt_1(f,\sigma')}{opt_2(f,\sigma')}$. More precisely, $U(A,f)$ performs the steps described in Figure 2.

`INPUT: algorithm` A `for the` k`-server problem, selection function` f `and graph` $G=(V,E)$ `with edge weights` $(w_1(e), w_2(e))$ $\forall e \in E$

- `Let` $\lambda = 1$
- `Upon the arrival of a new request` r
  - `let` σ' `be the current sequence of requests (`r `included)`
  - `if` $\lambda \le \frac{1}{2}\frac{opt_1(f,\sigma')}{opt_2(f,\sigma')}$ `or` $\lambda \ge 2\frac{opt_1(f,\sigma')}{opt_2(f,\sigma')}$ `then let` $\lambda = \frac{opt_1(f,\sigma')}{opt_2(f,\sigma')}$
  - `serve` r `moving the servers in` G `as algorithm` A `in` G_λ

Figure 2: The algorithm $U(A,f)$.

By construction, the sequence σ can be partitioned in phases during which λ is not modified by the algorithm. Let λ_i be the value of λ used during the i-th phase. We call a 1-phase (resp. 2-phase) a phase after which λ doubles (resp. halves) its value, i.e., $\lambda_{i+1} \ge 2\lambda_i$ (resp. $\lambda_{i+1} \le \lambda_i/2$). Moreover, let $\sigma[i-1,i]$ be the subsequence of σ corresponding to the requests of the i-th phase and $\sigma[i] = \sigma[0,i]$ the prefix of σ constituted by all the requests until the i-th phase included. Since f is monotone, if the i-th phase is a 1-phase (resp. 2-phase) opt_1 (resp. opt_2) during the phase has at least to double its value, that is $opt_1(f,\sigma[i]) \ge 2opt_1(f,\sigma[i-1])$ (resp. $opt_2(f,\sigma[i]) \ge 2opt_2(f,\sigma[i-1])$). Notice that the strategy selected by f in $\mathcal{O}(\sigma[i])$ in general is not an extension of the one in $\mathcal{O}(\sigma[i-1])$.

The following lemma is an easy consequence of the fact that the real value of λ is always comprised between $1/W$ and W.

Lemma 3.2 *Let h_1 and h_2 be the number of 1- and 2-phases of σ, respectively. Then, $|h_1 - h_2| \le \log W$.*

The costs $cost_1(U(A,f),\sigma[i-1,i])$ and $cost_2(U(A,f),\sigma[i-1,i])$ paid by the algorithm $U(A,f)$ during the i-th phase, from now on denoted simply as $cost_1(U,\sigma[i-1,i])$ and $cost_2(U,\sigma[i-1,i])$, can be suitably bounded.

Lemma 3.3 *During the i-th phase, $cost_1(U,\sigma[i-1,i]) \leq 3c \cdot opt_1(f,\sigma[i]) + \alpha_1$ and $cost_2(U,\sigma[i-1,i]) \leq 3c \cdot opt_2(f,\sigma[i]) + \alpha_2$, where α_1 and α_2 are suitable constants.*

Proof. Similarly as in Lemma 3.1, the cost incurred by A on G_{λ_i} for $\sigma[i-1,i]$ can be written as

$$cost(A,\sigma[i-1,i]) = cost_1(U,\sigma[i-1,i]) + \lambda_i cost_2(U,\sigma[i-1,i]) \leq$$

$$c \cdot opt(\lambda_i,\sigma[i-1,i]) + \alpha \leq c \cdot opt(\lambda_i,\sigma[i]) + \alpha \leq$$

$$c(opt_1(f,\sigma[i]) + \lambda_i opt_2(f,\sigma[i])) + \alpha,$$

where α is a suitable constant.

Since $\frac{1}{2}\frac{opt_1(f,\sigma[i])}{opt_2(f,\sigma[i])} \leq \lambda_i \leq 2\frac{opt_1(f,\sigma[i])}{opt_2(f,\sigma[i])}$, $cost_1(U,\sigma[i-1,i]) \leq 3c \cdot opt_1(f,\sigma[i]) + \alpha$ and $cost_2(U,\sigma[i-1,i]) \leq 3c \cdot opt_2(f,\sigma[i]) + \alpha/\lambda_i$. The thesis follows by observing that $1/W \leq \lambda_i \leq W$ and thus $\frac{\alpha}{\lambda_i}$ is bounded by a constant. □

Summing up the costs paid by the algorithm over all the phases, we get the following theorem.

Theorem 3.4 *The algorithm $U(A,f)$ defined above is $((3\log W + 13)c, (3\log W + 13)c)$-competitive for the bicriteria k-server problem versus any monotone selection function f.*

Proof. Let h_1 and h_2 be the number of 1-phases and 2-phases in the final sequence σ. Then σ has $h_1 + h_2$ phases plus a last possibly incomplete phase.

The cost paid by the algorithm on the first cost measure is

$$cost_1(U,\sigma) = \sum_{i=1}^{h_1+h_2+1} cost_1(U,\sigma[i-1,i]) \leq \sum_{i=1}^{h_1+h_2+1} (3c \cdot opt_1(f,\sigma[i]) + \alpha_1)\,.$$

At each 1-phase i opt_1 at least doubles its cost, i.e. $opt_1(f,\sigma[i+1]) \geq 2opt_1(f,\sigma[i])$. Therefore, informally speaking, if σ were composed only of 1-phases then the above summation would follow a geometric behavior and $cost_1(U,\sigma)$ would be roughly twice the last term $(3c \cdot opt_1(f,\sigma) + \alpha_1)$. Unfortunately, such a doubling of opt_1 does not hold for the 2-phases, and the worst case for $cost_1$ occurs when the last phases are all 2-phases. By Lemma 3.2, $|h_1 - h_2| \leq \log W$ and there are at most $\log W$ final 2-phases plus the incomplete last one. Before the last $\log W$ 2-phases the worst case is completed alternating 1- and 2-phases. Therefore,

$$cost_1(U,\sigma) \leq \sum_{i=1}^{h_1+h_2+1} (3c \cdot opt_1(f,\sigma[i]) + \alpha_1) \leq (\log W + 4)\,(3c \cdot opt_1(f,\sigma)) + \sum_{i=1}^{h_1+h_2+1} \alpha_1.$$

Since $h_1 \leq \log opt_1(f,\sigma)$ and $h_2 \leq h_1 + \log W \leq \log opt_1(f,\sigma) + \log W$,

$$cost_1(U,\sigma) \leq \left((\log W+4)+\alpha_1\frac{2(\log opt_1(f,\sigma))+(\log W)+1}{3c\cdot opt_1(f,\sigma)}\right)3c\cdot opt_1(f,\sigma) <$$

$$(3\log W+13)c\cdot opt_1(f,\sigma),$$

for $opt_1(f,\sigma)$ suitably large.

A symmetric argument holds for $cost_2(U,\sigma)$, thus proving the theorem. □

As a corollary of the above theorem and of the result in [9] with $c = 2k-1$, there exists a $((6\log W+26)k, (6\log W+26)k)$-competitive algorithm for the bicriteria k-server problem versus monotone selections. Since a trivial lower bound of k for both ratios comes directly from the single-criterion k-server by considering identical edge weights $w_1(e) = w_2(e)$ for each $e \in E$, our algorithm looses only an order of $\log W$ multiplicative factor.

We conclude this section with an example showing that the analysis of Theorem 3.4 is asymptotically tight. Even if we don't prove it explicitly, it is also possible to check that better approximations of λ might only decrease the competitive ratios of a small multiplicative constant and cannot get rid of the $\log W$ factor.

Example 3.5 Consider the bicriteria 1-server problem and let A be the trivial 1-competitive algorithm for the single-criterion formulation that always routes the server on shortest paths.

Given a suitable large integer h such that h is a power of 2, consider the graph $G = (V,E)$ depicted in Figure 3 with $n = h - \log h$ nodes $v_1,\ldots,v_n$ and such that v_1 and v_2 are connected by an edge of weights $(2^h,2^h)$, while for each i, $2 \leq i < n$, v_i and v_{i+1} are connected by two edges of weights $(2^{h-i+2},1)$ and $(1,2^h)$, respectively. Therefore, the maximum edge ratio is $W = 2^h$.

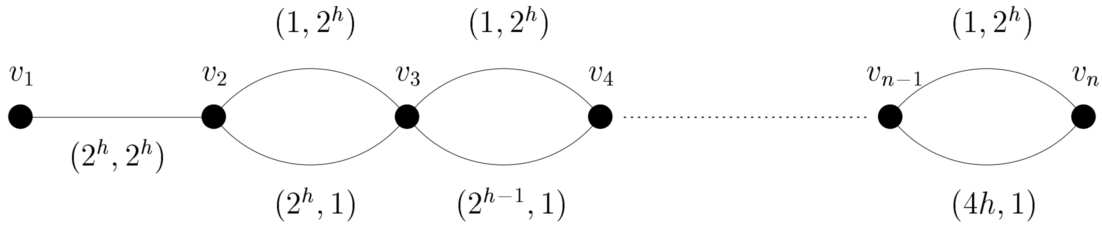

Figure 3: The tight instance for $U(A,f)$.

Assume that initially the server is in v_1 and let the sequence of requests $\sigma = \sigma_1\sigma_2\cdots\sigma_{n-1}$, where each σ_i, $1 \leq i < n$, is a subsequence of alternate requests between nodes v_{i-1} and v_i, starting with v_i and terminating with v_i. Every σ_i has a length l_i to be defined below.

The selection function f we compare our algorithm with is the one that minimizes $opt_1(f,\sigma)$, or analogously minimizes $\max(opt_1(f,\sigma), \beta opt_2(f,\sigma))$ or $|opt_1(f,\sigma) - \beta opt_2(f,\sigma)|$ with $\beta = \frac{1}{W}$. This function coincides with the selections f_1, f_2 and f_3 introduced in Example 2.7, that can be shown to be strictly monotone when $k = 1$. By definition, $f(\sigma)$ always uses edges of weights

$(1, 2^h)$, except during the first subsequence σ_1, in which the only available edge has weights $(2^h, 2^h)$.

The lengths of the subsequences of σ are such that l_1 is an arbitrarily large odd integer and at the end of every σ_i, $2 \leq i < n$, $\frac{opt_1(f,\sigma_1\cdots\sigma_i)}{opt_2(f,\sigma_1\cdots\sigma_i)} = \frac{1}{2^{i-1}}$. Clearly, this construction is correct if each l_i is an odd integer and is such that $\frac{opt_1(f,\sigma_1\cdots\sigma_i)}{opt_2(f,\sigma_1\cdots\sigma_i)}$ is exactly equal to $\frac{1}{2^{i-1}}$. However, this properties can be imposed by a slightly more complicated definition of G without substantially affecting the calculations below.

