8-2022

The on-line width of various classes of posets.

Israel R. Curbelo
University of Louisville

THE ON-LINE WIDTH OF VARIOUS CLASSES OF POSETS

By

Israel R. Curbelo
B.A., The College of New Jersey, 2016
M.A., University of Louisville, 2018

A Dissertation
Submitted to the Faculty of the
College of Arts and Sciences of the University of Louisville
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy
in Applied and Industrial Mathematics

Department of Mathematics
University of Louisville
Louisville, Kentucky

August 2022

THE ON-LINE WIDTH OF VARIOUS CLASSES OF POSETS

Submitted by

Israel R. Curbelo

A Dissertation Approved on

April 19, 2022

by the following Dissertation Committee:

Dr. Csaba Biró,
Dissertation Director

Dr. André Kézdy

Dr. Grzegorz Kubicki

Dr. David Wildstrom

Dr. Hichem Frigui

For my mother, my father and my wife

ACKNOWLEDGMENTS

I would like to thank...

Judit for opening my eyes to the beauty of mathematics,

Csaba for providing invaluable guidance in research and in life,

Andrew, Sida and Matt for being the brothers I never had,

Kaori for being a continuous source of inspiration and for never allowing me to give up,

my father for teaching me to always follow my dreams and for instilling in me the courage to do so,

my mother for always being there for me and for raising me to be the person I am today, and

my wife for pushing me to apply to grad school and for being by my side throughout the journey.

ABSTRACT

THE ON-LINE WIDTH OF VARIOUS CLASSES OF POSETS

Israel R. Curbelo

April 19, 2022

An on-line chain partitioning algorithm receives a poset, one element at a time, and irrevocably assigns the element to one of the chains. Over 30 years ago, Szemerédi proved that any on-line algorithm could be forced to use $\binom{w+1}{2}$ chains to partition a poset of width w. The maximum number of chains that can be forced on any on-line algorithm remains unknown. In the survey paper by Bosek et al., variants of the problem were studied where the class is restricted to posets of bounded dimension or where the poset is presented via a realizer of size d. We prove two results for this problem. First, we prove that any on-line algorithm can be forced to use $(2-o(1))\binom{w+1}{2}$ chains to partition a 2-dimensional poset of width w. Second, we prove that any on-line algorithm can be forced to use $(2-\frac{1}{d-1}-o(1))\binom{w+1}{2}$ chains to partition a poset of width w presented via a realizer of size d. Chrobak and Ślusarek considered variants of the on-line chain partitioning problem in which the elements are presented as intervals and intersecting intervals are incomparable. They constructed an on-line algorithm which uses at most $3w-2$ chains, where w is the width of the interval order, and showed that this algorithm is optimal. They also considered the problem restricted to intervals of unit-length and while they showed that first-fit needs at most $2w-1$ chains, over 30 years later, it remains unknown whether a more optimal algorithm exists. We improve upon previously known bounds and show that any on-line algorithm can be forced to use $\lceil \frac{3}{2}w \rceil$ chains to partition a semi-order presented in the form of its unit-interval representation. As a consequence, we completely solve the problem for $w = 3$. We also consider entirely new variants and present the results for those.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

OVERVIEW

This thesis focuses on the on-line width of various classes of posets. Dilworth's theorem states that an optimal off-line algorithm can partition any poset of width w into w chains. However, this is not always the case when the poset is presented in an on-line manner. Therefore, our goal is to find the efficiency of an optimal on-line algorithm given a class of posets with bounded width. We refer to this value as the on-line width of that class of posets.

The on-line width $\text{olw}(w)$ for the class of all posets was first considered in the 1970s, and it took nearly 30 years to confirm that the value was even bounded. Researchers continue to make progress to this day, and yet, the problem remains open. Meanwhile, researchers have studied many different classes of posets, as well as considered variants of the problem where the poset is presented with some form of geometric representation. To illustrate just how wide the field is, we provide all results presented in a recent survey with updated knowledge in the table below.

The R column indicated variants which require the poset to be presented with representation. The U column indicates variants which require the poset to be constructed in an upgrowing manner, meaning that new elements must be maximal elements of the poset. A ? in the On-Line Width column indicates that the problem remains open. On the right, we provide notation as well as the best known bounds for each of the open variants.

