Minimum Age TDMA Scheduling

Tung-Wei Kuo
Department of Computer Science, National Chengchi University, Taiwan
twkuo@cs.nccu.edu.tw

***Abstract*—We consider a transmission scheduling problem in which multiple systems receive update information through a shared Time Division Multiple Access (TDMA) channel. To provide timely delivery of update information, the problem asks for a schedule that minimizes the overall age of information. We call this problem the Min-Age problem. This problem is first studied by He *et al.* [IEEE Trans. Inform. Theory, 2018], who identified several special cases where the problem can be solved optimally in polynomial time. Our contribution is threefold. First, we introduce a new job scheduling problem called the Min-WCS problem, and we prove that, for any constant $r \geq 1$, every r-approximation algorithm for the Min-WCS problem can be transformed into an r-approximation algorithm for the Min-Age problem. Second, we give a randomized 2.733-approximation algorithm and a dynamic-programming-based exact algorithm for the Min-WCS problem. Finally, we prove that the Min-Age problem is NP-hard.**

I. Introduction

We consider systems whose states change upon reception of update messages. Such systems include, for example, web caches [1], intelligent vehicles [2], and real-time databases [3]. The timely delivery of update messages is often critical to the smooth and secure functioning of the system. Moreover, since any given update is likely dependent on previous updates, the update messages should not be delivered out of order. In most cases, the system does not have exclusive access to a communication channel. Instead, it must share the channel with other systems. Hence, the transmission schedule plays a crucial role in determining the performance of the systems that share the channel.

This scenario can be modeled by multiple sender-receiver pairs and a channel shared by these sender-receiver pairs. The sender sends update messages to the receiver through the shared channel, and the receiver changes its state upon reception of an update message.[1] This paper discusses the design of transmission scheduling algorithms for such channels. Specifically, we assume that the channel has a buffer in which the update messages are stored, and a transmission schedule for the messages in the buffer must be determined.[2] In this paper, we refer to a system that changes its state upon reception of an update message as a *receiver*.

To keep the state of a receiver as fresh as possible, it is important to keep the age of the receiver as small as possible. Specifically, the age of a receiver is the age of the receiver's most recently received message M, i.e., the difference between the current time and the time at which M is generated. Most prior research analyzes the age of a receiver through stochastic process models [4]–[12], where the randomness comes from the state of the channel or the arrival process of update messages. In this paper, we take a combinatorial optimization approach to minimize the overall age of all receivers on a reliable channel. In particular, we study the problem defined by He *et al.*, who considered a scenario in which the transmission scheduling algorithm is invoked repeatedly [13]. Specifically, after the scheduling algorithm computes a schedule, the channel then delivers the messages according to the schedule. New messages may arrive while the channel is delivering the scheduled messages. These new messages are stored in the buffer and scheduled for transmission during the next invocation of the algorithm.

The scheduling algorithm should be designed with the characteristics of the channel in mind. For example, He *et al.* considered a wireless channel, in which various senders might interfere with one another [13]. They also considered a Time Division Multiple Access (TDMA) channel, in which the channel delivers one message at a time. They identified some conditions in which optimal schedules can be obtained by sorting the sender-receiver pairs according to the number of messages to be sent to the receiver [13]. However, even if the channel is TDMA-based, it remained open whether the problem can be solved optimally in polynomial time. In this paper, we therefore focus on TDMA channels. In the remainder of this paper, we refer to this scheduling problem on a TDMA channel as the Min-Age problem.

In this paper, we cast the Min-Age problem as a job scheduling problem called the Min-WCS problem. The Min-WCS problem has a simple formulation inspired by a geometric interpretation of the Min-Age problem. The simplicity of the formulation also facilitates algorithm design. As we will see in Section VII, one may solve variants of the Min-Age problem by modifying the geometric interpretation and then solving the corresponding job scheduling problem.

Job scheduling has been studied for decades. In fact, the Min-WCS problem is a special case of single-machine scheduling with a non-linear objective function under precedence constraints, which has been studied by Schulz and Verschae [14] and Carrasco *et al.* [15]. Specifically, for any $\epsilon > 0$, the algorithm proposed by Schulz and Verschae approximates the optimum within a factor of $(2 + \epsilon)$ when the objective function is concave [14]. When the objective function is convex, Carrasco *et al.* proposed a $(4 + \epsilon)$-speed

This work is partially supported by the Ministry of Science and Technology of R.O.C. under contract No. MOST 106-2221-E-004-005-MY3.

[1]The sender may serve as a relay or hub for the system and thus may not be responsible for generating update messages.

[2]The buffer may be a logical one that stores the inputs to a scheduler.

1-approximation algorithm for any $\epsilon > 0$ [15].[3] The solutions proposed by Schulz and Verschae [14] and Carrasco *et al.* [15] are based on linear programming rounding. The objective function of the Min-WCS problem is convex, and we give a randomized 2.733-approximation algorithm for the Min-WCS problem without linear programming. We summarize our major results as follows:

Theorem 1: We introduce the Min-WCS problem and prove that, for any constant $r \geq 1$, every r-approximation algorithm of the Min-WCS problem can be transformed into an r-approximation algorithm for the Min-Age problem.

Theorem 2: We solve the Min-WCS problem by combining two feasible schedules. Specifically, we propose a deterministic 4-approximation algorithm and a randomized 2.733-approximation algorithm for the Min-WCS problem.

Theorem 3: We give a dynamic-programming-based exact algorithm for the Min-WCS problem. The result implies that the Min-Age problem can be solved optimally in polynomial time when the number of sender-receiver pairs is a constant. The result holds even if there are arbitrarily many messages.

Theorem 4: We show that the Min-Age problem is NP-hard.

II. Problem Definition

The studied problem is first considered by He *et al.*, and is referred to as the minimum age scheduling problem with TDMA [13]. Throughout this paper, we simply refer to this problem as the Min-Age problem. To make the paper self-contained, we rephrase the definition of the Min-Age problem.

Inputs: We consider n sender-receiver pairs, $(s_1, r_1), (s_2, r_2), \cdots, (s_n, r_n)$, where s_i and r_i are the sender and receiver of the ith sender-receiver pair, respectively. Time is indexed by non-negative integers, and the current time is T_0. These n sender-receiver pairs share one transmission channel, which can transmit one message in one unit of time (hence the name TDMA). Each sender s_i has a set of messages $\mathcal{M}_i$ to be sent to receiver r_i. Our task is to schedule the transmissions of messages in $\mathcal{M}_1 \cup \mathcal{M}_2 \cup \cdots \cup \mathcal{M}_n$.

We use $b(M)$ (the birthday of M) to indicate the time at which message M is generated. Let M_i^0 be the latest message that has been received by r_i so far.[4] Thus, $M_i^0 \notin \mathcal{M}_i$. Let M_i^j be the jth oldest message in $\mathcal{M}_i$. Thus, for all $1 \leq i \leq n$, $0 \leq b(M_i^0) < b(M_i^1) < b(M_i^2) < \cdots < b(M_i^{|\mathcal{M}_i|}) \leq T_0$.

Output and constraints: The goal is to find a schedule S of message transmissions so that the overall age of information (to be defined later) is minimized. Let $S(M_i^j)$ be the time at which message M_i^j is received by r_i under schedule S. Hence, by the channel capacity constraint, $S(M_i^j) - 1$ is the time at which the channel starts to send M_i^j under schedule S. Let $T = |\mathcal{M}_1| + |\mathcal{M}_2| + \cdots + |\mathcal{M}_n|$ be the time needed to send all the messages. A feasible schedule S has to satisfy the following constraints.

1) Due to the channel capacity constraint, S is a one-to-one and onto mapping from $\mathcal{M}_1 \cup \mathcal{M}_2 \cup \cdots \cup \mathcal{M}_n$ to $\{T_0 + 1, T_0 + 2, \cdots, T_0 + T\}$.
2) Since a message may depend on previous messages, the schedule must follow the order of message generation. Specifically, for all $1 \leq i \leq n$, $S(M_i^1) < S(M_i^2) < \cdots < S(M_i^{|\mathcal{M}_i|})$. In other words, for each sender-receiver pair, the transmission schedule must follow the first-come-first-served (FCFS) discipline.

