MPMD on Two Sources with Lookahead

Enze Sun[1], Bo Wang[1], Quan Xue[1], Mengshi Zhao[1], and Zixuan Zhu[1]

The University of Hong Kong, Hong Kong {sunenze,iswangbo, u3008434,zmsxsl,zhuzixua}@connect.hku.hk

Abstract. The min-cost perfect matching with delays problem on two sources (2-MPMD) captures the critical trade-off between match and wait, and has been extensively studied recently. A 3-competitive deterministic algorithm and a 2-competitive randomized algorithm were proposed, where 2 is also the lower bound. In this paper, we consider a variation where an algorithm is allowed to look ahead for certain time, exploring the power of future knowledge in online decisions. We devise a $\frac{3+\tau}{1+\tau}$-competitive deterministic algorithm and a randomized algorithm which is $\big(2-(2\sqrt{2}-2)\tau\big)$-competitive when $\tau \leq \frac{1}{2}$ and $\big(2+2\tau-\sqrt{4\tau^2+4\tau-1}\big)$-competitive otherwise, where τ is the length of the foreseen time period.

Keywords: online algorithm with lookahead · randomized algorithm · competitive analysis.

1 Introduction

The increasing popularity of online gaming platforms expands the demands of efficient online matching system. Consider the two-player games such as Chess, Hearthstone, or Asphalt 8. With online arrival of the players, the gaming platform tries to match each player with a suitable component to initiate a new game between them, and maintain short waiting times. These two desired criteria, though, are often conflicting, is it possible to strike for an optimal trade-off between match and wait?

Emek et al. [11] first formalized this challenge in terms of the min-cost perfect matching with delays (MPMD) problem. In this problem, n sources are given as points on a metric space. Requests arrive in a continuous time online fashion from the sources and should be served by matching with each other. The algorithm is allowed to delay the matching commitments, but at a cost. The total cost of the algorithm is the sum of metric distances between the matched pairs, plus the sum of times that each request waits from its arrival until being matched. When the metric space degenerates into two sources, we come to the min-cost perfect matching with delays on two sources (2-MPDP) problem, for which a 3-competitive deterministic algorithm [12] and a 2-competitive randomized algorithm [13] were proposed. Though both ratios are tight for deterministic and randomized algorithms on 2-MPDP respectively, it is natural to ask whether better performance can be achieved with more information in hand.

Online optimization with lookahead deals with sequential decision making under incomplete information where each decision must be made based on a

limited, but certain preview (lookahead) of future input data. In this paper, we study 2-MPDP with lookahead, where at time t, the requests during the time period $[t, t+\tau]$ are revealed to the algorithm. This variation provides a better description of a decision maker's informational state since not all may be known about the future, but also not nothing. Specifically for the online gaming platforms, based on the number of online players and the number of players that are currently in a game, the platform can predict the requests in the coming period accurately. This is because that a player who just logs in and a player who is going to finish a game will have certain probability to request for a new game soon. This probability and the buffer time can be learnt from historical behaviors. With information on future requests, the uncertainties are reduced significantly, and a better performance is promising. The main results of this paper include a $\frac{3+\tau}{1+\tau}$-competitive deterministic algorithm and a randomized algorithm which is $\big(2-(2\sqrt{2}-2)\tau\big)$-competitive when $\tau \leq \frac{1}{2}$ and $\big(2+2\tau-\sqrt{4\tau^2+4\tau-1}\big)$-competitive otherwise, where τ is the length of the foreseen time period. When $\tau = 0$, i.e., foresight is not allowed, those ratios match the known lower bounds in [12,13].

1.1 Related Work

The match-or-wait feature is fundamental to online decisions and thus, prominent in the study of online computations. Representative online problems encoding this difficulty include ski rental problem [16,17] and TCP knowledgement [6,7,15]. In particular, for the ski rental problem, the best deterministic algorithm is the $(2-\frac{1}{b})$-competitive break-even algorithm [17], where b is the ratio between buying and renting costs. On the other hand, there exists a $\frac{e}{e-1}$-competitive randomized algorithm, which is memoryless and strictly better than the best deterministic algorithm [16].

MPMD problem was first introduced by Emek et al. [11]. They presented an online randomized algorithm with competitive ratio $O(\log^2 n + \log \Delta)$, where n is the number of points in the metric space and Δ is the aspect ratio. Azar et al. [3] later devised another online algorithm with an improved logarithmic competitive ratio $O(\log n)$. They also provided a lower bound by showing that no online MPDP algorithm can have competitive ratio better than $\Omega(\sqrt{\log n})$. Ashlagi et al. [2] improved this lower bound to $\Omega(\frac{\log n}{\log \log n})$. They also showed that the algorithms of [11] and [3] can be modified to treat the bipartite version of the MPMD problem, where they obtained an upper bound $O(\log n)$ and a lower bound $\Omega(\sqrt{\frac{\log n}{\log \log n}})$. Azar et al. [4] devised the first deterministic algorithm achieving a sub-linear competitive ratio, which is $O(n^{\log(\frac{3}{2}+\epsilon)})$-competitive for any fixed ϵ.

For the special case of 2-MPMD problem, Emek et al. [12] presented a deterministic variant of the algorithm in [11], which is 3-competitive. They also proved that any deterministic online algorithm has a competitive ratio of at least 3. Recently, He et al. [13] showed that, unlike ski rental problem, memoryless randomized algorithms for 2-MPMD cannot do better than 3-competitive.

They devised a 2-competitive randomized algorithm and showed that no other randomized algorithm for 2-MPMD can further reduce the competitive ratio.

The concept of "lookahead" in online algorithms was formally introduced by Dunke et al. [8]. They proposed the method of distributional performance analysis to online algorithms endowed with various degrees of information preview and performed an exact analysis in basic settings of the ski rental, bin packing and traveling salesman problem. Later, Dunke et al. [9] provided a general modelling approach to online optimization with lookahead. Online problems with lookahead have also been widely studied in concrete models including online planning[10], online scheduling [18,19,14,5] and online paging [1].

1.2 Paper Organization

The rest of this paper is organized as follows. We formally introduce the model and a special class of requests sequences called paired sequences in Section 2. In Section 3, we characterize the structure of optimal solutions and present a deterministic online algorithm on paired sequences. In Section 4, we provide a randomized online algorithm on paired sequences, whose missing proofs are delayed to Appendix A. The generalization of the deterministic and the randomized algorithms from paired sequences to general sequences are provided in Appendix B.