In Chapter 2, we focus on the on-line width $\text{olw}(w)$ of the class of all posets. In Chapter 3, we focus on the on-line width $\text{olw}(w, d)$ and $\text{olw}_R(w, d)$ of the class of d-dimensional posets without and with representation respectively. We present new strategies which provide us with the currently best lower bounds for each. In Chapter 4, we focus on the on-line width for classes of interval orders both with and without representation. We finish by presenting a new strategy which provides us with the currently best lower bound on the on-line width $\text{olws}_R(w)$ of the class of semi-orders with representation. In Chapter 5, we introduce new subclasses of interval orders which we call L-interval orders. We prove properties of the classes and provide a non-trivial lower bound for the on-line width $\text{olwi}(L, w)$ for a specific set L. Lastly, in Chapter 6, we focus on the efficiency of greedy algorithms

for various classes of posets and refer to this value as the greedy width of that class of posets. We prove that many of the results on First Fit transfer to all greedy algorithms. We end by showing non-trivial bounds for the greedy width $\mathrm{gwa}(w)$ of the class of almost-semiorders.

Class	R	U	On-Line Width					
All Orders			?	$(2-o(1))\binom{w+1}{2}$	$\leq$	$\mathrm{olw}(w)$	$\leq$	$w^{O(\log\log w)}$
		+	$\binom{w+1}{2}$					
Interval Orders			$3w-2$					
	+		$3w-2$					
		+	$2w-1$					
	+	+	w					
Semi-Orders			$2w-1$					
	+		?	$\lceil \frac{3}{2}w \rceil$	$\leq$	$\mathrm{olws}_R(w)$	$\leq$	$2w-1$
		+	$\lfloor \frac{1+\sqrt{5}}{2}w \rfloor$					
	+	+	w					
2-Dimensional			?	$(2-o(1))\binom{w+1}{2}$	$\leq$	$\mathrm{olw}(w,2)$	$\leq$	$w^{O(\log\log w)}$
	+		$\binom{w+1}{2}$					
		+	$\binom{w+1}{2}$					
	+	+	$\binom{w+1}{2}$					
d-Dimensional	+		?	$(2-\frac{1}{d-1}-o(1))\binom{w+1}{2}$	$\leq$	$\mathrm{olw}_R(w,d)$	$\leq$	$\binom{w+1}{2}^{d-1}$

CHAPTER 2

INTRODUCTION

2.1 Motivation

Suppose we have a multiprocessing machine with a set of tasks $\{t_1, t_2, t_3, t_4\}$ of which some tasks depend on the output of other tasks.

task	dependencies
t_1	-
t_2	-
t_3	t_1, t_2
t_4	t_2

Table 2.1: Dependencies between tasks t_1, t_2, t_3, t_4.

Our goal is to assign each task t to a processor p in an efficient way. Consider the following example where we assign tasks t_1 and t_2 to processor p_1, and tasks t_1 and t_4 to processor p_2.

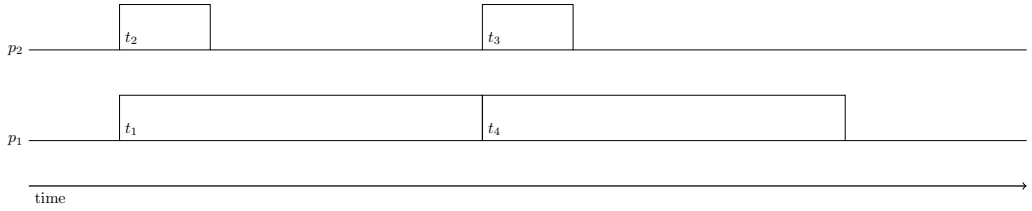


Figure 2.1: Suboptimal assignment of tasks to processors.

Notice that our first attempt results in task t_4 unnecessarily waiting for task t_1 to be completed by processor p_1. Furthermore, processor p_2 waists time waiting to begin task t_3 which depends on the output of t_1. We define a partial order on the set of tasks so that $t_i < t_j$ if and only if t_i depends on the output of t_j.

We avoid the issues from the first attempt by following the rule that tasks assigned to the same processor must form a chain. Hence, in our second attempt we assign tasks t_1 and t_3 to processor p_1, and tasks t_2 and t_4 to processor p_2.

