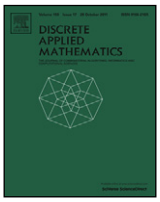
Discrete Applied Mathematics 160 (2012) 1453–1464

Contents lists available at SciVerse ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam



NP-hardness of the sorting buffer problem on the uniform metric[☆]

Yuichi Asahiro[a,*], Kenichi Kawahara[b], Eiji Miyano[b]

[a] *Department of Information Science, Kyushu Sangyo University, Fukuoka 813-8503, Japan*
[b] *Department of Systems Design and Informatics, Kyushu Institute of Technology, Fukuoka 820-8502, Japan*

A R T I C L E I N F O

Article history:
Received 5 June 2010
Received in revised form 24 November 2011
Accepted 7 February 2012
Available online 3 March 2012

Keywords:
Sorting buffer problem
Metric space
Uniform metric
NP-hardness

A B S T R A C T

An instance of the sorting buffer problem (SBP) consists of a sequence of requests for service, each of which is specified by a point in a metric space, and a sorting buffer which can store up to a limited number of requests and rearrange them. To serve a request, the server needs to visit the point where serving a request p following the service to a request q requires the cost corresponding to the distance $d(p, q)$ between p and q. The objective of SBP is to serve all input requests in a way that minimizes the total distance traveled by the server by reordering the input sequence. In this paper, we focus our attention to the uniform metric, i.e., the distance $d(p, q) = 1$ if $p \neq q$, $d(p, q) = 0$ otherwise, and present the first NP-hardness proof for SBP on the uniform metric.

1. Introduction

In the sorting buffer problem (SBP), an instance consists of (1) a sequence of requests for service, each of which is specified by a point in a metric space (V, d), where V is a set of points and d is a distance function, (2) a server which moves from a point to a point in order to serve these requests, and (3) a finite-capacity sorting buffer capable of storing up to L requests. To serve a request, the server needs to visit the request point where serving a request $p \in V$ following the service to a request $q \in V$ requires the cost corresponding to the distance $d(p, q)$ between p and q. The sorting buffer which is a random access buffer can be used to reorder the input sequence. The objective of SBP is to serve all input requests in a way that minimizes the total distance traveled by the server in the metric space by reordering the input sequence.

1.1. Previous and our results

The *buffering-rearranging mechanism* is widely used in many applications and very universal. Thus, several metric spaces are investigated in the literature: Räcke et al. [12] first introduced a *uniform metric*, in which points are either at distance 0 or 1. The uniform metric is motivated by the following manufacturing process: consider a paint shop in a car plant which receives a sequence of cars with specific colors. If consecutive two cars must be painted in different colors, then significant set-up and cleaning costs are incurred in changing colors. That is, the consecutive two requests are at distance 1 if the corresponding cars have to be painted in different colors, and at distance 0 otherwise. Kohrt and Pruhs [11] also consider the problem on the uniform metric and provide a polynomial-time 20-approximation algorithm. This has been subsequently improved by Bar-Yehuda and Laserson [4] to the approximation ratio of 9, but their objective is to maximize the reduction in the cost from that of the input sequence. Khandekar and Pandit [10] investigate a *line metric*, which is motivated by its

☆ A preliminary version of this paper appeared in Asahiro et al. (2008) [2].
* Corresponding author. Tel.: +81 92 673 5411; fax: +81 92 673 5454.
E-mail addresses: asahiro@is.kyusan-u.ac.jp (Y. Asahiro), kawahara@theory.ces.kyutech.ac.jp (K. Kawahara), miyano@ces.kyutech.ac.jp (E. Miyano).

0166-218X/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2012.02.005

application to a disk scheduling problem, and present a 15-approximation algorithm running in $O(|V| \cdot N \cdot L^{O(\log N)})$ time where N is the length of the input sequence. Englert et al. [6] study more general metric spaces.

For the general setting, SBP is known to be NP-hard by a simple reduction from the minimum weight Hamiltonian path problem, and can be solved optimally using dynamic programming in $O(N^{L+1})$ time and in $O(N^{|V|+1})$ time [10]. For the uniform metric, however, it is not known if the problem remains NP-hard. In this paper, we prove NP-hardness of SBP on the uniform metric. Note that the conference version [2] of this paper contained an error which is fixed in this full version; a brief description of the error correction will be included in the footnote of Section 3. Also note that Chan et al. have recently posted an independent proof of the NP-hardness for SBP on the uniform metric on arXiv.org [5].

1.2. *Related work*

Most of the work for SBP has been devoted to the online models; the performance of algorithms is analyzed by the *competitive analysis*. Räcke et al. [12] study several standard strategies on the uniform metric and prove that the competitive ratio of the First In First Out and the Least Recently Used strategies is $\Omega(\sqrt{L})$ and the competitive ratio of the Most Common First strategy is $\Omega(L)$. Also, a deterministic online algorithm with a competitive ratio of $O(\log^2 L)$ is provided in [12], and then Englert and Westermann [7] improved the competitive ratio to $O(\log L)$. Subsequently, Avigdor-Elgrabli and Rabani design a deterministic online algorithm with a competitive ratio of $O(\frac{\log L}{\log \log L})$ [3]. The current best known upper and lower bounds for the competitive ratio are $O(\sqrt{\log L})$ and $\Omega(\sqrt{\log L / \log \log L})$ proved by Adamaszek et al. [1].

Khandekar and Pandit [10] consider the line metric and present a randomized $O(\log^2 N)$ competitive algorithm for N uniformly-spaced points on a line, and then Gamzu and Segev [8] improve this by providing a deterministic $O(\log N)$ competitive algorithm. As for the more general class of metric spaces, Englert and Westermann [7] prove that *any* greedy strategy achieves a $2L - 1$ competitive ratio for *any* metric space and Englert et al. [6] show a polylogarithmic competitive ratio for general metric spaces.

2. Uniform metric model

In this paper, we consider the following paint shop scenario [11,12]: we are given a *service station*, a *(sorting) buffer*, and an input sequence $S = s_1 s_2 \cdots s_N$ of N requests which are characterized by their colors. For each request s_i, let $c(s_i)$ denote the color of s_i. Only for simplicity, we may identify the color $c(s_i)$ with s_i itself. Also, for a subsequence $s_i s_{i+1} \cdots s_{i+\ell-1}$ of ℓ requests, let $c(s_i s_{i+1} \cdots s_{i+\ell-1})$ denote its color sequence $c(s_i)c(s_{i+1}) \cdots c(s_{i+\ell-1})$. The buffer can hold up to L requests, which is used to reorder the input sequence as follows: when the service station receives a new request, the request is initially placed in the buffer. At any time, the buffer contains the first L requests of the input sequence that have not been served so far, and one of those L requests can be processed at the service station and removed from the buffer. The similar process is repeated until the buffer ends up empty. That is, the buffer can reorder the input sequence of N requests; the removed requests result in an *output* sequence $S_{\pi^{-1}} = s_{\pi^{-1}(1)} s_{\pi^{-1}(2)} \cdots s_{\pi^{-1}(N)}$, which is a permutation $\pi(S)$ of the input sequence S. Since the buffer can store only a limited number of requests, the output sequence $S_{\pi^{-1}}$ cannot be an arbitrary permutation of S; it is only possible to reorder S into a sequence which belongs to a restricted subset of the permutations depending on the size L of the buffer.

Let a *color block* denote a maximal subsequence including only requests with the same color in the output sequence. Let a cost $C(S)$ of a sequence S be the number of color blocks in S. The objective of SBP on the uniform metric is to minimize the cost $C(S_{\pi^{-1}})$ of the output sequence $S_{\pi^{-1}}$ by using the sorting buffer.

3. NP-hardness

Let us consider the following decision version of SBP on the uniform metric, SBPU(Z):

Problem SBPU(Z):

Instance: A sequence $S = s_1 s_2 \cdots s_N$ of N requests, their colors $c(s_1), c(s_2), \ldots, c(s_N)$, the capacity L of the sorting buffer, and a positive integer $Z \le N$.
Question: Is there a rearranged output sequence $S_{\pi^{-1}} = s_{\pi^{-1}(1)} s_{\pi^{-1}(2)} \cdots s_{\pi^{-1}(N)}$ such that $C(S_{\pi^{-1}}) \le Z$?

The proof of its NP-hardness is by a polynomial-time reduction from the following vertex cover problem $VC(k)$ [9]. Let $G = (V, E)$ be an undirected graph, where V and E denote the set of vertices and the set of edges, respectively.

Problem VC(k):

Instance: A graph $G = (V, E)$ and a positive integer $k \le |V|$.
Question: Is there a subset $VC \subseteq V$ with $|VC| \le k$ such that for each edge $\{u, v\} \in E$ at least one of u and v belongs to VC?

The subset *VC* is called a *vertex cover*. In the proof of the following theorem, from an instance $G = (V, E)$ and an integer k of $VC(k)$, we construct a sequence S of requests satisfying that there is an output sequence $S_{\pi^{-1}}$ such that $C(S_{\pi^{-1}}) \le Z$ if and only if there is a vertex cover $VC \subseteq V$ with $|VC| \le k$.

