Dynamic Interval Scheduling with Random Start and End Times*

Rui Gong† Alejandro Toriello

February 6, 2026

Abstract

We study sequential interval scheduling when task start and end times are random. The set of tasks and their weights are known in advance, while each task's start and end times are drawn from known discrete distributions and revealed only upon commitment; this also eliminates tasks that conflict with the committed task, and remaining tasks are those that do not conflict. The objective is to maximize the expected weight of a conflict-free schedule. We propose two models that differ in how conflicts are enforced, develop LP relaxations and bounds for each, and present a computational study.

1 Introduction

The *interval scheduling problem* is classical in operations research, industrial engineering and computer science. It considers a set of tasks $N := [n] = \{1, \dots, n\}$ and consecutive time slots $M := [m]$. Each task is represented by an interval $[s_i, e_i]$ describing the time slots it occupies if scheduled. The *maximum interval scheduling problem* seeks a set of pairwise nonoverlapping intervals (a schedule) of maximum total weight. When all tasks have the same weight, the greedy rule that repeatedly selects the earliest-ending task and discards conflicts is optimal; the weighted version is also solvable in $\Theta(n)$ time [5].

An *interval graph* is an undirected graph representing intervals on a line, where each vertex corresponds to an interval and an edge connects two vertices if their intervals overlap. Thus a feasible schedule corresponds exactly to an independent set of the associated interval graph. Since interval graphs are perfect, the maximum interval scheduling problem can be represented by the clique LP pair of the interval graph $G = (N, E)$,

$$\begin{aligned} \max_{x \geq 0} \quad & w^\top x \\ \text{s.t.} \quad & \sum_{i \in C} x_i \leq 1, \forall C \in \mathscr{C}(G) \end{aligned} \qquad \text{(LP-P)} \qquad\qquad \begin{aligned} \min_{\mu \geq 0} \quad & \sum_{C \in \mathscr{C}(G)} \mu_C \\ \text{s.t.} \quad & \sum_{C \ni i} \mu_C \geq w_i, \ \forall i \in N, \end{aligned} \qquad \text{(LP-D)}$$

where $\mathscr{C}(G)$ is the set of all maximal cliques of G. (LP-P) formulates the maximum independent set problem and (LP-D) formulates the corresponding minimum clique cover problem. For arbitrary perfect graphs, (LP-P) and (LP-D) are tight formulations, but not guaranteed to be solvable in polynomial time because there are possibly exponentially many maximal cliques. In contrast, in interval graphs the number of maximal cliques is at most $\min\{m, n\}$. Thus the deterministic maximum interval scheduling problem can be solved efficiently either directly or via these LP relaxations. Our goal is to extend this setting to incorporate uncertainty and sequential decisions while retaining tractable relaxations and algorithms.

Several stochastic interval scheduling models have been studied. A classical one is *online interval scheduling* [7], where intervals of fixed length are presented to the scheduler in the order of their start times and must be accepted or rejected irrevocably. The objective is to maximize total weight while maintaining feasibility. Recent work includes online interval scheduling with predictions of upcoming intervals [2] and models that allow overlaps but maximize a

*Partially funded by the U.S. Office of Naval Research, N00014-23-1-2631.

†Corresponding author (rgong44@gatech.edu)
Georgia Institute of Technology, Atlanta, GA 30332, USA

The authors' work was partially funded by the U.S. Office of Naval Research, N00014-23-1-2631.

satisfaction objective [6]. Other variants include interval scheduling with random delays [3] and settings where tasks may depart unexpectedly and service times are stochastic [9].

We study the setting where the set of tasks and their weights are known in advance but task start and end times are random. The scheduler commits to tasks sequentially, learns realized intervals as tasks are committed, and seeks to maximize the expected total weight.

2 Dynamic Scheduling with Random Start and End Times

We again let $M = [m]$ be the set of slots, $N = [n]$ the set of tasks, and $w \in \mathbb{R}_+^N$ the task weights. For each task $i \in N$, let D_i^s, D_i^e be the independent discrete distributions of i's start and end times, and denote $S_i, E_i \subseteq M$ as their supports. We assume $\max S_i \leq \min E_i$ and define $D := \{D_i^* : * \in \{s,e\}, i \in N\}$. When task i is scheduled, its start and end times s_i, e_i are realized according to D_i^s and D_i^e. For each remaining task j, we compute the probability p_{ij} that it conflicts with i based on s_i, e_i, D_j^s and D_j^e, and draw $B_{ij} \sim \text{Bernoulli}(p_{ij})$. If $B_{ij} = 1$, task j is deleted; otherwise, it remains available and its distributions are updated to condition on the slots occupied by i, $[s_i, e_i]$. The goal is to schedule tasks sequentially and maximize the expected total weight of scheduled tasks. We call this problem *dynamic scheduling with random start and end times (DSRSE)*.

DSRSE can be formulated as a dynamic program with the Bellman recursion

$$V^*(N, M, w, D) := \max_i \{w_i + \mathbb{E}[V^*(N', M', w, D')]\},$$

where V^* is the optimal value function and N', M', w, D' are the random variables representing the resulting state after scheduling task i. Like many dynamic programming problems, this problem has exponentially many states and each state has $O(n)$ actions, so it is unreasonable to solve it directly; moreover, the problem is NP-hard.