Age: Let $lm(S, i, t)$ be the latest message received by receiver r_i at or before time t under schedule S. The **age** of r_i at time t is the age of $lm(S, i, t)$ at time t, i.e., $t - b(lm(S, i, t))$. Like [13], we assume that, once r_i receives all messages in $\mathcal{M}_i$, the age of r_i becomes zero. Intuitively, under this assumption, a scheduling algorithm that minimizes the overall age would have the side benefit that the last message of each sender-receiver pair is sent as early as possible (under the FCFS discipline). More supporting arguments for this assumption can be found in [13]. Specifically, the age of r_i at time t under schedule S, $age(S, i, t)$, is defined as follows.

$$\begin{aligned} &age(S, i, t) = t - b(lm(S, i, t)), && \text{if } lm(S, i, t) \neq M_i^{|\mathcal{M}_i|}, \\ &age(S, i, t) = 0, && \text{otherwise.} \end{aligned}$$

Notice that $b(M_i^{|\mathcal{M}_i|})$ is not used when evaluating the age of r_i. Moreover, $age(S, i, T_0) = T_0 - b(M_i^0)$ is referred to as the *initial age* of receiver r_i. In Section VII, we will discuss the case where the age of r_i does not become zero even if r_i receives all messages in $\mathcal{M}_i$.

Objective function: In the Min-Age problem, the goal is to minimize the overall age, which adds up the ages of all receivers at all time indices. Specifically, the goal is to find a feasible schedule S that minimizes

$$Age(S) = \sum_{i=1}^{n} \sum_{t=T_0}^{T_0+T} age(S, i, t).$$

Example 1 (Min-Age Problem)**.** *We give an example in [13] with our notation.*[5] *We consider two sender-receiver pairs, where $|\mathcal{M}_1| = 3$ and $|\mathcal{M}_2| = 2$. Specifically,*

$$\begin{aligned} &T_0 = 15 \\ &b(M_1^0) = 3, b(M_1^1) = 6, b(M_1^2) = 7, b(M_1^3) = 8 \\ &b(M_2^0) = 3, b(M_2^1) = 5, b(M_2^2) = 10. \end{aligned}$$

Consider the schedule S shown in Fig. 1 with

$$\begin{aligned} &S(M_1^1) = 16, S(M_1^2) = 19, S(M_1^3) = 20 \\ &S(M_2^1) = 17, S(M_2^2) = 18. \end{aligned}$$

Observe that S is a one-to-one and onto mapping from $\mathcal{M}_1 \cup \mathcal{M}_2$ to $\{T_0 + 1, T_0 + 2, \cdots, T_0 + T\}$, where $T_0 = 15$ and $T = 5$. Moreover, S follows the first-come-first-served policy.

[3] Specifically, let OPT be the optimal objective value. An s-speed r-approximation algorithm for a minimization problem finds a solution of objective value at most $r \cdot OPT$ when using a machine that is s times faster than the original machine.

[4] Recall that a receiver is defined as a system that changes its state upon reception of an update message. The system is first assigned a state during the initialization phase. Thus, if r_i has not received any message sent from s_i, M_i^0 is the initial information installed on r_i during the initialization phase.

[5] The example is shown in Fig. 5 in [13].

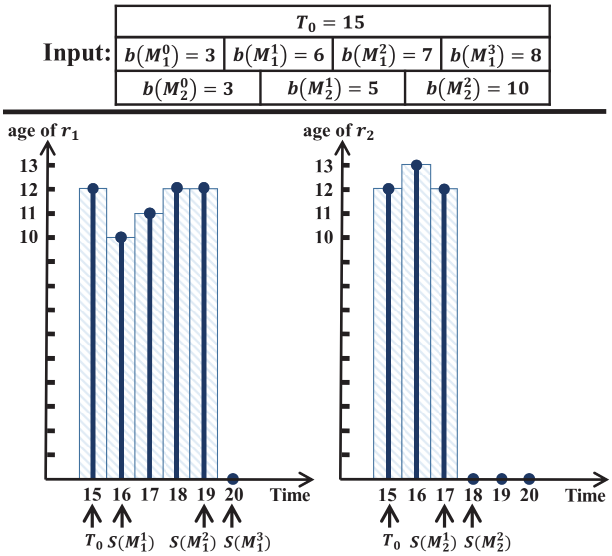

Fig. 1. An example of the Min-Age problem.

Hence, S is a feasible schedule. $\sum_{t=T_0}^{T_0+T} age(S,1,t) = (15-3)+(16-6)+(17-6)+(18-6)+(19-7)+0 = 57$. $\sum_{t=T_0}^{T_0+T} age(S,2,t) = (15-3)+(16-3)+(17-5)+0+0+0 = 37$. *Hence,* $Age(S) = 57+37 = 94$.

III. A Corresponding Job Scheduling Problem and Problem Transformation

In this paper, we cast the Min-Age problem as a job scheduling problem called the Min-WCS problem. We first give the definition of the Min-WCS problem in Section III-A. We then show that the Min-Age problem can be transformed into the Min-WCS problem in Section III-B.

A. The Min-WCS Problem

We consider a job scheduling problem with precedence constraints. That is, the order of job completion has to follow a given precedence relation $\to$. Specifically, for any two jobs J_1 and J_2, if $J_1 \to J_2$, then $S(J_1) < S(J_2)$, where $S(J)$ is the completion time of job J under schedule S. We consider chain-like precedence constraints. Specifically, the set of all jobs is divided into n_{chain} **job chains**, $\mathcal{C}_1, \mathcal{C}_2, \cdots, \mathcal{C}_{n_{chain}}$, where $\mathcal{C}_i$ is a chain of $|\mathcal{C}_i|$ jobs, $J_i^1 \to J_i^2 \to \cdots \to J_i^{|\mathcal{C}_i|}$. For any feasible job schedule S and any $1 \le i \le n_{chain}$, $S(J_i^1) < S(J_i^2) < \cdots < S(J_i^{|\mathcal{C}_i|})$. Throughout this paper, J_i^j denotes the jth job of job chain $\mathcal{C}_i$. J_i^j is called a **leaf job** if $j = |\mathcal{C}_i|$; otherwise, it is called an **internal job**.

We are now ready to define the job scheduling problem considered in this paper. The input consists of n_{chain} job chains, where each job J_i^j is associated with a non-negative weight w_i^j. The processing time of every job is one unit of time, and the system only has one machine, which starts processing jobs at time 0. All jobs are non-preemptive. Hence, the completion time of the last completed job is $T_{chain} = |\mathcal{C}_1| + |\mathcal{C}_2| + \cdots + |\mathcal{C}_{n_{chain}}|$. Since the processing time of each job is one unit of time, a feasible schedule is a one-to-one and onto mapping from the set of all jobs to $\{1, 2, \cdots, T_{chain}\}$. The goal is to find a feasible schedule S that minimizes $wcs(S) = wc(S) + cs(S)$, where $wc(S)$ is the total weighted completion time of all jobs under S, and $cs(S)$ is the total completion time squared of all leaf jobs under S. Specifically,

$$wc(S) = \sum_{\text{All jobs } J_i^j} (w_i^j \cdot S(J_i^j)),$$

and

$$cs(S) = \sum_{\text{All leaf jobs } J_i^{|\mathcal{C}_i|}} (S(J_i^{|\mathcal{C}_i|}) \cdot S(J_i^{|\mathcal{C}_i|})).$$

In this paper, we refer to this job scheduling problem as the Min-WCS problem.