Before giving the proof, to make the basic ideas of the reduction clear, we first give its intuitive explanation: consider the problem $VC(1)$ and two graphs $G = (V, E)$ and $G' = (V, E')$ as its instances, where $V = \{a, b, c\}$, $E = \{\{a, b\}, \{b, c\}\}$, and $E' = \{\{a, b\}, \{b, c\}, \{a, c\}\}$. For G, there is a vertex cover $VC = \{b\}$, but there is no vertex cover of size 1 for G'. From these graphs and the integer $k = 1$, we construct two sequences S and S' such that $S = S_h S_{\{a,b\}} S_{\{b,c\}} S_t$ and $S' = S_h S_{\{a,b\}} S_{\{b,c\}} S_{\{a,c\}} S_t$, where each S_i is a subsequence of a certain length for $i \in \{h, t\} \cup E \cup E'$. The capacity L of the sorting buffer and the target costs Z and Z' respectively for S and S' are determined based on $|V|$, $|E|$, and $k(=1)$. Here, if $i \in \{h, t\}$, the subsequence S_i is a concatenation of subsequences represented by $T_V^{i,1} T_B^{i,1} T_V^{i,2} T_B^{i,2}$, otherwise, i.e., if $i \in E \cup E'$, it is $T_V^{i,1} T_B^{i,1} T_i T_V^{i,2} T_B^{i,2}$, where the subsequences $T_V^{i,1}$ and $T_V^{i,2}$ correspond to the vertex set V of G, and the subsequence T_i corresponds to the edge i. Each of $T_B^{i,1}$ and $T_B^{i,2}$ is a long sequence which plays an important role in the proof, and briefly explained in the following. Since the capacity L of the sorting buffer is limited, every algorithm is forced to choose a set of requests from $T_V^{h,1}$ in S_h. The set of requests chosen here corresponds to a subset $U \subseteq V$ which is considered as a candidate of *VC*. Then, once the algorithm has chosen U, we can show that it is impossible to change U when processing, e.g., $T_V^{\{a,b\},1}$, based on the property of the subsequences $T_B^{i,j}$'s; the subsequence $T_B^{i,j}$ is a kind of *barrier* to change the candidate U of *VC*. Namely, if the algorithm changes U, say, when processing $T_V^{\{a,b\},1}$, either of the subsequences $T_V^{\{a,b\},1}$ or $T_B^{\{a,b\},1}$ has to be processed in a non-optimal way that incurs an extra cost. In addition, for $i \in E \cup E'$, the optimal cost for the subsequence T_i is achieved by processing at least one of the requests corresponding to an end point of the edge i right before processing a request corresponding to a vertex which belongs to U (if U is a vertex cover), meaning that the edge i is covered by U. By the construction when $k = 1$, the algorithms can choose a set of requests from $T_V^{h,1}$ in S_h, which corresponds to one vertex in V. Since there is a vertex cover $VC = \{b\}$ for G, a set of requests corresponding to $\{b\}$ can be chosen at this moment for the sequence S. The set is kept unchanged until the end of processing, because of the barriers $T_B^{i,j}$'s, and then $S_{\{a,b\}}$ and $S_{\{b,c\}}$ are processed by the manner of covering described above, which achieves the optimal cost. On the other hand, there is no vertex cover of size 1 for G'. Let us consider an example that $\{b\}$ is also chosen as a candidate U of *VC* at the beginning. Based on this choice, the constructed sequence S' can be processed "optimally" from S_h through $S_{\{b,c\}}$ by the manner of covering. However, since the edge $\{a, c\}$ is not covered by $\{b\}$, the subsequence $S_{\{a,c\}}$ cannot be processed by the manner of covering if we do not change U. As a result, an extra cost is necessary to process $S_{\{a,c\}}$. Since changing U at some intermediate step also incurs an extra cost as described above, we can see that the cost $C(S')$ of S' is larger than the target value Z'.

In this section, we prove the following theorem.

Theorem 1. *SBPU(Z) is NP-complete.*

Proof. From an instance $G = (V, E)$ and an integer k of $VC(k)$, we construct a sequence S of requests satisfying that there is an output sequence $S_{\pi^{-1}}$ such that $C(S_{\pi^{-1}}) \le Z$ if and only if there is a vertex cover $VC \subseteq V$ with $|VC| \le k$. Assume that $|V| = n$ and $|E| = m$ and let $V = \{v_1, v_2, \ldots, v_n\}$ and $E = \{\{v_{i_1}, v_{j_1}\}, \{v_{i_2}, v_{j_2}\}, \ldots, \{v_{i_m}, v_{j_m}\}\}$. Brief explanation of the construction is as follows.

1. We associate each vertex to a color.
2. There are four types of subsequences, two of which correspond to the vertex set V and an edge in the edge set E of the instance of $VC(k)$, and the remaining two types are extra ones.
3. We construct a subsequence for each edge in E, which contains all the four types of subsequences, and then concatenate them.
4. Finally we add special subsequences S_0 and S_{m+1} respectively to the head and the tail of the sequence S, which do not contain the type of subsequence corresponding to an edge.

As a result, the constructed sequence S is a concatenation of $m + 2$ subsequences, a special subsequence S_0 at the head, S_1 through S_m, corresponding to m edges, and another special subsequence S_{m+1} at the tail, i.e., $S = S_0 S_1 \cdots S_m S_{m+1}$. The detailed construction is given below.

First of all, we consider $2(m+2)$ subsequences Σ_i^1 and Σ_i^2 of requests for $i = 0, 1, \ldots, m+1$, referred to as *vertex requests*. Each of the vertex requests consists of n different colors, say, $\sigma_1, \sigma_2, \ldots, \sigma_n$, associated with n vertices, $v_1, v_2, \ldots, v_n$. Then, each vertex request includes two requests per color, and so has length $2n$. In summary,

$$c(\Sigma_i^1) = c(\Sigma_i^2) = \sigma_1\sigma_1\ \sigma_2\sigma_2\ \cdots\ \sigma_n\sigma_n, \quad \text{for } i = 0, 1, \ldots, m+1.$$

Second of all, associated with ℓ-th edge $\{v_{i_\ell}, v_{j_\ell}\} \in E$, we prepare a sequence $e_\ell e'_\ell$ of two requests referred to as an *edge request*, such that $c(e_\ell) = \sigma_{i_\ell}$ and $c(e'_\ell) = \sigma_{j_\ell}$, respectively.

Now we describe the subsequence S_0 at the head in detail, which contains vertex requests and *extra requests* as follows:

$$S_0 = \Sigma_0^1\, A_{0,1}^1 A_{0,1}^2\, A_{0,2}^1 A_{0,2}^2\ \cdots\ A_{0,p}^1 A_{0,p}^2\, \Sigma_0^2\, B_{0,1}^1 B_{0,1}^2\, B_{0,2}^1 B_{0,2}^2\ \cdots\ B_{0,p}^1 B_{0,p}^2.$$

Here, p is some fixed, sufficiently large integer, say, $p \geq n^2$ and, for $j = 1, 2, \ldots, p$, $A^1_{0,j}$ and $A^2_{0,j}$ are sequences of the same colors, each of which consists of $2n + 1$ different colors:

$$c(A^1_{0,j}) = c(A^2_{0,j}) = a_{0,j,1}\, a_{0,j,2}\, \cdots\, a_{0,j,2n+1},$$

and, for any j, the colors in $c(A^1_{0,j})$ and $c(A^2_{0,j})$ do not appear in any other part in the input sequence. Also, for $j = 1, 2, \ldots, p$, $B^1_{0,j}$ and $B^2_{0,j}$ are sequences of the same colors, each of which consists of $4k + 1$ different colors:

$$c(B^1_{0,j}) = c(B^2_{0,j}) = b_{0,j,1}\, b_{0,j,2}\, \cdots\, b_{0,j,4k+1},$$

and, for any j, the colors in $c(B^1_{0,j})$ and $c(B^2_{0,j})$ do not appear in any other part in the input sequence.

The ℓ-th subsequence S_ℓ ($\ell = 1, 2, \ldots, m$) is similar to S_0:

$$S_\ell = \Sigma^1_\ell\, A^1_{\ell,1}A^2_{\ell,1}\, A^1_{\ell,2}A^2_{\ell,2}\, \cdots\, A^1_{\ell,p}A^2_{\ell,p} e_\ell\, e'_\ell\, \Sigma^2_\ell\, B^1_{\ell,1}B^2_{\ell,1}\, B^1_{\ell,2}B^2_{\ell,2}\, \cdots\, B^1_{\ell,p}B^2_{\ell,p},$$

where Σ^1_ℓ and Σ^2_ℓ are vertex requests, $e_\ell e'_\ell$ is an edge request, and $A^1_{\ell,j}, A^2_{\ell,j}, B^1_{\ell,j}$ and $B^2_{\ell,j}$ are extra requests. For $j = 1, 2, \ldots, p$, $A^1_{\ell,j}$ and $A^2_{\ell,j}$ are sequences of the same colors, each of which consists of $2n + 1$ different colors:

$$c(A^1_{\ell,j}) = (A^2_{\ell,j}) = a_{\ell,j,1}\, a_{\ell,j,2}\, \cdots\, a_{\ell,j,2n+1},$$

and colors of $c(A^1_{\ell,j})$ and $c(A^2_{\ell,j})$ do not appear in any other part in the sequence. Also, for $j = 1, 2, \ldots, p$, $B^1_{\ell,j}$ and $B^2_{\ell,j}$ are sequences of the same colors, each of which consists of $4k$ different colors:

$$c(B^1_{\ell,j}) = (B^2_{\ell,j}) = b_{\ell,j,1}\, b_{\ell,j,2}\, \cdots\, b_{\ell,j,4k},$$

and, colors of $c(B^1_{\ell,j})$ and $c(B^2_{\ell,j})$ do not appear in any other part in the sequence.

The last subsequence S_{m+1} includes only vertex and extra requests again:

$$S_{m+1} = \Sigma^1_{m+1}\, A^1_{m+1,1}A^2_{m+1,1}\, \cdots\, A^1_{m+1,p}A^2_{m+1,p}\Sigma^2_{m+1}\, B^1_{m+1,1}B^2_{m+1,1}\, \cdots\, B^1_{m+1,p}B^2_{m+1,p},$$

where, for $j = 1, 2, \ldots, p$, $c(A^1_{m+1,j}) = c(A^2_{m+1,j})$ and $c(B^1_{m+1,j}) = c(B^2_{m+1,j})$ include $2n + 1$ and $4k + 1$ different colors, respectively, and those colors do not appear in any other part in the sequence.

The important differences between S_0 (or S_{m+1}) and S_ℓ ($1 \leq \ell \leq m$) are: (i) $B^i_{\ell,j}$ has only $4k$ requests for $i = 1, 2$, and (ii) S_ℓ includes an edge request $e_\ell e'_\ell$, which play a key role in this proof.