Now suppose that a sequence tasks $t_1, \ldots, t_n$ are presented to the machine one at a time and

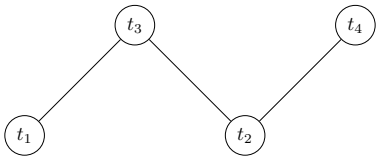


Figure 2.2: Diagram of the partial order on the set of tasks.

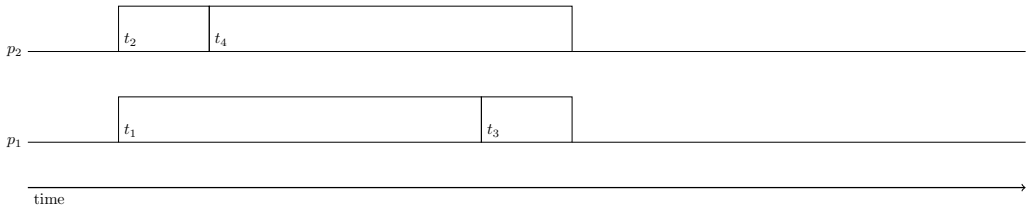


Figure 2.3: An efficient assignment of tasks to processors following the rule.

that we have no knowledge of the remaining tasks. We only know the dependencies between the tasks which have already been presented. Furthermore, we must immediately and irrevocably assign the tasks to a processor as they are presented. Then consider the following scenario. Tasks t_1 and t_2 are presented and neither task depends on the other. By our rule, we must assign the tasks to different processors p_1 and p_2 respectively. Next, a task t_3 is presented which depends on the output of both t_1 and t_2. At this point we essentially have the freedom to choose whether we assign task t_3 to processor p_1 or to processor p_2. However, regardless of our decision, a fourth task t_4 could always be presented which forces us to use a third processor p_3.

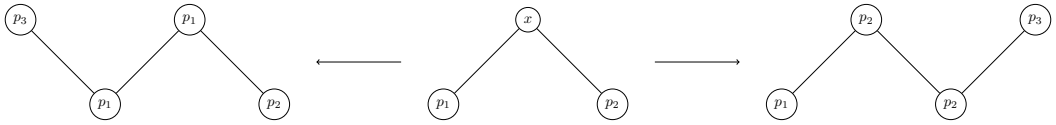


Figure 2.4: Forcing 3 processors on a set of tasks which only need 2.

Moreover, we could have used just two processors if we had knowledge of all tasks from the start. Thus our goal becomes finding the minimum number of processors needed to efficiently assign tasks to processors, or more generally, finding the minimum number of chains needed to partition a partially ordered set.

2.2 Introduction to Posets

A partially ordered set or *poset* is a pair (X, P) where X is a set and P is a reflexive, antisymmetric and transitive relation on X, that is, P satisfies the following conditions:

1. $(x, x) \in P$ for all $x \in X$ (reflexive,)

2. $(x, y) \in P$ and $(y, x) \in P \implies x = P$ (antisymmetric,) and

3. $(x, y) \in P$ and $(y, z) \in P \implies (x, z) \in P$ (transitive.)

We call X the *ground set* and P a *partial order on* X. We often refer to elements of X as points.

Example 2.2.1. *If* $X = \{a, b, c, d\}$ *and* $P = \{(a, b), (a, c), (b, d), (c, d), (a, d)\}$*, then* (X, P) *is a poset.*

Let (X, P) be a poset. For two points x and y, we write $x \leq y$ in P whenever $(x, y) \in P$ and omit the P whenever it is clear from context. We write $x < y$ whenever $x \leq y$ and $x \neq y$. We say that two distinct points x and y are *comparable* whenever $x < y$ or $y < x$. Conversely, we say that x and y are *incomparable* whenever x and y are not comparable. We use the notation $x \perp y$ to denote that x and y are comparable and use the notation $x \parallel y$ to denote that x and y are incomparable. For any two distinct points x and y, we say y covers x or $x <: y$ if $x < y$ and there is no other point z such that $x < z < y$. It is often convenient to use a graphical representation of a poset.