B. Transformation from the Min-Age Problem to the Min-WCS Problem

In this subsection, we give a method to solve the Min-Age problem by transforming it into the Min-WCS problem. The high-level idea is to construct a corresponding job J_i^j for each message $M_i^j \in \mathcal{M}_i$. Specifically, given a problem instance I_{age} of the Min-Age problem, we construct a corresponding instance I_{job} of the Min-WCS problem, where

$$n_{chain} = n, \tag{1}$$

and

$$|\mathcal{C}_i| = |\mathcal{M}_i|, \text{ for all } 1 \le i \le n. \tag{2}$$

The job weight is determined by T_0 and $b(M)$. Specifically,

$$w_i^j = 2(b(M_i^j) - b(M_i^{j-1})), \text{ if } j \in \{1, 2, \cdots, |\mathcal{C}_i| - 1\}, \tag{3}$$

and

$$w_i^{|\mathcal{C}_i|} = 2(T_0 - 0.5 - b(M_i^{|\mathcal{M}_i|-1})). \tag{4}$$

Note that, since $b(M_i^{|\mathcal{M}_i|-1}) \le b(M_i^{|\mathcal{M}_i|}) - 1 \le T_0 - 1$, all weights are non-negative, and thus this is a valid problem instance of the Min-WCS problem. Since we have $n_{chain} = n$ and $T_{chain} = T$ in the transformation, in what follows, we omit the subscript of n_{chain} and T_{chain}.

Example 2 (The transformation)**.** *Consider the Min-Age problem instance I_{age} in Example 1. We transform I_{age} into the following instance I_{job} of the Min-WCS problem. I_{job} has two jobs chains. The first job chain has three jobs, and the second job chain has two jobs. The weights of the first two jobs in $\mathcal{C}_1$ are*

$$w_1^1 = 2(b(M_1^1) - b(M_1^0)) = 2(6-3) = 6,$$

and

$$w_1^2 = 2(b(M_1^2) - b(M_1^1)) = 2(7-6) = 2.$$

The weight of the last job in $\mathcal{C}_1$ is

$$w_1^3 = 2(T_0 - 0.5 - b(M_1^2)) = 2(15 - 0.5 - 7) = 15.$$

Similarly, we have $w_2^1 = 4$ and $w_2^2 = 19$. Recall that, in Fig. 1, $Age(S) = 94$. Consider a schedule S_{job} such that $S_{job}(J_i^j) = S(M_i^j) - T_0$ for all $1 \le i \le 2$, $1 \le j \le |\mathcal{C}_i|$. We

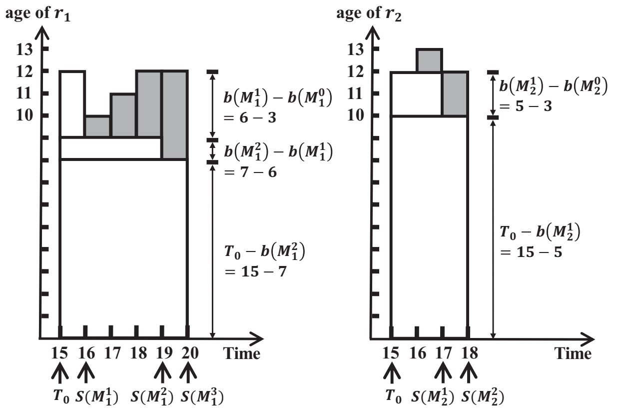

Fig. 2. A geometric interpretation of $Age(S)$.

then have $wc(S_{job}) = 6\cdot 1 + 4 \cdot 2 + 19 \cdot 3 + 2 \cdot 4 + 15 \cdot 5 = 154$ *and* $cs(S_{job}) = 5 \cdot 5 + 3 \cdot 3 = 34$*. Notice that* $wcs(S_{job}) = wc(S_{job}) + cs(S_{job}) = 154 + 34 = 188 = 2 \cdot Age(S)$*.*

The rationale behind the transformation: We give a geometric interpretation of $Age(S)$.[6] We use Fig. 2 to explain the idea. Notice that in Fig. 1, $Age(S)$ is the total area of rectangles shown in Fig. 1. In Fig. 2, we divide the overall age of r_i into white rectangles and gray rectangles. Since we only consider the total area, we right-shift all rectangles by 0.5 unit. For r_i, there are $|\mathcal{M}_i|$ white rectangles, and the width of the jth white rectangle is $S(M_i^j) - T_0$. The height of the jth white rectangle is $b(M_i^j) - b(M_i^{j-1})$ (if $1 \le j \le |\mathcal{M}_i| - 1$) or $T_0 - b(M_i^{|\mathcal{M}_i|-1})$ (if $j = |\mathcal{M}_i|$). The height can be interpreted as the age reduction after receiving message M_i^j. Note that, after receiving the last message, the age becomes zero. Hence, the total height of the white rectangles should be $T_0 - b(M_i^0)$, i.e., the initial age of r_i. Therefore, the height of the bottom white rectangle is $T_0 - b(M_i^0) - \sum_{j=1}^{|\mathcal{M}_i|-1} (b(M_i^j) - b(M_i^{j-1})) = T_0 - b(M_i^{|\mathcal{M}_i|-1})$. After considering age reduction, we still need to increase the age by one after each unit of time. This is captured by the gray rectangles. The width of every gray rectangle is one, and the heights of gray rectangles are $1, 2, \cdots, S(M_i^{|\mathcal{M}_i|}) - T_0 - 1$. Hence, the total area of the gray rectangles is $\frac{(S(M_i^{|\mathcal{M}_i|}) - T_0)(S(M_i^{|\mathcal{M}_i|}) - T_0 - 1)}{2} = \frac{(S(M_i^{|\mathcal{M}_i|}) - T_0)^2}{2} - \frac{S(M_i^{|\mathcal{M}_i|}) - T_0}{2}$. Let S_{age} be any feasible schedule of a Min-Age problem instance I_{age}. We have

$$\begin{aligned}
&Age(S_{age})\\
&= \sum_{i=1}^{n} \sum_{j=1}^{|\mathcal{M}_i|-1} (b(M_i^j) - b(M_i^{j-1}))(S_{age}(M_i^j) - T_0)\\
&+ \sum_{i=1}^{n} (T_0 - b(M_i^{|\mathcal{M}_i|-1}))(S_{age}(M_i^{|\mathcal{M}_i|}) - T_0)\\
&+ \sum_{i=1}^{n} (\frac{(S_{age}(M_i^{|\mathcal{M}_i|}) - T_0)^2}{2} - \frac{S_{age}(M_i^{|\mathcal{M}_i|}) - T_0}{2}).
\end{aligned}$$

Let I_{job} be I_{age}'s corresponding job scheduling problem instance. Specifically, I_{age} and I_{job} satisfy Eq. (1) to Eq. (4). Let S_{job} be any feasible schedule of I_{job}. We have

$$\begin{aligned}
wcs(S_{job}) &= \sum_{i=1}^{n} \sum_{j=1}^{|\mathcal{C}_i|-1} 2(b(M_i^j) - b(M_i^{j-1}))S_{job}(J_i^j)\\
&+ \sum_{i=1}^{n} 2(T_0 - \frac{1}{2} - b(M_i^{|\mathcal{M}_i|-1}))S_{job}(J_i^{|\mathcal{C}_i|})\\
&+ \sum_{i=1}^{n} (S_{job}(J_i^{|\mathcal{C}_i|}))^2.
\end{aligned}$$

Thus, if $S_{job}(J_i^j) = S_{age}(M_i^j) - T_0$ holds for all $1 \le i \le n$, $1 \le j \le |\mathcal{M}_i|$, we then have $2Age(S_{age}) = wcs(S_{job})$.

The above result then suggests the following method to construct a schedule S_{age} for I_{age}. First, obtain a schedule S_{job} of the corresponding Min-WCS problem instance I_{job}. We then view $S_{job}(J_i^j)$ as the transmission order of M_i^j in S_{age}. Specifically, we set $S_{age}(M_i^j) = S_{job}(J_i^j) + T_0$. The following lemma establishes the relation between S_{job} and S_{age}. Throughout this paper, we use I_{age} and I_{job} to denote problem instances of the Min-Age problem and the Min-WCS problem, respectively.