The lengths of the subsequences σ_i can be easily calculated and expressed as function of l_1. In fact, since at the end of each σ_{i-1} with $i \geq 2$ we have $\frac{opt_1(f,\sigma_1\cdots\sigma_{i-1})}{opt_2(f,\sigma_1\cdots\sigma_{i-1})} = \frac{l_1 2^h + l_2 + \cdots + l_{i-1}}{(l_1+l_2+\cdots+l_{i-1})2^h} = \frac{1}{2^{i-2}}$ and at the end of σ_i $\frac{opt_1(f,\sigma_1\cdots\sigma_i)}{opt_2(f,\sigma_1\cdots\sigma_i)} = \frac{l_1 2^h + l_2 + \cdots + l_i}{(l_1+l_2+\cdots+l_i)2^h} = \frac{1}{2^{i-1}}$, a trivial computation shows that $l_i = (1 + \frac{1}{2^{h-i}-1})(l_1 + \cdots + l_{i-1})$.

Since $l_i > l_1 + \cdots + l_{i-1}$ for $i \geq 2$, $l_i > 2^{i-2} l_1$. Moreover, as $i < n = h - \log h$, $l_i < (1 + \frac{1}{2^{\log h}-1})(l_1 + \cdots + l_{i-1}) = (1 + \frac{1}{h-1})(l_1 + \cdots + l_{i-1})$. Therefore for each i, $2 \leq i < n$, $l_1 + \cdots + l_i < (2 + \frac{1}{h-1})(l_1 + \cdots + l_{i-1}) < (2 + \frac{1}{h-1})^2 (l_1 + \cdots + l_{i-2}) < \ldots < (2 + \frac{1}{h-1})^{i-1} l_1 < 2^{i-1}(1 + \frac{1}{h-1})^{i-1} l_1 < 2^{i-1} e^{\frac{i-1}{h-1}} l_1$, as $1 + x < e^x$ for x converging to 0, where e is the natural number.

By the above inequalities,

$$opt_1(f,\sigma) = 2^h l_1 + l_2 + l_3 + \ldots + l_{n-1} < 2^h l_1 + 2^{n-1} e^{\frac{n-1}{h-1}} l_1 =$$
$$(2^h + 2^{h-\log h-1} e^{\frac{h-\log h-1}{h-1}}) l_1 = 2^h (1 + O(1/h)) l_1.$$

The algorithm $U(A,f)$, starting with $\lambda = 1$, while serving σ updates the value of λ halving it at the end of each subsequence σ_i with $i \geq 2$. Therefore, during every σ_i it always uses edges of weights $(2^{h-i+2}, 1)$, except during the first subsequence σ_1 in which the only available edge has weights $(2^h, 2^h)$. Thus,

$$cost_1(U(A,f),\sigma) = 2^h l_1 + \sum_{i=2}^{n-1} 2^{h-i+2} l_i > 2^h l_1 + \sum_{i=2}^{n-1} (2^{i-2} 2^{h-i+2} l_1) =$$
$$2^h n l_1 = 2^h (h - \log h) l_1.$$

Hence, for h sufficiently large, $\frac{cost_1(U(A,f),\sigma)}{opt_1(f,\sigma)} \approx h - \log h = \log W - \log\log W$.

4 Extension to (Δ_1, Δ_2)-monotone selections

In this section we briefly show how to extend our algorithm $U(A,f)$ to the class of the (Δ_1, Δ_2)-monotone selection functions for any $\Delta_1, \Delta_2 \leq 1$.

Let us define for each sequence of requests σ, $maxopt_1(f,\sigma) = \max_{\sigma' \, prefix \, of \, \sigma} opt_1(f,\sigma')$ and $maxopt_2(f,\sigma) = \max_{\sigma' \, prefix \, of \, \sigma} opt_2(f,\sigma')$.

Clearly the $maxopt_1$ and $maxopt_2$ functions are monotone in the sense that both cannot decrease as the sequence of requests grows. Substituting in $U(A,f)$ opt_1 and opt_2 with $maxopt_1$ and $maxopt_2$, we obtain a new algorithm with the same competitive ratios with respect to

$maxopt_1$ and $maxopt_2$. Since by definition of (Δ_1, Δ_2)-monotone function for every σ it results $opt_1(f,\sigma) + \alpha_1 \geq \Delta_1 maxopt_1(f,\sigma)$ and $opt_2(f,\sigma) + \alpha_2 \geq \Delta_2 maxopt_2(f,\sigma)$ for two suitable constants α_1 and α_2, the following theorem can be easily proven. We don't give a formal proof here, as a more general statement will be proven in Theorem 6.2.

Theorem 4.1 *$U(A,f)$ is $(\frac{(3\log W+13)c}{\Delta_1}, \frac{(3\log W+13)c}{\Delta_2})$-competitive for the bicriteria k-server problem versus any (Δ_1, Δ_2)-monotone selection function f with $\Delta_1, \Delta_2 \leq 1$.*

Again, using the result of [9], there exists a $(\frac{(6logW+26)k}{\Delta_1}, \frac{(6\log W+26)k}{\Delta_2})$-competitive algorithm. Moreover, as shown in the following theorem, also in this case our algorithm is only an order of $\log W$ multiplicative factor away from the lower bound.

Theorem 4.2 *For any Δ_1, Δ_2 such that $\frac{1}{W^2} \leq \Delta_1\Delta_2 \leq 1$, there exist instances of the bicriteria k-server problem with (Δ_1, Δ_2)-monotone selection functions for which no algorithm can be (c_1, c_2)-competitive with $c_1\Delta_1 + c_2\Delta_2 < k$.*

Proof. Consider a complete multigraph with $k+1$ nodes, where between each pair of nodes there are two arcs: one with weights $(\frac{1}{\Delta_1}, \frac{1}{W\Delta_1})$ and the other one with weights $(1, \frac{1}{W\Delta_1\Delta_2})$. Since by hypothesis $\Delta_1\Delta_2 \geq \frac{1}{W^2}$, the maximum edge ratio is at most W.

Given any (c_1, c_2)-competitive algorithm A, the strategy used by the adversary to construct the input sequence σ is to request at each step the node that is not occupied by one of the servers of A. By construction, for any σ all the nondominated strategies in $\mathcal{O}(\sigma)$ move the servers an equal number of times, as otherwise from one with the lowest number of moves it would be possible to construct strategies that dominate the others having more moves. Let us denote as $m(\sigma)$ such a number of moves. Clearly, $m(\sigma) \leq \lceil\frac{|\sigma|}{k}\rceil$.

The selection function f we compare the online algorithm with is such that if $m(\sigma)$ is even, then $f(\sigma)$ uses always the edges with weights $(\frac{1}{\Delta_1}, \frac{1}{W\Delta_1})$, i.e., $(opt_1(f,\sigma), opt_2(f,\sigma)) = (\frac{m(\sigma)}{\Delta_1}, \frac{m(\sigma)}{W\Delta_1})$, otherwise $f(\sigma)$ it uses only the edges with weights $(1, \frac{1}{W\Delta_1\Delta_2})$, that is $(opt_1(f,\sigma), opt_2(f,\sigma)) = (m(\sigma), \frac{m(\sigma)}{W\Delta_1\Delta_2})$.

Clearly, f is (Δ_1, Δ_2)-monotone. In fact, for any σ and prefix σ' of σ, $opt_1(f,\sigma') \leq \frac{m(\sigma')}{\Delta_1} \leq \frac{m(\sigma)}{\Delta_1}$ and thus $opt_1(f,\sigma) \geq m(\sigma) \geq \Delta_1 opt_1(f,\sigma')$. Similarly, $opt_2(f,\sigma') \leq \frac{m(\sigma')}{W\Delta_1\Delta_2} \leq \frac{m(\sigma)}{W\Delta_1\Delta_2}$ and $opt_2(f,\sigma) \geq \frac{m(\sigma)}{W\Delta_1} \geq \Delta_2 opt_2(f,\sigma')$.

Let σ be an arbitrarily long sequence generated by the adversary such that $m(\sigma)$ is odd. Since A is (c_1, c_2)-competitive, there exists a constant α_1 such that $cost_1(A,\sigma) \leq c_1 m(\sigma) + \alpha_1 \leq c_1\frac{|\sigma|}{k} + \alpha_1 + 1$. This means that the number of times A uses edges of weights $(\frac{1}{\Delta_1}, \frac{1}{W\Delta_1})$ to serve σ is at most $\Delta_1 cost_1(A,\sigma) \leq \Delta_1(c_1\frac{|\sigma|}{k} + \alpha_1 + 1)$. Hence, since A must use at least $|\sigma| - c_1\Delta_1(\frac{|\sigma|}{k} - \alpha_1 - 1)$ edges of weights $(1, \frac{1}{W\Delta_1\Delta_2})$, we have that $cost_2(A,\sigma) \geq (|\sigma| - c_1\Delta_1(\frac{|\sigma|}{k} - \alpha_1 - 1))\frac{1}{W\Delta_1\Delta_2}$. Let σ' be the shortest subsequence of successive requests generated by the adversary such that $m(\sigma\sigma') = m(\sigma) + 1$ and thus $m(\sigma\sigma')$ is even. Then,

$$\frac{cost_2(A,\sigma\sigma')}{opt_2(f,\sigma\sigma')} \geq \frac{cost_2(A,\sigma)}{\frac{m(\sigma)+1}{W\Delta_1}} \geq \frac{(|\sigma| - \Delta_1(c_1\frac{|\sigma|}{k} - \alpha_1 - 1))\frac{1}{W\Delta_1\Delta_2}}{\frac{\frac{|\sigma|}{k}+1}{W\Delta_1}} =$$

$$\frac{(|\sigma| - \Delta_1(c_1\frac{|\sigma|}{k} - \alpha_1 - 1))\frac{1}{\Delta_2}}{\frac{|\sigma|}{k} + 1}.$$

Therefore, as a limit for $|\sigma| \to \infty$, $c_2 \geq \frac{k}{\Delta_2} - c_1\frac{\Delta_1}{\Delta_2}$, that is $c_1\Delta_1 + c_2\Delta_2 \geq k$. □

Notice that, if $c_1\Delta_1 + c_2\Delta_2 \geq k$, $c_1 \leq \frac{k}{2\Delta_1}$ implies $c_2 \geq \frac{k}{2\Delta_2}$. Then, if $\Delta_1\Delta_2 = \frac{1}{W^2}$, the monotonicity of f is so poor that the simple $COMB(A, \lambda)$ algorithm where A is the $(2k-1)$-competitive algorithm of [9] and $\lambda = \frac{1}{W\Delta_1}$ is asymptotically optimal. In fact, it has competitive ratios $c_1 = (2k-1)(W\lambda + 1) = O(\frac{k}{\Delta_1})$ and $c_2 = (2k-1)(W/\lambda + 1) = O(kW^2\Delta_1) = O(\frac{k}{\Delta_2})$.

Clearly, the optimality of $COMB(A, \lambda)$ holds also when $\Delta_1\Delta_2 < \frac{1}{W^2}$. In fact, informally speaking, as $\Delta_2 < \frac{1}{W^2\Delta_1}$, any (Δ_1, Δ_2)-monotone function can admit sequences on which it is $(\Delta_1, \frac{1}{W^2\Delta_1})$-monotone. Therefore, by substituting in the above proof Δ_2 with $\frac{1}{W^2\Delta_1}$, it results $c_1\Delta_1 + \frac{c_2}{W^2\Delta_1} \geq k$. Again $c_1 \leq \frac{k}{2\Delta_1}$ implies $c_2 \geq \frac{kW^2\Delta_1}{2}$, so that $COMB(A, \lambda)$ with $\lambda = \frac{1}{W\Delta_1}$ is asymptotically optimal. A symmetric argument holds fixing Δ_2 and taking $\lambda = W\Delta_2$.