2 Preliminaries

2.1 2-MPMD with Lookahead

In an instance of 2-MPMD with lookahead problem, there are two sources a and b with a unit distance, and a request sequence R. Each request comes from one source, denoted as $r \in \{a, b\}$. The requests in R arrive in a continuous-time online fashion so that each request $r \in R$ is unknown until its arrival.

Assume there is an even number of requests, say, $|R| = 2n$. Denote the set of sources as $S = \{a, b\}$. Then each request $r_i = (t_i, s_i) \in \mathbb{R}_+ \times S$ comes at time t_i from source s_i, where $i \in [2n]$. We need to match all the requests into pairs, so the result should be a perfect matching of R. At time t, the algorithm is able to foresee the requests in time period $[t, t+\tau]$ and delay the matching commitment for any request.

Given two request $r_1, r_2 \in R$, a match between them is formally defined by a tuple $M = (r_1, r_2, t) \in R \times R \times \mathbb{R}_+$, where $t \geq \max\{t_1, t_2\}$ is the time that r_1 and r_2 are matched with each other. The matching cost for a match M comes from two parts: delay cost and distance cost. The delay costs incurred by $M = (r_1, r_2, t)$ on r_1 and r_2 are $t - t_1$ and $t - t_2$ respectively. If requests r_1 and r_2 originate from different sources, the distance cost for each request is $\frac{1}{2}$, and 0 otherwise. This is because the unit distance between the two sources is equally charged on r_1 and r_2. The matching cost of M, denoted as $\text{cost}(M)$, is defined to be the sum of its delay costs and space costs of the involved requests.

2.2 Competitive Ratio

Given any request sequence R, an online algorithm $\mathcal{A}$ produces a set of matches, denoted as $\mathcal{M}_{\mathcal{A}}^{R}$. Then the total cost incurred by algorithm $\mathcal{A}$ on R is defined to be $\text{cost}_{\mathcal{A}}^{R} = \sum_{M \in \mathcal{M}_{\mathcal{A}}^{R}} \text{cost}(M)$. The cost ratio of algorithm $\mathcal{A}$ on request sequence R is defined to be the ratio of $\text{cost}_{\mathcal{A}}^{R}$ to $\text{cost}_{\mathcal{A}^*}^{R}$, where $\mathcal{A}^*$ is an offline optimal algorithm, who knows the whole request sequence in advance and minimizes the total cost.

A deterministic algorithm $\mathcal{A}$ is *α-competitive* if $\text{cost}_{\mathcal{A}}^{R} \leq \alpha \cdot \text{cost}_{\mathcal{A}^*}^{R}$ for any request sequence R; a randomized algorithm $\mathcal{A}$ is *α-competitive* if $\mathbb{E}[\text{cost}_{\mathcal{A}}^{R}] \leq \alpha \cdot \text{cost}_{\mathcal{A}^*}^{R}$ for any request sequence R, where the expectation is taken over the internal randomness of the algorithm $\mathcal{A}$.

2.3 Smart Algorithm

Smart Algorithm. We say that an algorithm is *smart* if it matches a new request with an open request from the same source immediately whenever there is any.

Note that during the execution of a smart algorithm, there will be at most one open request at a single source. The following lemma further indicates that we can focus on the smart algorithm for further investigation.

Lemma 1. *([12]) Any algorithm can be transformed to a smart algorithm using an online method without increasing the total matching cost.*

2.4 Paired Sequences

Inspired by [12,13], we first restrict our focus to a special class of input sequences. Then, we will generalize our smart algorithms 2 and 3 from restricted input sequences into arbitrary input sequences, with the same competitive ratios.

Paired Sequence A request sequence $R = \{r_i = (s_i, t_i)\}_{1 \leq i \leq 2n}$ with $t_1 \leq t_2 \leq ... \leq t_{2n}$ is a *paired sequence* if $t_{2k} = t_{2k-1}$ and $s_{2k} \neq s_{2k-1}$ for all $k = 1, 2, ..., n$.

Note that in a smart algorithm, if a pair of requests arrive simultaneously from the same source, they will be matched immediately without a cost. Therefore, it suffices to only consider the pairs with two requests from different sources. From now on, we assume that the given request sequence is a paired sequence, unless further clarification.

A paired sequence can be characterized by a sequence of request pairs $\{p_i = (r_{2i}, r_{2i-1}, t_i)\}_{1 \leq i \leq n}$. The states for paired requests are defined as follows: A pair is said to be *foreseen* if its requests have not arrived but have been foreseen by the algorithm, *open* if its requests have already arrived but have not been matched, and *matched* otherwise.

In a smart algorithm $\mathcal{A}$, two requests in a pair must be matched at the same time. At time t, $\mathcal{A}$ is said to be *waiting* if there is an open pair; and *idle* otherwise. When $\mathcal{A}$ is waiting, it can either match the current open pair internally by matching its two requests at a distance cost, or wait at a delay

cost until a new pair arrives and match these two pairs externally, such that requests from the same source can be matched to avoid a distance cost. Now, the difficulty of balancing the trade-off between wait and match is translated into finding a proper matching time to alternate $\mathcal{A}$ from waiting to idle.

3 Deterministic Algorithm

In this section, we introduce a deterministic algorithm for 2-MPMP with lookahead. In Section 3.1, we introduce a Continuous Phase Identification (CPI) subroutine and relate the cost of an offline optimal solution to the length of phases. In Section 3.2, we propose a Deterministic State-based Algorithm with Lookahead (DSAL) for paired sequences based on CPI subroutine and show that it is $\frac{3+\tau}{1+\tau}$-competitive. In Section 3.3, we generalize DSAL for all request sequences with the same competitive guarantee.

3.1 State-based Phases Decomposition

A Phase Identification (PI) subroutine was introduced in [13] to divide R into several independent phases. However, this PI subroutine only assigns state values to pairs, which cannot fully encode the foreseen information. Suppose at time t, the foreseen period $(t, t+\tau]$ does not have any arriving pair, then no state value will be assigned to this period and the foreseen information is wasted. In order to make full use of lookahead, state values have to be defined more finely. Here, we introduce a continuous *state function* $S(t)$ to expand the definition of state value to encompass the entire time horizon. The original PI is generalized into a Continuous Phase Identification (CPI) subroutine, which divides $[t_1, t_n+1]$ into a set of disjoint intervals $\mathcal{P} = \{P_1, P_2, ..., P_k\}$ and maintains a state function $S(t)$, along with a variable trend(t) that can take on the values of *decreasing* or *increasing* at each time $t \in \bigcup_i P_i$.