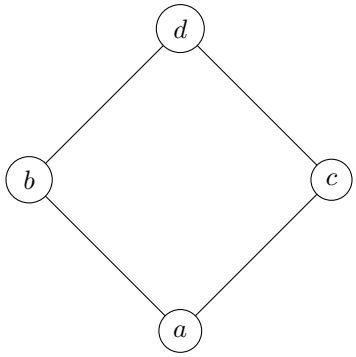


Figure 2.5: Diagram for Example 2.2.1

In Figure 2.2, we see an example of a *Hasse diagram* or simply *diagram* for Example 2.2.1. An edge between two points implies that the vertically higher point covers the vertically lower point. We see that $a <: b$, $a <: c$, $b <: d$ and $c <: d$. Notice that there is no edge between a and d. While $a < d$, it is not true that $a <: d$ since $a < b < d$.

A poset (X, P) is called a *chain* if every pair of distinct elements in X is comparable in P. A poset (X, P) is called an *antichain* if every pair of elements in X is incomparable in P. The *size* of a poset (X, P) is the cardinality of X. A point $x \in X$ is called a *maximal point* if for any point $y \in X$, $y \leq x$ or $y \parallel x$. A point $x \in X$ is called a *greatest point* if for any point $y \in X$, $y \leq x$. Similarly, a point $x \in X$ is called a *minimal point* if for any point $y \in X$, $y \geq x$ or $y \parallel x$, and a point $x \in X$ is called a least point if for any point $y \in X$, $y \leq x$. The set of all chains of a poset (X, P) is partially ordered by set inclusion and a *maximal chain* is a chain that is maximal in this poset. A chain C

is a *maximum chain* if there is no chain with more points than C. Similarly, an antichain A is a *maximum antichain* if there is no antichain with more points than A. The *height* of a poset (X, P) is the size of a maximum chain. The *width* of a poset (X, P) is the size of a maximum antichain.

Example 2.2.1 (continued)**.** *Since* $\{a, b, c\}$ *induces a maximum chain of size* 3*, the height of* (X, P) *is* 3*. Since* $\{b, c\}$ *induces a maximum antichain of size* 2*, the width of* (X, P) *is* 2*.*

For every point $x \in X$, the *downset* $D(x)$ of x is defined by $D(x) = \{y \in X : y < x \text{ in } P\}$. We call $D[x] = \{y \in X : y \leq x \text{ in } P\}$ the *closed downset* of x. Conversely, the *upset* $U(x)$ of x is defined by $U(x) = \{y \in X : x < y \text{ in } P\}$, and we call $U[x] = \{y \in X : x \leq y \text{ in } P\}$ the *closed upset* of x. When Y is a nonempty subset of X, the restriction of P to Y, denoted by $P|_Y$ is a partial order on Y and we call $(Y, P|_Y)$ a subposet of (X, P). The dual of a partial order P is denoted P^* and is defined by $P^* = \{(y, x) : (x, y) \in P\}$. The *dual* of a poset (X, P) is (X, P^*).

The following theorem is fundamental for the study of on-line chain partitioning of posets, and the reason for the terminology *on-line width*.

Theorem 2.2.1 (Dilworth [7])**.** *If* (X, P) *is a poset of width* w*, then there exists a partition* $X = C_1 \cup \ldots \cup C_w$*, where* C_i *is a chain for* $i \in \{1, \ldots, w\}$*.*

Proof. Let (X, P) be a poset of width w. We argue by induction on $|X|$. If $|X| = 1$, then the result is trivial. Assume that any poset (X_0, P_0) of width w with $|X_0| \leq k$ can be partitioned into w chains and suppose $|X| = k + 1$. We may assume that $w > 1$ since otherwise the trivial partition $X = C_1$ would satisfy the conclusion. Let C be a nonempty chain in (X, P). If $(X \setminus C, P|_{X \setminus C})$ has width less than w, then $X \setminus C$ can be partitioned into k chains $C_1, \ldots, C_k$ with $k < w$ and $C, C_1, \ldots, C_k$ would be a valid partition of X. Hence, we may assume that if C is a nonempty chain in (X, P), then $(X \setminus C, P|_{X \setminus C})$ also has width w.