Finally, similar considerations show that $COMB(A, \lambda)$ asymptotically improves upon $U(A, f)$ when $\Delta_1\Delta_2 = o(\frac{\log^2 W}{W^2})$, as it gets rid of the two $\log W$ multiplicative factors.

5 Merging properties of nondominated strategies

In this section we show some basic properties of the nondominated strategies for the bicriteria k-server problem that will be useful in the sequel. In particular, we prove some results related to their mixing or interleaving. In fact, the existence of suitable nondominated strategies can be inferred by the merge of other ones.

Given a strategy $f(\sigma)$ for serving a sequence of requests σ, let us denote as $C(f, \sigma) \subseteq V$ the final configuration of $f(\sigma)$, that is the subset of the k vertices in G with a server at the end of $f(\sigma)$. Moreover, for any two configurations C_1 and C_2, let $(d_1(C_1, C_2), d_2(C_1, C_2))$ be a nondominated pair of costs needed to move the k servers from C_1 to C_2. Clearly in every case $d_1(C_1, C_2) < (k-i)nw_{max}$ and $d_2(C_1, C_2) < (k-i)nw_{max}$, where $i = |C_1 \cap C_2|$.

Given a sequence of requests σ, a prefix σ' of σ and two (non necessarily nondominated) strategies $g_1(\sigma)$ and $g_2(\sigma)$ for σ, we define their merging $merge(g_1, g_2, \sigma')(\sigma)$ as the strategy whose restriction on the prefix σ' of σ coincides with $g_1^\sigma(\sigma')$, then changes the servers from the configuration $C(g_1^\sigma, \sigma')$ to $C(g_2^\sigma, \sigma')$, and finally it coincides with the restriction of $g_2(\sigma)$ on the remaining suffix of σ.

For what concerns the costs paid by $merge(g_1, g_2, \sigma')(\sigma)$, by the above definition,

$$cost_1(merge(g_1, g_2, \sigma')(\sigma)) < cost_1(g_1^\sigma, \sigma') + (cost_1(g_2, \sigma) - cost_1(g_2^\sigma, \sigma')) + (k-1)nw_{max}$$

and

$$cost_2(merge(g_1, g_2, \sigma')(\sigma)) < cost_2(g_1^\sigma, \sigma') + (cost_2(g_2, \sigma) - cost_2(g_2^\sigma, \sigma')) + (k-1)nw_{max},$$

as $|C(g_1^\sigma, \sigma') \cap C(g_2^\sigma, \sigma')| \geq 1$.

Informally, $merge(g_1, g_2, \sigma')(\sigma)$ corresponds to the concatenation of the restriction of $g_1(\sigma)$ to σ' and the restriction of $g_2(\sigma)$ to the remaining suffix of σ. The intermediate change of

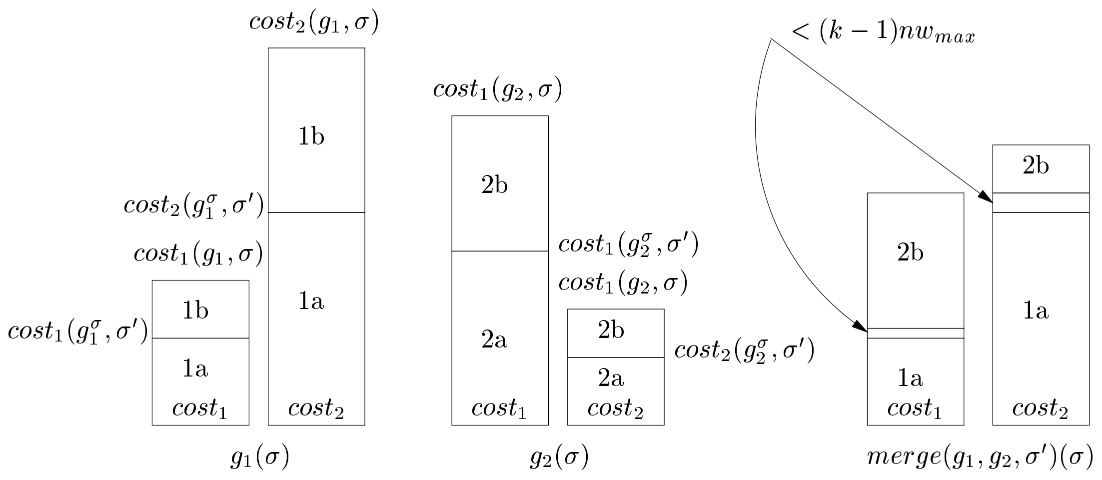

Figure 4: The merging of two strategies $g_1(\sigma)$ and $g_2(\sigma)$.

configuration is necessary since the final configuration $C(g_1^\sigma,\sigma')$ of $g_1^\sigma(\sigma')$ might be different from the initial configuration $C(g_2^\sigma,\sigma')$ of the restriction of $g_2(\sigma)$ on the remaining suffix of σ. Clearly, $merge(g_1,g_1,\sigma')(\sigma)=g_1(\sigma)$.

An example of merging is depicted in Figure 4, where the costs associated to each strategy are represented by a pair of vertical bars.

The following lemma is useful for obtaining merging strategies that try to average the costs paid on the two measures.

Lemma 5.1 *Given an input sequence σ and two (not necessarily nondominated) strategies $g_1(\sigma)$ and $g_2(\sigma)$, there exists a strategy $f(\sigma)$ such that $cost_1(f,\sigma) < \frac{cost_1(g_1,\sigma)+cost_1(g_2,\sigma)}{2} + knw_{max}$ and $cost_2(f,\sigma) < \frac{cost_2(g_1,\sigma)+cost_2(g_2,\sigma)}{2} + knw_{max}$.*

Proof. Since any request costs less than nw_{max} for both the two cost measures, given two prefixes σ_1 and σ_2 of σ with $0 \le |\sigma_1| = |\sigma_2|-1 \le m$, $|cost_1(g_1^\sigma,\sigma_2)+(cost_1(g_2,\sigma)-cost_1(g_2^\sigma,\sigma_2)-(cost_1(g_1^\sigma,\sigma_1)+(cost_1(g_2,\sigma)-cost_1(g_2^\sigma,\sigma_1))| < nw_{max}$. Therefore, since for any prefix σ' of σ if $|\sigma'|=m$ $cost_1(g_1^\sigma,\sigma')+(cost_1(g_2,\sigma)-cost_1(g_2^\sigma,\sigma')) = cost_1(g_1,\sigma)$ and if $|\sigma'|=0$ $cost_1(g_1^\sigma,\sigma')+(cost_1(g_2,\sigma)-cost_1(g_2^\sigma,\sigma')) = cost_1(g_2,\sigma)$, there must exist σ' with $0\le|\sigma'|\le m$ and $\frac{cost_1(g_1,\sigma)+cost_1(g_2,\sigma)}{2} - nw_{max} < cost_1(g_1^\sigma,\sigma')+(cost_1(g_2,\sigma)-cost_1(g_2^\sigma,\sigma')) < \frac{cost_1(g_1,\sigma)+cost_1(g_2,\sigma)}{2} + nw_{max}$.

Then, since $cost_1(g_1^\sigma,\sigma') + (cost_1(g_2,\sigma) - cost_1(g_2^\sigma,\sigma')) + cost_1(g_2^\sigma,\sigma') + (cost_1(g_1,\sigma) - cost_1(g_1^\sigma,\sigma')) = cost_1(g_1,\sigma)+cost_1(g_2,\sigma)$ and $cost_2(g_1^\sigma,\sigma') + (cost_2(g_2,\sigma) - cost_2(g_2^\sigma,\sigma')) + cost_2(g_2^\sigma,\sigma') + (cost_2(g_1,\sigma) - cost_2(g_1^\sigma,\sigma')) = cost_2(g_1,\sigma)+cost_2(g_2,\sigma)$, also $\frac{cost_1(g_1,\sigma)+cost_1(g_2,\sigma)}{2} - nw_{max} < cost_1(g_2^\sigma,\sigma') + (cost_1(g_1,\sigma) - cost_1(g_1^\sigma,\sigma')) < \frac{cost_1(g_1,\sigma)+cost_1(g_2,\sigma)}{2} + nw_{max}$, and $cost_2(g_1^\sigma,\sigma')+(cost_2(g_2,\sigma)-cost_2(g_2^\sigma,\sigma')) \le$

$\frac{cost_2(g_1,\sigma)+cost_2(g_2,\sigma)}{2}$ or $cost_2(g_2^\sigma,\sigma') + (cost_2(g_1,\sigma) - cost_2(g_1^\sigma,\sigma')) \leq \frac{cost_2(g_1,\sigma)+cost_2(g_2,\sigma)}{2}$.

If $cost_2(g_1^\sigma,\sigma') + (cost_2(g_2,\sigma) - cost_2(g_2^\sigma,\sigma')) \leq \frac{cost_2(g_1,\sigma)+cost_2(g_2,\sigma)}{2}$ then let $f(\sigma) = merge(g_1,g_2,\sigma')(\sigma)$, otherwise $f(\sigma) = merge(g_2,g_1,\sigma')(\sigma)$.

In every case, $cost_1(f,\sigma) < \frac{cost_1(g_1,\sigma)+cost_1(g_2,\sigma)}{2} + nw_{max} + (k-1)nw_{max} = \frac{cost_1(g_1,\sigma)+cost_1(g_2,\sigma)}{2} + knw_{max}$ and $cost_2(f,\sigma) < \frac{cost_2(g_1,\sigma)+cost_2(g_2,\sigma)}{2} + (k-1)nw_{max}$, hence the lemma. □

Given an input sequence σ and two nondominated strategies $g_1(\sigma)$ and $g_2(\sigma)$ in $\mathcal{O}(\sigma)$ such that $opt_1(g_1,\sigma) > opt_1(g_2,\sigma)$, we say that $g_1(\sigma)$ and $g_2(\sigma)$ are adjacent if no strategy $f(\sigma) \in \mathcal{O}(\sigma)$ is such that $opt_1(g_2,\sigma) < opt_1(f,\sigma) < opt_1(g_1,\sigma)$ (or analogously $opt_2(g_1,\sigma) < opt_2(f,\sigma) < opt_2(g_2,\sigma)$).

The following lemma relates the costs paid by adjacent strategies.

Lemma 5.2 *Given an input sequence σ and two nondominated adjacent strategies $g_1(\sigma) \in \mathcal{O}(\sigma)$ and $g_2(\sigma) \in \mathcal{O}(\sigma)$, $|opt_1(g_1,\sigma) - opt_1(g_2,\sigma)| < 2knw_{max}$ or $|opt_2(g_1,\sigma) - opt_2(g_2,\sigma)| < 2knw_{max}$.*

Proof. Assume by contradiction that $|opt_1(g_1,\sigma) - opt_1(g_2,\sigma)| \geq 2knw_{max}$ and $|opt_2(g_1,\sigma) - opt_2(g_2,\sigma)| \geq 2knw_{max}$, and without loss of generality let $opt_1(g_1,\sigma) > opt_1(g_2,\sigma)$.