Finally, we set the capacity L of the sorting buffer and the positive integer Z as follows:

$$L = 2n + 2k + 1$$
$$Z = (m + 2)(n + (2n + 1)p + 4kp) + 2p + n - k.$$

This completes the reduction from $VC(k)$ to SBPU(Z).[1]

We show Lemmas 1 and 2 below in the following two subsections, from which Theorem 1 follows. □

Lemma 1. *If there is a vertex cover VC with $|VC| \leq k$, then there is an output sequence $S_{\pi^{-1}}$ such that $C(S_{\pi^{-1}}) \leq Z$.*

Lemma 2. *If there is an output sequence $S_{\pi^{-1}}$ such that $C(S_{\pi^{-1}}) \leq Z$, then there is a vertex cover VC with $|VC| \leq k$.*

3.1. Proof of Lemma 1

In this subsection, we give the proof for Lemma 1. Without loss of generality, assume that $VC = \{v_1, v_2, \ldots, v_k\}$, and thus $V \setminus VC = \overline{VC} = \{v_{k+1}, v_{k+2}, \ldots, v_n\}$. Regarding these VC and $\overline{VC}$, let the corresponding set of the colors be $c_{VC} = \{\sigma_1, \sigma_2, \ldots, \sigma_k\}$ and $c_{\overline{VC}} = \{\sigma_{k+1}, \sigma_{k+2}, \ldots, \sigma_n\}$. We describe subsequences of the output sequence $S_{\pi^{-1}}$ step by step.

(Step 0–1) Since $L = 2n+2k+1$, the whole vertex requests Σ^1_0 of length $2n$ and the first $2k+1$ requests from $A^1_{0,1}$ are stored in the sorting buffer during the first $2n+2k+1$ steps. First, $n-k$ colors $\sigma_{k+1}, \sigma_{k+2}, \ldots, \sigma_n$ ($2(n-k)$ requests) of Σ^1_0, corresponding to $\overline{VC}$, are processed and removed from the sorting buffer. Let $\Sigma_{\overline{VC}_0} = \sigma_{k+1}\sigma_{k+1}\, \sigma_{k+2}\sigma_{k+2}\, \cdots\, \sigma_n\sigma_n$. Then, $C(\Sigma_{\overline{VC}_0}) = n - k$.

(Step 0–2) Now, in the sorting buffer there remain $\sigma_1, \sigma_1, \sigma_2, \sigma_2, \ldots, \sigma_k, \sigma_k$ corresponding to VC and the whole $A^1_{0,1}$ of length $2n + 1$. We process the head $a_{0,1,1}$ of $A^1_{0,1}$ and remove it from the sorting buffer. At this moment, the same

1 In the conference version [2] of the paper, the lengths of subsequences were different. Roughly speaking, the lengths were about half of those in this paper: (i) the length of Σ^1_i and Σ^2_i for $0 \leq i \leq m + 1$ is n, where one request per color is included, (ii) the length of $A^1_{i,j}$ and $A^2_{i,j}$ for $0 \leq i \leq m + 1$ and $1 \leq j \leq p$ is $n + 1$, (iii) the length of $B^1_{0,j}, B^2_{0,j}, B^1_{m+1,j}$, and $B^2_{m+1,j}$ is $2k + 1$, and then (iv) the length of $B^1_{i,j}$ and $B^2_{i,j}$ for $1 \leq i \leq m$ and $1 \leq j \leq p$ is $2k$. Although this difference caused an error in a part of the proof, we could have fixed the error just by doubling the subsequences.

color $a_{0,1,1}$ which is the head of $A^2_{0,1}$ is placed in the buffer and thus $a_{0,1,1}$ is removed. Similarly, each pair of the same color in $A^1_{0,1}$ and $A^2_{0,1}$ is processed consecutively. That is, $2(2n+1)p$ requests of $A^1_{0,1}, A^2_{0,1}, A^1_{0,2}, A^2_{0,2}, \ldots, A^1_{0,p}, A^2_{0,p}$ are permuted as follows:

$$\begin{aligned} A_0 = {} & a_{0,1,1}a_{0,1,1}\; a_{0,1,2}a_{0,1,2}\; \cdots\; a_{0,1,2n+1}a_{0,1,2n+1}a_{0,2,1}a_{0,2,1}\; a_{0,2,2}a_{0,2,2}\; \cdots\; a_{0,2,2n+1}a_{0,2,2n+1} \cdots \\ & a_{0,p,1}a_{0,p,1}\; a_{0,p,2}a_{0,p,2}\; \cdots\; a_{0,p,2n+1}a_{0,p,2n+1}, \end{aligned}$$

where the first $a_{0,1,1}$ comes from $A^1_{0,1}$, the second $a_{0,1,1}$ comes from $A^2_{0,1}$, the third $a_{0,1,2}$ from $A^1_{0,2}$, and so on. The cost $C(A_0)$ is $(2n+1)p$.

(Step 0–3) The next output subsequence of length $4k$ is

$$\Sigma_{VC_0} = \sigma_1\sigma_1\sigma_1\sigma_1\; \sigma_2\sigma_2\sigma_2\sigma_2\; \cdots\; \sigma_k\sigma_k\sigma_k\sigma_k,$$

and its cost $C(\Sigma_{VC_0})$ is k.

(Step 0–4) Currently, $2(n-k)$ requests, $\sigma_{k+1}, \sigma_{k+1}, \sigma_{k+2}, \sigma_{k+2}, \ldots, \sigma_n, \sigma_n$, and the whole $B^1_{0,1}$ of length $4k+1$ are stored in the sorting buffer, from which the following subsequence of length $2(4k+1)p$ is obtained by merging $B^1_{0,1}$ and the subsequence $B^2_{0,1}B^1_{0,2}B^2_{0,2}\cdots B^1_{0,p}B^2_{0,p}$:

$$\begin{aligned} B_0 = {} & b_{0,1,1}b_{0,1,1}\; b_{0,1,2}b_{0,1,2}\; \cdots\; b_{0,1,4k+1}b_{0,1,4k+1}b_{0,2,1}b_{0,2,1}\; b_{0,2,2}b_{0,2,2}\; \cdots\; b_{0,2,4k+1}b_{0,2,4k+1} \cdots \\ & b_{0,p,1}b_{0,p,1}\; b_{0,p,2}b_{0,p,2}\; \cdots\; b_{0,p,4k+1}b_{0,p,4k+1}, \end{aligned}$$

where the first $b_{0,1,1}$ comes from $B^1_{0,1}$, the second $b_{0,1,1}$ comes from $B^2_{0,1}$, the third $b_{0,1,2}$ comes from $B^1_{0,2}$, and so on. The cost $C(B_0)$ is $(4k+1)p$.

At this moment, σ_{k+1} through σ_n (each of which has two requests and $2(n-k)$ requests in total) from S_0 and the first $4k+1$ requests from S_1 are placed in the sorting buffer. In the following, from $\ell = 1$ to $\ell = m$, we repeat (Step ℓ-1) through (Step ℓ-4), which are very similar to (Step 0–1) through (Step 0–4).

(Step ℓ-1) The following sequence $\Sigma_{\overline{VC}_\ell}$ of colors corresponding to $\overline{VC}$ comes out of the buffer, merging σ_{k+1} through σ_n in $S_{\ell-1}$ and the same colors in Σ^1_ℓ of S_ℓ. If a color σ_j in $S_{\ell-1}$ which corresponds to one color of an edge request not in c_{VC} remains in the buffer,[2]

$$\Sigma_{\overline{VC}_\ell} = \sigma_{k+1}\sigma_{k+1}\sigma_{k+1}\sigma_{k+1}\; \sigma_{k+2}\sigma_{k+2}\sigma_{k+2}\sigma_{k+2}\; \cdots \sigma_j\sigma_j\sigma_j\sigma_j\sigma_j\; \cdots\; \sigma_n\sigma_n\sigma_n\sigma_n,$$

where five σ_j's are output consecutively, or, otherwise,

$$\Sigma_{\overline{VC}_\ell} = \sigma_{k+1}\sigma_{k+1}\sigma_{k+1}\sigma_{k+1}\; \sigma_{k+2}\sigma_{k+2}\sigma_{k+2}\sigma_{k+2}\; \cdots\; \sigma_n\sigma_n\sigma_n\sigma_n.$$

The cost $C(\Sigma_{\overline{VC}_\ell})$ is $n-k$.

(Step ℓ-2) For $j = 1, \ldots, p$, $A^1_{\ell,j}$ and $A^2_{\ell,j}$ are merged in a similar way to (Step 0–2), and thus the following output subsequence is obtained:

$$\begin{aligned} A_\ell = {} & a_{\ell,1,1}a_{\ell,1,1}\; a_{\ell,1,2}a_{\ell,1,2}\; \cdots\; a_{\ell,1,2n+1}a_{\ell,1,2n+1}a_{\ell,2,1}a_{\ell,2,1}\; a_{\ell,2,2}a_{\ell,2,2}\; \cdots\; a_{\ell,2,2n+1}a_{\ell,2,2n+1} \cdots \\ & a_{\ell,p,1}a_{\ell,p,1}\; a_{\ell,p,2}a_{\ell,p,2}\; \cdots\; a_{\ell,p,2n+1}a_{\ell,p,2n+1} \end{aligned}$$

and its cost $C(A_\ell)$ is $(2n+1)p$.