Lemma 1. *Let S_{age} and S_{job} be any two schedules of I_{age} and I_{job}, respectively. If I_{age} and I_{job} satisfy Eq.* (1) *to Eq.* (4)*, and $S_{age}(M_i^j) = S_{job}(J_i^j) + T_0$, for all $1 \le i \le n$, $1 \le j \le |\mathcal{C}_i|$, then*

1) S_{age} is feasible if and only if S_{job} is feasible.
2) $2Age(S_{age}) = wcs(S_{job})$.

Proof. By the above discussion, we already have $2Age(S_{age}) = wcs(S_{job})$. Since $S_{age}(M_i^j) = S_{job}(J_i^j) + T_0$ for all $1 \le i \le n$, $1 \le j \le |\mathcal{C}_i|$, S_{age} is a one-to-one and onto mapping from $\bigcup_{i=1}^{n} \mathcal{M}_i$ to $\{T_0+1, T_0+2, \cdots, T_0+T\}$ if and only if S_{age} is a one-to-one and onto mapping from the set of all jobs to $\{1, 2, \cdots, T\}$. On the other hand, it is easy to see that S_{age} follows the first-come-first-served policy for each sender-receiver pair if and only if S_{job} follows the chain-like precedence constraint. Thus, S_{age} is feasible if and only if S_{job} is feasible. □

The next lemma establishes the relation between the optimums of a Min-Age problem instance and the corresponding Min-WCS problem instance.

[6] He *et al.* also gave a geometric interpretation of $Age(S)$ [13]. The geometric interpretation proposed in this paper is different from that in [13], and our interpretation naturally suggests a transformation into the job scheduling problem defined in this paper.

Lemma 2. *Let S^*_{age} and S^*_{job} be the optimal schedules of I_{age} and I_{job}, respectively. If I_{age} and I_{job} satisfy Eq.* (1) *to Eq.* (4)*, then $2Age(S^*_{age}) = wcs(S^*_{job})$.*

Proof. Let S'_{age} be a schedule such that $S'_{age}(M_i^j) = S^*_{job}(J_i^j) + T_0$ for all $1 \leq i \leq n$, $1 \leq j \leq |\mathcal{C}_i|$. Similarly, let S'_{job} be a schedule such that $S^*_{age}(M_i^j) = S'_{job}(J_i^j) + T_0$ for all $1 \leq i \leq n$, $1 \leq j \leq |\mathcal{C}_i|$. By Lemma 1, we have

$$2Age(S'_{age}) = wcs(S^*_{job}) \text{ and } 2Age(S^*_{age}) = wcs(S'_{job}).$$

Finally, since

$$2Age(S^*_{age}) \leq 2Age(S'_{age}) = wcs(S^*_{job})$$

and

$$wcs(S^*_{job}) \leq wcs(S'_{job}) = 2Age(S^*_{age}),$$

we have $wcs(S^*_{job}) = 2Age(S^*_{age})$. □

Theorem 1. *For any constant $r \geq 1$, if there exists a polynomial-time r-approximation algorithm for the Min-WCS problem, then there exists a polynomial-time r-approximation algorithm for the Min-Age problem.*

Proof. The r-approximation algorithm for the Min-Age problem proceeds as follows. First, given a problem instance I_{age} of the Min-Age problem, the algorithm constructs a corresponding instance I_{job} of the Min-WCS problem by the aforementioned transformation. Obviously, the transformation can be done in polynomial time. We then apply the r-approximation algorithm for the Min-WCS problem on I_{job} to get a schedule S_{job}. We construct a schedule S_{age} for I_{age} by setting $S_{age}(M_i^j) = S_{job}(J_i^j) + T_0$ for all $1 \leq i \leq n$, $1 \leq j \leq |\mathcal{M}_i|$. By Lemmas 1 and 2, S_{age} is feasible and $Age(S_{age}) = \frac{wcs(S_{job})}{2} \leq r \cdot \frac{wcs(S^*_{job})}{2} = r \cdot Age(S^*_{age})$. □

IV. Approximation Algorithms for the Min-WCS Problem

By Theorem 1, to solve the Min-Age problem, it suffices to solve the Min-WCS problem. Notice that the objective function of the Min-WCS problem is the sum of two functions, wc and cs. When the objective function becomes wc (respectively, cs), we refer to the problem as the Min-WC problem (respectively, the Min-CS problem). Both the Min-WC problem and the Min-CS problem can be solved optimally in polynomial time. Given an instance of the Min-WCS problem, the high-level idea of our algorithm is to first solve the corresponding instances of the Min-WC problem and the Min-CS problem. Throughout this paper, we use S^*_{wc} (respectively, S^*_{cs}) to denote the optimal schedule of the Min-WC problem (respectively, the Min-CS problem). We then *interleave* S^*_{cs} with S^*_{wc} to approximate the Min-WCS problem. We first discuss the solutions of the Min-WC problem and the Min-CS problem in Section IV-A. We then present our algorithm for the Min-WCS problem in Section IV-B.

Algorithm 1: An Algorithm for the Min-WC Problem

for $t \leftarrow 1$ **to** $|\mathcal{C}_1| + |\mathcal{C}_2| + \cdots + |\mathcal{C}_n|$ **do**
  $\mathcal{U} \leftarrow$ the set of the first unscheduled job in each job chain
  $J \leftarrow \arg\max_{J_i^j \in \mathcal{U}} \rho_i^j$
  $S^*_{wc}(J) \leftarrow t$

Algorithm 2: An Algorithm for the Min-CS Problem

$t \leftarrow 1$
$\mathcal{U} \leftarrow \{1, 2, \cdots, n\}$
while $\mathcal{U} \neq \varnothing$ **do**
  $i^* \leftarrow \arg\min_{i \in \mathcal{U}} |\mathcal{C}_i|$
  $\mathcal{U} \leftarrow \mathcal{U} \setminus \{i^*\}$
  for $j \leftarrow 1$ **to** $|\mathcal{C}_{i^*}|$ **do**
    $S^*_{cs}(J_{i^*}^j) \leftarrow t$
    $t \leftarrow t + 1$

A. Algorithms for the Min-WC Problem and the Min-CS Problem

1) The Min-WC Problem: The Min-WC problem is a special case of the minimum total weighted completion time scheduling problem subject to precedence constraints, which has been studied over many years. When the precedence constraints are chain-like, the problem can be solved in polynomial time [16]. Recall that, in our problem, the processing time of every job is one. The algorithm for the Min-WC problem proceeds as follows. For each job J_i^j, define the job's priority ρ_i^j as $\max_{k: j \leq k \leq |\mathcal{C}_i|} \frac{w_i^j + w_i^{j+1} + \cdots + w_i^k}{k-j+1}$. To minimize the total weighted completion time, the machine should first process the job with the highest priority. We still need to follow the precedence constraints. Hence, to determine the next processing job, we only consider the first unprocessed job in each job chain, and we choose the one that has the highest priority. Algorithm 1 summarizes the pseudocode.

Lemma 3 (Lawler [16])**.** *Algorithm 1 solves the Min-WC problem optimally in polynomial time.*

Example 3 (Algorithm 1)**.** *Consider the problem instance in Example 2. We have $\rho_1^1 = \max\left(\frac{6}{1}, \frac{6+2}{2}, \frac{6+2+15}{3}\right) = \frac{23}{3}$ and $\rho_2^1 = \max\left(\frac{4}{1}, \frac{4+19}{2}\right) = \frac{23}{2}$. Since $\rho_2^1 > \rho_1^1$, Algorithm 1 first schedules J_2^1 and sets $S^*_{wc}(J_2^1) = 1$. The job completion order under S^*_{wc} is $J_2^1, J_2^2, J_1^1, J_1^2, J_1^3$.*

2) The Min-CS Problem: By a simple interchange argument, it is easy to see that the shortest job chain should be completed first in the Min-CS problem. Algorithm 2 summarizes the pseudocode. We have the following lemma.