Proposition 2.1. *DSRSE is NP-hard.*

Proof. Suppose there are $n+1$ tasks and $2n$ slots. Let 0 be a task taking positions $1, \ldots, n$ with probability 1, and task i taking position i with probability p_i and $n+i$ with probability $(1-p_i)$. Then this problem is equivalent to the dynamic maximum stable set problem over a star graph, where 0 is the center; this problem is NP-hard [8]. □

Example 2.2. *Consider an instance with $N = [2], M = [3]$, and unit weights, where $D_1^s = \mathbb{1}_{\{1\}}, D_1^e = \mathbb{1}_{\{2\}}$, $D_2^s = U(\{2,3\}), D_2^e = \mathbb{1}_{\{3\}}$. If the scheduler commits to task 1 first, then task 1 occupies $[1,2]$. If task 1 is scheduled, task 2 has a probability of $1/2$ of conflicting and $1/2$ of remaining. If it remains, task 2's distributions are updated to condition on the fact that it does not conflict with task 1: $D_2^s = \mathbb{1}_{\{3\}}, D_2^e = \mathbb{1}_{\{3\}}$. On the other hand, if task 2 is scheduled first and its start time realizes to 2, task 1 is deleted; if it realizes to 3, task 1 remains.*

When s_i, e_i are deterministic for every task, DSRSE reduces to the maximum independent set problem of the corresponding interval graph. As discussed above, it can be represented by (LP-P) and (LP-D), so the deterministic problem can be solved efficiently via these LP relaxations. For an interval graph, maximal cliques are elements of M, so the constraint $\sum_{i \in C} x_i \leq 1, \forall C \in \mathscr{C}(G)$ in (LP-P) simply enforces that each slot is occupied by at most one task. We consider analogous constraints for DSRSE.

2.1 Relaxations

For DSRSE, consider $P^s, P^e, P^o \in \mathbb{R}^{n \times m}$, where $P_{ik}^s, P_{ik}^e, P_{ik}^o$ are the probabilities that task i starts at, ends at, and occupies slot k, respectively. Let $T = [\min\{m,n\}]$ be the stages of the dynamic program. For any policy, let $x_{ik}^{ts}, x_{ik}^{te}, x_{ik}^{to}$ be the probabilities that i is scheduled and starts at, ends at, or occupies slot k at stage t, respectively. Let $x_i^t = \sum_{k=1}^m x_{ik}^{ts} = \sum_{k=1}^m x_{ik}^{te}$ be the probability of scheduling i at stage t. The objective can then be written as $\max \sum_t \sum_i w_i x_i^t$. Analogous to the deterministic case, we impose that the probability a slot is occupied is at most 1. We relax DSRSE to

$$\begin{aligned}
\max \ & \sum_t \sum_i w_i x_i^t \\
& x_i^t = \sum_{k=1}^m x_{ik}^{ts} = \sum_{k=1}^m x_{ik}^{te}, \forall i, \forall t \in T \\
& x_{ik}^{ts} = 0, \forall t \in T, \forall k \notin S_i \\
& x_{ik}^{te} = 0, \forall t \in T, \forall k \notin E_i
\end{aligned}$$

$$\sum_t \sum_i x^{to}_{ik} \le 1, \forall k \in M$$
$$\sum_t x^{ts}_{ik} \le P^s_{ik},\ \sum_t x^{te}_{ik} \le P^e_{ik}\ \forall i \in N, \forall k \in M$$
$$0 \le x^t_i, x^{ts}_{ik}, x^{te}_{ik}, x^{to}_{ik},$$

where $x^s_{ik} := \sum_t x^{ts}_{ik} \le P^s_{ik}$ indicates that the probability of i being scheduled and starting at k is at most the marginal probability P^s_{ik}, and similarly for $x^e_{ik} := \sum_t x^{te}_{ik} \le P^e_{ik}$. For each $k \in M$, we can rewrite $\sum_t \sum_i x^{to}_{ik} \le 1$ as

$$\begin{aligned}
&\sum_t \sum_i x^{to}_{ik} \\
=&\sum_t \left(\sum_{i,a_i \le k \le b_i} x^{to}_{ik} + \sum_{i,c_i \le k \le d_i} x^{to}_{ik} + \sum_{i,b_i < k < c_i} x^{to}_{ik} \right) \\
=&\sum_t \left(\sum_{i,a_i \le k \le b_i - 1} \sum_{\ell=a_i}^{k} x^{ts}_{i\ell} + \sum_{i,c_i+1 \le k \le d_i} \sum_{\ell=k}^{d_i} x^{te}_{i\ell} + \sum_{i,b_i \le k \le c_i} x^t_i \right) \\
=&\sum_t \sum_{i,b_i \le k \le c_i} x^t_i + \sum_t \left(\sum_{i,a_i \le k \le b_i - 1} \sum_{\ell=a_i}^{k} x^{ts}_{i\ell} + \sum_{i,c_i+1 \le k \le d_i} \sum_{\ell=k}^{d_i} x^{te}_{i\ell} \right) \\
\le& 1,
\end{aligned} \tag{1}$$