(Step ℓ-3) Currently, σ_1 through σ_k from Σ^1_ℓ ($2k$ requests), e_ℓ and e'_ℓ of an edge request $e_\ell e'_\ell$, and two σ_1's through σ_{n-1}'s, and one σ_n from Σ^2_ℓ are stored in the sorting buffer. It is important to note that at least one of $c(e_\ell) = \sigma_{i_\ell}$ and $c(e'_\ell) = \sigma_{j_\ell}$ is the same as one of σ_1 through σ_k since at least one of e_ℓ and e'_ℓ must be in VC. Now, for example, assume that $e'_\ell \in VC$. Then, the following sequence is removed:

$$\Sigma_{VC_\ell} = \sigma_1\sigma_1\sigma_1\sigma_1\; \sigma_2\sigma_2\sigma_2\sigma_2\; \cdots\; \sigma_{j_\ell}\sigma_{j_\ell}\sigma_{j_\ell}\sigma_{j_\ell}\sigma_{j_\ell}\; \cdots\; \sigma_k\sigma_k\sigma_k\sigma_k,$$

and its cost $C(\Sigma_{VC_\ell})$ is k. If both of e_ℓ and e'_ℓ are in VC, then Σ_{VC_ℓ} is as follows, assuming $i_\ell \le j_\ell$:

$$\Sigma_{VC_\ell} = \sigma_1\sigma_1\sigma_1\sigma_1\; \cdots\; \sigma_{i_\ell}\sigma_{i_\ell}\sigma_{i_\ell}\sigma_{i_\ell}\sigma_{i_\ell}\; \cdots\; \sigma_{j_\ell}\sigma_{j_\ell}\sigma_{j_\ell}\sigma_{j_\ell}\sigma_{j_\ell}\; \cdots\; \sigma_k\sigma_k\sigma_k\sigma_k,$$

and its cost is still k.

(Step ℓ-4) For $j = 1, \ldots, p$, $B^1_{\ell,j}$ and $B^2_{\ell,j}$ are merged in a similar way to (Step 0–4). At the beginning of this step, there remain $2(n-k)$ requests, $\sigma_{k+1}, \sigma_{k+1}, \sigma_{k+2}, \sigma_{k+2}, \ldots, \sigma_n, \sigma_n$ in the sorting buffer. Also in the sorting buffer there may exist one request, say, σ_{j_ℓ} of the edge request $e_\ell e'_{ell}$ because of (Step ℓ-3), i.e., currently $2(n-k)+1$ requests may be

[2] Such a color does not exist in the case $\ell = 1$ by the definitions of Step 0–1–Step 0–4.

stored in the buffer. However, differently to (Step 0–4), since the length of $B^1_{\ell,j}$ (and $B^2_{\ell,j}$) for $1 \le j \le p$ is $4k$, the whole of $B^1_{\ell,j}$ is also stored in the sorting buffer. Hence by merging B^1_ℓ and B^2_ℓ similarly to (Step 0–4), the following output subsequence can be obtained:

$$B_\ell = b_{\ell,1,1}b_{\ell,1,1}\, b_{\ell,1,2}b_{\ell,1,2} \cdots b_{\ell,1,4k}b_{\ell,1,4k}b_{\ell,2,1}b_{\ell,2,1} \cdots b_{\ell,2,4k}b_{\ell,2,4k} \cdots b_{\ell,p,1}b_{\ell,p,1} \cdots b_{\ell,p,4k}b_{\ell,p,4k}$$

and its cost $C(B_\ell)$ is $4kp$ since $B^1_{\ell,j}$ and $B^2_{\ell,j}$ have $4k$ different colors.

After (Step m-4), σ_{k+1} through σ_n ($2(n-k)$ requests) remain in the sorting buffer. We execute a similar phase once more, and generate four output subsequences, $\Sigma_{\overline{VC}_{m+1}}$, A_{m+1}, $\Sigma_{VC_{m+1}}$, and B_{m+1}: if a color σ_j of the edge request in S_m is not in c_{VC}, then it remains in the sorting buffer and hence

$$\Sigma_{\overline{VC}_{m+1}} = \sigma_{k+1}\sigma_{k+1}\sigma_{k+1}\sigma_{k+1}\, \sigma_{k+2}\sigma_{k+2}\sigma_{k+2}\sigma_{k+2} \cdots \sigma_j\sigma_j\sigma_j\sigma_j\sigma_j \cdots \sigma_n\sigma_n\sigma_n\sigma_n,$$

where five σ_j's are output consecutively, or, otherwise,

$$\Sigma_{\overline{VC}_{m+1}} = \sigma_{k+1}\sigma_{k+1}\sigma_{k+1}\sigma_{k+1}\, \sigma_{k+2}\sigma_{k+2}\sigma_{k+2}\sigma_{k+2} \cdots \sigma_n\sigma_n\sigma_n\sigma_n,$$

$$\begin{aligned} A_{m+1} &= a_{m+1,1,1}a_{m+1,1,1} \cdots a_{m+1,1,2n+1}a_{m+1,1,2n+1}a_{m+1,2,1}a_{m+1,2,1} \cdots a_{m+1,2,2n+1}a_{m+1,2,2n+1} \cdots \\ &\quad a_{m+1,p,1}a_{m+1,p,1} \cdots a_{m+1,p,2n+1}a_{m+1,p,2n+1}, \end{aligned}$$

$$\Sigma_{VC_{m+1}} = \sigma_1\sigma_1\sigma_1\sigma_1\, \sigma_2\sigma_2\sigma_2\sigma_2 \cdots \sigma_k\sigma_k\sigma_k\sigma_k\ (=\Sigma_{VC_0}),$$

$$\begin{aligned} B_{m+1} &= b_{m+1,1,1}b_{m+1,1,1} \cdots b_{m+1,1,4k+1}b_{m+1,1,4k+1}b_{m+1,2,1}b_{m+1,2,1} \cdots b_{m+1,2,4k+1}b_{m+1,2,4k+1} \cdots \\ &\quad b_{m+1,p,1}b_{m+1,p,1} \cdots b_{m+1,p,4k+1}b_{m+1,p,4k+1}. \end{aligned}$$

Finally, there remain $n-k$ colors ($2(n-k)$ requests), σ_{k+1} through σ_n, each of which is removed from the sorting buffer. Let $\Sigma_{\overline{VC}_{m+2}}$ be the sequence of those $n-k$ colors. As a result, the output sequence $S_{\pi^{-1}}$ is as follows:

$$S_{\pi^{-1}} = \Sigma_{\overline{VC}_0}A_0\Sigma_{VC_0}B_0\ \Sigma_{\overline{VC}_1}A_1\Sigma_{VC_1}B_1 \cdots \Sigma_{\overline{VC}_{m+1}}A_{m+1}\Sigma_{VC_{m+1}}B_{m+1}\ \Sigma_{\overline{VC}_{m+2}}.$$

Therefore, the lemma holds since the total cost $C(S_{\pi^{-1}})$ is Z:

$$\begin{aligned} C(S_{\pi^{-1}}) &= \sum_{\ell=0}^{m+1}(C(\Sigma_{VC_\ell}) + C(\Sigma_{\overline{VC}_\ell})) + \sum_{\ell=0}^{m+1}(C(A_\ell) + C(B_\ell)) + C(\Sigma_{\overline{VC}_{m+2}}) \\ &= (m+2)(n + (2n+1)p + 4kp) + 2p + n - k. \end{aligned}$$

3.2. *Proof of Lemma 2*

In this subsection, we prove Lemma 2. First we introduce the notation only used in this subsection, and then show several propositions on important properties and the optimal cost of subsequences rearranged in the sorting buffer. Let ΞX denote the multiset of colors in the buffer when the head request of a (sub)sequence X (of the input sequence) is ready to enter into the sorting buffer. For simplicity we say that X *is ready* instead of "the head request of X is ready to enter into the sorting buffer". Similarly, a request is *ready* if it is ready to enter into the sorting buffer. Let $\{T\}$ for a sequence T be the multiset of colors in T. For a multiset M of colors, $\#(M)$ denotes the number of colors in M. For a pair of sequences X and Y, $[Y]X$ denotes the situation that $\Xi X = \{Y\}$, in which we are implicitly assuming that $|\{Y\}|$ is at most the buffer size. If the buffer is empty, we write $[\emptyset]X$. Note that the sequence of colors in the buffer is not important because we can remove them in an arbitrary order.

Let $OPT(X)$ (or $OPT(X_{\pi^{-1}})$ if we want to clarify that X is reordered) denote the optimal cost for a sequence X, under the assumption that the buffer is empty before X arrives, i.e., $OPT(X) = \min_\pi\{C(X_{\pi^{-1}})\}$. For a pair of sequences X and Y, $OPT([Y]X)$ denotes the optimal cost to process X with the sorting buffer containing $\{Y\}$ initially, i.e., $OPT([Y]X) = OPT([\emptyset]YX) = OPT(YX)$.

An output sequence $s_1s_2s_3s_4 \cdots s_l$ of length l is a *pairwise output* if $c(s_i) = c(s_{i+1})$ for any odd i, or *quadwise* if $c(s_i) = c(s_{i+1}) = c(s_{i+2}) = c(s_{i+3})$ for any $i = 1 \bmod 4$. For example, A_0 and Σ_{VC_0} in the proof of Lemma 1 are pairwise and quadwise, respectively. We define $A^{(\ell)}$ and $B^{(\ell)}$ are as follows:

$$A^{(\ell)} = A^1_{\ell,1}A^2_{\ell,1}\, A^1_{\ell,2}A^2_{\ell,2} \cdots A^1_{\ell,p}A^2_{\ell,p} \quad \text{and}$$

$$B^{(\ell)} = B^1_{\ell,1}B^2_{\ell,1}\, B^1_{\ell,2}B^2_{\ell,2} \cdots B^1_{\ell,p}B^2_{\ell,p}.$$

First of all, in Propositions 1–3, we show quite simple but important properties on the optimal cost of (sub)sequences rearranged in the sorting buffer. The following proposition gives a trivial lower bound on the optimal cost.

Proposition 1. *For any sequence X, it holds that $OPT(X) \ge \#(X)$.* □

Propositions 2 and 3 are on the optimal cost to process a sequence using the buffer initially holding another sequence.

Proposition 2. *For any pair of sequences X and Y, it holds that $OPT([Y]X) \ge OPT(X)$.* □

Proposition 3. *For any two sequences X and Y such that $\{X\} \cap \{Y\} = \emptyset$, $OPT([Y]X) = OPT(Y) + OPT(X)$.* □

Based on the above propositions, we prove Lemma 2 in the following. The main idea of the proof is to show that the output sequence $S_{\pi^{-1}}$ given in the previous section is essentially the unique choice to minimize the cost.