By Lemma 5.1, there exists $f(\sigma)$ with $cost_1(f,\sigma) < \frac{opt_1(g_1,\sigma)+opt_1(g_2,\sigma)}{2} + knw_{max} \leq opt_1(g_1,\sigma)$ and $cost_2(f,\sigma) < \frac{opt_2(g_1,\sigma)+opt_2(g_2,\sigma)}{2} + knw_{max} \leq opt_2(g_2,\sigma)$.

Let $g(\sigma) = f(\sigma)$ if $f(\sigma) \in \mathcal{O}(\sigma)$, otherwise $g(\sigma) \in \mathcal{O}(\sigma)$ is a nondominated strategy dominating $f(\sigma)$. Then, since $g_1(\sigma) \in \mathcal{O}(\sigma)$ and $g_2(\sigma) \in \mathcal{O}(\sigma)$, it must be $opt_1(g_2,\sigma) < opt_1(g,\sigma) < opt_1(g_1,\sigma)$ and $opt_2(g_1,\sigma) < opt_2(g,\sigma) < opt_2(g_2,\sigma)$: a contradiction since $g_1(\sigma)$ and $g_2(\sigma)$ are adjacent. □

Starting from a nondominated strategy $f(\sigma) \in \mathcal{O}(\sigma)$, the following lemma states that it is not possible to have a small increase of the first cost measure with a big decrease of the second one and simultaneously a small increase of the second cost measure with a big decrease of the first one.

Lemma 5.3 *Given an input sequence σ, a nondominated strategy $g(\sigma) \in \mathcal{O}(\sigma)$ and a real number $\alpha > 0$, let $g_1(\sigma) \in \mathcal{O}(\sigma)$ and $g_2(\sigma) \in \mathcal{O}(\sigma)$ be two nondominated strategies with $opt_2(g_1,\sigma) < opt_2(g,\sigma) + \alpha$ and $opt_1(g_2,\sigma) < opt_1(g,\sigma) + \alpha$. Then, $opt_1(g_1,\sigma) \geq opt_1(g,\sigma) - \alpha - 2knw_{max}$ or $opt_2(g_2,\sigma) \geq opt_2(g,\sigma) - \alpha - 2knw_{max}$.*

Proof. Assume by contradiction that there exist $g_1(\sigma) \in \mathcal{O}(\sigma)$ and $g_2(\sigma) \in \mathcal{O}(\sigma)$ with $opt_2(g_1,\sigma) < opt_2(g,\sigma) + \alpha$, $opt_1(g_2,\sigma) < opt_1(g,\sigma) + \alpha$, $opt_1(g_1,\sigma) < opt_1(g,\sigma) - \alpha - 2knw_{max}$ and $opt_2(g_2,\sigma) < opt_2(g,\sigma) - \alpha - 2knw_{max}$. By Lemma 5.1, there exists $f(\sigma)$ with $cost_1(f,\sigma) < \frac{opt_1(g_1,\sigma)+opt_1(g_2,\sigma)}{2} + knw_{max} < opt_1(g,\sigma)$ and $cost_2(f,\sigma) < \frac{opt_2(g_1,\sigma)+opt_2(g_2,\sigma)}{2} + knw_{max} < opt_2(g,\sigma)$. Therefore, $f(\sigma)$ dominates $g(\sigma)$: a contradiction since $g(\sigma) \in \mathcal{O}(\sigma)$. □

6 Results for specific selections

In this section we give results related to the specific selections introduced in Example 2.7. As shown in the sequel, even if such functions have a poor monotonicity behavior, good competitive ratios can be accomplished by slight modifications of the algorithm $U(A, f)$.

Let f_1, f_2, f_3, f_4 and f_5 be defined as in Example 2.7. The reason why $U(A, f_i)$, $i = 1, \ldots, 5$, has high competitive ratios for f_i is due the fact that, when a prefix σ' of the input sequence σ is processed, $U(A, f_i)$ refers to $f_i(\sigma')$ and not the restriction $f_i^\sigma(\sigma')$ of $f_i(\sigma)$ on σ'. Indeed, by the poor monotonicity behavior, $f_i(\sigma')$ and $f_i^\sigma(\sigma')$ can be extremely different and the algorithm pays both its missing knowledge of the future requests and of the offline nondominated strategy it should compare with.

One way to circumvent this problem is to consider at each step a function g that, even if different from the given f_i, compares more favorably with respect to the restrictions of f_i. This concept is formalized in the following definition.

Definition 6.1 *Given $\Delta_1 \leq 1$, $\Delta_2 \leq 1$ and two selection functions f and g, g is a (Δ_1, Δ_2)-monotonization of f if and only if, for any sequence of requests σ and for every prefix σ' of σ, $opt_1(f, \sigma) + \alpha_1 \geq \Delta_1 \cdot opt_1(g, \sigma')$ and $opt_2(f, \sigma) + \alpha_2 \geq \Delta_2 \cdot opt_2(g, \sigma')$, where α_1 and α_2 are suitable constants.*

As we will see in the following subsections, the basic idea downstanding the construction of good monotonizations stands in amortizing high variations of one cost measure with low variations of the other one. In fact, as also shown in Section 5 (see Lemma 5.3), nondominated strategies have the basic property that it is not possible to have a small increase of the first cost measure with a big decrease of the other one and simultaneously a small increase of the second cost measure with a big decrease of the first one.

Theorem 6.2 *If g is a (Δ_1, Δ_2)-monotonization of f, then $U(A, g)$ is $(\frac{(3 \log W + 13)c}{\Delta_1}, \frac{(3 \log W + 13)c}{\Delta_2})$-competitive for the bicriteria k-server problem versus f, where c is the competitive ratio of A.*

Proof. Every sequence of requests σ can be partitioned in phases during which λ is not modified by the algorithm. Let λ_i be the value of λ used during the i-th phase, $\sigma[i-1, i]$ be the corresponding subsequence of requests and $\sigma[i] = \sigma[0, i]$ be the prefix of σ constituted by all the requests till the i-th phase included.

Similarly as in Lemma 3.3, during each phase the cost incurred by A on G_{λ_i} for $\sigma[i-1, i]$ can be written as

$$cost(A, \sigma[i-1, i]) = cost_1(U, \sigma[i-1, i]) + \lambda_i cost_2(U, \sigma[i-1, i]) \leq$$
$$c(opt_1(g, \sigma[i]) + \lambda_i opt_2(g, \sigma[i])) + \alpha \leq$$
$$c(maxopt_1(g, \sigma[i]) + \lambda_i maxopt_2(g, \sigma[i])) + \alpha,$$

where α is a suitable constant.

Since $\frac{1}{2} \frac{maxopt_1(g, \sigma[i])}{maxopt_2(g, \sigma[i])} \leq \lambda_i \leq 2 \frac{maxopt_1(g, \sigma[i])}{maxopt_2(g, \sigma[i])}$, $cost_1(U, \sigma[i-1, i]) \leq 3c \cdot maxopt_1(g, \sigma[i]) + \alpha$ and $cost_2(U, \sigma[i-1, i]) \leq 3c \cdot maxopt_2(g, \sigma[i]) + \alpha/\lambda_i$.

Summing up the costs paid by the algorithm over all the phases, repeating the steps in the proof of Theorem 3.4 with $maxopt_1$ instead of opt_1 it results

$$cost_1(U, \sigma) < (3 \log W + 13)c \cdot maxopt_1(g, \sigma),$$

for $maxopt_1(g, \sigma)$ suitably large.

A symmetric argument holds for $cost_2(U, \sigma)$, and the theorem follows by observing that $opt_1(f, \sigma) + \alpha_1 \geq \Delta_1 maxopt_1(g, \sigma)$ and $opt_2(f, \sigma) + \alpha_2 \geq \Delta_2 maxopt_2(g, \sigma)$. □

Notice that, since a (Δ_1, Δ_2)-monotone function f is a (Δ_1, Δ_2)-monotonization of itself, Theorem 4.1 can be derived as a simple corollary of the above theorem.

Motivated by Theorem 6.2, in the following we will focus on the determination of good monotonizations for the selection functions defined in Example 2.7. For the sake of clarity, we will consider each case separately.

6.1 The selection function f_1

The selection function f_1 chooses a nondominated solution in $\mathcal{O}(\sigma)$ that minimizes $opt_1(f_1, \sigma)$.

Clearly, f_1 is (Δ_1, Δ_2)-monotone with $\Delta_1 = 1$. Moreover, it is easy to see that it is $(1,1)$-monotone for $k = 1$. In this subsection we show that for $k \geq 2$ f_1 is poorly monotone with respect to the second cost measure, but it admits a $(1,1)$-monotonization.

Lemma 6.3 *f_1 is not $(1, \Delta_2)$-monotone for $\Delta_2 > \frac{2}{W+1}$.*

Proof. In order to prove the claim it suffices to show that there exist a graph G and two arbitrarily long sequences of requests σ' and σ such that σ' is a prefix of σ and as a limit for $|\sigma'| \to \infty$ (and thus $|\sigma| \to \infty$) $\frac{opt_2(f_1,\sigma)}{opt_2(f_1,\sigma')} = \frac{2}{W+1}$. We show that the claim holds even for $k = 2$ servers.

Consider a ring of four nodes v_1, v_2, v_3, v_4, where the two edges between v_1 and v_2 and between v_3 and v_4 have weights $(1, W)$, while the remaining two ones between v_2 and v_3 and between v_1 and v_4 have weights are $(1, 1)$ (see Figure 5).

Assume that the two servers are initially on v_1 and v_4, that is the initial configuration is $\{v_1, v_4\}$.

The sequence σ' is constituted by an arbitrarily large number of identical phases, each consisting of the subsequence of calls $\langle v_2, v_3, v_2, v_1, v_4, v_1 \rangle$. σ is obtained from σ' adding three last requests $\langle v_2, v_1, v_2 \rangle$.

The basic idea downstanding the above construction is that, if the initial configuration were $\{v_1, v_2\}$ instead of $\{v_1, v_4\}$, $f_1(\sigma')$ would serve each phase in such a way that one server always moves between v_1 and v_4 paying twice $(1, 1)$, while the other one between v_2 and v_3, again paying twice $(1, 1)$. This would give total costs per phase equal to $(4, 4)$. The last three requests of σ would be for free, since the two servers at the end of σ' would be on v_1 and v_2.