where $a_i := \min\{k : k \in S_i\}, b_i := \max\{k : k \in S_i\}, c_i := \min\{k : k \in E_i\}, d_i := \max\{k : k \in E_i\}$. The first equality comes from the fact that the probability i occupies a slot k not in $S_i \cup E_i \cup (b_i, c_i)$ is 0. For the second equality, by definition, any $k \in [b_i, c_i]$ is always occupied by i if it is scheduled, so $x^{to}_{ik} = x^t_i$; for $k \in [a_i, b_i)$, the probability of i being scheduled and occupying k at t is equivalent to the probability of i being scheduled and starting at or before k at t, and similarly for the ending slots. The third equality is derived by rearranging terms. Since $\sum_{i,a_i \le k \le b_i - 1} \sum_{\ell=a_i}^{k} x^{ts}_{i\ell} = \sum_{i,a_i \le k \le b_i - 1} (x^t_i - \sum_{\ell=k+1}^{b_i} x^{ts}_{i\ell})$, and similarly for $x^{te}_{i\ell}$, we have for every $k \in M$,

$$\sum_t \left(\sum_{i,a_i \le k \le d_i} x^t_i - \sum_{i,a_i \le k \le b_i} \sum_{\ell=k+1}^{b_i} x^{ts}_{i\ell} - \sum_{i,c_i \le k \le d_i} \sum_{\ell=c_i}^{k-1} x^{te}_{i\ell} \right) \le 1. \tag{2}$$

Then we have the following primal-dual pair:

$$\begin{aligned}
\max_{x \ge 0}\ & \sum_t \sum_i w_i x^t_i \\
& x^t_i = \sum_{k=1}^{m} x^{ts}_{ik} = \sum_{k=1}^{m} x^{te}_{ik}, \forall i \in N, \forall t \in T \\
& x^{ts}_{ik} = 0, \forall t \in T, \forall k \notin [a_i, b_i] \\
& x^{te}_{ik} = 0, \forall t \in T, \forall k \notin [c_i, d_i] \\
& \sum_t \Big(\sum_{i,a_i \le k \le d_i} x^t_i - \sum_{i,a_i \le k \le b_i} \sum_{\ell=k+1}^{b_i} x^{ts}_{i\ell} - \sum_{i,c_i \le k \le d_i} \sum_{\ell=c_i}^{k-1} x^{te}_{i\ell} \Big) \le 1,\ \forall k \in M \\
& \sum_t x^{ts}_{ik} \le P^s_{ik},\ \sum_t x^{te}_{ik} \le P^e_{ik}, \forall i \in N, \forall k \in M
\end{aligned} \tag{DLP-P}$$

$$\begin{aligned}
\min_{\mu, \nu, u \ge 0, y, z}\ & \sum_{r=1}^{m} \mu_r + \sum_{i=1}^{n} \sum_{k \in S_i} P^s_{ik} \nu^s_{ik} + \sum_{i=1}^{n} \sum_{k \in E_i} P^e_{ik} \nu^e_{ik} + \sum_{i=1}^{n} \sum_{t=1}^{T} u_{it} \\
\text{s.t.}\ & y_{it} + z_{it} + u_{it} + \sum_{r=a_i}^{d_i} \mu_r \ge w_i, \forall\, i \in N,\ \forall\, t \in T, \\
& -y_{it} - \sum_{r=a_i}^{k-1} \mu_r + \nu^s_{ik} \ge 0, \forall\, i \in N,\ \forall\, t \in T,\ \forall\, k \in S_i, \\
& -z_{it} - \sum_{r=k+1}^{d_i} \mu_r + \nu^e_{ik} \ge 0, \forall\, i \in N,\ \forall\, t \in T,\ \forall\, k \in E_i.
\end{aligned} \tag{DLP-D}$$

By setting y, z, u to 0, (DLP-D) is equivalent to,

$$\min_{\mu, u \geq 0} \quad \sum_{r=1}^{m} \mu_r + \sum_{i=1}^{n} \sum_{k \in S_i} P_{ik}^s \sum_{r=a_i}^{k-1} \mu_r + \sum_{i=1}^{n} \sum_{k \in E_i} P_{ik}^e \sum_{r=k+1}^{d_i} \mu_r \tag{LP-Dpes}$$
$$\text{s.t.} \quad \sum_{r=a_i}^{d_i} \mu_r \geq w_i, \ \forall i \in N.$$