For a while, we focus on the first subsequence S_0 of S (but the following claims and propositions on S_0 also hold for S_{m+1}). First we show the optimal cost of S_0 in Proposition 4 below. Let $Z_0 = (2n + 4k + 2)p + 2n - k$.

Proposition 4. *$OPT(S_0) = Z_0$.* □

Proof. We first show that $OPT(S_0) \geq Z_0$, and then present an output sequence of the cost Z_0. We divide S_0 into two subsequences S_0^H and S_0^T such that

$$S_0^H = \Sigma_0^1 A_{0,1}^1 \quad \text{and}$$
$$S_0^T = A_{0,1}^2 A_{0,2}^1 A_{0,2}^2 \cdots A_{0,p}^1 A_{0,p}^2 \Sigma_0^2 B^{(0)}.$$

For the number of the colors in S_0^T, we observe that

$$\begin{aligned}\#(S_0^T) &= \#(A_{0,1}^2 A_{0,2}^1 A_{0,2}^2 \cdots A_{0,p}^1 A_{0,p}^2 \Sigma_0^2 B^{(0)})\\ &= (2n+1)p + n + (4k+1)p\\ &= (2n+4k+2)p + n. \end{aligned} \tag{1}$$

Since $\{\Sigma_0^1\} \cap \{A_{0,1}^1\} = \emptyset$ and $|\{S_0^H\}| = 4n + 1$, we must always remove $|\{S_0^H\}| - L = 2n - 2k$ requests of S_0^H before S_0^T is ready. Here, these $2n - 2k$ requests include at least $n - k$ colors since each color appears at most twice in Σ_0^1 and $A_{0,1}^1$, and thus the following holds.

$$\begin{aligned} OPT(S_0) &\geq (n-k) + OPT([\Xi S_0^T] S_0^T)\\ &\geq (n-k) + OPT(S_0^T)\\ &\geq n - k + (2n+4k+2)p + n\\ &= (2n+4k+2)p + 2n - k \ (= Z_0), \end{aligned}$$

where the second inequality comes from Proposition 2, and the third one comes from Proposition 1 and Eq. (1).

Next we present an output sequence which achieves the minimum cost Z_0 for S_0. The output sequence is very similar to a concatenation of the output sequence given by (Step 0–1) through (Step 0–4) in the proof of Lemma 1 and an output sequence obtained from the requests remaining in the sorting buffer at the end of (Step 0–4). We describe only the difference from the sequences of (Step 0–1) through (Step 0–4) in the following.

(Step 0–1$'$) The multiset of $n - k$ colors (with $2(n - k)$ requests) removed first can be chosen *arbitrarily* from Σ_0^1 although in the previous section we selected $n - k$ colors corresponding to $\overline{VC}$. Let the multiset be $\overline{W_0}$ and the multiset of remaining k colors be W_0 such that $|W_0| = 2k$. Without loss of generality, we can assume that $W_0 = \{\sigma_1, \sigma_1, \ldots, \sigma_k, \sigma_k\}$ as well, and then the output subsequence $\Sigma_{\overline{W_0}}$ is the same as $\Sigma_{\overline{VC_0}}$ in (Step 0–1).

(Step 0–2$'$) Exactly the same as (Step 0–2).

(Step 0–3$'$) Almost all the same as (Step 0–3) except that we denote the output subsequence by Σ_{W_0} to distinguish.

(Step 0–4$'$) Exactly the same as (Step 0–4).

(Step 0–5$'$) Currently, since the buffer contains $\overline{W_0}$, we remove them by $\Sigma_{\overline{W_0}}$, again. The number of color blocks of the output subsequence $\Sigma_{\overline{W_0}}$ is $n - k$.

The whole output sequence obtained in (Step 0–1$'$) through (Step 0–5$'$) is

$$S_{0,\pi^{-1}} = \Sigma_{\overline{W_0}} A_0 \Sigma_{W_0} B_0 \Sigma_{\overline{W_0}},$$

and its cost $C(S_{0,\pi^{-1}})$ is

$$\begin{aligned} C(S_{0,\pi^{-1}}) &= 2C(\Sigma_{\overline{W_0}}) + C(A_0) + C(\Sigma_{W_0}) + C(B_0)\\ &= 2(n-k) + (2n+1)p + k + (4k+1)p\\ &= (2n+4k+2)p + 2n - k \ (= Z_0). \end{aligned}$$

Therefore we can conclude that $OPT(S_0) = Z_0$. □

Based on the above proposition, we next show the necessary conditions on optimal output sequences for S_0; more precisely, any optimal output sequence must obey (Step 0–1$'$), (Step 0–2$'$), and (Step 0–4$'$), which are shown by Claims 1, 2 and 4, respectively.

Claim 1. *For an output sequence U_0 of S_0, if $C(U_0) = Z_0$, then $|\{\Sigma_0^1\} \cap \Xi \Sigma_0^2| = 2k$ and $\#(\{\Sigma_0^1\} \cap \Xi \Sigma_0^2) = k$ must be satisfied.*

Proof. We prove this claim by contradiction, considering three cases (Case 1) $|\{\Sigma_0^1\} \cap \Xi \Sigma_0^2| \geq 2k + 1$, (Case 2) $|\{\Sigma_0^1\} \cap \Xi \Sigma_0^2| \leq 2k - 1$, and (Case 3) $|\{\Sigma_0^1\} \cap \Xi \Sigma_0^2| = 2k$ but $\#(\{\Sigma_0^1\} \cap \Xi \Sigma_0^2) \geq k + 1$. (Note that if $|\{\Sigma_0^1\} \cap \Xi \Sigma_0^2| = 2k$, then $\#(\{\Sigma_0^1\} \cap \Xi \Sigma_0^2) \geq k$ holds since each color is included at most twice in $\{\Sigma_0^1\} \cap \Xi \Sigma_0^2$.)

(*Case* 1). Since $\{\Sigma_0^1\} \cap \Xi\Sigma_0^2 \subseteq \{\Sigma_0^1\} \cap \Xi A_{0,1}^2$ and $\{A_{0,j}^i\} \cap \{\Sigma_0^1\} = \emptyset$ for $i = 1, 2$ and $j = 1, \ldots, p$, it holds that $2k+1 \le |\{\Sigma_0^1\} \cap \Xi\Sigma_0^2| \le |\{\Sigma_0^1\} \cap \Xi A_{0,1}^2|$. Assume, for simplicity, that $|\{\Sigma_0^1\} \cap \Xi A_{0,1}^2| = 2k+1$, by which we observe that there exist $2n$ requests of $A_{0,1}^1$ in $\Xi A_{0,1}^2$ since the buffer size is $L = 2n + 2k + 1$. When the head request of $A_{0,1}^2$ is ready, the number of color blocks so far is at least $n - k + 1$ according to the argument of (Step 0–1$'$). Namely, at least one request (color) of $A_{0,1}^1$ is not included in $\Xi A_{0,1}^2$. Let such a color be $a_{0,1,q}$ for some q. There are two more subcases; the next output from the current buffer is (Case 1–1) a request of $\{\Sigma_0^1\}$, or (Case 1–2) a request of $\{A_{0,1}^1\}$.

(*Case* 1–1). After removing the color $\sigma \in \{\Sigma_0^1\}$, $a_{0,1,1}$ of $A_{0,1}^2$ enters into the buffer. Since $OPT(S_0^T) \ge (2n + 4k + 2)p + n$ as in the proof of Proposition 4, the total number of color blocks must be at least $(n - k + 1) + (2n + 4k + 2)p + n = (2n + 4k + 2)p + 2n - k + 1 > Z_0$, which contradicts the assumption $C(U_0) = Z_0$.

(*Case* 1–2). Let us divide S_0 into four subsequences S_0^1, S_0^2, S_0^3, and S_0^4:

$$\begin{aligned} S_0^1 &= \Sigma_0^1\, A_{0,1}^1\, a_{0,1,1}\, a_{0,1,2} \cdots a_{0,1,q-1}, \\ S_0^2 &= a_{0,1,q}\, a_{0,1,q+1} \cdots a_{0,1,2n+1}, \\ S_0^3 &= A_{0,2}^1 A_{0,2}^2 \cdots A_{0,p}^1 A_{0,p}^2, \quad \text{and} \\ S_0^4 &= \Sigma_0^2\, B_{0,1}^1 B_{0,1}^2\, B_{0,2}^1 B_{0,2}^2 \cdots B_{0,p}^1 B_{0,p}^2. \end{aligned}$$

When $a_{0,1,q}$ of S_0^2 is ready, the buffer contains $2k+1$ requests of Σ_0^1 by the assumption $|\{\Sigma_0^1\} \cap \Xi\Sigma_0^2| = 2k+1$. Recall that there exists only space which can hold up to other $2n$ requests. Since the distance between $a_{0,1,q}$ in S_0^1 and $a_{0,1,q}$ in S_0^2 is $2n+1$, they must be removed separately from the sorting buffer. Since we can consider that the colors of S_0^2 will be removed before the colors of S_0^3 from Proposition 3, a possible output subsequence to minimize the cost for removing colors of $\{A_{0,1}^1\} \cup \{A_{0,1}^2\}$ is like this:

$$\cdots a_{0,1,q} \cdots a_{0,1,1} a_{0,1,1}\, a_{0,1,2} a_{0,1,2} \cdots a_{0,1,q} \cdots a_{0,1,2n+1} a_{0,1,2n+1}.$$

The optimal cost incurred for this portion is at least $2n+2$. Thus the number of color blocks so far is at least $(n-k+1) + (2n+2) = 3n - k + 3$ where $(n-k+1)$ comes from the cost of $n-k+1$ colors in Σ_0^1, and $(2n+2)$ comes from the above output sequence.