Starting from $\{v_1, v_4\}$, an initial change of configuration to $\{v_1, v_2\}$ obtained by moving the server of v_4 on v_2 has costs $(2, W + 1)$. Such an extra cost equal to 2 on $cost_1$ can be recovered during the processing of σ but not of σ'. In fact, a simple case analysis shows that the best that $f_1(\sigma')$ can do during a phase is moving only the server of v_1 to serve all the requests, thus

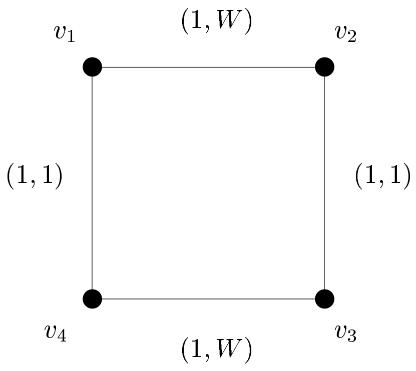

Figure 5: The ring of the upper bound on the monotonicity of f_1.

falling again in the final configuration $\{v_1, v_4\}$. Since such a server has to traverse twice an edge of weights $(1, W)$ and twice one of weights $(1, 1)$, the total cost per phase is $(4, 2W + 2)$.

Trying to extend $f_1(\sigma')$ to serve the last three calls of σ now requires an extra cost equal to $(2, W+1)$. This means that $f_1(\sigma)$ can initially pay $(2, W+1)$ to change the initial configuration from $\{v_1, v_4\}$ to $\{v_1, v_2\}$, and then recover this initial extra cost on $cost_1$ during the last three calls of σ, for which it does not incur any additional cost.

As a consequence, if p is the number of phases in σ, $(opt_1(f_1, \sigma'), opt_2(f_1, \sigma')) = (4p, (2W + 2)p)$ and $(opt_1(f_1, \sigma), opt_2(f_1, \sigma)) = (4p + 2, 4p + W + 1)$. Therefore, as a limit for $p \to \infty$ (and thus $|\sigma'| \to \infty$ and $|\sigma| \to \infty$), $\frac{opt_2(f_1,\sigma)}{opt_2(f_1,\sigma')} = \frac{2}{W+1}$.

$\square$

Given any sequence σ and prefix σ' of σ, consider the restriction $f_1^\sigma(\sigma')$ of $f_1(\sigma)$ on σ'. Then $f_1^\sigma(\sigma')$ minimizes $cost_1(f_1^\sigma, \sigma')$ among all the possible solutions for σ' terminating in the configuration $C(f_1^\sigma, \sigma')$, i.e., the one identified by the positions of the k vertices at the end of $f_1^\sigma(\sigma')$. In fact, assume by contradiction that there exists a strategy f' with final configuration $C(f_1^\sigma, \sigma')$ such that $cost_1(f', \sigma') < cost_1(f_1^\sigma, \sigma')$, and let f'' be the concatenation of $f'(\sigma')$ with the restriction of f_1 on the suffix $\sigma - \sigma'$. Then $cost_1(f'', \sigma) < cost_1(f_1, \sigma)$, that clearly contradicts the definition of f_1. Therefore, $cost_1(f_1^\sigma, \sigma') \leq opt_1(f_1, \sigma') + d_1(C(f_1, \sigma'), C(f_1^\sigma, \sigma')) < opt_1(f_1, \sigma') + knw_{max}$.

Let g be the selection function defined as follows. For each sequence σ, $g(\sigma)$ chooses a strategy in $\mathcal{O}(\sigma)$ which minimizes $opt_2(g, \sigma)$ (and thus maximizes $opt_1(g, \sigma)$) among all the strategies with $opt_1(g, \sigma) < opt_1(f_1, \sigma) + knw_{max}$ (see Figure 6). More precisely, g is such that $opt_1(g, \sigma) < opt_1(f_1, \sigma) + knw_{max}$ and no other strategy g' in $\mathcal{O}(\sigma)$ with $opt_1(g', \sigma) < opt_1(f_1, \sigma) + knw_{max}$ satisfies $opt_2(g', \sigma) < opt_2(g, \sigma)$.

Then, if σ' is a prefix of σ and f is a selection function such that $f(\sigma') \in \mathcal{O}(\sigma')$ coincides with $f_1^\sigma(\sigma')$ if $f_1^\sigma(\sigma') \in \mathcal{O}(\sigma')$ and dominates $f_1^\sigma(\sigma')$ otherwise, since $opt_1(f, \sigma') \leq cost_1(f_1^\sigma, \sigma') < opt_1(f_1, \sigma') + knw_{max}$, $opt_2(f_1, \sigma) \geq cost_2(f_1^\sigma, \sigma') \geq opt_2(f, \sigma') \geq opt_2(g, \sigma')$. Hence, the following lemma is proved.

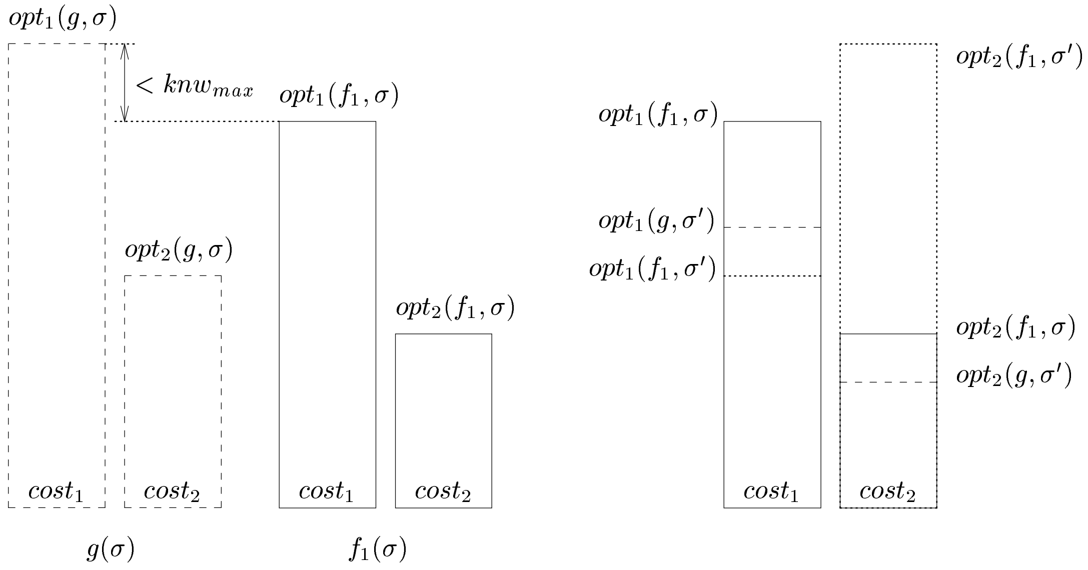

Figure 6: The $(1,1)$-monotonization g of f_1 and comparison between $f_1(\sigma)$ and $g(\sigma')$ for a prefix σ' of σ.

Lemma 6.4 *The selection function g defined above is a $(1,1)$-monotonization of f_1.*

It is now possible to see how g amortizes high variations of one measure with low variations of the other one. In fact, a constant additive cost with respect to $opt_1(f_1,\sigma)$ avoids sudden final decreases of $opt_2(f_1,\sigma)$ like in Lemma 6.3, thus improving the overall performance of the algorithm.

As a direct consequence of the previous lemma and Theorem 6.2, the following corollary holds.

Corollary 6.5 *$U(A,g)$ is $(O(c\log W),O(c\log W))$-competitive for the bicriteria k-server problem versus f_1, where c is the competitive ratio of A.*

6.2 The selection function f_2

Given a positive real number β, for any sequence σ the selection f_2 chooses a nondominated solution in $\mathcal{O}(\sigma)$ such that $\max(opt_1(f_2,\sigma), \beta opt_2(f_2,\sigma))$ is minimized.

It is not difficult to show that for $k=1$ f_2 is $(1,1)$-monotone. However, since f_2 coincides with f_1 for $\beta \leq \frac{1}{W}$, when $k \geq 2$ also f_2 in general is not (Δ_1,Δ_2)-monotone for $\Delta_1 > \frac{2}{W+1}$ or $\Delta_2 > \frac{2}{W+1}$. We now show that in every case f_2 admits a good monotonization.

The monotonization g of f_2 is defined as follows. Let $g_1(\sigma)$ be a strategy in $\mathcal{O}(\sigma)$ that minimizes $opt_1(g_1,\sigma)$ among all the strategies with $opt_2(g_1,\sigma) < opt_2(f_2,\sigma) + 4knw_{max}$, and let $g_2(\sigma)$ be a strategy similarly defined exchanging opt_1 and opt_2. Moreover, let $\alpha_1 = opt_1(f_2,\sigma) - opt_1(g_1,\sigma)$ and $\alpha_2 = opt_2(f_2,\sigma) - opt_2(g_2,\sigma)$. Then $g(\sigma) = g_1(\sigma)$ if $\alpha_1 \geq \alpha_2$, otherwise $g(\sigma) =$

$g_2(\sigma)$ (see Figure 7). Informally, $g(\sigma)$ chooses a nondominated strategy in $\mathcal{O}(\sigma)$ that with a limited increase (at most $4knw_{max}$) on one cost measure yields the highest decrease of the other measure.

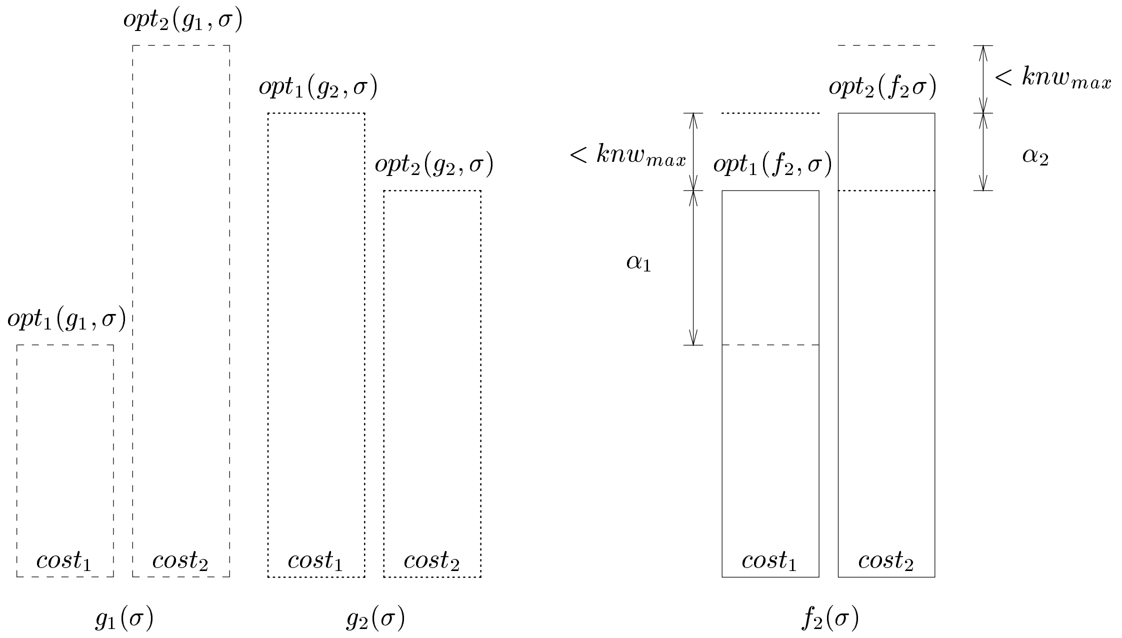

Figure 7: The $(1/2, 1/2)$-monotonization of f_2.