Consider the pessimistic interval graph G_{pes}, where each task i occupies $[a_i, d_i]$, the widest possible interval, and the corresponding LP (LP-D) defined on G_{pes}; let μ_{pes} be its optimal solution, and $\alpha_{\text{pes}} := \alpha(G_{\text{pes}}) = \sum_r [\mu_{\text{pes}}]_r$ its optimal value. Since $\mu = \mu_{\text{pes}}, y = z = u = 0$ is feasible for (DLP-D), the optimal value of (DLP-D), α^*, is at most

$$\alpha_{\text{pes}} + \sum_{i=1}^{n} \sum_{k \in S_i} P_{ik}^s \sum_{r=a_i}^{k-1} [\mu_{\text{pes}}]_r + \sum_{i=1}^{n} \sum_{k \in E_i} P_{ik}^e \sum_{r=k+1}^{d_i} [\mu_{\text{pes}}]_r,$$

where

$$\begin{aligned} \sum_{i=1}^{n} \sum_{k \in S_i} P_{ik}^s \sum_{r=a_i}^{k-1} [\mu_{\text{pes}}]_r &= \sum_r [\mu_{\text{pes}}]_r \sum_i \sum_{k, r \in S_i, k > r} P_{ik}^s \\ &= \sum_r [\mu_{\text{pes}}]_r \sum_{i \text{ s.t. } r \in S_i} (1 - P_{ir}^o), \end{aligned}$$

and

$$\sum_{i=1}^{n} \sum_{k \in E_i} P_{ik}^e \sum_{r=k+1}^{d_i} [\mu_{\text{pes}}]_r = \sum_r [\mu_{\text{pes}}]_r \sum_{i \text{ s.t. } r \in E_i} (1 - P_{ir}^o).$$

Thus,

$$\sum_r [\mu_{\text{pes}}]_r \leq \alpha^* \leq \sum_r [\mu_{\text{pes}}]_r \left(1 + \sum_{i \text{ s.t. } r \in S_i} (1 - P_{ir}^o) + \sum_{i \text{ s.t. } r \in E_i} (1 - P_{ir}^o) \right).$$

The bound is tight when the problem is deterministic because for every $r \in M$, $\sum_{i \text{ s.t. } r \in S_i} (1 - P_{ir}^o) = \sum_{i \text{ s.t. } r \in E_i} (1 - P_{ir}^o) = 0$.

We conclude this section by considering the special case where tasks only occupy one slot, $s_i = e_i$.

Proposition 2.3. *If each task always occupies only a single slot, a greedy policy is optimal.*

Proof. We use an adversarial argument. Consider an adversary that places tasks N into $[m]$ positions in advance according to their distributions D_i^s, D_i^e for each $i \in N$. Since the length of each task is 1, if the positions of the tasks are known, it is optimal to pick the largest weighted task of each position. Without loss of generality, assume the weights of the largest weighted task of each position to be $w_1 \geq w_2 \geq \ldots w_m \geq 0$. Without knowing the positions, the greedy policy would pick $w_1, w_2, \ldots, w_m$ which is equivalent to the case where positions are known. Thus, for any realization, the greedy policy returns the set of non-conflicting tasks with the maximum weights. □

3 Conservative DSRSE

In this section, we consider another version of DSRSE, where a task is deleted whenever it possibly conflicts with the committed tasks.

We have M, N, w, D, like in DSRSE. Once s_i, e_i of a committed task are determined, for any remaining task j, instead of flipping a coin, we delete j if any slot in its support is occupied by committed tasks; otherwise, keep j. The goal remains to schedule tasks sequentially and maximize the expected total weight of scheduled tasks. We call this problem *conservative dynamic scheduling with random start and end times (CDSRSE)*.

Example 3.1. *Consider an instance with $N = [2], M = [3]$, and unit weights, where $D_1^s = \mathbb{1}_{\{1\}}, D_1^e = \mathbb{1}_{\{2\}}$, $D_2^s = U(\{2,3\}), D_2^e = \mathbb{1}_{\{3\}}$. If we commit to task 1 first, then task 1 occupies $[1,2]$. For CDSRSE, task 2 is always deleted, since position 2 is in its support, which is now occupied by task 1.*

Theorem 3.2. *CDSRSE is also NP-hard.*

Proof. As in the previous proof, consider the instance with $n+1$ tasks and $2n$ slots. Task 0 occupies slots $1,\dots,n$ with probability 1, and task i occupies slot i with probability p_i and $n+i$ with probability $q_i := 1-p_i$. As in [8], any solution for this problem is determined by when to select task 0, and what is selected before trying to select task 0. Then the problem can be formulated as

$$\max_{S\subseteq N}\left\{\sum_{i\in S} w_i + (1-\prod_{i\in S} q_i)\sum_{j\notin S} w_j + \prod_{i\in S} q_i w_o\right\}. \tag{Conserv-JS}$$

By rearranging, we get

$$\max_{S\subseteq N}\left\{\sum_{i\in N} w_i + \prod_{i\in S} q_i(w_0 - \sum_{j\notin S} w_j)\right\},$$

and by dropping constant terms and applying the natural logarithm, we get

$$\max_{S\subseteq N}\left\{\sum_{i\in S} \ln q_i + \ln(w_0 - \sum_{j\notin S} w_j)\right\}.$$

Given a set of numbers N with positive weights a_i , $i\in N$, the partitioning problem asks for a subset $S\subseteq N$ such that $\sum_{i\in S} a_i = \sum_{j\notin S} a_j$; without loss of generality, we assume that $\sum_{i\in N} a_i = 2$. By setting parameters of the problem above as $q_i = e^{-a_i}$, $p_i = 1-e^{-a_i}$, $w_i = a_i$,and $w_0 = \sum_{i\in N} a_i$,

$$\max_{S\subseteq N}\left\{-\sum_{i\in S} a_i + \ln(\sum_{i\in S} a_i).\right\}$$