CPI starts at t_1 and ends before $t_{n+1} = t_n + 1$, where t_1 and t_n are the arrival times of p_1 and p_n respectively. Initially, there is only one phase P_1 containing the first pair p_1, with $S(t_1) = 1$ and trend$(t) = \textit{decreasing}$. Subsequently, $S(t)$ and trend(t) are updated in an inductive manner. For any time $t \in (t_k, t_{k+1})$, trend(t) = trend(t_k) and

$$S(t) = \min(S(t_k) - (t - t_k), 0). \tag{1}$$

Once $S(t) = 0$ for some $t \in (t_k, t_{k+1})$, CPI concludes the current phase and starts a new phase from the arrival of next pair p_{k+1} with $S(t_{k+1}) = 1$ and trend$(t_{k+1}) = \textit{decreasing}$. Otherwise, we stay in the same phase with trend(t_{k+1}) flipped and $S(t_{k+1}) = 1 - S(t_k) + (t_{k+1} - t_k)$.

Let $-\text{trend}(t)$ be the flipped value of trend(t). The following Algorithm 1 implements this CPI subroutine. After CPI, each phase $[t_{start}, t_{end}]$ begins with $S(t_{start}) = 1$ and ends with $S(t_{end}) = 0$. Our CPI and the PI in [13] are similar in the sense that at each time t_i when a pair p_i arrives, our $S(t_i)$ equals to

Algorithm 1 Continuous Phase Identification (CPI)

$t_{start} = t_1$, $t_{end} = 0$, $\mathcal{P} = \emptyset$, $\text{trend}(t_1) = \textit{decreasing}$, $k = 1$
while $k < n$ **do**
  for $t \in [t_k, t_{k+1})$ **do**
    $trend(t) \leftarrow \text{trend}(t_k)$
    $S(t) \leftarrow \min(S(t_k) - (t - t_k), 0)$
    if $t \neq t_{start}$ and $S(t) = 0$. **then**
      $t_{end} \leftarrow t$,$t_{start} \leftarrow t_{k+1}$, $S(t_{start}) \leftarrow 1$,
      $\mathcal{P} \leftarrow \mathcal{P} \cup \{[t_{start}, t_{end}]\}$, $\text{trend}(t_{start}) \leftarrow \textit{decreasing}$
      Break out the *for* loop
    end if
  end for
  $S(t_{k+1}) \leftarrow 1 - S(t_k) + (t_{k+1} - t_k)$, $\text{trend}(t_{k+1}) \leftarrow -\text{trend}(t_k)$, $k \leftarrow k + 1$.
end while

their state value, and for each phase $P = [t_{start}, t_{end}]$ generated by CPI, the pairs arrive during P form a phase f in their PI. Nevertheless, we have an extra "trend" variable to identify the future trend of the state value a certain interval, as well as complete state values at each time. Additionally, instead of dividing R, we divide the whole time horizon to identify each phase more precisely. Consider two consecutive requests p_k and p_{k+1}, the interval $[t_k, t_{k+1})$ is an *increasing interval* if $\text{trend}(t_k) = \textit{increasing}$, and is a *decreasing interval* otherwise. Let $\mathcal{I}_{inc}$ be the union of all the increasing intervals, and $\mathcal{I}_{dec}$ be the union of all the decreasing intervals. Given a time interval $I = [t_0, t_1]$, we define the length of all increasing intervals in I as $U(I) := |I \cap \mathcal{I}_{inc}|$, and the total length of all decreasing intervals in I as $D(I) := |I \cap \mathcal{I}_{dec}|$. The following lemma characterizes the cost of an offline optimal solution $\mathcal{A}^*$ using the length of phases.

Lemma 2. *[13] For each phase $P = [t_{start}, t_{end}]$, the cost of an optimal solution $\mathcal{A}^*$ on P is the length of P, i.e. $\text{cost}_{\mathcal{A}^*}(P) = |P|$. And the cost of the optimal solution for the whole sequence is the sum of the cost for each phase.*

3.2 Deterministic Algorithm

First note that the CPI subroutine can be implemented in an online manner. As time goes by, CPI calculates $S(t)$ and $\text{trend}(t)$ inductively. When a certain moment is reached, CPI concludes the current phase and waits for the next arriving pair to start a new phase. With the help of this subroutine, at each time t, the algorithm can tell whether t belongs to the current phase, or t does not belong to any phase. Further if t belongs to the current phase, the algorithm can foresee whether this phase will end before $t + \tau$, together with the state values and the trends during $[t, t+\tau]$. The ending time of each phase is set to be a reasonable cut-off time to turn an algorithm from waiting to idle. Therefore, DSAL will deal with each phase separately so that all the pairs in the same phase are matched before this phase ends. At a non-phase time, DSAL is simply idle and will do nothing. The algorithm is outlined in Algorithm 2, which proceeds

the whole time horizon phase by phase. The key idea behind DSAL is to make decisions by evaluating the potential future profit, ensuring that each phase is handled at a low cost. This is done by comparing the current state value with the state value of future time. The algorithm utilizes the following parameters when making decisions.

Definition 1. *Suppose $P = [t_{start}, t_{end}]$ is a phase. Given two different times t_0, t_1 such that $t_{start} \leq t_0 \leq t_1 \leq t_{end}$. The profit function $\Delta(t_0, t_1)$ is defined as*

$$\Delta(t_0, t_1) = \begin{cases} S(t_0) - S(t_1) & \text{if } \mathrm{trend}(t_0) = \mathrm{trend}(t_1) \\ S(t_0) - (1 - S(t_1)) & \text{if } \mathrm{trend}(t_0) \neq \mathrm{trend}(t_1). \end{cases}$$

Let $\mathrm{open}(t)$ and $\mathrm{foreseen}(t)$ denote the number of open pairs and foreseen pairs at time t respectively. DSAL decides to match internally or externally based on the parity of $\mathrm{open}(t)$ and $\mathrm{foreseen}(t)$. Note that a phase not longer than τ is fully foreseen by the algorithm, hence DSAL can handle it optimally by following an offline optimal algorithm, e.g. TOPI [1].