Lemma 4. *Algorithm 2 solves the Min-CS problem optimally in polynomial time.*

Example 4 (Algorithm 2)**.** *Consider the problem instance in Example 2. Since $|\mathcal{C}_2| < |\mathcal{C}_1|$, the job completion order under S^*_{cs} is $J_2^1, J_2^2, J_1^1, J_1^2, J_1^3$.*

Observe that in Example 3 and Example 4, $S^*_{wc} = S^*_{cs}$. It

Algorithm 3: An Algorithm for the Min-WCS Problem with Parameter p

$S_{wc}^{*} \leftarrow$ the schedule obtained by Algorithm 1
$S_{cs}^{*} \leftarrow$ the schedule obtained by Algorithm 2
$S_{cs}^{int} \leftarrow S_{cs}^{*}$
$T \leftarrow |\mathcal{C}_1| + |\mathcal{C}_2| + \cdots + |\mathcal{C}_n|$
for $i \leftarrow 1$ **to** $T-1$ **do**
  X_i is set to 1 with probability p and is set to 0 with probability $1-p$
  if $X_i = 1$ **then**
    forall *Job J such that $S_{cs}^{*}(J) > i$* **do**
      $S_{cs}^{int}(J) \leftarrow S_{cs}^{int}(J) + 1$
for $i \leftarrow 1$ **to** T **do**
  $J \leftarrow$ the ith completed job under S_{wc}^{*}
  $S_{wc}^{int}(J) \leftarrow$ the finish time of the ith idle time slot in S_{cs}^{int}
forall $J \in \mathcal{C}_1 \cup \mathcal{C}_2 \cup \cdots \cup \mathcal{C}_n$ **do**
  $S'_{wcs}(J) \leftarrow \min\{S_{cs}^{int}(J), S_{wc}^{int}(J)\}$
for $i \leftarrow 1$ **to** T **do**
  $J \leftarrow$ the ith completed job under S'_{wcs}
  $S_{wcs}(J) = i$

is easy to see that S_{wc}^{*} and S_{cs}^{*} are thus optimal schedules of the Min-WCS problem. Therefore, the optimal message transmission order in Example 1 is $M_2^1, M_2^2, M_1^1, M_1^2, M_1^3$, and the optimal overall age is $(12+13+14+12+12)+(12+11) = 86$.

Proposition 1. *Let I_{job} be any instance of the Min-WCS problem, and let S_{wc}^{*} and S_{cs}^{*} be the optimal schedules of the corresponding instances of the Min-WC problem and the Min-CS problem, respectively. If $S_{wc}^{*} = S_{cs}^{*}$, then S_{wc}^{*} and S_{cs}^{*} are optimal schedules of I_{job}.*

B. Interleaving S_{wc}^{*} and S_{cs}^{*} Randomly: A Randomized Approximation Algorithm for the Min-WCS Problem

While S_{wc}^{*} and S_{cs}^{*} solve the Min-WC problem and the Min-CS problem, respectively, neither S_{wc}^{*} nor S_{cs}^{*} can approximate the Min-WCS problem well. Specifically, we have the following results.

Proposition 2. *The approximation ratio of Algorithm 1 for the Min-WCS problem is $\Omega(n)$.*

Proposition 3. *The approximation ratio of Algorithm 2 for the Min-WCS problem is $\Omega(n)$.*

The proof idea of Proposition 2 (respectively, Proposition 3) is to consider an instance such that, for any feasible schedule S, $\frac{wc(S)}{cs(S)} \approx 0$ (respectively, $\frac{cs(S)}{wc(S)} \approx 0$). We omit the detailed proof due to the space limit.

Despite the above negative results, we will show that interleaving S_{wc}^{*} and S_{cs}^{*} gives an $O(1)$-approximation algorithm for the Min-WCS problem. A critical observation of the Min-WC problem (respectively, the Min-CS problem) is that, if we multiply the optimal scheduled completion time $S_{wc}^{*}(J)$ (respectively, $S_{cs}^{*}(J)$) of every job J by a factor $c > 1$ (i.e., we *delay* the optimal schedule by a multiplicative *delay factor* of c), then the total weighted completion time (respectively, the total completion time squared of all leaf jobs) is increased by a multiplicative factor of c (respectively, c^2). This immediately suggests the following deterministic 4-approximation algorithm: For each job J, set $S_{cs}^{int}(J) = 2S_{cs}^{*}(J) - 1$. Hence, S_{cs}^{int} is a delayed version of S_{cs}^{*} with a delay factor less than two[7], and the time period $[2k-1, 2k]$ is idle for any integer $k \geq 1$. We call such an idle time period an *idle time slot*. Moreover, define the *finish time* of an idle time slot $[t-1, t]$ as t. Consider another schedule S_{wc}^{int} obtained by setting $S_{wc}^{int}(J) = 2S_{wc}^{*}(J)$ for each job J. Hence, S_{wc}^{int} is a delayed version of S_{wc}^{*} with a delay factor of two. We can view S_{wc}^{int} as a schedule obtained by inserting jobs one by one following the order specified in S_{wc}^{*} to the idle time slots in S_{cs}^{int}. For each job J, set $S'_{wcs}(J) = \min\{S_{wc}^{int}(J), S_{cs}^{int}(J)\}$. We will show that S'_{wcs} satisfies the precedence constraints. Finally, we remove the idle time slots in S'_{wcs} to obtain the final schedule S_{wcs}. We then have

$$wc(S_{wcs}) \leq wc(S_{wc}^{int}) \leq 2 \cdot wc(S_{wc}^{*})$$

and

$$cs(S_{wcs}) \leq cs(S_{cs}^{int}) \leq 2^2 \cdot cs(S_{cs}^{*}).$$

Thus,

$$wcs(S_{wcs}) = wc(S_{wcs}) + cs(S_{wcs}) \leq 4(wc(S_{wc}^{*}) + cs(S_{cs}^{*})).$$

Since $wc(S_{wc}^{*}) + cs(S_{cs}^{*})$ is a lower bound of the optimum of the Min-WCS problem, S_{wcs} is a 4-approximation solution.

In hindsight, we first insert idle time slots to S_{cs}^{*} and then insert jobs to the idle time slots following the order specified in S_{wc}^{*}. To improve the algorithm, we insert idle time slots to S_{cs}^{*} randomly. Specifically, let p be a number in $[0, 1]$. Initially, $S_{cs}^{int} = S_{cs}^{*}$. For every two jobs J_1 and J_2 that are processed contiguously in S_{cs}^{*} (i.e., $|S_{cs}^{*}(J_2) - S_{cs}^{*}(J_1)| = 1$), we insert an idle time slot between $S_{cs}^{int}(J_1)$ and $S_{cs}^{int}(J_2)$ with probability p. Notice that, in S_{cs}^{int}, we never insert two or more contiguous idle time slots, which is a critical property that will be used in the analysis. Algorithm 3 summarizes the pseudocode. Observe that this randomized algorithm degenerates to Algorithm 2 when $p = 0$, and this randomized algorithm degenerates to the aforementioned deterministic 4-approximation algorithm when $p = 1$.

Example 5 (Algorithm 3)**.** *Consider a Min-WCS problem instance with two job chains where $|\mathcal{C}_1| = 2$ and $|\mathcal{C}_2| = 3$. Hence, the job completion order under S_{cs}^{*} is $J_1^1, J_1^2, J_2^1, J_2^2, J_2^3$. Assume that the job completion order under S_{wc}^{*} is $J_2^1, J_1^1, J_2^2, J_1^2, J_2^3$. Assume $X_1 = X_3 = 1, X_2 = X_4 = 0$. $S_{wc}^{int}, S_{cs}^{int}, S'_{wcs}$ and S_{wcs} are shown in Fig. 3.*

Since S_{wc}^{int} and S_{cs}^{int} do not overlap, we never execute two jobs at the same time in S'_{wcs}. Thus, to prove that S_{wcs} is feasible, it remains to prove the following lemma.

Lemma 5. *S'_{wcs} follows the precedence constraints.*

[7]Although different jobs have different delay factors, every job has a delay factor less than two.