Proceeding similarly to [1], there is a set $S\subseteq N$ with $\sum_{i\in S} a_i = 1$ if and only if the optimum of this problem is -1. □

Unlike DSRSE, in CDSRSE the distributions D_i^s, D_i^e are not updated for remaining tasks. Thus, D_i^s, D_i^e are independent of the stage t when i is scheduled; x_{ik}^{ts} can always be expressed as $x_{ik}^{ts} = P_{ik}^s x_i^t$, and similarly for x_{ik}^{te}, x_{ik}^{to}. Therefore, we can express the constraint $\Pr[k \text{ occupied by any } i]\le 1$ as $\sum_{i=1}^n P_{ik}^o x_i \le 1$. Consider the LP relaxation and its dual,

$$\begin{aligned}\max\quad & \sum_{i=1}^n w_i x_i \\ & \sum_{i=1}^n P_{ik}^o x_i \le 1 \\ & x\ge 0,\end{aligned} \tag{CDLP-P}$$

$$\begin{aligned}\min\quad & \sum_{r=1}^m \mu_r \\ & \sum_{r=1}^m P_{ir}^o \mu_r \ge w_i,\ \forall i\in N \\ & \mu\ge 0.\end{aligned} \tag{CDLP-D}$$

Let $p^* := \min\{P_{ir}^o : P_{ir}^o > 0, i\in N, r\in M\}$. As above, consider the pessimistic interval graph G_{pes} and the corresponding optimal solution μ_{pes} of (LP-D) defined on G_{pes}. Let $\hat{\mu} := \mu_{\text{pes}}/p^*$; then for each $i\in N$,

$$\sum_r P_{ir}^o \hat{\mu}_r \ge \sum_{r\in[a_i,d_i]} [\mu_{\text{pes}}]_r \ge w_i.$$

Thus $\hat{\mu}$ is feasible to (CDLP-D) and $\sum_r \hat{\mu}_r = \frac{1}{p^*}\alpha_{\text{pes}}$. We have $\alpha_{\text{pes}} \le OPT \le \frac{1}{p^*}\alpha_{\text{pes}}$. However, this bound is weak when p^* is small.

Remark 3.3. *We emphasize that $\sum_{i=1}^n P_{ik}^o x_i \le 1$ is not a valid inequality for the original DSRSE because the distribution of start and end times of i depends on the stage at which it is committed. Consider Example 2.2 and a policy that always schedules task 1 first and then task 2 if possible. Under this policy, the probability that slot 2 is occupied is $1\cdot x_1 = 1$, but $P_{12}^o x_1 + P_{22}^o x_2 = 1\cdot x_1 + \frac{1}{2}x_2 = 1+\frac{1}{4} > 1$. For Example 3.1, after task 1 is scheduled, the probability of scheduling task 2 is 0, so $P_{12}^o x_1 + P_{22}^o x_2 = 1$.*

4 Uniform Weights

In this section, we consider the case where task weights are uniform, $w_i = 1$ for $i\in N$. Let α_{pes} be the optimal value of the pessimistic interval graph; for G_{pes}, because of the cardinality objective there is a minimum clique cover, a set

$\mathscr{J} \subseteq M$ with $|\mathscr{J}| = \alpha_{\text{pes}}$ such that each task intersects with exactly one element in $\mathscr{J}$; this clique clique cover can be derived from (LP-D). Specifically, for each task i there exists exactly one $k \in \mathscr{J}$ such that $k \in [a_i, d_i]$. Let x be an optimal solution of (DLP-P), and α be its optimal value; by (2), we have for each $k \in \mathscr{J}$,

$$\begin{aligned}\alpha = \sum_t \sum_i x_i^t &\le 1 + \sum_t \sum_{i, P^o_{ik}=0} x_i^t + \sum_t \left(\sum_{i, a_i \le k \le b_i} \sum_{\ell=k+1}^{b_i} x_{i\ell}^{ts} + \sum_{i, c_i \le k \le d_i} \sum_{\ell=c_i}^{k-1} x_{i\ell}^{te} \right) \\ &= 1 + \sum_{i, a_i \le k \le b_i} \sum_{\ell=k+1}^{b_i} x_{i\ell}^s + \sum_{i, c_i \le k \le d_i} \sum_{\ell=c_i}^{k-1} x_{i\ell}^e + \sum_{i, P^o_{ik}=0} x_i.\end{aligned}$$