It is not hard to see that for each pair of $A_{0,h}^1$ and $A_{0,h}^2$ for $2 \le h \le p$, the cost of $(2n+2)$ is required due to the limited space in the sorting buffer. Thus, the number of color blocks so far when Σ_0^2 is ready is at least $(n-k+1) + (2n+2)p$. In addition to that, $OPT(S_0^4) \ge n + (4k+1)p$ from Proposition 1. Therefore, the total number of color blocks is at least

$$(n-k+1) + (2n+2)p + n + (4k+1)p = (2n+4k+2)p + 2n - k + p + 1\ (>Z_0),$$

which contradicts the assumption.

In the above argument, we assumed that $|\{\Sigma_0^1\} \cap \Xi A_{0,1}^2| = 2k+1$ for simplicity. However, in the case that $|\{\Sigma_0^1\} \cap \Xi A_{0,1}^2| > 2k+1$, i.e., the buffer contains more than $2k+1$ requests of Σ_0^1 when $a_{0,1,q}$ of S_0^2 is ready, the situation gets worse since the available space in the sorting buffer is smaller. [End of Case 1.]

(*Case* 2). Since $|\{\Sigma_0^1\} \cap \Xi\Sigma_0^2| \le 2k-1$, at least $2n - 2k + 1$ requests of Σ_0^1 have been already removed when Σ_0^2 is ready. This produces at least $n-k+1$ color blocks. Then since the lower bound of the optimal cost for the subsequence $A_{0,1}^1\, A_{0,1}^2 \cdots A_{0,p}^1\, A_{0,p}^2$ is $(2n+1)p$ and that for S_0^4 is $n + (4k+1)p$ from Proposition 1, the total cost is at least

$$(n-k+1) + (2n+1)p + n + (4k+1)p = (2n+4k+2)p + 2n - k + 1\ (>Z_0),$$

which contradicts the assumption. [End of Case 2.]

(*Case* 3). The fact $\#(\{\Sigma_0^1\} \cap \Xi\Sigma_0^2) \ge k+1$ implies that some of the colors from $\{\Sigma_0^1\} \cap \Xi\Sigma_0^2$ has only one request in $\{\Sigma_0^1\} \cap \Xi\Sigma_0^2$, because $|\{\Sigma_0^1\} \cap \Xi\Sigma_0^2| = 2k$ and each color is included at most twice in the buffer. Let $r = \#(\{\Sigma_0^1\} \cap \Xi\Sigma_0^2) - k\ (> 0)$. In this case the number of color blocks already produced when $A_{0,1}^1$ is ready is at least $n-k+r$. Since $\#(A^{(0)}\ \Sigma_0^2\ B^{(0)}) = (2n+1)p + n + (4k+1)p$, in this case the total number of color blocks turns to be at least $(n-k+r) + (2n+1)p + n + (4k+1)p = (2n+4k+2)p + 2n - k + r > Z_0$. [End of Case 3.]

From the above cases (Cases 1–3) we conclude that $|\{\Sigma_0^1\} \cap \Xi\Sigma_0^2| = 2k$ and $\#(\{\Sigma_0^1\} \cap \Xi\Sigma_0^2) = k$. □

From the above claim, we obtain the following corollaries in a straightforward way.

Corollary 1. *The multiset of colors W_0 determined by (Step 0–1$'$) remains in $\Xi\Sigma_0^2$.* □

Corollary 2. *For an output sequence U_0 of S_0, if $C(U_0) = Z_0$, $|\{\Sigma_0^1\} \cap \Xi A_{0,1}^2| = 2k$ and $\#(\{\Sigma_0^1\} \cap \Xi A_{0,1}^2) = k$ must be satisfied.*

Proof. From Claim 1 it holds that $|\{\Sigma_0^1\} \cap \Xi\Sigma_0^2| = 2k$. First we assume that $|\{\Sigma_0^1\} \cap \Xi A_{0,1}^2| \ge 2k+1$, i.e., at least one request in Σ_0^1 remains in the buffer when $A_{0,1}^2$ is ready but is removed while processing $A^{(0)}$ before Σ_0^2 is ready. In this case, by a similar discussion to the (Case 1–2) of the proof of Claim 1, we observe that the total cost of U_0 is greater than Z_0, because there is not enough room in the buffer for storing the whole $A^i_{0,j}$ for each i and j. Therefore $|\{\Sigma^1_0\} \cap \Xi A^2_{0,1}| = 2k$ as well. To consider the other case, suppose that $|\{\Sigma^1_0\} \cap \Xi A^2_{0,1}| = 2k$ and $\#(\{\Sigma^1_0\} \cap \Xi A^2_{0,1}) \geq k+1$. In this case, the total cost of U_0 is also greater than Z_0, because the number of color blocks before $A^2_{0,1}$ is ready is at least $n-k+1$. □

In conjunction with the above Claim 1, the following claim guarantees the optimality of the output subsequence for S_0 that is determined by (Step 0–1′) and (Step 0–2′).

Claim 2. *For an output sequence U_0 of S_0, if $C(U_0) = Z_0$, the output subsequence for $A^{(0)}$ must be pairwise.*

Proof. From Corollary 2, $\Xi A^2_{0,1}$ contains $2k$ requests of Σ^1_0 and the whole $A^1_{0,1}$. Thus, we can reorder $A^{(0)}$ into A_0 using the space of size $2n+1$, which is occupied by $A^1_{0,1}$ currently in the buffer. If we remove two $a_{0,i,j}$'s in $A^{(0)}$ separately, it wastes the cost so that the cost at the end of $A^{(0)}$ is greater than $(n-k) + (2n+1)p$. Moreover, the optimal cost for S^4_0 is at least $n + (4k+1)p$. Therefore the final cost is greater than $(2n+4k+2)p + 2n - k = Z_0$ from Proposition 3. □

Next we show the intuitive validity of (Step 0–3′), while the following claim only guarantees that a subset of $\overline{W_0}$ remains in the buffer.

Claim 3. *For an output sequence U_0 of S_0, if $C(U_0) = Z_0$, $|\{\Sigma^2_0\} \cap \Xi B^2_{0,1}| \leq 2n - 2k$ and $\{\Sigma^2_0\} \cap \Xi B^2_{0,1} \subseteq \overline{W_0}$ must be satisfied.*

Proof. Assume that $|\{\Sigma^2_0\} \cap \Xi B^2_{0,1}| \geq 2n - 2k + 1$, so that less than $2k$ colors ($4k$ requests) of $B^1_{0,1}$ can be stored in the buffer when the last request $b_{0,1,4k+1}$ of $B^1_{0,1}$ is ready. At this moment, we must remove one color of $B^1_{0,1}$, say, $b_{0,1,i}$ for some i. As a result, the coming $b_{0,1,i}$ in $B^2_{0,1}$ cannot be removed right after $b_{0,1,i}$ in $B^1_{0,1}$. Namely, by the assumption, the subsequence $B^{(0)}$ cannot be reordered into a pairwise sequence. Thus, even if the output subsequence so far is optimal satisfying Claims 1 and 2, the total cost is greater than Z_0. This contradicts the assumption on optimality, and so $|\{\Sigma^2_0\} \cap \Xi B^2_{0,1}| \leq 2n - 2k$ must be satisfied.

Next, assume that $|\{\Sigma^2_0\} \cap \Xi B^2_{0,1}| \leq 2n - 2k$ and $\{\Sigma^2_0\} \cap \Xi B^2_{0,1} \not\subseteq \overline{W_0}$ hold. From Corollary 1, we observe that $W_0 \subseteq \Xi\, \Sigma^2_0$. After removing $A^{(0)}$ based on Claim 2, if we output a subsequence differ from Σ_{W_0}, it produces more than k color blocks, because the number of remaining requests must be at most $2(n-k)$, in which only one request of some color is included. As a result, the total cost is again greater than Z_0, which again contradicts the assumption on optimality. Therefore the claim holds. □

The proof of the above claim indicates the optimality of the sequence determined by (Step 0–4′).

Claim 4. *For an output sequence U_0 of S_0, if $C(U_0) = Z_0$, the output subsequence for $B^{(0)}$ must be pairwise.* □

Now we know that the optimal output sequence has to satisfy the conditions (Step 0–1′), (Step 0–2′), and (Step 0–4′) for the subsequence S_0, from Claims 1, 2 and 4. However (Step 0–3′) is just a candidate of optimal output sequences. In the following, we will prove that (Step 0–3′) has to be also satisfied in order to obtain an optimal output sequence for the entire input. Before proving it, we need to deal with each subsequence S_ℓ for $1 \leq \ell \leq m$.

We define an output sequence for S_ℓ as in (Step 0–1′) through (Step 0–5′). The main difference can be seen in the steps (Step ℓ-3′) and (Step ℓ-5′) in which respectively five and three requests in identical colors corresponding to a color of the edge request are removed consecutively.

(Step ℓ-1′) Almost all the same as (Step ℓ-1) except that we denote the output subsequence by $\Sigma_{\overline{W_\ell}}$, and the remaining requests of Σ^1_ℓ by W_ℓ.
(Step ℓ-2′) Exactly the same as (Step ℓ-2) (and so as (Step 0–2′)).
(Step ℓ-3′) Almost all the same as (Step ℓ-3) except that

- we denote the output subsequence by Σ_{W_ℓ} to distinguish, and
- if neither of the two colors σ_{i_ℓ} and σ_{j_ℓ} of the edge request $e_\ell e_{\ell'}$ are in W_ℓ, the output subsequence is as follows:

$$\Sigma_{W_\ell} = \sigma_1\sigma_1\sigma_1\sigma_1\ \sigma_2\sigma_2\sigma_2\sigma_2\ \cdots\ \sigma_k\sigma_k\sigma_k\sigma_k\ \sigma_{i_\ell}.$$

(Step ℓ-4′) Exactly the same as (Step ℓ-4).
(Step ℓ-5′) If $e'_\ell \notin W_\ell$, then the buffer contains $\overline{W_\ell}$ and σ_{j_ℓ}, the color of the edge request which is not removed in (Step ℓ-3′); otherwise (both colors of the edge request are removed in (Step ℓ-3′)), the buffer contains $\overline{W_\ell}$ only. In the former case, the output subsequence is

$$\Sigma_{\overline{W_\ell}} = \sigma_{k+1}\sigma_{k+1}\ \sigma_{k+2}\sigma_{k+2}\ \cdots\ \sigma_{j_\ell}\sigma_{j_\ell}\sigma_{j_\ell}\ \cdots\ \sigma_n\sigma_n,$$

and in the latter case

$$\Sigma_{\overline{W_\ell}} = \sigma_{k+1}\sigma_{k+1}\ \sigma_{k+2}\sigma_{k+2}\ \cdots\ \sigma_n\sigma_n.$$

Similar to the above discussion for S_0, we state several propositions and claims for S_ℓ in the following. Since the proofs are very similar to those for S_0, we will omit the details. Let $Z' = Z_0 - p = (2n+4k+1)p + 2n - k$.