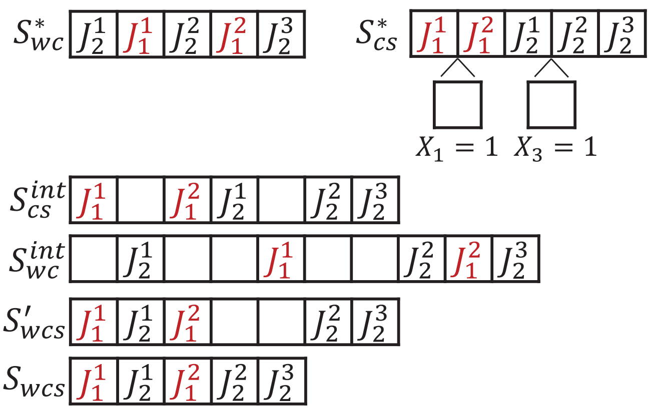

Fig. 3. An example of Algorithm 3.

Proof. For the sake of contradiction, assume that there are two jobs J_i^a and J_i^b from the same job chain such that $S'_{wcs}(J_i^a) > S'_{wcs}(J_i^b)$ but $a < b$. We first consider the case where $S'_{wcs}(J_i^a) = S^{int}_{wc}(J_i^a)$. Hence, we must have $S'_{wcs}(J_i^b) = S^{int}_{cs}(J_i^b)$ (otherwise, S^{int}_{wc} and S^*_{wc} would violate the precedence constraints). Since S^{int}_{cs} follows the precedence constraints, $S^{int}_{cs}(J_i^b) > S^{int}_{cs}(J_i^a)$. Finally, since $S'_{wcs}(J_i^a) > S'_{wcs}(J_i^b) = S^{int}_{cs}(J_i^b)$, we have $S'_{wcs}(J_i^a) > S^{int}_{cs}(J_i^a)$, which contradicts to the definition of S'_{wcs}. The case where $S'_{wcs}(J_i^a) = S^{int}_{cs}(J_i^a)$ can be proved in a similar way. □

Throughout this paper, we use $\mathbf{E}[X]$ to denote the expected value of X. The following theorem expresses the approximation ratio of Algorithm 3 as a function of p.

Theorem 2. $\frac{\mathbf{E}[wcs(S_{wcs})]}{wc(S^*_{wc})+cs(S^*_{cs})} \le \max\{1+\frac{1}{p}, 1+3p\}$.

Proof. It is sufficient to prove

$$\mathbf{E}[wc(S^{int}_{wc})] \le (1+\frac{1}{p})wc(S^*_{wc}) \qquad (5)$$

and

$$\mathbf{E}[cs(S^{int}_{cs})] \le (1+3p)cs(S^*_{cs}). \qquad (6)$$

This is because, by Eq. (5) and Eq. (6), we have

$$\begin{aligned}
&\mathbf{E}[wcs(S_{wcs})] = \mathbf{E}[wc(S_{wcs}) + cs(S_{wcs})] \\
&= \mathbf{E}[wc(S_{wcs})] + \mathbf{E}[cs(S_{wcs})] \le \mathbf{E}[wc(S^{int}_{wc})] + \mathbf{E}[cs(S^{int}_{cs})] \\
&\le (1+1/p)wc(S^*_{wc}) + (1+3p)cs(S^*_{cs}) \\
&\le \max\{1+1/p, 1+3p\} \cdot (wc(S^*_{wc}) + cs(S^*_{cs})).
\end{aligned}$$

Let J^i_{wc} be the ith completed job under S^*_{wc}. Hence, $S^*_{wc}(J^i_{wc}) = i$. Let Y_i be the completion time of J^i_{wc} under S^{int}_{wc}, i.e., $Y_i = S^{int}_{wc}(J^i_{wc})$. Similarly, let J^i_{cs} be the ith completed job under S^*_{cs}, and let $Z_i = S^{int}_{cs}(J^i_{cs})$.

To prove Eq. (5), it suffices to show that $\mathbf{E}[Y_i] \le i(1+1/p)$ holds for all $1 \le i \le T$. Let G_p be a random variable indicating the number of trials required to get the first success where the probability of success in each independent trial is p. By the setting of S^{int}_{wc}, we have $Y_1 = \min\{G_p+1, T+1\}$. Moreover, for all $2 \le i \le T$, $Y_i = \min\{Y_{i-1}+G_p+1, T+i\} = \min\{i(G_p+1), T+i\}$. Hence,

$$\mathbf{E}[Y_i] \le \mathbf{E}[i(G_p+1)] = i(1+\mathbf{E}[G_p]) = i(1+1/p).$$

To prove Eq. (6), it suffices to prove that $\mathbf{E}[Z_i^2] \le (1+3p)i^2$ holds for all $1 \le i \le T$. Let B_p be a random variable such that $B_p = 1$ with probability p and $B_p = 0$ with probability $1-p$. By the setting of S^{int}_{cs}, we have $Z_1 = 1$, and $Z_2 = Z_1 + B_p + 1 = 2 + B_p$. In general, for all $2 \le i \le T$, we have $Z_i = Z_{i-1} + B_p + 1 = i + (i-1)B_p$. Hence,

$$\begin{aligned}
&\mathbf{E}[Z_i^2] = \mathbf{E}[(i+(i-1)B_p)^2] \\
&= \mathbf{E}[i^2 + 2i(i-1)B_p + (i-1)^2B_p^2] \\
&= i^2 + 2i(i-1)\mathbf{E}[B_p] + (i-1)^2\mathbf{E}[B_p^2] \\
&= i^2 + 2i(i-1)p + (i-1)^2p = (1+3p)i^2 - 4ip + p \\
&\le (1+3p)i^2 \text{ (since } i \ge 1).
\end{aligned}$$

□

When $p = \frac{1}{\sqrt{3}}$, $1+\frac{1}{p} = 1+3p = 1+\sqrt{3}$. Note that, $\frac{1}{\sqrt{3}} \approx 0.57735$ and $1+\sqrt{3} \approx 2.73205$.

Corollary 1. *Algorithm 3 is a randomized 2.733-approximation algorithm for the Min-WCS problem when* $p = 0.57735$.

Corollary 2. *Algorithm 3 is a deterministic 4-approximation algorithm for the Min-WCS problem when* $p = 1$.

V. An Exact Algorithm for the Min-WCS Problem

Next, we solve the Min-WCS problem by dynamic programming. The objective function wcs can be stated as follows.

$$wcs(S) = \sum_{i=1}^{n}\sum_{j=1}^{|\mathcal{C}_i|} f_i^j(S(J_i^j)),$$

where $f_i^j(t) = w_i^j \cdot t$ if $j \ne |\mathcal{C}_i|$, and $f_i^j(t) = w_i^j \cdot t + t^2$ if $j = |\mathcal{C}_i|$. In other words, $f_i^j(t)$ is the cost incurred by job J_i^j if J_i^j is completed at time t, and $wcs(S)$ is simply the total cost incurred by all jobs under schedule S.

In the dynamic program, we consider subproblems of the Min-WCS problem where job chains can be executed partially. Specifically, for each job chain $\mathcal{C}_i$, we only need to schedule the first $\mathbf{L}[i]$ jobs $J_i^1, J_i^2, \cdots, J_i^{\mathbf{L}[i]}$, where $\mathbf{L}[i] \le |\mathcal{C}_i|$ and $\mathbf{L}[i]$ may be zero. When $\mathbf{L}[i] = 0$, we do not need to schedule any job in $\mathcal{C}_i$. More formally, let $\mathbf{L}$ be any vector of length n such that, for all $1 \le i \le n$, the ith element of $\mathbf{L}$, denoted by $\mathbf{L}[i]$, is in $\{0, 1, 2, \cdots, |\mathcal{C}_i|\}$. In this paper, for any vector $\mathbf{V}$, we use $\mathbf{V}[i]$ to denote the ith element of $\mathbf{V}$. Define

$$wcs_{\mathbf{L}}(S) = \sum_{i=1}^{n}\sum_{j=1}^{\mathbf{L}[i]} f_i^j(S(J_i^j)).$$

Let $\mathcal{J}(\mathbf{L})$ be the set of jobs whose costs are considered in $wcs_{\mathbf{L}}$. Let $\mathbf{MinWCS}(\mathbf{L})$ be a subproblem of the Min-WCS problem where we only need to schedule jobs in $\mathcal{J}(\mathbf{L})$ with objective function $wcs_{\mathbf{L}}$. Hence, the Min-WCS problem is equivalent to $\mathbf{MinWCS}(\mathbf{L})$ when $\mathbf{L}[i] = |\mathcal{C}_i|$ for all $1 \leq i \leq n$. Let $S^*_{\mathbf{L}}$ be the optimal schedule of $\mathbf{MinWCS}(\mathbf{L})$.