Algorithm 2 Deterministic State-based Algorithm with Lookahead (DSAL)

```
for each phase P = [t_start, t_end] do
    if |P| ≤ τ then Match as an optimal offline algorithm does
    else
        Match externally once there are 2 open pairs until t = t_end − τ
        Waiting until open(t) = 1
        if foreseen(t) is even then
            Match internally at once
        end if
        Match externally once there are 2 open pairs until t = t_end
    end if
end for
```

Theorem 1. *For any paired sequence R, the competitive ratio of DSAL is $\left(\frac{3+\tau}{1+\tau}\right)$ for 2-MPMD problem with foresight τ.*

Proof. The competitive ratio is proved on each phase P. We start with a trivial case when $|P| \leq \tau$. Then DSAL works optimally as all the information are foreseen. Otherwise, if $|P| > \tau$, by the definition of increasing and decreasing intervals, at any time t such that $t_{start} \leq t \leq t_{end}$, we have $U([t_{start}, t]) + D([t_{start}, t]) = t - t_{start}$ and $D([t_{start}, t]) - U([t_{start}, t]) = \Delta(t_{start}, t)$. If the total number of requests in P is even, $\Delta(t_{start}, t_{end}) = 0$ and only on the decreasing intervals will a pair be open. Hence the cost of externally matching all pairs is exactly $|P|$, which is also optimal. Therefore, it remains to consider phases P's such that $|P| > \tau$ and the total number of requests in P is odd.

[1] The Optimal Phase Identification Algorithm proposed in [13]

Let t be the earliest time after $t^* = t_{end} - \tau$ such that $\text{open}(t) = 1$. This t is also the only time when DSAL matches internally on P. Let $I_1 = [t_{start}, t)$, $I_2 = [t, t_{end}]$. Then the waiting cost on I_1 and I_2 are

$$2D(I_1) = t - t_{start} + \Delta(t_{start}, t), \ \textit{and} \ \ 2U(I_2) = t_{end} - t - \Delta(t, t_{end}).$$

Summing over two intervals plus the cost of an internal match, we have

$$\begin{aligned}\text{cost}_{\mathcal{D}}(P) =& t - t_{start} + \Delta(t_{start}, t) + t_{end} - t - \Delta(t, t_{end}) + 1\\ =& t_{end} - t_{start} + 2\Delta(t_{start}, t) - \Delta(t_{start}, t_{end}) + 1\\ =& t_{end} - t_{start} + 2\Delta(t_{start}, t),\end{aligned}$$

where the last line comes from the fact that $\Delta(t_{start}, t_{end}) = 1$ as the total number of requests is odd.

The trend on $[t^*, t)$ is always *increasing* as there is no open pair. Therefore, $\Delta(t_{start}, t) \le \Delta(t_{start}, t^*)$. Then the total cost of DSAL on P is bounded by

$$\text{cost}_{\mathcal{D}}(P) \le t_{end} - t_{start} + 2\Delta(t_{start}, t^*) = t^* - t_{start} + 2\Delta(t_{start}, t^*) + \tau.$$

On the other hand, $\text{cost}_{\mathcal{A}^*}(P) = |P| = t^* + \tau - t_{start}$ by Lemma 2. Hence the competitive ratio on P is bounded by

$$\alpha = \frac{\text{cost}_{\mathcal{D}}(P)}{\text{cost}_{\mathcal{A}^*}(P)} \le \frac{t^* - t_{start} + 2\Delta(t_{start}, t^*) + \tau}{t^* - t_{start} + \tau} = 1 + \frac{2\Delta(t_{start}, t^*)}{t^* - t_{start} + \tau}.$$

Further by the facts that $t^* - t_{start} \ge \Delta(t_{start}, t^*)$ and $\Delta(t_{start}, t^*) \le 1$, we obtain

$$\alpha \le 1 + \frac{2\Delta(t_{start}, t^*)}{\Delta(t_{start}, t^*) + \tau} \le 3 - \frac{2\tau}{1+\tau} = \frac{3+\tau}{1+\tau}.$$

3.3 Deterministic Algorithm for Generalized Sequences

For general request sequences, we will construct a modified CPI subroutine, based on which we can further construct a modified deterministic algorithm with the same competitive guarantee. The construction highly depends on the following fact: Suppose $R = \{r_i = (s_i, t_i)\}_{i \in [2n]}$ is a generalized request sequence. On all intervals of the form (t_{2i-1}, t_{2i}), $i \in [n]$, any smart algorithm has to pay the same cost. We will set the state value on (t_{2i-1}, t_{2i}) to be $S(t_{2i-1})$, and assign a new trend *waiting* for this interval. The state value and trend for other intervals will be calculated in the same way as the pair input. The details of the construction are delayed to Appendix B.

Corollary 1. *There is a* $\left(\frac{3+\tau}{1+\tau}\right)$*-competitive deterministic online algorithm for the 2-MPMD problem with foresight* τ *for any request sequence.*

4 Randomized Algorithm

In this section, we present the Randomized State-based Algorithm with Lookahead (RSAL) for 2-MPMD with prediction. The algorithm is outlined in Algorithm 3. Similar to DSAL, RSAL also processes phase by phase. However, RSAL takes more lookahead information into account instead of only using the foresight on the last τ time of a phase.

Algorithm 3 Randomized State-based Algorithm with Lookahead (RSAL)

for Phase $P = [t_{start}, t_{end}]$ **do**
  $t = t_{start}$
  while $t \leq t_{end}$ **do**
    if $\exists t' \in [t, t+\tau]$ such that $\Delta(t, t') = 0$ **then** $t^* = t'$
    else $t^* = \min(t+\tau, t_{end})$
    end if
    if $open(t) = 1$ **then**
      Internal match the open pair with probability $\frac{\Delta(t,t^*)}{S(t)}$
    end if
    External match all the pairs until t^*, $t = t^*$
  end while
end for

Theorem 2. *For any paired sequence R, the competitive ratio of RSAL is $2 - (2\sqrt{2}-2)\tau$ when $\tau \leq \frac{1}{2}$, and $2 + 2\tau - \sqrt{4\tau^2 + 4\tau - 1}$ when $\tau > \frac{1}{2}$ for 2-MPMD problem with foresight τ.*

Similar to the proof of Theorem 1, we analyze the performance of RSAL on each phase independently. However, RSAL performances differently on different time intervals even in the same phase, which inspires the necessity of classifying the intervals into certain types. During the implementation of RSAL, the t^*'s divide each phase into consecutive time intervals, where each time interval belongs to one of the following three categories:

- **Lossless Interval:** $I_l = [t_0, t_1)$ is a lossless interval if $\Delta(t_0, t_1) = 0$;
- **Critical Interval:** $I_c = [t_0, t_1)$ is a critical interval if $t_1 - t_0 = \tau$ and $\forall t \in [t_0, t_1]$, $\Delta(t_0, t_1) > 0$;
- **End interval:** $I_e = [t_0, t_1]$ is an end interval if $t_1 = t_{end}$ and $\forall t \in [t_0, t_1]$, $\Delta(t_0, t_1) > 0$.