Let k_i be the unique element in $\mathscr{J}$ covering $i \in [n]$. Summing both sides over $\mathscr{J}$, we get

$$\begin{aligned}\alpha * \alpha_{\text{pes}} &\le \alpha_{\text{pes}} + \sum_{k \in \mathscr{J}} \left(\sum_{i, a_i \le k \le b_i} \sum_{\ell=k+1}^{b_i} x_{i\ell}^s + \sum_{i, c_i \le k \le d_i} \sum_{\ell=c_i}^{k-1} x_{i\ell}^e \right) + (\alpha_{\text{pes}} - 1) \sum_i x_i \\ \iff \alpha &\le \alpha_{\text{pes}} + \sum_{k \in \mathscr{J}} \left(\sum_{i, a_i \le k \le b_i} \sum_{\ell=k+1}^{b_i} x_{i\ell}^s + \sum_{i, c_i \le k \le d_i} \sum_{\ell=c_i}^{k-1} x_{i\ell}^e \right) \\ &= \alpha_{\text{pes}} + \sum_i \sum_{\ell=k_i+1}^{b_i} x_{i\ell}^s + \sum_{\ell=c_i}^{k_i-1} x_{i\ell}^e \\ &\le \alpha_{\text{pes}} + \sum_i (1 - P^o_{ik_i}),\end{aligned}$$

where the first inequality and the equality are from the fact that i is not covered by $\mathscr{J} \setminus \{k_i\}$, whose cardinality is $\alpha_{\text{pes}} - 1$. The equivalence between inequalities is derived from the uniform weights, i.e. $\sum_i x_i = \alpha$. The last line is from the constraint $\sum_t x_{i\ell}^{ts} \le P_{i\ell}^s$, and the analogous constraint for $x_{i\ell}^e$.

For CDSRSE, we have $x_{i\ell}^s = x_i P_{i\ell}^s$ and $x_{i\ell}^e = x_i P_{i\ell}^e$. Thus we have,

$$\alpha \le \alpha_{\text{pes}} + \sum_i \sum_{\ell=k_i+1}^{b[i]} x_{i\ell}^s + \sum_{\ell=c[i]}^{k_i-1} x_{i\ell}^e = \alpha_{\text{pes}} + \sum_i (1 - P^o_{ik_i}) x_i \le \alpha_{\text{pes}} + \alpha * (1 - P^o_{i^* k[i^*]}),$$

where $P^o_{i^* k[i^*]} = \min_i \{P^o_{ik_i}\} > 0$, so $\alpha \le \frac{1}{P^o_{i^* k[i^*]}} \alpha_{\text{pes}}$.

5 Computational Experiments

In this section, we test the bounds derived above. We use the following (n, m) combinations: $(8, 12)$, $(10, 15)$, $(14, 21)$, $(16, 24)$, $(18, 27)$, $(19, 29)$, $(20, 30)$, $(40, 60)$, $(80, 120)$. For each (n, m), we generate 30 instances and report average results.

For each task i, we first sample x, y uniformly and independently from $[m]$, and let $a_i = \min\{x, y\}, d_i = \max\{x, y\}$. Similarly, sample u, v uniformly and independently from $[a_i, d_i]$, and set $b_i = \min\{u, v\}, c_i = \max\{u, v\}$. The starting and ending times follow uniform distributions on $[a_i, b_i]$, $[c_i, d_i]$, respectively. Finally, we sample task weights uniformly from $[0, 1]$

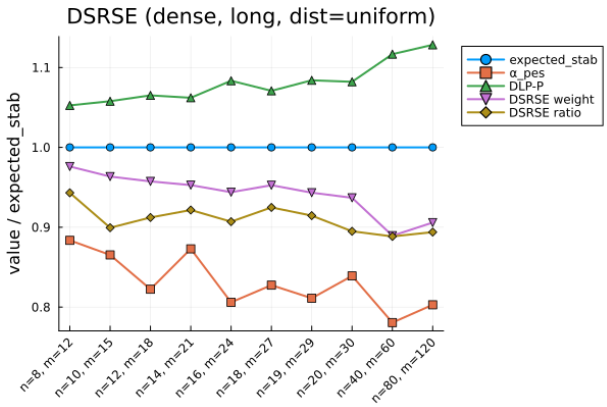
We conducted all of our computational experiments on a laptop computer with an Apple M1 Pro CPU and 16 GB of RAM, solving LPs using HiGHS [4] and performing other computations using Julia.

Since computing the optimal values of DSRSE and CSDRSE is impractical, we estimate the expected maximum-weight schedule $\mathbb{E}[\alpha(G)]$ by repeatedly sampling the start and end times of each task and solving the (deterministic) maximum independent set 1,000 times; this expected value is an upper bound for the optimal values of DSRSE and CDSRSE. We compare it with the optimal values of (DLP-P), (CDLP-P), and with α_{pes}. Due to the poor performance of the bounds $\sum_r [\mu_{\text{pes}}]_r \left(1 + \sum_{i \text{ s.t. } r \in S_i} (1 - P^o_{ir}) + \sum_{i \text{ s.t. } r \in E_i} (1 - P^o_{ir})\right)$ and α_{pes}/p^*, we do not include them in the plots.