Lemma 6.6 *The selection function g is a $(1/2, 1/2)$-monotonization of f_2.*

Proof. We prove the claim for $\beta = 1$. The case $\beta \neq 1$ can be reduced to the one with $\beta = 1$ transforming the bicriteria instance of the problem in one where the first cost measure is the same and the second one is multiplied times β. The proof is then obtained from the one below simply by substituting w_{max} with $\max(w_{max}, \beta w_{max})$.

We show that $opt_1(f_2, \sigma) + 8knw_{max} \geq \frac{1}{2} opt_1(g, \sigma')$ and $opt_2(f_2, \sigma) + 8knw_{max} \geq \frac{1}{2} opt_2(g, \sigma')$ for every sequence of requests σ and prefix σ' of σ.

Without loss of generality let $opt_1(f_2, \sigma) \geq opt_1(f_2, \sigma')$ (the case $opt_2(f_2, \sigma) \geq opt_2(f_2, \sigma')$ is symmetric).

Since $opt_1(g, \sigma') < opt_1(f_2, \sigma') + 4knw_{max}$, $opt_1(f_2, \sigma) + 8knw_{max} > opt_1(f_2, \sigma') + 4knw_{max} > opt_1(g, \sigma')$.

Concerning the second cost measure, we distinguish among the following cases.

1) $opt_2(f_2, \sigma) \geq \frac{1}{2} opt_2(f_2, \sigma') - 6knw_{max}$:

   Since $opt_2(g, \sigma') < opt_2(f_2, \sigma') + 4knw_{max}$, $opt_2(f_2, \sigma) + 8knw_{max} \geq \frac{1}{2} opt_2(f_2, \sigma') + 2knw_{max} > \frac{1}{2} opt_2(g, \sigma')$.

2) $opt_2(f_2, \sigma) < \frac{1}{2}opt_2(f_2, \sigma') - 6knw_{max}$ and $cost_1(f_2^\sigma, \sigma') < opt_1(f_2, \sigma') + 4knw_{max}$:

Recalling the above definition of the selections g_1 and g_2 used to construct g, if f is a selection function such that $f(\sigma') \in \mathcal{O}(\sigma')$ coincides with $f_2^\sigma(\sigma')$ if $f_2^\sigma(\sigma') \in \mathcal{O}(\sigma')$, otherwise $f(\sigma')$ dominates $f_2^\sigma(\sigma')$, then $opt_1(g_2, \sigma') \geq opt_1(f, \sigma')$. As a consequence, $opt_2(f_2, \sigma) \geq opt_2(f, \sigma') \geq opt_2(g_2, \sigma')$ and thus $opt_2(f_2, \sigma) + 8knw_{max} > \frac{1}{2}opt_2(g_2, \sigma')$. The claim then follows by observing that in this case $g(\sigma') = g_2(\sigma')$. In fact, $opt_2(g_2, \sigma') \leq opt_2(f_2, \sigma) < \frac{1}{2}opt_2(f_2, \sigma') - 6knw_{max} < opt_2(f_2, \sigma') - 6knw_{max}$, and by applying Lemma 5.3 to $f_2(\sigma')$ with $\alpha = 4knw_{max}$, as $opt_2(g_1, \sigma') < opt_2(f_2, \sigma') + 4knw_{max}$ and $opt_1(g_2, \sigma') < opt_1(f_2, \sigma') + 4knw_{max}$, $opt_1(g_1, \sigma') > opt_1(f_2, \sigma') - 6knw_{max}$. Hence $\alpha_2 > \alpha_1$ and $g(\sigma) = g_2(\sigma)$.

3) $opt_2(f_2, \sigma) < \frac{1}{2}opt_2(f_2, \sigma') - 6knw_{max}$ and $cost_1(f_2^\sigma, \sigma') \geq opt_1(f_2, \sigma') + 4knw_{max}$:

By Lemma 5.1 applied to $f_2^\sigma(\sigma')$ and $f_2(\sigma')$, there exists a strategy $f(\sigma')$ such that $cost_1(f, \sigma') < \frac{cost_1(f_2^\sigma, \sigma') + opt_1(f_2, \sigma')}{2} + knw_{max}$ and $cost_2(f, \sigma') < \frac{cost_2(f_2^\sigma, \sigma') + opt_2(f_2, \sigma')}{2} + knw_{max}$.

Let $f_c(\sigma)$ the strategy that behaves like $f(\sigma')$ while processing σ', then changes the servers from the configuration $C(f, \sigma')$ to the configuration $C(f_2^\sigma, \sigma')$, and finally proceeds like $f_2(\sigma)$ on the remaining suffix of σ.

We now show that if case 3) holds, $\max(cost_1(f_c, \sigma), cost_2(f_c, \sigma)) < \max(opt_1(f_2, \sigma), opt_2(f_2, \sigma))$: a contradiction since by definition $f_2(\sigma)$ minimizes $\max(opt_1(f_2, \sigma), opt_2(f_2, \sigma))$.

By the definition of the strategy $f_c(\sigma)$,

$$cost_1(f_c, \sigma) < cost_1(f, \sigma') + (opt_1(f_2, \sigma) - cost_1(f_2^\sigma, \sigma')) + knw_{max} <$$
$$\frac{cost_1(f_2^\sigma, \sigma') + opt_1(f_2, \sigma')}{2} + knw_{max} + opt_1(f_2, \sigma) - cost_1(f_2^\sigma, \sigma') + knw_{max} \leq$$
$$cost_1(f_2^\sigma, \sigma') - 2knw_{max} + knw_{max} + opt_1(f_2, \sigma) - cost_1(f_2^\sigma, \sigma') + knw_{max} =$$
$$opt_1(f_2, \sigma),$$

as by the hypothesis of case 3) $cost_1(f_2^\sigma, \sigma') \geq opt_1(f_2, \sigma') + 4knw_{max}$.

Moreover, since $opt_2(f_2, \sigma) < \frac{1}{2}opt_2(f_2, \sigma') - 6knw_{max}$,

$$cost_2(f_c, \sigma) < cost_2(f, \sigma') + (opt_2(f_2, \sigma) - cost_2(f_2^\sigma, \sigma')) + knw_{max} <$$
$$\frac{cost_2(f_2^\sigma, \sigma') + opt_2(f_2, \sigma')}{2} + knw_{max} + opt_2(f_2, \sigma) - cost_2(f_2^\sigma, \sigma') + knw_{max} =$$
$$\frac{opt_2(f_2, \sigma') - cost_2(f_2^\sigma, \sigma')}{2} + opt_2(f_2, \sigma) + 2knw_{max} <$$
$$\frac{opt_2(f_2, \sigma') - cost_2(f_2^\sigma, \sigma')}{2} + \frac{1}{2}opt_2(f_2, \sigma') - 4knw_{max} < opt_2(f_2, \sigma').$$

Since $cost_2(f_2^\sigma, \sigma') \leq opt_2(f_2, \sigma) < \frac{1}{2} opt_2(f_2, \sigma') - 6knw_{max} < opt_2(f_2, \sigma')$, it must be $cost_1(f_2^\sigma, \sigma') \geq opt_2(f_2, \sigma')$, otherwise, by the two above inequalities bounding the costs of $f_c(\sigma)$, $\max(cost_1(f_2^\sigma, \sigma'), cost_2(f_2^\sigma, \sigma')) < opt_2(f_2, \sigma') \leq \max(opt_1(f_2, \sigma'), opt_2(f_2, \sigma'))$, thus contradicting the definition of $f_2(\sigma')$.

Therefore, as $opt_1(f_2, \sigma) \geq cost_1(f_2^\sigma, \sigma')$, $\max(cost_1(f_c, \sigma), cost_2(f_c, \sigma)) < \max(opt_1(f_2, \sigma), opt_2(f_2, \sigma')) = opt_1(f_2, \sigma) \leq \max(opt_1(f_2, \sigma), opt_2(f_2, \sigma))$: a contradiction, since $f_2(\sigma)$ minimizes $\max(opt_1(f_2, \sigma), opt_2(f_2, \sigma))$.

□

Again, by applying the previous lemma and Theorem 6.2, the following corollary holds.

Corollary 6.7 *$U(A, g)$ is $(O(c \log W), O(c \log W))$-competitive for the bicriteria k-server problem versus f_2, where c is the competitive ratio of A.*

6.3 The selection function f_3

Given a positive real number β, for any sequence σ the selection f_3 chooses a nondominated solution in $\mathcal{O}(\sigma)$ such that $opt_1(f_3, \sigma) \approx \beta opt_2(f_3, \sigma)$, i.e., minimizing $|opt_1(f_3, \sigma) - \beta opt_2(f_3, \sigma)|$.

Similarly as for f_2, it is not difficult to show that f_3 is $(1,1)$-monotone for $k = 1$. Again, since f_3 coincides with f_1 for $\beta \leq \frac{1}{W}$, when $k \geq 2$ also f_3 in general is not (Δ_1, Δ_2)-monotone for $\Delta_1 > \frac{2}{W+1}$ or $\Delta_2 > \frac{2}{W+1}$. However, as stated in the following lemma, also f_3 admits a good monotonization.

Lemma 6.8 *There exists a $(1/2, 1/2)$-monotonization of f_3.*

Proof. Again we prove the claim for $\beta = 1$. The case $\beta \neq 1$ can be reduced to the one with $\beta = 1$ by multiplying the second cost measure times β and by substituting w_{max} with $\max(w_{max}, \beta w_{max})$.

Given $\Delta_1 \leq 1$, $\Delta_2 \leq 1$, we show that every (Δ_1, Δ_2)-monotonization g of f_2 is also a (Δ_1, Δ_2)-monotonization of f_3. The lemma then follows from Lemma 6.6.

Let σ be a given sequence of requests and without loss of generality assume that $opt_1(f_2, \sigma) \geq opt_2(f_2, \sigma)$. Then $f_3(\sigma)$ either has costs coincident with $f_2(\sigma)$, or is a nondominated strategy adjacent to $f_2(\sigma)$ such that $opt_1(f_3, \sigma) < opt_2(f_3, \sigma)$.

In the first case, since g is a (Δ_1, Δ_2)-monotonization of f_2, for every prefix σ' of σ, $opt_1(f_3, \sigma) + \alpha_1 \geq \Delta_1 \cdot opt_1(g, \sigma')$ and $opt_2(f_3, \sigma) + \alpha_2 \geq \Delta_2 \cdot opt_2(g, \sigma')$, where α_1 and α_2 are suitable constants.

If $f_3(\sigma)$ is adjacent to $f_2(\sigma)$, since $opt_1(f_2, \sigma) > opt_1(f_3, \sigma)$ (otherwise $f_2(\sigma)$ would dominate $f_3(\sigma)$), by Lemma 5.2 $opt_1(f_2, \sigma) - opt_1(f_3, \sigma) < 2knw_{max}$ or $opt_2(f_3, \sigma) - opt_2(f_2, \sigma) < 2knw_{max}$.