We first give the costs of RSAL on these three types of intervals respectively, which is a generalized version of Lemma 3 in [13]. The proofs of the following lemmas are delayed to Appendix A.

Lemma 3. *If $I_l = [t_0, t_1)$ is a lossless interval, and the probability that there is an open pair at t_0 is $S(t_0)$, i.e. $\Pr[open(t_0) = 1] = S(t_0)$, then the cost of RSAL in this lossless interval is*

$$\text{cost}_{\mathcal{R}}(I_l) = S(t_1)(1 - S(t_1)) - S(t_0)(1 - S(t_0)) + |I_l| = |I_l|.$$

Lemma 4. *If $I_c = [t_0, t_1)$ is a critical interval, and the probability that there is an open pair at t_0 is $S(t_0)$, i.e.* $\Pr[open(t_0) = 1] = S(t_0)$*, then the cost of RSAL in this critical interval is upper bounded by*

$$\text{cost}_{\mathcal{R}}(I_c) \leq S(t_1)(1 - S(t_1)) - S(t_0)(1 - S(t_0)) + \begin{cases} (1 + \frac{1}{4\tau})|I_c| & \text{if } \tau > \frac{1}{2} \\ (2 - \tau)|I_c| & \text{if } \tau \leq \frac{1}{2} \end{cases}.$$

Lemma 5. *If $I_e := [t_0, t_1]$ is an end interval, and the probability that there is an open pair at t_0 is $S(t_0)$, i.e.* $\Pr[open(t_0) = 1] = S(t_0)$*, then the cost of RSAL in this end interval is*

$$\text{cost}_{\mathcal{R}}(I_e) = |I_e|.$$

Next lemma shows that the conditions in Lemma 3 4 and 5 are satisfied during the execution of RSAL. Since the time intervals proceeded by RSAL on each phase are consecutive, Lemma 6 can be proved inductively by showing that once the starting time of an interval satisfies the condition, then the ending time of this interval (i.e. the starting time of next interval) will also satisfy the condition. The details of the proof are provided in Appendix A.

Lemma 6. *Suppose RSAL divides phase $P = [t_{start}, t_{end}]$ into intervals $[t_1, t_2)$, $[t_2, t_3)$, ..., $[t_{k-1}, t_k]$ with $t_1 = t_{start}$ and $t_k = t_{end}$, then the probability that there is an open pair at each t_i is $S(t_i)$ for $1 \leq i \leq k$, i.e.* $\Pr[open(t_i) = 1] = S(t_i)$ *for* $1 \leq i \leq k$.

Proof (Proof of Theorem 2). We will prove the competitive ratio phase by phase. First fix any phase $P = [t_{start}, t_{end}]$. Suppose RSAL partitions P into intervals $\{I_1, I_2, ..., I_m\}$, where I_m is the only end interval. Let $\mathcal{C}$ be the collection of all the critical intervals, and $\mathcal{L}$ be the collection of all the lossless intervals.

If $\mathcal{C} = \emptyset$, by Lemma 3 and 5, we have

$$\text{cost}_{\mathcal{R}}(P) = \sum_{I \in \mathcal{L}} \text{cost}_{\mathcal{R}}(I) + \text{cost}_{\mathcal{R}}(I_m) = \sum_{I \in \mathcal{L}} |I| + |I_m| = |P|,$$

which is optimal by Lemma 2. It remains to consider the case when $\mathcal{C} \neq \emptyset$. Let t_m be the starting time of I_m, then we have $t_m - t_{start} \geq \tau$.

When $\tau > \frac{1}{2}$, the cost of RSAL is

$$\begin{aligned} \text{cost}_{\mathcal{R}}(P) =& \sum_{I \in \mathcal{C}} \text{cost}_{\mathcal{R}}(I) + \sum_{I \in \mathcal{L}} \text{cost}_{\mathcal{R}}(I) + \text{cost}_{\mathcal{R}}(I_e) \\ =& \sum_{I \in \mathcal{C}} (1 + \frac{1}{4\tau})|I| + \sum_{I \in \mathcal{L}} |I| + |I_m| + S(t_m)(1 - S(t_m)) \\ \leq& (1 + \frac{1}{4\tau})(t_m - t_{start}) + (t_{end} - t_m) + S(t_m)(1 - S(t_m)). \end{aligned}$$

Denote $t_m - t_{start}$ as T_1 and $t_{end} - t_m$ as T_2. We naturally have $T_1 \geq \tau$ and $S(t_m) \leq T_2 \leq \tau$. Then the competitive ratio on P is upper bounded by

$$\begin{aligned}\alpha = \frac{\text{cost}_{\mathcal{R}}(P)}{\text{cost}_{\mathcal{A}^*}(P)} \leq & \frac{1}{T_1 + T_2}\left(T_1(1 + \frac{1}{4\tau}) + T_2 + S(t_m)(1 - S(t_m)))\right) \\ \leq & \frac{\tau + 1/4 + S(t_m)(2 - S(t_m))}{\tau + S(t_m)},\end{aligned}$$

which is optimized when $S(t_m) = \sqrt{\tau^2 + \tau - 1/4} - \tau$. Therefore, we obtain

$$\alpha \leq 2 + 2\tau - \sqrt{4\tau^2 + 4\tau - 1}.$$

When $\tau \leq \frac{1}{2}$, with the same reasoning,

$$\begin{aligned}\text{cost}_{\mathcal{R}}(P) = & \sum_{I \in \mathcal{C}} (2 - \tau)|I| + \sum_{I \in \mathcal{L}} |I| + |I_m| + S(t_m)(1 - S(t_m)) \\ \leq & (2 - \tau)T_1 + T_2 + S(t_m)(1 - S(t_m)).\end{aligned}$$

By taking $S(m) = (\sqrt{2} - 1)\tau$, the competitive ratio is bounded by

$$\alpha = \frac{\text{cost}_{\mathcal{R}}(P)}{\text{cost}_{\mathcal{A}^*}(P)} \leq \frac{2\tau - \tau^2 + S(t_m)(2 - S(t_m))}{\tau + S(t_m)} \leq 2 - (2\sqrt{2} - 2)\tau.$$

Randomized Algorithm for Generalized Input Similar to the deterministic algorithm, RSAL can be modified for generalized sequence with the same competitive ratio. The details can be found in Appendix B.