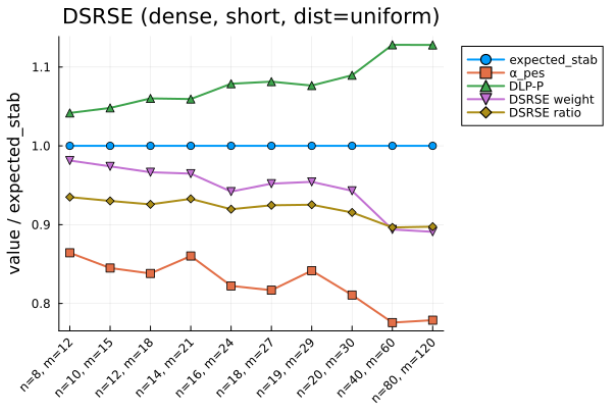
On the primal side, we consider the heuristic that picks the remaining task with the largest ratio $w_i / \mathbb{E}[\text{length of } i] = w_i / \sum_r P^o_{ir}$, which can be interpreted as the weight gained per expected occupied position; we call this heuristic *(C)DSRSE ratio*. Also, we consider the heuristic that, in stage t, maximizes $w_i - \sum_{\text{available } j \ne i} \Pr[j \text{ deleted by } i | i \text{ selected at } t] w_j$. That is, we maximize the net gained weight, accounting for potential lost weight from conflicts; we call this heuristic *(C)DSRSE weight*. For each instance, we run 1,000 simulations per policy to estimate expected performance.

We test four sets of instances for each (n, m) pair. We first generate n instances and treat this as a long, dense instance. Then we consider a sparser instance where we only keep the first $n/2$ of the generated tasks. We also double

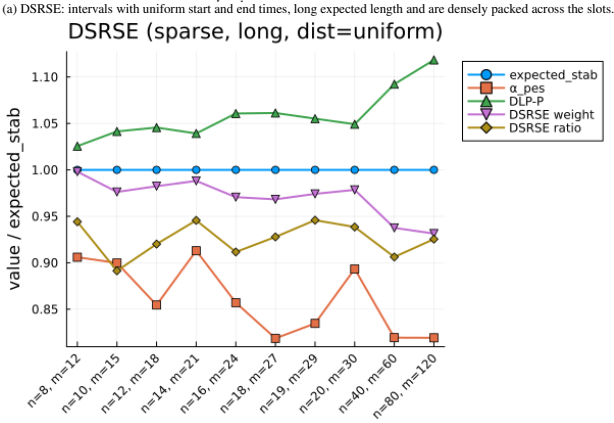
the expected length of each task by doubling $[b_i, c_i]$ from its midpoint first and then doubling $[a_i, b_i]$ and $[c_i, d_i]$ from new b_i and c_i; $[m]$ is extended if necessary to accommodate all tasks. For this case, we also consider both dense and sparse instances of n and $n/2$ tasks, respectively.



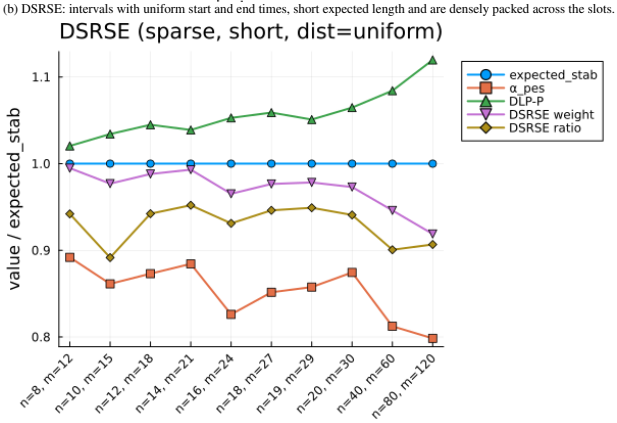
(a) DSRSE: intervals with uniform start and end times, long expected length and are densely packed across the slots.



(b) DSRSE: intervals with uniform start and end times, short expected length and are densely packed across the slots.



(c) DSRSE: intervals with uniform start and end times, long expected length and are sparsely packed across the slots.



(d) DSRSE: intervals with uniform start and end times, short expected length and are sparsely packed across the slots.

Figure 1: DSRSE

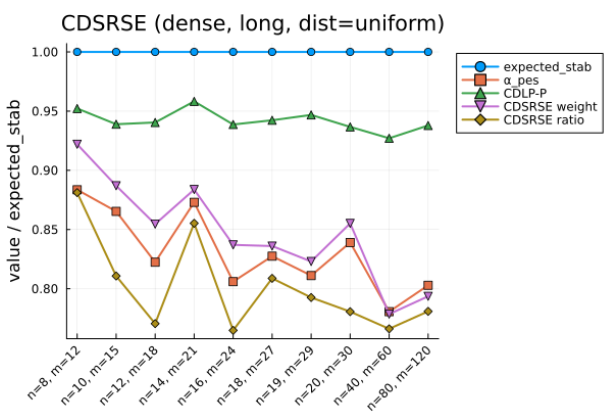
In Figure 1, we present the value of each bound and heuristic on DSRSE instances after being normalized by the expected stability number. DLP-P denotes the optimal value of (DLP-P), DSRSE-weight is the expected value of the greedy policy that schedules the largest-gained-weight task, and DSRSE-ratio is the expected value of the greedy policy that schedules the largest $w_i/\mathbb{E}[\text{length of } i]$; α_{pes} denotes the optimal value of the pessimistic realization and expected_stab denotes the expected stability number. The (DLP-P) bound is slightly looser than the expected stability number for these instances, with performance gradually worsening as the instance size increases; on average, it is roughly 7% above. We observe similar behavior for both heuristics; in addition, DSRSE-weight consistently outperforms DSRSE-ratio. On average, the former's gap is 4.5% and the latter's is 8.9%. The pessimistic stability number, α_{pes}, has an average gap with the expected stability number of almost 16%.