If $opt_1(f_2, \sigma) - opt_1(f_3, \sigma) < 2knw_{max}$, for every prefix σ' of σ, $opt_1(f_3, \sigma) + \alpha_1 + 2knw_{max} > opt_1(f_2, \sigma) + \alpha_1 \geq \Delta_1 \cdot opt_1(g, \sigma')$ and $opt_2(f_3, \sigma) + \alpha_2 > opt_2(f_2, \sigma) + \alpha_2 \geq \Delta_2 \cdot opt_2(g, \sigma')$.

If $opt_2(f_3, \sigma) - opt_2(f_2, \sigma) < 2knw_{max}$, as $opt_2(f_3, \sigma) \geq opt_1(f_2, \sigma)$ by the definition of $f_2(\sigma)$ and $f_3(\sigma)$ minimizes $|opt_2(f_3, \sigma) - opt_1(f_3, \sigma)|$, $opt_1(f_2, \sigma) - opt_1(f_3, \sigma) \leq opt_2(f_3, \sigma) - opt_2(f_2, \sigma) < 2knw_{max}$, and we fall again in the previous case. □

Again, by applying the previous lemma and Theorem 6.2, the following corollary holds.

Corollary 6.9 *$U(A, g)$ is $(O(c \log W), O(c \log W))$-competitive for the bicriteria k-server problem versus f_2, where c is the competitive ratio of A.*

6.4 The selection function f_4

Given a positive real number β, for any sequence σ the selection f_4 chooses a nondominated solution in $\mathcal{O}(\sigma)$ that minimizes $opt_1(f_4, \sigma) + \beta opt_2(f_4, \sigma)$.

Lemma 6.10 *f_4 is not (Δ_1, Δ_2)-monotone for $\Delta_1 > \frac{\beta+W}{W(\beta W+1)}$ or $\Delta_2 > \frac{\beta W+1}{W(\beta+W)}$.*

Proof. We prove that the claim holds even for $k = 1$ servers. Let $\Delta_1 = \frac{\beta+W}{W(\beta W+1)}$ and $\Delta_2 = \frac{\beta W+1}{W(\beta+W)}$. Consider a graph consisting of two nodes u and v connected by two edges of weights $(\frac{1}{\Delta_1}, \frac{1}{W\Delta_1})$ and $(1, \frac{1}{W\Delta_1\Delta_2})$, respectively. As by hypothesis $\Delta_1\Delta_2 = \frac{1}{W^2}$, the maximum edge ratio is W.

Since by construction $\frac{1}{\Delta_1} + \beta\frac{1}{W\Delta_1} = 1 + \beta\frac{1}{W\Delta_1\Delta_2}$, any nondominated strategy $f(\sigma) \in \mathcal{O}(\sigma)$ for a sequence σ has the same value $opt_1(f, \sigma) + \beta opt_2(\sigma)$. Therefore, f_4 can be such that if $|\sigma|$ is even, then $f_4(\sigma)$ uses always the edge of weights $(\frac{1}{\Delta_1}, \frac{1}{W\Delta_1})$, otherwise it uses only the one of weights $(1, \frac{1}{W\Delta_1\Delta_2})$.

We now provide two arbitrarily long sequences of requests σ' and σ such that σ' is a prefix of σ and, as a limit for $|\sigma'| \to \infty$, $\frac{opt_1(f_1,\sigma)}{opt_1(f_1,\sigma')} = \Delta_1$.

Assuming that the single server initially is on v, σ is constituted by an arbitrarily long sequence of alternate requests of u and v, starting with u. Moreover, σ' is the prefix of σ of length $|\sigma| - 1$, i.e., σ without the last request.

Then, if $|\sigma|$ is odd,

$$\frac{opt_1(f_4, \sigma)}{opt_1(f_4, \sigma')} = \frac{|\sigma|}{\frac{|\sigma'|}{\frac{1}{\Delta_1}}} = \frac{\Delta_1|\sigma|}{|\sigma| - 1},$$

and as $|\sigma| \to \infty$ (and thus $|\sigma'| \to \infty$) $\frac{opt_1(f_4,\sigma)}{opt_1(f_4,\sigma')} = \Delta_1$.

Analogously, if $|\sigma|$ is even, $\frac{opt_2(f_4,\sigma)}{opt_2(f_4,\sigma')} = \Delta_2$ for $|\sigma'| \to \infty$, hence the lemma. □

Extending the arguments of Lemma 6.10, it is possible to show that f_4 is (Δ_1, Δ_2)-monotone with $\Delta_1 = \frac{\beta+W}{W(\beta W+1)}$ and $\Delta_2 = \frac{\beta W+1}{W(\beta+W)}$ for every $k \geq 1$. Moreover, a proof identical to that of Theorem 4.2 can be used to derive the following lemma.

Lemma 6.11 *Any (c_1, c_2)-competitive algorithm for the bicriteria k-server problem versus f_4 is such that $c_1\Delta_1 + c_2\Delta_2 \geq k$, where $\Delta_1 = \frac{\beta+W}{W(\beta W+1)}$ and $\Delta_2 = \frac{\beta W+1}{W(\beta+W)}$.*

Hence, f_4 is a worst case example in the family of the (Δ_1, Δ_2)-monotone functions with $\Delta_1 = \frac{\beta+W}{W(\beta W+1)}$ and $\Delta_2 = \frac{\beta W+1}{W(\beta+W)}$. As a consequence, by the same considerations after Theorem 4.2, the simple $COMB(A, \lambda)$ algorithm where A is the $(2k-1)$-competitive algorithm of [9] and $\lambda = \frac{1}{W\Delta_1}$ is asymptotically optimal versus f_4.

6.5 The selection function f_5

Given a positive real number β, f_5 chooses a nondominated solution in $\mathcal{O}(\sigma)$ such that $opt_1(f_5, \sigma)$ does not exceed a given budget $\beta \cdot m$ proportional to the length m of σ and $opt_2(f_5, \sigma)$ is minimum. If such strategy does not exist, then f_5 minimizes $opt_1(f_5, \sigma)$.

Lemma 6.12 *f_5 is not (Δ_1, Δ_2)-monotone for $\Delta_1 > \frac{2}{W+1}$ or $\Delta_2 > \frac{1}{W^2}$.*

Proof. Consider the construction obtained from the one of Lemma 6.3 depicted in Figure 5 by exchanging the two edge weightings of G and dividing both of them by a suitable large number such that, for every prefix σ' of the given sequence of requests σ, f_5 minimizes $opt_2(f_5, \sigma')$ without exceeding the budget. Then, $\Delta_1 > \frac{2}{W+1}$ follows by the same considerations of Lemma 6.3 applied on the first cost measure.

In order to bound Δ_2, we show that for any $\epsilon > 0$ there exist a graph G and two arbitrarily long sequences of requests σ' and σ such that σ' is a prefix of σ and $\frac{opt_2(f_5,\sigma)}{opt_2(f_5,\sigma')} \leq \frac{1}{W^2(1-\epsilon)}$. Let G consist of two nodes u and v connected by two edges with weights respectively $(\frac{\beta}{1-\epsilon}, \frac{\beta}{W(1-\epsilon)})$ and $(\beta, \beta W)$. Assuming there is only a single server initially on v, let σ be the concatenation of an arbitrarily long sequence σ' of alternate calls of the nodes u and v starting with u and terminating with v, plus a sequence of at least $|\sigma'|\frac{\epsilon}{1-\epsilon}$ "dummy" calls of v. Then, $opt_1(f_5, \sigma') = |\sigma'|\beta$ and $opt_2(f_5, \sigma') = |\sigma'|\beta W$, since in order not to exceed the budget it is possible to use only edges with weights $(\beta, \beta W)$. The final dummy calls of σ do not increase the costs paid by any strategy for σ after σ', since they are all from the node occupied by the server. However, they have the effect of lengthening the overall sequence σ, making available a total budget at least equal to $|\sigma|\beta \geq |\sigma'|\frac{\beta}{1-\epsilon}$. Therefore, since to serve σ it is possible to use edges of weights $(\frac{\beta}{1-\epsilon}, \frac{\beta}{W(1-\epsilon)})$, $opt_1(f_5, \sigma) = |\sigma'|\frac{\beta}{1-\epsilon} \leq |\sigma|\beta$ and $opt_2(f_5, \sigma') = |\sigma'|\frac{\beta}{W(1-\epsilon)}$. As a consequence, $\frac{opt_2(f_5,\sigma)}{opt_2(f_5,\sigma')} \leq \frac{1}{W^2(1-\epsilon)}$, and the claim derives by the arbitrariety of ϵ and of the lengths of the sequence σ and its prefix σ'. $\square$

A lower bound on the competitive ratios achievable by any algorithm versus f_5 is established in the following lemma.

Lemma 6.13 *Any (c_1, c_2)-competitive algorithm for the bicriteria k-server problem versus f_5 is such that $c_1 c_2 \geq \frac{(k-2)W^2}{2}$.*

Proof. We prove the claim for $\beta = 1$. The result for general β can be obtained by multiplying all the edge weights times β.

Given an odd number of servers k, we prove that $c_1 c_2 \geq \frac{(k-1)W^2}{2}$. The case k even can be derived by a slight modification of the construction below that forces one server to stay into an additional node and allows the movement of the remaining $k-1$ servers, thus yielding $c_1 c_2 \geq \frac{(k-2)W^2}{2}$.

Consider a complete graph $G = (V, E)$ with $V = \{u_1, v_1, u_2, v_2, \ldots, u_{\frac{k+1}{2}}, v_{\frac{k+1}{2}}\}$ (hence $|V| = k+1$). For any i, $1 \leq i \leq \frac{k+1}{2}$, we say that u_i and v_i form the i-th horizontal component, that is connected by a horizontal edge of weights $(1, W)$. All the other edges in E are called vertical and have weights (xW, x), where x is a fixed positive integer.

Given any (c_1, c_2)-competitive algorithm A, the adversary generates the sequence σ in such a way that each request of an arbitrarily long prefix σ' of σ is always to the node that is not occupied by one of the servers of A, while the remaining requests of σ are dummy requests, all corresponding to the last node of σ'. Such requests do not increase the costs paid by the adversary and by A after σ', but have the effect of lengthening the overall sequence making available a total budget equal to $|\sigma|$ instead of $|\sigma'|$. Hence, by allowing such dummy sequence to be suitably long, we can assume that the selection function f_5 is able to minimize $opt_2(f_5, \sigma)$ without exceeding the total budget.

Consider the following offline strategy used by the adversary: it uses only vertical edges and when needed moves a server from the node that will be called farthest in the future.

According to such a strategy, let us say that at a given step during σ' the i-th horizontal component is covered by the adversary if before serving the corresponding request it has a server on u_i and another one on v_i. Then, the adversary can pay only when there is a call from a node of an uncovered component and in this case it covers it. Except from the first request of σ', this can happen only when the algorithm A uses a vertical edge, as all the requests of a subsequence served by horizontal edges must be from the same horizontal component. Since the last server moved was on the node called farthest in the future, the adversary while serving σ' pays at most once every $\frac{k-1}{2}$ vertical moves of A (like for $\frac{k-1}{2}$ servers on a complete uniform graph of $\frac{k-1}{2}+1 = \frac{k+1}{2}$ nodes). Therefore, as A can perform at most $\frac{cost_1(A,\sigma')}{xW}$ vertical moves, the total number of (vertical) moves executed by the adversary is at most $\frac{\frac{cost_1(A,\sigma')}{xW}}{\frac{k-1}{2}}+1 = \frac{2cost_1(A,\sigma')}{(k-1)xW}+1$.