Proposition 5. *$OPT(S_\ell) = Z'$ for any ℓ.* □

Note that the above optimal output is done by choosing a multiset of colors as W_ℓ, that includes k colors and $2k$ requests such that at least one of the k colors is the same as one of the two edge requests in S_ℓ.

Claim 5. *For an output sequence U_ℓ, if $C(U_\ell) = Z'$, then $|\{\Sigma_\ell^1\} \cap \Xi e_\ell| = 2k$ and $\#(\{\Sigma_\ell^1\} \cap \Xi e_\ell) = k$ must be satisfied.* □

The following corollary follows from the above claim.

Corollary 3. *The multiset W_ℓ of colors determined by (Step ℓ-1′) remains in Ξe_ℓ.* □

Moreover, the following Claims 6–8 can be proved by similar discussions as in the proofs of Claims 2–4, respectively. One difference in Claim 7 is that the right hand side of the inequality is one greater than that in Claim 3, which is caused by the difference between the length $4k$ of $B_{\ell,j}^i$'s and the length $4k+1$ of $B_{0,j}^i$'s.

Claim 6. *For an output sequence U_ℓ, if $C(U_\ell) = Z'$, then the output subsequence for $A^{(\ell)}$ must be pairwise.* □

Claim 7. *For an output sequence U_ℓ of S_ℓ, if $C(U_\ell) = Z'$, $|\{\Sigma_\ell^2\} \cap \Xi B_{\ell,1}^2| \le 2n - 2k + 1$ and $\{\Sigma_\ell^2\} \cap \Xi B_{\ell,1}^2 \subseteq \overline{W_\ell}$ must be satisfied.* □

Claim 8. *For an output sequence U_ℓ, if $C(U_\ell) = Z'$, the output subsequence for $B^{(\ell)}$ must be pairwise.* □

Now we are prepared to prove Lemma 2. Let a sequence $S^{(\ell)}$ be defined by $S^{(\ell)} = S_0 S_1 \cdots S_\ell S_{m+1}$ for $1 \le \ell \le m$.

Proposition 6. *For an output sequence $U^{(\ell)}$ of $S^{(\ell)}$ $(1 \le \ell)$, if $C(U^{(\ell)}) = Z_0 + \ell(Z' - (n-k)) + (Z_0 - (n-k))$, then W_0 must contain at least one color corresponding to an edge request for every S_i $(1 \le i \le \ell)$.*

Proof. The proof is by induction on ℓ.

(Base case). We prove that if $C(U^{(1)}) = Z_0 + Z' - (n-k) + (Z_0 - (n-k))$, then W_0 surely contains either e_1 or e_1' by contradiction. Recall that colors of $A^{(0)}$ and $B^{(0)}$ in S_0 appear neither in S_1 nor in S_{m+1}. Then, one can see that if ΞS_1 contains those colors, then we can remove all of them immediately by a pairwise output before putting the head request of S_1 into the buffer, which does not incur any redundant cost. A similar argument can be applied for $A^{(1)}$, $B^{(1)}$, and ΞS_{m+1}.

We prove that $\{\Sigma_0^2\} \cap \Xi S_1 = \overline{W_0}$ if $C(U^{(1)}) = Z_0 + Z' - (n-k) + (Z_0 - (n-k))$: for a while, we consider a subsequence $S' = S_0 S_1$ of $S^{(1)}$ and show that if $OPT(S') = Z_0 + Z' - (n-k)$ is obtained, then $\{\Sigma_0^2\} \cap \Xi S_1 = \overline{W_0}$. First we show that $|\{\Sigma_0^2\} \cap \Xi S_1| = 2n - 2k$ by proving that assumptions $|\{\Sigma_0^2\} \cap \Xi S_1| \le 2n - 2k - 1$ and $|\{\Sigma_0^2\} \cap \Xi S_1| \ge 2n - 2k + 1$ lead us to contradictions. Suppose for contradiction that $|\{\Sigma_0^2\} \cap \Xi S_1| \le 2n - 2k - 1$ in an optimal output. Recall that $OPT(S_0) = Z_0$ from Proposition 4. Then, the number of color blocks is at least $Z_0 - (n-k-1)$ when S_1 is ready, because at least $2k+1$ requests of Σ_0^2 in more than k colors are removed. Since $OPT(S_1) = Z'$ from Proposition 5, the cost must be at least $Z_0 - (n-k-1) + Z' = Z_0 + Z' - (n-k) + 1$, which is a contradiction.

Next suppose that $|\{\Sigma_0^2\} \cap \Xi S_1| = 2n - 2k + q$ for $1 \le q \le 4k+1$. The number of color blocks when S_1 is ready is at least $Z_0 - (n-k+q/2) + p$, because $B^{(0)}$ in S_0 is not removed by a pairwise output due to the lack of the room to store the whole $B_{0,j}^i$'s in the sorting buffer. Again, since $OPT(S_1) = Z'$, the cost must be at least $Z_0 - (n-k+q/2) + p + Z' = Z_0 + Z' - (n-k) + p - q/2$, which is also a contradiction since $q < p$ and $OPT(S') = Z_0 + Z' - (n-k)$. Thus, it holds that $|\{\Sigma_0^2\} \cap \Xi S_1| = 2n - 2k$.

Here we claim that $\{\Sigma_0^2\} \cap \Xi S_1 = \overline{W_0}$ by observing that $\Xi B_{0,1}^2$ contains W_0 and $\{\Sigma_0^2\}$ $(= W_0 \cup \overline{W_0})$ based on the fact $|\{\Sigma_0^2\} \cap \Xi S_1| = 2n - 2k$ and Claim 3. If we remove the colors of two W_0's (four requests for each color) by a quadwise output, then the number of color blocks is k. Otherwise, more than k color blocks are indispensable to decrease the number of requests from $W_0 \cup \{\Sigma_0^2\}$ remaining in the buffer to $2n - 2k$. Therefore, again the number of color blocks by such an output is greater than $Z_0 - (n-k) + OPT(S_1) = Z_0 + Z' - (n-k)$. Furthermore, one can see that the output sequence given by (Step 0–1′)–(Step 0–4′) and (Step 1–1′)–(Step 1–5′) has cost $Z_0 + Z' - (n-k)$, and so, it is the optimal cost of the subsequence S'.

By the above argument, we can assume $\Xi S_1 = \overline{W_0}$ considering that all requests $B^{(0)}$ are removed from the buffer, and the number of color blocks so far when S_1 is ready is $Z_0 - (n-k)$. Hence in order to achieve the cost $Z_0 + Z' - (n-k) + Z_0 - (n-k)$ for the entire sequence $S_0 S_1 S_{m+1}$, we have to process S_1 at the cost $OPT(S_1) - (n-k)$ (before S_{m+1} is ready), and then process S_{m+1} at the cost $OPT(S_{m+1})$ using the buffer initially filled with some requests of S_0 and S_1. It follows that we need to first output two $\overline{W_0}$'s from ΞS_1 and Σ_1^1 by a quadwise output. Otherwise, $|\{\Sigma_1^1\} \cap \Xi A_{1,1}^1| \ge 2k+1$, which does not meet the condition on Claim 6, or more than $n-k$ color blocks are produced to store the whole $A_{1,1}^1$ in the buffer, which does not meet the condition to achieve the cost Z' for S_1 from Proposition 5. Similarly, it can be shown that two $\overline{W_0}$'s from ΞS_{m+1} and Σ_{m+1}^1 must be removed by a quadwise output, which leads us to the proposition.

Then the last thing to show is that W_0 contains at least one of e_1 and e_1'. By the above argument, we assume that whole $A^{(1)}$ is output and the set of remaining requests in the buffer is W_0 only, i.e., $\Xi e_1 e_1' = W_0$. Since $|\Xi e_1 e_1' \cup \{e_1, e_1'\} \cup \{\Sigma_1^2\}| = 2n + 2k + 2$ with n colors, we need to remove at least $4k+1$ requests from the buffer before $B_{1,1}^1$ is ready, in order to process

$B^{(1)}$ in S_1 by a pairwise output based on Claim 8. Moreover, to achieve the cost $OPT(S_1) - (n-k)$ for S_1 before S_{m+1} is ready, only k color blocks are allowed for this removal of $4k+1$ requests. Such an output can be done only if there are five requests in one color, and at least $4(k-1)$ requests in $k-1$ colors, that is, W_0 must contain $c(e_1)$ or $c(e_1')$ (or both). [End of Base case.]

(Induction Step). As shown in Lemma 1, if W_0 contains at least one color corresponding to at least one edge request for every S_i, then the output sequence such that the cost is equal to $Z_0 + m(Z' - (n-k)) + (Z_0 - (n-k))$ can be obtained by applying (Step 0–$1'$)–(Step 0–$4'$) for S_0, (Step ℓ-1)–(Step ℓ-$4'$) for every S_i $(1 \le i \le \ell)$, and then (Step 0–$1'$)–(Step 0–$5'$) for S_{m+1}. In this case, $W_0 = W_i$ holds for $1 \le i \le m+1$, and the obtained sequence has the optimal cost. We call this type of the output sequences is *canonical*. Also we say that a subsequence is canonical if it is a part of a canonical sequence.