Let x and y be a maximal point and a minimal point respectively with $y < x$ in P. Since $C = x, y$ is a nonempty chain, we may assume that $(X \setminus C, P|_{X \setminus C})$ also has width w and has an antichain $A = \{a_1, \ldots, a_w\}$. Since $y \notin U[A]$, $U[A] \neq X$. Since $x \notin D[A]$, $D[A] \neq X$. Also, note that $U[A] \cap D[A] = A$.

By the induction hypothesis, we can partition $U[A]$ into w chains $U_1, \ldots, U_w$. Similarly, we can partition $D[A]$ into w chains $D_1, \ldots, D_w$. Without loss of generality, we may assume that $U_i \cap D_i = a_i$ for $i \in \{1, \ldots, w\}$. Thus, $C_1, \ldots, C_w$ is the desired partition where $C_i = U_i \cup D_i$ for $i \in \{1, \ldots, w\}$. □

2.3 On-Line Width

An *on-line chain partitioning algorithm* receives a poset (X, P) in the order of its elements $x_1, \ldots, x_n$ and constructs an *on-line chain partition* $X = C_1 \cup \ldots \cup C_t$. This means that whenever a new element x_i is introduced, the chain to which it is assigned to depends solely on the poset (X, P) restricted to $\{x_1, \ldots, x_n\}$ and on the chains to which $x_1, \ldots, x_{i-1}$ were assigned to. The *efficiency* or *performance* of an on-line chain partitioning algorithm is measured in terms of the number of chains needed by an optimal off-line algorithm. By Theorem 2.2.1, every poset of width w can be partitioned into w chains by an optimal off-line chain partitioning algorithm. However, it is not always possible to partition posets of width w into w chains when the poset is presented in an on-line manner (as we saw in Section 2.1.)

We may consider each problem as a two-player coloring game. We call the first player Anna and the second player Beth. In this game, Anna constructs a poset one point at a time and Beth constructs a chain partition in an on-line manner. During round i, Anna introduces a new point x_i to the poset and describes the subposet $(X_i, P|_{X_i})$ induced by the subset $X_i = \{x_1, \ldots, x_i\}$. Beth responds by assigning x_i to one of the chains in the chain partition. We often consider the chains $C_1, \ldots, C_t$ as being different colors $1, \ldots, t$ and say that Beth assigns x_i the color t whenever x_i is assigned the chain C_t.

Therefore, the *on-line width* $\mathrm{olw}(w)$ of the class of posets of width at most w is the largest integer k for which there exists a strategy for Anna that forces any on-line chain partitioning algorithm to use k chains on a poset of width at most w. It is sometimes defined as the least integer k for which there exists an on-line chain partitioning algorithm which partitions posets of width at most w into at most k chains. It is easy to see that these definitions are equivalent.

Since a poset of width 1 is a chain, $\mathrm{olw}(1) = 1$. In 1981, Kierstead [13] proved that $5 \leq \mathrm{olw}(2) \leq 6$. Sixteen years later, Felsner [9] constructed an on-line chain partitioning algorithm which uses at most 5 chains.

Theorem 2.3.1 (Kierstead [13], Felsner [9])**.** $\mathrm{olw}(2) = 5$.

The exact value of $\mathrm{olw}(w)$ remains unknown for $w \geq 3$. In the 1970's, Schmerl asked the question of whether $\mathrm{olw}(w)$ was bounded for all $w \in \mathbb{N}$. We see in Chapter 5 that this is not a naive question. Kierstead [13] was the first to answer the question in the affirmative.

Theorem 2.3.2 (Kierstead [13])**.** *For every $w \geq 1$,* $\mathrm{olw}(w) \leq (5^w - 1)/4$

Proof. We argue inductively on the width w that Beth has a strategy $S(w)$ which uses at most

$(5^w - 1)/4$ colors. If $w = 1$, then Beth simply assigns every element to the same color. We first show that $S(2)$ uses at most 6 colors and extend the argument to prove the theorem. The strategy $S(2)$ consists of two stages.

Stage 1. In Stage 1, Beth constructs a greedy chain C. Suppose that Anna introduces a new point x. Then Beth inserts it into C if $C \cup \{x\}$ is a chain. For each $x \in X \setminus C$, let $\|_{(C,x)}$ denote the set of elements from C which are incomparable to x. Notice that the set $\|_{(C,x)}$ consists of consecutive points in C.