Corollary 2. *There is a randomized online algorithm for the 2-MPMD problem with foresight τ for any request sequence, and its competitive ratio is $2 - (2\sqrt{2} - 2)\tau$ when $\tau \leq \frac{1}{2}$, and $2 + 2\tau - \sqrt{4\tau^2 + 4\tau - 1}$ when $\tau > \frac{1}{2}$.*

Acknowledgments E. Sun and Z. Zhu are supported by the National Natural Science Foundation of China (Grant No. 6212290003).

References

1. Albers, S.: On the influence of lookahead in competitive paging algorithms. Algorithmica **18**(3), 283–305 (1997)
2. Ashlagi, I., Azar, Y., Charikar, M., Chiplunkar, A., Geri, O., Kaplan, H., Makhijani, R., Wang, Y., Wattenhofer, R.: Min-cost bipartite perfect matching with delays. In: Jansen, K., Rolim, J.D.P., Williamson, D., Vempala, S.S. (eds.) Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017, August 16-18, 2017, Berkeley, CA, USA. LIPIcs, vol. 81, pp. 1:1–1:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2017)

3. Azar, Y., Chiplunkar, A., Kaplan, H.: Polylogarithmic bounds on the competitiveness of min-cost perfect matching with delays. In: Klein, P.N. (ed.) Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19. pp. 1051–1061. SIAM (2017)
4. Azar, Y., Fanani, A.J.: Deterministic min-cost matching with delays. Theory Comput. Syst. **64**(4), 572–592 (2020)
5. Böhm, M., Chrobak, M., Jez, L., Li, F., Sgall, J., Veselý, P.: Online packet scheduling with bounded delay and lookahead. Theor. Comput. Sci. **776**, 95–113 (2019)
6. Dooly, D.R., Goldman, S.A., Scott, S.D.: TCP dynamic acknowledgment delay: Theory and practice (extended abstract). In: Vitter, J.S. (ed.) Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998. pp. 389–398. ACM (1998)
7. Dooly, D.R., Goldman, S.A., Scott, S.D.: On-line analysis of the TCP acknowledgment delay problem. J. ACM **48**(2), 243–273 (2001)
8. Dunke, F.: Online optimization with lookahead. Ph.D. thesis, Karlsruhe, Karlsruher Institut für Technologie (KIT), Diss., 2014 (2014)
9. Dunke, F., Nickel, S.: A general modeling approach to online optimization with lookahead. Omega **63**, 134–153 (2016)
10. Efroni, Y., Ghavamzadeh, M., Mannor, S.: Online planning with lookahead policies. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds.) Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual (2020)
11. Emek, Y., Kutten, S., Wattenhofer, R.: Online matching: haste makes waste! In: Wichs, D., Mansour, Y. (eds.) Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016. pp. 333–344. ACM (2016)
12. Emek, Y., Shapiro, Y., Wang, Y.: Minimum cost perfect matching with delays for two sources. Theor. Comput. Sci. **754**, 122–129 (2019)
13. He, K., Li, S., Sun, E., Wang, Y., Wattenhofer, R., Zhu, W.: Randomized algorithm for MPMD on two sources. In: Garg, J., Klimm, M., Kong, Y. (eds.) Web and Internet Economics - 19th International Conference, WINE 2023, Shanghai, China, December 4-8, 2023, Proceedings. Lecture Notes in Computer Science, vol. 14413, pp. 348–365. Springer (2023)
14. Jiao, C., Yuan, J., Feng, Q.: Online algorithms for scheduling unit length jobs on unbounded parallel-batch machines with linearly lookahead. Asia Pac. J. Oper. Res. **36**(5), 1950024:1–1950024:8 (2019)
15. Karlin, A.R., Kenyon, C., Randall, D.: Dynamic TCP acknowledgement and other stories about e/(e-1). In: Vitter, J.S., Spirakis, P.G., Yannakakis, M. (eds.) Proceedings on 33rd Annual ACM Symposium on Theory of Computing, July 6-8, 2001, Heraklion, Crete, Greece. pp. 502–509. ACM (2001)
16. Karlin, A.R., Manasse, M.S., McGeoch, L.A., Owicki, S.S.: Competitive randomized algorithms for nonuniform problems. Algorithmica **11**(6), 542–571 (1994)
17. Karlin, A.R., Manasse, M.S., Rudolph, L., Sleator, D.D.: Competitive snoopy caching. Algorithmica **3**, 77–119 (1988)
18. Motwani, R., Saraswat, V.A., Torng, E.: Online scheduling with lookahead: Multipass assembly lines. INFORMS J. Comput. **10**(3), 331–340 (1998)
19. Zheng, F., Cheng, Y., Liu, M., Xu, Y.: Online interval scheduling on a single machine with finite lookahead. Comput. Oper. Res. **40**(1), 180–191 (2013)

A Missing proofs in Section 4

Proof (Proof of Lemma 3). If $open(t_0) = 1$, by the algorithm, this pair will be matched to the next arriving pair. After that, the algorithm will be idle until a new pair arrives. Based on this observation, the algorithm will be waiting at t's such that $\text{trend}(t) = \text{trend}(t_0)$ and idle at t''s such that $\text{trend}(t') \neq \text{trend}(t_0)$. By the definition of a lossless interval, we have $U(I_l) = D(I_l)$. Therefore, no matter what $\text{trend}(t_0)$ is, the waiting time and the idling time both equal to $\frac{|I_l|}{2}$. As a consequence, we obtain that $\text{cost}_{\mathcal{D}}(I_l) = |I_l|$.