Observe that the last completed job under $S^*_{\mathbf{L}}$ must be $J_k^{\mathbf{L}[k]}$ for some $1 \leq k \leq n$. To find $S^*_{\mathbf{L}}$, we try every possible last completed job $J_k^{\mathbf{L}[k]}$ and consider the subproblem obtained by removing $J_k^{\mathbf{L}[k]}$ from $\mathbf{MinWCS}(\mathbf{L})$. Define $\mathbf{L}_{-k}$ as a vector of length n such that $\mathbf{L}_{-k}[j] = \mathbf{L}[j]$ if $j \neq k$ and $\mathbf{L}_{-k}[k] = \mathbf{L}[k] - 1$. Thus, the subproblem $\mathbf{MinWCS}(\mathbf{L}_{-k})$ can be obtained by removing $J_k^{\mathbf{L}[k]}$ from $\mathbf{MinWCS}(\mathbf{L})$. We then compute $S^*_{\mathbf{L}}$ based on $S^*_{\mathbf{L}_{-1}}, S^*_{\mathbf{L}_{-2}}, \cdots, S^*_{\mathbf{L}_{-n}}$. Specifically, define $S^*_{\mathbf{L}_{-k}} \oplus J_k^{\mathbf{L}[k]}$ as the schedule obtained by setting the completion time of job $J_k^{\mathbf{L}[k]}$ to $\sum_{1 \leqslant i \leqslant n} \mathbf{L}[i]$ and setting the completion time of every job J' in $\mathcal{J}(\mathbf{L}_{-k})$ to $S^*_{\mathbf{L}_{-k}}(J')$. Since $wcs_{\mathbf{L}}$ simply adds up the cost incurred by each job in $\mathcal{J}(\mathbf{L})$, it is easy to see that

$$S^*_{\mathbf{L}} = \underset{S \in \{S^*_{\mathbf{L}_{-k}} \oplus J_k^{\mathbf{L}[k]} | 1 \leq k \leq n, \mathbf{L}[k] \geq 1\}}{\arg\min} wcs_{\mathbf{L}}(S).$$

Given the values of $\sum_{1 \leqslant i \leqslant n} \mathbf{L}[i]$ and $wcs_{\mathbf{L}_{-k}}(S^*_{\mathbf{L}_{-k}})$, $wcs_{\mathbf{L}}(S^*_{\mathbf{L}_{-k}} \oplus J_k^{\mathbf{L}[k]})$ can be computed in $O(1)$ time. Hence, given all the required $S^*_{\mathbf{L}_{-k}}$ and $wcs_{\mathbf{L}_{-k}}(S^*_{\mathbf{L}_{-k}})$, $\min_{S \in \{S^*_{\mathbf{L}_{-k}} \oplus J_k^{\mathbf{L}[k]} | 1 \leq k \leq n, \mathbf{L}[k] \geq 1\}} wcs_{\mathbf{L}}(S)$ can be computed in $O(n) = O(T)$ time. We can then construct $S^*_{\mathbf{L}}$ in $O(\sum_{1 \leqslant i \leqslant n} \mathbf{L}[i]) = O(T)$ time. The base case where $\mathbf{L}[i] = 0$ for all $1 \leq i \leq n$ is trivial, and we can solve the dynamic program by a bottom-up approach. Since there are $\prod_{i=1}^{n} (|\mathcal{C}_i| + 1)$ subproblems, we have the following results.

Theorem 3. *The Min-WCS problem can be solved optimally in* $O(T \prod_{i=1}^{n} (|\mathcal{C}_i| + 1))$ *time.*

Thus, the Min-WCS problem can be solved optimally in polynomial time when the number of job chains is a constant (i.e., $n = O(1)$), even if there are arbitrarily many jobs.

Corollary 3. *The Min-WCS problem can be solved optimally in polynomial time if* $n = O(1)$. *This is true even if there are arbitrarily many jobs.*

VI. NP-Hardness of the Min-Age Problem

In this section, we prove that the Min-Age problem is NP-hard. He *et al.* proved that a certain generalization of the Min-Age problem is NP-hard [13]. However, this result does not preclude the possibility of solving the Min-Age problem optimally in polynomial time. Specifically, He *et al.* studied a generalization of the Min-Age problem where senders in the same *candidate group* can send messages simultaneously. The list of candidate groups are either explicitly specified in the inputs or can be derived from an interference model based on SINR. This generalization greatly increases the complexity of the scheduling problem. In fact, He *et al.* proved that, even if every sender s_i has only one message to be scheduled, (i.e., $|\mathcal{M}_i| = 1$ for all $1 \leq i \leq n$), the generalization is still NP-hard [13]. However, this special case can be solved optimally in polynomial time for the Min-Age problem.[8] On the other hand, we transform the Min-Age problem into the Min-WCS problem, where the processing time of every job is one, and the precedence constraints are chain-like. Given such a simple setting, one may suspect that the Min-WCS problem is in P, and thus the Min-Age problem is in P as well. Nevertheless, in this paper, we prove that the Min-Age problem is NP-hard. Hence, unless P = NP, the best polynomial-time algorithm for the Min-Age problem is an approximation algorithm.

Theorem 4. *The Min-Age problem is NP-hard.*

The proof of Theorem 4 is quite involved, and we only give the proof sketch due to the space limit. We use $P_1 \to P_2$ to indicate a reduction from problem P_1 to problem P_2.

1) SPC-Min-WCS $\to$ Min-Age: In this reduction, we reverse the transformation in Section III-B so that the given instance of the Min-WCS problem and the constructed instance of the Min-Age problem satisfy Eq. (1) to Eq. (4). Note that, to make the reverse transformation yield a valid Min-Age instance, we only consider a special case of the Min-WCS problem. We call this special-case problem the SPC-Min-WCS problem.

2) 3-Partition $\to$ EX-Min-WCS: To prove that the SPC-Min-WCS problem is NP-hard, we consider an extension of the Min-WCS problem in which the processing time of a job may exceed one. We call this extension the EX-Min-WCS problem. We prove that this extension is NP-hard by a reduction from 3-Partition, which is defined as follows.

Definition 1. *Given a set* L *of* $3m$ *positive integers,* $a_1, a_2, \cdots, a_{3m}$, *and a positive integer* B *such that* $\sum_{a \in L} a = mB$ *and* $\frac{B}{4} < a < \frac{B}{2}$ *for all* $a \in L$, ***3-Partition*** *asks for a partition of* L *into* m *subsets of* L, $P_1, P_2, \cdots, P_m$, *such that* $\sum_{a \in P_i} a = B$ *for all* $1 \leq i \leq m$.

The idea of the reduction is the following.

- For each $a_i \in L$, we create a corresponding job chain $\mathcal{C}_i$. The processing time of the first job in $\mathcal{C}_i$ is a_i. We thus refer to the first job in $\mathcal{C}_i$ as an a**-job**.
- We create $m-1$ job chains $\mathcal{C}^*_1, \cdots, \mathcal{C}^*_{m-1}$. Each job chain $\mathcal{C}^*_i$ has only one job. We call these jobs the **separating jobs**. We fine-tune the job weights so that jobs completed between separating jobs are a-jobs. Thus, these $m-1$ separating jobs partition the set of a-jobs into m sets, $A_1, \cdots, A_m$, based on the scheduled completion time. The processing time of a separating job is one. Thus, if the completion times of these separating jobs are $1(B+1), 2(B+1), \cdots, (m-1)(B+1)$, then the total processing time of each A_i is B, which implies that 3-Partition has a feasible partition. In the reduction, we fine-tune the job weights, so that the completion times of these separating jobs under the optimal schedule are $1(B+$

[8] To see this, consider the corresponding Min-WCS problem instance, where every job chain has only one job. Then, for any two feasible schedules S_1 and S_2 of the Min-WCS problem instance, $cs(S_1) = cs(S_2)$. Hence, the corresponding Min-WCS problem instance can be solved optimally in polynomial time by Algorithm 1. This implies that the Min-Age problem can be solved optimally in polynomial time if $|\mathcal{M}_i| = 1$ for all $1 \leq i \leq n$.