Figure 2 presents analogous results for CDSRSE. In this case, the bound (CDLP-P) is uniformly better than the expected stability number, with values 5% lower on average. The heuristics do not perform as well because conflicts are enforced more conservatively in this model. DSRSE-weight is still usually superior to DSRSE-ratio, but the latter does better for the largest dense instances. The heuristics' average gaps with respect to (CDLP-P) are now 9% and 14.7%.

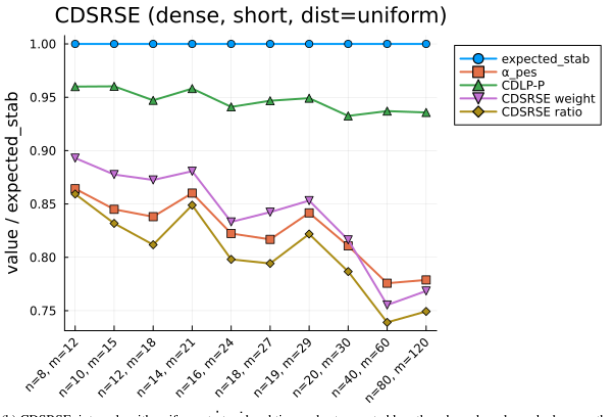
References

[1] S. Ahmed and A. Atamtürk, *Maximizing a class of submodular utility functions*, Mathematical Programming, 128 (2011), pp. 149–169.

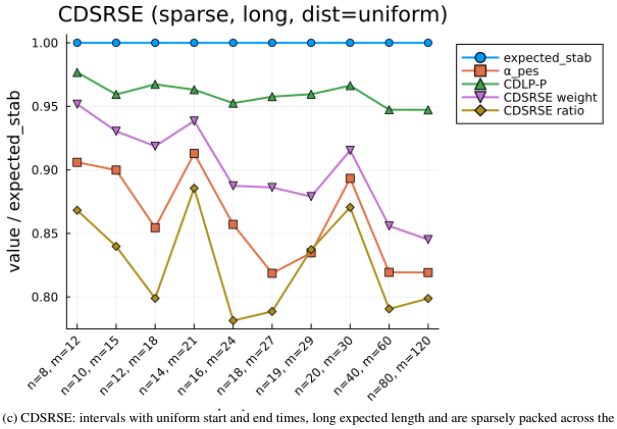
[2] J. Boyar, L. M. Favrholdt, S. Kamali, and K. S. Larsen, *Online interval scheduling with predictions*, 2025.



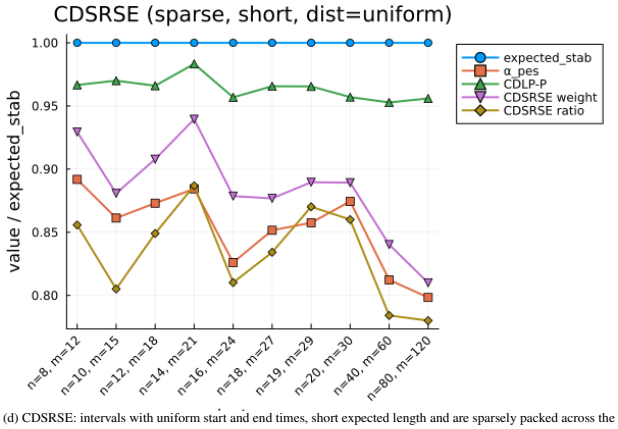
(a) CDSRSE: intervals with uniform start and end times, long expected length and are densely packed across the slots.



(b) CDSRSE: intervals with uniform start and end times, short expected length and are densely packed across the slots.



(c) CDSRSE: intervals with uniform start and end times, long expected length and are sparsely packed across the slots.



(d) CDSRSE: intervals with uniform start and end times, short expected length and are sparsely packed across the slots.

Figure 2: CDSRSE

[3] M. BRANDA AND M. MATOUŠKOVÁ, *A lagrangian relaxation algorithm for stochastic fixed interval scheduling problem with non-identical machines and job classes*, Computers & Operations Research, 164 (2024), p. 106542.

[4] Q. HUANGFU AND J. A. J. HALL, *Parallelizing the dual revised simplex method*, Mathematical Programming Computation, 10 (2018), pp. 119–142.

[5] J. KLEINBERG AND E. TARDOS, *Algorithm Design*, Addison Wesley, 2006.

[6] K. M. KOBAYASHI, *Online interval scheduling to maximize total satisfaction*, Theoretical Computer Science, 806 (2020), pp. 673–688.

[7] R. J. LIPTON AND A. TOMKINS, *Online interval scheduling*, in ACM-SIAM Symposium on Discrete Algorithms, 1994.

[8] C. MUIR AND A. TORIELLO, *Dynamic node packing*, Mathematical Programming, 196 (2022), pp. 875–906.

[9] Y. XU, R. GHUGE, AND S. PEREZ-SALAZAR, *Stochastic scheduling with abandonments via greedy strategies*, 2024.