What we would like to prove here is that non-canonical output sequences cannot achieve the optimal cost. Suppose that $S^{(\ell)}$ satisfies the proposition and an optimal output sequence is denoted by U^*. We partition U^* into two subsequences such that $U^* = U^*_\ell U^*_{m+1}$, where U^*_ℓ is a subsequence before when S_{m+1} is ready. Then we consider $S^{(\ell+1)}$ and an optimal output sequence U. The sequence U is also partitioned into three subsequences such that $U = U_\ell U_{\ell+1} U_{m+1}$, where U_ℓ is the subsequence before when $S_{\ell+1}$ is ready, and $U_{\ell+1}$ is one between the time when $S_{\ell+1}$ is ready and the time when S_{m+1} is ready. We will prove that if U is not canonical, then its cost is greater than the desired value $Z_0 + (\ell+1)(Z' - (n-k)) + (Z_0 - (n-k))$, by contradiction.

There are two cases to consider: (Case 1) $U_\ell = U^*_\ell$ and (Case 2) $U_\ell \ne U^*_\ell$.

(*Case* 1). When $S_{\ell+1}$ is ready, there remain $\overline{W_\ell}$ $(=\overline{W_0})$ including $2n-2k$ requests in $n-k$ colors in the sorting buffer. To achieve the optimal cost, we need to achieve the optimal costs for subsequences $S_{\ell+1}$ and S_{m+1}. The only way to process is outputting $4n-4k$ requests in at most $n-k$ colors, i.e., $\overline{W_\ell}$ (two requests of $\overline{W_\ell}$ and two requests of $\Sigma^1_{\ell+1}$ in each color) must be removed from the buffer. This determines $W_{\ell+1}$ such that $W_{\ell+1} = W_0$. Then in $S_{\ell+1}$, there must be at least one edge request whose color is included in $W_{\ell+1}(=W_0)$; otherwise, the optimal cost cannot be obtained. Note here that there is a possibility to remove both of the colors of the edge request of $S_{\ell+1}$ and then an "extra" free space of size one is made in the buffer when $B^{(\ell)}$ is ready and also when S_{m+1} is ready. This free space can store one request in $\Sigma^2_{\ell+1}$ (or, $W_{\ell+1}$), or one request in $B^{(0)}$. However this fact does not contribute to reduce the optimal cost for S_{m+1}: first, $B^{(\ell+1)}$ must be removed by a pairwise output to have the optimal cost from Claim 8. Then even if one request σ in $\Sigma^2_{\ell+1}$ (or, $W_{\ell+1}$) remains in the buffer, another request σ' in $\Sigma^2_{\ell+1}$ having the same color $c(\sigma)$ is already removed from the buffer, and thus wastefully increases the cost by one. This contradicts the assumption that U is optimal. Thus in this case we observe that $U_{\ell+1}$ is canonical. Since $U_{\ell+1}$ is canonical and so is $U_\ell U_{\ell+1}$, U_{m+1} is also canonical to achieve the optimal cost, by which we conclude U is canonical. As a result, W_0 must contain at least one color corresponding to an edge request for every $S_i (1 \le i \le \ell)$. [End of (Case 1).]

(*Case* 2). As in the above discussion for (Case 1), there may be a chance not to follow the rules determined by (Step ℓ-$1'$)–(Step ℓ-$5'$) by storing some request in the possible extra free space of the buffer.

Let i be the smallest index such that the extra free space appears, namely, both of the colors of the edge request from S_i are included in W_i. Similar to (Case 1), putting a request of $B^{(i)}$ into the free space does not reduce the total cost. Hence, we assume in the following that a request δ from Σ^2_i (also from W_i) is stored in the free space, that is, $\Xi B^{(i)} \supseteq \overline{W_i} \cup \{\delta\}$ and so $\Xi S_{i+1} \supseteq \overline{W_i} \cup \{\delta\}$. Again, note here that the other request in Σ^2_i which has the same color $c(\delta)$ is already removed from the buffer and consumes one color block.

We will see that we cannot change $\overline{W_{i+1}}$ from $\overline{W_i}$ without any penalty cost, i.e., W_{i+1} is identical to W_i. To obtain the optimal cost $Z' - (n-k)$ for S_{i+1} (before S_{i+2} is ready), we must remove $4n-4k+1$ requests in at most $n-k$ colors from $\overline{W_i} \cup \{\delta\} \cup \Sigma^1_{i+1}$, and then keep $2k$ requests in the buffer. (Otherwise, i.e., if there remain at most $2n$ spaces in the buffer, then $A^{(i+1)}$ of length $2n+1$ cannot be removed by a pairwise output from Claim 6).

If δ and all of the requests having the same color $c(\delta)$ are removed at this time, the cost increases. The reason is that the number of such requests is only three and also at most four requests in every color exist in $\overline{W_i} \cup \{\delta\} \cup \Sigma^1_{i+1}$, that is, if we remove $4n-4k+1$ requests from the buffer and keep $2k$ requests in the buffer, then it produces at least $n-k+1$ color blocks. Thus, even if the rest of the output sequence has the optimal cost by this change, the total cost is greater than the optimal value. When we decide to keep one or two requests of color $c(\delta)$ in the buffer before when S_{i+1} is ready, the others (two requests or one request, resp.) having color $c(\delta)$ is removed at this time and consumes one color block. The last choice is to keep those three requests of color $c(\delta)$ until when S_{i+1} is ready. There are further two cases at (Step $(i+1)$-$0'$): (i) some of the three requests are included in W_{i+1}, and (ii) all the three requests are removed together with requests of the same color in Σ^1_{i+1}.

For the case (i), if we include in W_{i+1} all the three requests of color $c(\delta)$, then at least $n-k+1$ color blocks are produced at the step. The reason is that the three requests of color $c(\delta)$ and at least one color $c'(\ne c(\delta))$ in W_{i+1} has only one request in the buffer, which implies that the other request having color c' is already removed. If we include in W_{i+1} one or two requests of color $c(\delta)$, then we must remove at least one request of color $c(\delta)$ and waste a color block. Hence, keeping the extra space empty is better. For the case (ii), recall that we need $4n-4k+1$ requests in $n-k$ colors from $\overline{W_i} \cup \{\delta\} \cup \Sigma^1_{i+1}$ to achieve the optimal cost. This can be done only when every removed color has at least four requests, while currently the number of the requests of color $c(\delta)$ is three. Hence, again, keeping the extra space empty is better.

Therefore, the sequences considered in (Case 2) have cost greater than the optimal one and it contradicts the assumption that U is optimal. [End of (Case 2).]

From the discussion for (Case 2), if $U = U_\ell U_{\ell+1} U_{m+1}$ is optimal, then U can differ from canonical ones only for $U_{\ell+1}$ and U_{m+1}. However, as seen above in (Case 1), the difference for $U_{\ell+1}$ does not reduce the cost either, by which we conclude that if the total cost of the sequence is the desired value, the sequence must be canonical and so it holds that $W_0 = W_i$ for all i. It is easy to observe that a canonical sequence has the optimal cost only if W_0 contains at least one color of the edge request for every S_i, because otherwise, i.e., neither two colors of the edge request, e.g., from S_j are included in $W_j (=W_0)$, the cost for S_j is greater than the optimal value. □

If there is an output sequence $S_{\pi^{-1}}$ such that $C(S_{\pi^{-1}}) = C(U^{(m)}) = Z$, then there is W_0 which contains at least one color of two edge requests for every S_i from Proposition 6. This ensures that the corresponding subset V_0 of vertices includes at least one vertex of each edge and $|V_0| = k$, i.e., V_0 must be a vertex cover.

4. Conclusion

We have shown the NP-hardness of the sorting buffer problem on the uniform metric. However, our proof does not give any inapproximability result, e.g., APX-hardness.

Acknowledgments

The authors would like to thank Anna Adamaszek for pointing out a flaw in the preliminary version of this paper, and anonymous referees for their very helpful comments that improved the presentation of this paper. This work is partially supported by Grant-in-Aid for Scientific Research (KAKENHI), 16092223, 18700015, 20500017, 22700019, and 23500020.

References

[1] A. Adamaszek, A. Czumaj, M. Englert, H. Räcke, Almost tight bounds for reordering buffer management, in: Proc. STOC 2011, 2011, pp. 607–616.
[2] Y. Asahiro, K. Kawahara, E. Miyano, NP-hardness of the sorting buffer problem on the uniform metric, in: Proc. the 2008 International Conference on Foundations of Computer Science, FCS08, 2008, pp. 137–143.
[3] N. Avigdor-Elgrabli, Y. Rabani, An improved competitive algorithm for reordering buffer management, in: Proc. SODA 2010, 2010, pp. 13–21.
[4] R. Bar-Yehuda, J. Laserson, Exploiting locality: approximating sorting buffers, J. Discrete Algorithms 5 (4) (2007) 729–738.
[5] H.-L. Chan, N. Megow, R. van Stee, R. Sitters, The sorting buffer problem is NP-hard, CoRR http://arxiv.org/abs/1009.4355 [abs/1009.4355], 2010.
[6] M. Englert, H. Räcke, M. Westermann, Reordering buffers for general metric spaces, Theory Comput. 6 (1) (2010) 27–46.
[7] M. Englert, M. Westermann, Reordering buffer management for non-uniform cost models, in: Proc. ICALP'05, 2005, pp. 627–638.
[8] I. Gamzu, D. Segev, Improved online algorithms for the sorting buffer problem on line metric, ACM Trans. Algorithms 6 (1) (2009) 13. Article 15.
[9] M.J. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman, 1979.
[10] R. Khandekar, V. Pandit, Online and offline algorithms for the sorting buffers problem on the line metric, J. Discrete Algorithms 8 (1) (2010) 24–35.
[11] J.S. Kohrt, K. Pruhs, A constant approximation algorithm for sorting buffers, in: Proc. LATIN'04, 2004, pp. 193–202.
[12] H. Räcke, C. Sohler, M. Westermann, Online scheduling for sorting buffers, in: Proc. ESA'02, 2002, pp. 820–832.