By the competitive ratios of A, $cost_1(A, \sigma') \leq c_1 opt_1(f_5, \sigma') + \alpha_1 \leq c_1|\sigma'| + \alpha_1$, as there exists a strategy for σ' not exceeding the budget $|\sigma'|$ (for instance the one using only horizontal edges). Moreover, since every vertical move costs (xW, x) and $f_5(\sigma)$ minimizes $opt_2(f_5, \sigma)$, it results

$$opt_2(f_5, \sigma) \leq \left(\frac{2cost_1(A, \sigma')}{(k-1)xW} + 1\right) x = \frac{2cost_1(A, \sigma')}{(k-1)W} + x.$$

Hence, as A performs at least $|\sigma'| - \frac{cost_1(A,\sigma')}{xW}$ horizontal moves,

$$\frac{cost_2(A, \sigma)}{opt_2(f_5, \sigma)} \geq \frac{(|\sigma'| - \frac{cost_1(A,\sigma')}{xW})W}{\frac{2cost_1(A,\sigma')}{(k-1)W} + x} \geq \frac{|\sigma'|W}{\frac{2(c_1|\sigma'|+\alpha_1)}{(k-1)W} + x} - \frac{\frac{cost_1(A,\sigma')}{x}}{\frac{2cost_1(A,\sigma')}{(k-1)W} + x},$$

and as a limit for $|\sigma'| \to \infty$ (and hence $cost_1(A, \sigma') \to \infty$),

$$\frac{cost_2(A, \sigma)}{opt_2(f_5, \sigma)} \geq \frac{(k-1)W^2}{2c_1} - \frac{(k-1)W}{2x}.$$

Therefore, by the arbitrariety of x, $c_2 \geq \frac{(k-1)W^2}{2c_1}$ and thus $c_1c_2 \geq \frac{(k-1)W^2}{2}$. $\square$

Since in general f_5 is only $(\frac{2}{W+1}, \frac{1}{W^2})$-monotone, finding monotonizations that realize good trade-offs does not yield better results than the ones accomplished by the $COMB(A, \lambda)$ algorithm, that achieves competitive ratios $c_1 = c(W\lambda + 1)$ and $c_2 = c(W/\lambda + 1)$.

Again, by exploiting the $(2k-1)$-competitive algorithm of [9] for the single-criterion k-server problem, $c_1 = (2k-1)(W\lambda + 1)$ and $c_2 = (2k-1)(W/\lambda + 1)$. Since $c_1c_2 = O(k^2W^2)$, as a comparison with the lower bound stated in the above lemma, this result is asymptotically optimal for a low number of servers k.

7 Generalizations

The results shown in the previous sections can be extended to many other bicriteria formulations of classical online problems. In particular, if we redefine W as the maximum ratio over each possible selection function f and sequence σ between $opt_1(f,\sigma)$ and $opt_2(f,\sigma)$ and vice versa, the same framework works under the following conditions:

1. the optimization criteria must be *homogeneous*, that is, the minimization of a same function under two different cost measures, so that their linear combination into a corresponding single-criterion problem is well defined;

2. any solution for the combined single-criterion combination must directly translate into a feasible solution for both the two cost measures of the bicriteria formulation;

3. the ratio W must be bounded and independent of the length m of the sequence σ.

An example of problem satisfying such properties is Metrical Tasks Systems, where each request consists in the execution of one of n tasks $t_1, t_2, \ldots, t_n$. Each t_i can be processed in a state s_j belonging to a finite set S incurring a cost equal to $t_i(j)$. Given a metric function defining a distance $d_{j,h}$ for each pair of states s_j and s_h in S, the objective of the algorithm is to determine the state in which to execute each task in order to minimize the overall cost of the schedule, i.e., the sum of the costs due to the transitions between the states and the processing of the tasks (see [3]).

In the bicriteria version of the problem two different costs $t_i^1(j)$ and $t_i^2(j)$ are associated to each task t_i, and two different distances $d_{j,h}^1$ and $d_{j,h}^2$ to each pair of states s_j and s_h in S. As it can be easily checked, all the above conditions hold with $W = \max_{j,h,i}\left\{\frac{d_{j,h}^1+t_i^1(h)}{d_{j,h}^2+t_i^2(h)}, \frac{d_{j,h}^2+t_i^2(h)}{d_{j,h}^1+t_i^1(h)}\right\}$. Here our results are still valid if there are cases such that both $d_{j,h}^1+t_i^1(h) = 0$ and $d_{j,h}^2+t_i^2(h) = 0$. This situation can be taken into account either excluding such ratios in the definition of W or considering them simply equal to 1.

When suitable subsets of (single-criterion) instances of Metrical Task Systems are considered for which better competitive ratios can be achieved (like instances that model the k-server problem), our results can be applied if such subsets are closed under linear combination, that is when the combination according to any possible λ of any two instances in a subset still belongs to the subset.

Another example of problem satisfying the above properties is the bicriteria version of Multiprocessor Scheduling, where jobs arrive one by one and are to be assigned to one of p processors in order to minimize the maximum completion time (see for instance [7, 1]). Associated with each job i are two, possibly different, processing times $t_{1,i}$ and $t_{2,i}$. Again, all the above conditions are satisfied with $W = \max_i\left\{\frac{t_{1,i}}{t_{2,i}}, \frac{t_{2,i}}{t_{1,i}}\right\}$. However, unlike Metrical Task Systems, the merging techniques and the results for the specific selection functions f_1, f_2, f_3, f_4 and f_5 of Example 2.7 cannot be trivially extended and require further investigation.

Multiprocessor scheduling with the additional constraint that each job has to meet a given deadline does not satisfy the above properties. In fact, a job executed before its deadline in the single-criterion combined version does not necessarily meet the original two deadlines in the bicriteria formulation. Therefore, the single-criterion solution might not be feasible for the bicriteria, thus violating condition 2.

8 Conclusion

We have provided universal online algorithms for the multicriteria k-server problem and other problems sharing similar properties.

We have considered a *fair* adversary which seems to give a more realistic estimation of the algorithm performances and to our opinion corresponds to a more appropriate view of the multicriteria setting. However, one could compare the online algorithm also with an unfair adversary that can serve the sequence differently for the different cost measures. As already mentioned in the introduction, this basically requires standard online techniques with additional competitive constraints.

A natural question left open in the paper concerns the elimination of the order of $\log W$ multiplicative gap between the shown lower and upper bounds on the competitive ratios. Moreover, it would be nice to extend our results to any number of objectives.

Another interesting issue to be considered concerns the development of algorithms with improved performances versus specific selection functions like those introduced in Example 2.7. Our results have shown that the technique introduced in this paper related to the determination of good monotonizations overcomes in many cases the structural monotonic limitations of specific selections, thus considerably reducing the competitive ratios. To our opinion, such a technique can be used also when dealing with other bicriteria problems.

Finally, we considered only a limited number of selections, but other reasonable examples are possible.

References

[1] Susanne Albers. Better bounds for online scheduling. *SIAM Journal on Computing*, 29(2):459–473, 1999.

[2] Allan Borodin and Ran El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998.

[3] Allan Borodin, Nathan Linial, and Michael E. Saks. An optimal online algorithm for metrical task systems. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing —STOC '87*, pages 373–382. ACM Press, 1987.

[4] Marek Chrobak and Lawrence L. Larmore. An optimal on-line algorithm for k servers on trees. *SIAM Journal on Computing*, 20(1):144–148, 1991.

[5] Amos Fiat, Yuval Rabani, and Yiftach Ravid. Competitive k-server algorithms. In *Proceedings of the 31st Annual Symposium on Foundations of Computer Science —FOCS '90*, pages 454–463. IEEE Computer Society Press, 1990.

[6] Ashish Goel, Adam Meyerson, and Serge Plotkin. Combining fairness with throughput: Online routing with multiple objectives. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing —STOC '00*, pages 670–679. ACM Press, 2000.

[7] Ronald L. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal on Applied Mathematics*, 17(2):416–429, 1969.

[8] Anna R. Karlin, Mark S. Manasse, Larry Rudolph, and Daniel D. Sleator. Competitive snoopy caching. *Algorithmica*, 3:79–119, 1988.

[9] Elias Koutsoupias and Christos H. Papadimitriou. On the k-server conjecture. *Journal of the ACM*, 42(5):971–983, 1995.

[10] Sven Oliver Krumke, Hartmut Noltemeier, Madhav V. Marathe, R. Ravi, S. S. Ravi, Ravi Sundaram, and Hans-Christoph Wirth. Improving spanning trees by upgrading nodes. *Theoretical Computer Science*, 221(1-2):139–155, 1999.

[11] Jyh-Han Lin and Jeffrey Scott Vitter. ϵ-approximations with minimum packing constraint violation. In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing –STOC'92*, pages 771–782. ACM Press, 1992.

[12] Mark S. Manasse, Lyle A. McGeoch, and Daniel D. Sleator. Competitive algorithms for server problems. *Journal of Algorithms*, 11(2):208–230, 1990.

[13] Madhav V. Marathe, R. Ravi, and Ravi Sundaram. Service-constrained network design problems. In *Proceedings of the 5th Scandinavian Workshop on Algorithm Theory — SWAT '96*, volume 1097 of *Lecture Notes in Computer Science*, pages 28–40. Springer-Verlag, 1996.

[14] Madhav V. Marathe, R. Ravi, Ravi Sundaram, S.S. Ravi, Daniel J. Rosenkrantz, and Harry B. Hunt III. Bicriteria network design problems. In *Proceedings of the 22nd International Colloquium on Automata, Languages and Programming —ICALP '95*, volume 944 of *Lecture Notes in Computer Science*, pages 487–498. Springer-Verlag, 1995.

[15] R. Ravi. Rapid rumor ramification: Approximating the minimum broadcast time. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science —FOCS '94*, pages 202–213. IEEE Computer Society Press, 1994.

[16] R. Ravi and Michel X. Goemans. The constrained minimum spanning tree problem. In *Proceedings of the 5th Scandinavian Workshop on Algorithm Theory —SWAT '96*, volume 1097 of *Lecture Notes in Computer Science*, pages 66–75. Springer-Verlag, 1996.

[17] R. Ravi, Madhav V. Marathe, S. S. Ravi, Daniel J. Rosenkrantz, and Harry B. Hunt III. Many birds with one stone: Multi-objective approximation algorithms (extended abstract). In *Proceedings of the 25th Annual ACM Symposium on Theory of Computing –STOC'93*, pages 438–447. ACM Press, 1993.

[18] Daniel D. Sleator and Robert E. Tarjan. Amortized Efficiency of List Update and Paging Rules. *Communications of the ACM*, 28(2):202–208, 1985.

[19] Artur Warburton. Approximation of Pareto optima in multiple-objective, shortest path problems. *Operations Research*, 35(1):70–79, 1987.