Suppose that $x \parallel y$. Then it must be the case that $\|_{(C,x)} \cap \|_{(C,y)} = \varnothing$, otherwise, $\{x, y, z\}$ would form an antichain of size 3 where $z \in \|_{(C,x)} \cap \|_{(C,y)}$. Hence, if $x \parallel y$, either $\|_{(C,x)} < \|_{(C,y)}$ or $\|_{(C,y)} < \|_{(C,x)}$.

Next, Beth defines a partial order R on $X \setminus C$ as follows so that $x < y$ in R if

(1) $x < y$ in P, or

(2) $x \parallel y$ and $I(x) < I(y)$,

and, $y < x$ in R if

(3) $y < x$ in P, or

(4) $x \parallel y$ and $I(y) < I(x)$.

Since for any distinct x and y in $X \setminus C$, either $x < y$, $y < x$ or $x \parallel y$, and $x \parallel y$ implies that $\|_{(C,x)} < \|_{(C,y)}$ or $\|_{(C,y)} < \|_{(C,x)}$, R is a linear extension of P.

Lastly, Beth defines an equivalence relation on $X \setminus C$ so that

(a) each equivalence class is a set of consecutive elements of $(X \setminus C, R)$, and

(b) if x y and ($x <: y$ or $y <: x$), then $\|_{(C,x)} \cap \|_{(C,y)} \neq \varnothing$

Whenever Beth inserts a new point x into $X \setminus C$, Beth puts x in the same equivalence class as y if $x <: y$ in R and $\|_{(C,x)} \cap \|_{(C,y)} \neq \varnothing$. If no such y exists, Beth puts x in the same class as z if $z <: x$ in R and $\|_{(C,x)} \cap \|_{(C,y)} \neq \varnothing$. Otherwise, Beth assigns x to its own equivalence class.

With some effort, one can verify that if S_1 and S_2 are equivalence classes of $X \setminus C$, and there are at least two other equivalence classes between them in R, then $S_1 \cup S_2$ is a chain in P.

Stage 2. In Stage 2, Beth constructs an on-line partition of $X \setminus C$ into 5 chains $C_1, \ldots, C_5$. Each chain is the union of equivalence classes, and any two classes which are subsets of the same chain have at least two other classes between them in R. When Beth assigns a point x to an entirely new class, Beth assigns it to a chain which does not contain the two classes above x nor the two classes below x. Next we generalize to $S(w)$ for $w \geq 3$.

Stage 1'. Beth begins by constructing a greedy chain C just like before. Instead of a linear extension, Beth defines an extension R of P on $X \setminus C$ so that $(X \setminus C, R)$ is a poset of width $w - 1$.

When a new point x is inserted into $X \setminus C$, Beth defines R so that $x < y$ in R if (1), (2),

(1') there exists $u \in X \setminus C$ so that $x < u$ in P and $u < y$ in R, or

(2') there exists $v \in X \setminus C$ so that $x < v$ in R and $v < y$ in P,

and, $y < x$ in R if (3), (4),

(3') there exists $u \in X \setminus C$ so that $y < u$ in P and $u < x$ in R, and

(4') there exists $v \in X \setminus C$ so that $y < v$ in R and $v < x$ in P.

Beth adds conditions (1'),(2'),(3'),(4') in order to insure that R maintains transitivity.

Stage 2'. Hence, the width of $(X \setminus C, R)$ is at most $w - 1$, for if $A = a_1, \ldots, a_m$ is an antichain in $(X \setminus C, R)$, then $\bigcap_{i=1}^{m} \|_{(C,a_i)} \neq \emptyset$. If $x \in C$ and $x \parallel a_i$ for $i = 1, \ldots, m$, then $\{x\} \cup A$ is an antichain in P. By the induction hypothesis, $S(w-1)$ partitions $X \setminus C$ into $(5^{w-1} - 1)$ subsets each of which is a chain in R. The strategy $S(2)$ will then partition a chain in R into 5 chains in P. Then Beth needs at most $(5^w - 1)/4 = 1 + 5(5^{w-1} - 1)/4$ colors. □

Nearly 30 years later, Bosek and Krawcyk [1] presented the first subexponential upper bound.