Similarly, if $open(t_0) = 0$, the cost $\text{cost}_{\mathcal{D}}(I_l)$ is still the length of I_l. Thus the expected cost in this interval is

$$\text{cost}_{\mathcal{D}}(I_l) = S(t_0)|I| + (1 - S(t_0))|I| = |I|.$$

Proof (Proof of Lemma 4). Without loss of generality, we assume that trend $t_0 =$ *decreasing*. Then we have $U(t_0, t_1) + D(t_0, t_1) = \tau$ and $D(t_0, t_1) - U(t_0, t_1) = \Delta(t_0, t_1)$. We consider the following cases separately.

When $open(t_0) = 1$, if the algorithm matches this open pair internally, the cost will be the delay cost on the increasing intervals plus 1 (the internal match's cost); if the algorithm does not match this open pair internally, the cost will be the delay cost on the decreasing intervals.

When $open(t_0) = 0$, the cost is the delay cost on the decreasing intervals.

The expected cost of RSAL on I_c is thus the summation of the above cases,

$$\begin{aligned}\text{cost}_{\mathcal{D}}(I_c) =& S(t_0)\left[\frac{\Delta(t_0,t_1)}{S(t_0)}(2U(t_0,t_1)+1) + (1-\frac{\Delta(t_0,t_1)}{S(t_0)}(2D(t_0,t_1))\right] \\ &+ (1-S(t_0))(2U(t_0,t_1)) \\ =& -2\Delta(t_0,t_1)^2 + 2S(t_0)\Delta(t_0,t_1) + \tau\end{aligned}$$

1. If $\Delta(t_0, t_1) = S(t_0) - S(t_1)$, we have

$$\begin{aligned}\text{cost}_{\mathcal{D}}(I_c) =& -2\Delta(t_0,t_1)^2 + 2S(t_0)\Delta(t_0,t_1) + \tau \\ =& -(S(t_0)-S(t_1))^2 + S(t_0)^2 - S(t_1)^2 + \tau \\ =& S(t_1)(1-S(t_1)) - S(t_0)(1-S(t_0)) \\ &+ (S(t_0)-S(t_1)) - (S(t_0)-S(t_1))^2 + \tau\end{aligned}$$

2. If $\Delta(t_0, t_1) = S(t_0) - (1 - S(t_1))$, similarly, we have

$$\text{cost}_{\mathcal{D}}(I_c) = S(t_1)(1-S(t_1)) - S(t_0)(1-S(t_0)) + (S(t_0)-(1-S(t_1))) - (S(t_0)-(1-S(t_1)))^2 + \tau$$

If $\tau \leq \frac{1}{2}$, by choosing $(S(t_0)-S(t_1)) = \tau$ for case 1 and $(S(t_0)-(1-S(t_1))) = \tau$ for case 2, we obtain

$$\text{cost}_{\mathcal{D}}(I_c) \leq S(t_1)(1-S(t_1)) - S(t_0)(1-S(t_0)) + (2-\tau)|I_c|.$$

If $\tau \geq \frac{1}{2}$, by choosing $(S(t_0)-S(t_1)) = \frac{1}{2}$ for case 1 and $(S(t_0)-(1-S(t_1))) = \frac{1}{2}$ for case 2, we obtain

$$\text{cost}_{\mathcal{D}}(I_c) \leq S(t_1)(1-S(t_1)) - S(t_0)(1-S(t_0)) + (1+\frac{1}{4\tau})|I_c|.$$

Proof (Proof of Lemma 5). The proof is similar to the proof of Lemma 3. Without loss of generality, we assume that $\text{trend}(t_0) = \textit{decreasing}$. By definition, $\text{trend}(t_0) = \text{trend}(t_1)$ and $S(t_1) = 0$. Otherwise, there will be a lossless interval lying in $[t_0, t_1]$. Hence, $\Delta(t_0, t_1) = S(t_0)$ and the expected cost is

$$\text{cost}_{\mathcal{D}}(I_e) = S(t_0)(2U(I) + 1) + (1 - S(t_0))(2D(I_e)) = |I_e|.$$

Proof (Proof of Lemma 6). we will prove by induction.

Base case: For the first interval $I_1 = [t_1, t_2)$, the first pair of the phase must be open at the beginning.

Inductive step: Suppose the induction hypothesis is true for first i intervals. Consider the $(i+1)$-th interval $I_i = [t_i, t_{i+1})$.

If $\text{trend}(t_i) = \text{trend}(t_1)$, then $\textit{open}(t_{i+1}) = 1$ if and only if $\textit{open}(t_i) = 1$ and the algorithm does not internally match this open pair. Therefore,

$$\Pr[\textit{open}(t_{i+1}) = 1] = \Pr[\textit{open}(t_i) = 1](1 - \frac{\Delta(t_i, t_{i+1})}{S(t_i)}) = S(t_{i+1}).$$

If $\text{trend}(t_i) \neq \text{trend}(t_1)$, then $\textit{open}(t_{i+1}) = 1$ if and only if one of the following two situations happens:

1. $\textit{open}(t_i) = 0$ and the algorithm matches externally during $[t_i, t_{i+1}]$.
2. $\textit{open}(t_i) = 1$ and the algorithm matches internally for this open pair.

Then the probability that at the time t_1 there is an open pair is

$$\begin{aligned}\Pr[\textit{open}(t_{i+1}) = 1] =& (1 - \Pr[\textit{open}(t_i) = 1]) + \Pr[\textit{open}(t_i) = 1]\frac{\Delta(t_i, t_{i+1})}{S(t_i)} \\ =& 1 - S(t_i) + [S(t_i) - (1 - S(t_{i+1})] = S(t_{i+1}).\end{aligned}$$

B From Paired Sequence to General Sequence

In this section, we generalize the smart algorithms DSAL and RSAL to arbitrary request sequences. First, we introduce a modified CPI subroutine to find generalize state values and trends. In the original CPI, the trend can only be either *increasing* or *decreasing*. We need to modify the update rules for state function and trends to accommodate general sequences.