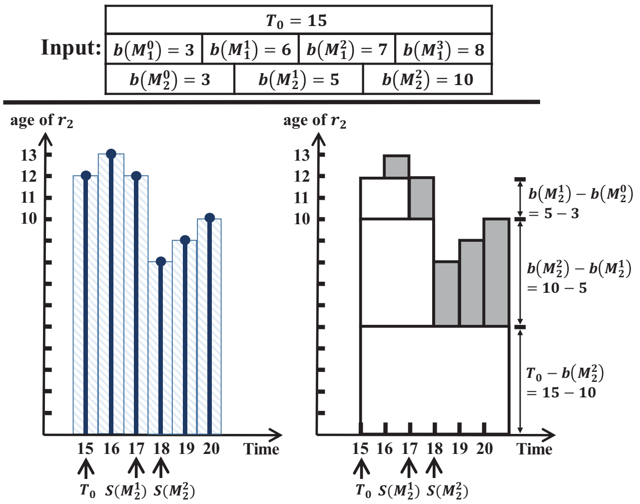

Fig. 4. A geometric interpretation of a special receiver's age.

$1), 2(B+1), \cdots, (m-1)(B+1)$, if and only if 3-Partition has a feasible partition.

Note that 3-Partition is NP-hard even if integers are encoded in unary. Specifically, if an integer n is encoded in unary, then the space required to store n is $\Theta(n)$. Thus, this reduction shows that the EX-Min-WCS problem is NP-hard even if the processing time is encoded in unary.

3) EX-Min-WCS $\rightarrow$ SPC-Min-WCS: In this reduction, for each job J in the EX-Min-WCS instance, if its processing time is p, we simulate the job by creating a job chain of length p in the corresponding SPC-Min-WCS instance. This reduction can be done in polynomial time since the job processing time in the EX-Min-WCS problem is encoded in unary.

VII. Concluding Remarks

In this paper, we assume that the age of a receiver r_i becomes zero once r_i receives all messages in $\mathcal{M}_i$. One of the rationales behind the design is to make the scheduling algorithm transmit the last message for each sender-receiver pair as early as possible. Nevertheless, we can solve the problem even if the age of some receiver r_i is not set to zero after receiving all messages in $\mathcal{M}_i$. We call such a receiver a **special** receiver. Hence, for a special receiver r_i, its age is always the age of the most recently received message. To solve the Min-Age problem with special receivers, we adjust the geometric interpretation given in Section III-B accordingly. An example of the new geometric interpretation is shown in Fig. 4, where r_2 is a special receiver. Recall that $T = |\mathcal{M}_1| + \cdots + |\mathcal{M}_n|$. Compared to Fig. 2, we have three critical observations for each special receiver r_i:

1) The number of white rectangles is increased by one, and the area of the bottom white rectangle is $(T+1)(T_0 - b(M_i^{|\mathcal{M}_i|}))$, which is fixed regardless of the schedule.
2) The height of the $|\mathcal{M}_i|$th white rectangle becomes $b(M_i^{|\mathcal{M}_i|}) - b(M_i^{|\mathcal{M}_i|-1})$. Therefore, we need to modify the job weight setting in the transformation accordingly.
3) The total area of the gray rectangles is $1 + 2 + \cdots + T$, which is fixed regardless of the schedule.

We thus update the objective function of the Min-WCS problem accordingly. Specifically, $wcs(S) = wc(S) + cs'(S) + C$, where C is a non-negative number specified in the input. Moreover, $cs'(S) = \sum_{i=1}^{n_{chain}} (I_i \cdot S(J_i^{|\mathcal{C}_i|}) \cdot S(J_i^{|\mathcal{C}_i|}))$, where I_i is an input that can be zero or one. In the problem transformation, if r_i is a special receiver, we set $I_i = 0$. Otherwise, we set $I_i = 1$. Finally, we use C to capture the total fixed rectangle area for the special receivers. To minimize $cs'(S)$, all job chains $\mathcal{C}_i$ with $I_i = 0$ are completed lastly in the schedule, and all job chains $\mathcal{C}_i$ with $I_i = 1$ are scheduled by Algorithm 2. Hence, we can still compute two optimal schedules that minimize wc and cs', respectively, and then apply Algorithm 3 to approximate the modified Min-WCS problem. Since the constant C in wcs is non-negative, the approximation ratio cannot be worse than that in Theorem 2. One may also solve other variants of the Min-Age problem by modifying the geometric interpretation and then considering the corresponding job scheduling problem.

References

[1] H. Yu, L. Breslau, and S. Shenker, "A scalable web cache consistency architecture," in *ACM SIGCOMM*, 1999.

[2] P. Papadimitratos, A. D. L. Fortelle, K. Evenssen, R. Brignolo, and S. Cosenza, "Vehicular communication systems: Enabling technologies, applications, and future outlook on intelligent transportation," *IEEE Communications Magazine*, vol. 47, no. 11, pp. 84–95, November 2009.

[3] M. Xiong and K. Ramamritham, "Deriving deadlines and periods for real-time update transactions," *IEEE Transactions on Computers*, vol. 53, no. 5, pp. 567–583, May 2004.

[4] S. Kaul, M. Gruteser, V. Rai, and J. Kenney, "Minimizing age of information in vehicular networks," in *IEEE SECON*, 2011.

[5] S. Kaul, R. Yates, and M. Gruteser, "Real-time status: How often should one update?" in *IEEE INFOCOM*, 2012.

[6] L. Huang and E. Modiano, "Optimizing age-of-information in a multi-class queueing system," in *IEEE ISIT*, 2015.

[7] C. Kam, S. Kompella, and A. Ephremides, "Age of information under random updates," in *IEEE ISIT*, 2013.

[8] I. Kadota, A. Sinha, and E. Modiano, "Optimizing age of information in wireless networks with throughput constraints," in *IEEE INFOCOM*, 2018.

[9] Y. P. Hsu, E. Modiano, and L. Duan, "Age of information: Design and analysis of optimal scheduling algorithms," in *IEEE ISIT*, 2017.

[10] I. Kadota, E. Uysal-Biyikoglu, R. Singh, and E. Modiano, "Minimizing the age of information in broadcast wireless networks," in *Allerton*, 2016.

[11] S. K. Kaul and R. D. Yates, "Status updates over unreliable multiaccess channels," *CoRR*, vol. abs/1705.02521, 2017.

[12] R. Talak, S. Karaman, and E. Modiano, "Distributed scheduling algorithms for optimizing information freshness in wireless networks," *CoRR*, vol. abs/1803.06469, 2018.

[13] Q. He, D. Yuan, and A. Ephremides, "Optimal link scheduling for age minimization in wireless systems," *IEEE Transactions on Information Theory*, vol. 64, no. 7, pp. 5381–5394, 2018.

[14] A. S. Schulz and J. Verschae, "Min-Sum Scheduling Under Precedence Constraints," in *ESA*, 2016.

[15] R. A. Carrasco, G. Iyengar, and C. Stein, "Single machine scheduling with job-dependent convex cost and arbitrary precedence constraints," *Operations Research Letters*, vol. 41, no. 5, pp. 436 – 441, 2013.

[16] E. Lawler, "Sequencing jobs to minimize total weighted completion time subject to precedence constraints," in *Algorithmic Aspects of Combinatorics*, ser. Annals of Discrete Mathematics, B. Alspach, P. Hell, and D. Miller, Eds. Elsevier, 1978, vol. 2, pp. 75 – 90.