Suppose we have a request sequence $R = \{r_i = (s_i, t_i)\}_{i \in [2n]}$. For each $i \in [2n]$, in the interval (t_{2i-1}, t_{2i}), the trend is set to be *waiting*, and the state value is the same as $S(t_{2i-1})$. We call an interval a waiting interval if in the whole interval the trend is *waiting*. Let $\mathcal{I}_w$ be the union of all waiting intervals. In the interval $[t_{2i}, t_{2i+1}]$ and the time after t_{2n}, the trend is *increasing* if the latest non-*waiting* trend is *decreasing*, and otherwise the trend is *decreasing*. For each t_{2i}, $S(t_{2i}) = 1 - S(t_{2i-1})$. The state value in the interval $(t_{2i}, t_{2i+1}]$ is still defined as in Equation 1. A phase $P = [t_{start}, t_{end}]$ will be an interval with $S(t_{start}) = 1$ and $S(t_{end}) = 0$.

Lemma 7. *For a request sequence $R = \{r_i = (s_i, t_i)\}_{i\in[2n]}$, the cost of an optimal offline algorithm equals to the sum of the length of each phase.*

Proof. Let k be the smallest integer such that $t_{2k-1} < t_{2k}$. Let $\delta = t_{2k} - t_{2k-1}$. We consider the request sequence $\hat{R} = \{\hat{r}_i = (s_i, \hat{t}_i)\}_{1\le i\le 2n}$ such that $\forall i \le 2k-1$, $\hat{r}_i = r_i$, and $\forall i \ge 2k$, $\hat{r}_i = (s_i, t_i - \delta)$.

Let $\mathcal{A}^*$ be an offline optimal algorithm. When $\mathcal{A}^*$ proceeds to the pair r_{2i-1}, since the current number of requests is odd, there must be at least one open request. By shifting all the points after $2k$ earlier by δ, the optimal algorithm $\mathcal{A}^*$ will output the same matches, but the cost will reduce δ. Reductively applying this process until all the requests are paired yields a paired sequence R'. The cost of $\mathcal{A}^*$ on R' is the sum of the lengths of all the increasing and decreasing intervals. Furthermore, $\text{cost}^R_{\mathcal{A}^*} = \text{cost}^{R'}_{\mathcal{A}^*} + \sum_{i\in[n]}(t_{2i} - t_{2i-1})$. The right-hand side is exactly the sum of the lengths of the phases.

Next we introduce the generalized internal match and external match. We say there is an open pair at time t if both sources have an open request. Internal match will match an open pair and will only be valid when there is such an open pair. External match of a request means matching it with the next request from the same source. The generalized DSAL performs the same as DSAL in Section 3.1 with modified CPI subroutine, together with the modified concepts of open pair, external and internal match.

Equipped with the above definitions, we're ready to show the competitive ratio of generalized algorithms.

Corollary 3. *The competitive ratio of generalized DSAL is $\left(\frac{3+\tau}{1+\tau}\right)$-competitive for the 2-MPMD problem with foresight τ.*

Proof. In any waiting interval $I = (t_0, t_1)$, no matter what decision the algorithm makes, there must be one open request. To see this, by the definition, there must be a request r coming at t_0. If all the requests before r are matched, then from t_0 to t_1, r has to be open. If there are two open requests on r's arrival, once r arrives, it will be externally matched with the request from the same source. Then there is also one request remaining open during I.

Given any interval, the cost on this interval is the sum of the costs of all increasing, decreasing, and waiting intervals. The costs on increasing and decreasing intervals are the same as the original DSAL. Hence by Theorem 1, summing over the increasing and decreasing intervals leads to

$$\frac{\text{cost}_{\mathcal{D}}(\mathcal{I}_{inc} \cup \mathcal{I}_{dec})}{\text{cost}_{\mathcal{A}^*}(\mathcal{I}_{inc} \cup \mathcal{I}_{dec})} \le \frac{3+\tau}{1+\tau}.$$

The cost on each waiting interval is the length of this interval for both optimal solution and DSAL. So the competitive ratio on the whole phase is

$$\alpha = \frac{\text{cost}_{\mathcal{D}}(\mathcal{I}_{inc} \cup \mathcal{I}_{dec}) + |\mathcal{I}_w|}{\text{cost}_{\mathcal{A}^*}(\mathcal{I}_{inc} \cup \mathcal{I}_{dec}) + |\mathcal{I}_w|} \le \frac{3+\tau}{1+\tau}.$$

Similarly, the generalized RSAL performs the same as DSAL in Section 4 with modified CPI subroutine, together with the modified concepts of open pair, external and internal match. Next lemma shows the competitive ratio of the generalized RSAL.

Corollary 4. *The competitive ratio of generalized RSAL is* $(2-(2\sqrt{2}-2)\tau$ *when* $\tau \leq \frac{1}{2}$*, and* $2+2\tau-\sqrt{4\tau^2+4\tau-1}$ *when* $\tau > \frac{1}{2}$ *for the 2-MPMD problem with foresight* τ*.*

Proof. Following the same argument as in the proof of Lemma 3, on any waiting interval $I = (t_0, t_1)$, no matter what decision the algorithm makes, there must be one open request.

Given any interval, the cost on this interval is the sum of the costs of all increasing, decreasing, and waiting intervals. The costs on increasing and decreasing intervals are the same as the original DSAL. Hence by Theorem 1, summing over the increasing and decreasing intervals leads to

$$\frac{\text{cost}_{\mathcal{R}}(\mathcal{I}_{inc} \cup \mathcal{I}_{dec})}{\text{cost}_{\mathcal{A}^*}(\mathcal{I}_{inc} \cup \mathcal{I}_{dec})} \leq \begin{cases} 2-(2-\sqrt{2})\tau & \text{if } \tau \leq \frac{1}{2} \\ 2+2\tau-\sqrt{4\tau^2+4\tau-1} & \text{if } \tau > \frac{1}{2} \end{cases}.$$

The cost on each waiting interval is the length of this interval for both $\mathcal{A}^*$ and DSAL. Therefore, the competitive ratio on the whole phase is

$$\alpha = \frac{\text{cost}_{\mathcal{R}}(\mathcal{I}_{inc} \cup \mathcal{I}_{dec}) + |\mathcal{I}_w|}{\text{cost}_{\mathcal{A}^*}(\mathcal{I}_{inc} \cup \mathcal{I}_{dec}) + |\mathcal{I}_w|} \leq \begin{cases} 2-(2-\sqrt{2})\tau & \text{if } \tau \leq \frac{1}{2} \\ 2+2\tau-\sqrt{4\tau^2+4\tau-1} & \text{if } \tau > \frac{1}{2} \end{cases}.$$