Theorem 2.3.3 (Bosek and Krawcyk [1]). *For every $w \geq 1$,* $\mathrm{olw}(w) \leq w^{13 \log w}$

This algorithm was quite involved. Bosek, Kierstead, Krawcyk, Matecki, and Smith [2] presented another subexponential bound which was easier to both implement and analyze. Moreover, it was slightly better than the previous bound.

Theorem 2.3.4 (Bosek,Kierstead,Matecki and Smith [2]). *For every $w \geq 1$,* $\mathrm{olw}(w) \leq w^{6.5 \log w + 7}$

Currently, the best upper bound came in 2021 by Bosek and Krawcyk [3].

Theorem 2.3.5 (Bosek and Krawcyk [3]). *For every $w \geq 1$,* $\mathrm{olw}(w) \leq w^{O(\log \log w)}$.

It remains unknown if there exists a polynomial bound, specifically, a $k \in \mathbb{N}$ such that $\mathrm{olw}(w) \leq w^k$ eventually.

On the other hand, Kierstead [13] also provided the first non-trivial lower bound.

Theorem 2.3.6 (Kierstead [13]). *For every $w \geq 1$,* $\mathrm{olw}(w) \geq 4w - 3$.

An unpublished argument by Szemerédi (See [14]) showed that $\mathrm{olw}(w)$ is not linear.

Theorem 2.3.7 (Szemerédi). *For every $w \geq 1$,* $\mathrm{olw}(w) \geq \binom{w+1}{2}$

Proof. We prove that any on-line algorithm can be forced to use $\binom{w+1}{2}$ chains to partition a poset of width w inductively on w. If $w = 1$, then we simply introduce a single point. If $w > 1$, then the strategy $S(w)$ consists of two stages.

Stage 1

We initialize a chain C with a point x_0. Now suppose in round $i-1$, we introduced a point x_{i-1} and the algorithm assigns it the color c_{i-1}. Let $c(C)$ denote the set of colors used to color elements of C. If $c_{i-1} \in c(C)$, then in round i, we introduce a new point x_i so that x_i is greater than every element in C but incomparable to every element not in C. If $c_{i-1} \notin c(C)$, then we update C to include c_{i-1}. If $|C| = w$, then we move onto Stage 2. Otherwise, in round i, we introduce a new point x_i so that x_i is greater than every element in C but incomparable to every element not in C, and repeat Stage 1.

Stage 2

Let S_1 denote the set of points introduced in Stage 1. In Stage 2 we play $S(w-1)$ so that every point introduced in Stage 2 is less than every point in $S_1 \setminus C$ and incomparable to every point in C. This is possible since every element introduced during Stage 1 was a maximal point the turn that it was introduced. By the inductive hypothesis, $S(w-1)$ forces $\binom{w}{2}$ colors on a poset of width at most $w-1$, and hence, (X, P) is a poset of width at most w. All that is left to show is that Stage 1 concludes with C being a rainbow chain, i.e. a chain with all distinct colors, of size w.

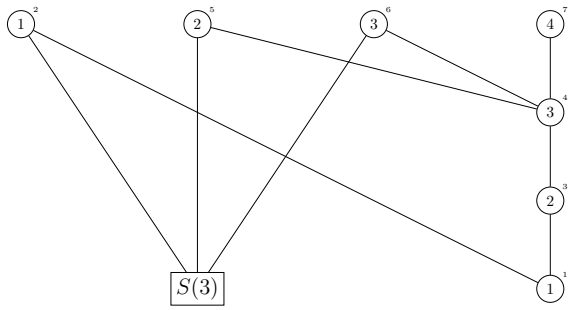


Figure 2.6: The strategy $S(4)$. The number inside the circle represents the color and the superscript represents the order in which the points were presented.

Result

Points only get added to C given that they were assigned a color not previously in C. Since the points not in C form an antichain, a new point will eventually get added to C. Hence, Stage 1 ends once C is a chain of size w where every point in C is assigned a distinct color. The width of $(S_1, P|_{S_1})$ is at most w since if it contained an antichain of size $w+1$, the antichain would contain a point colored $w+1$ and that point would have been added to C. Since the subposet $(S_2, P|_{S_2})$ constructed in Stage 2 by $S(w-1)$ is of width at most $w-1$ and is comparable to every element of S_1 except for a chain, (X, P) is a poset of width at most w. Since points in S_2 are incomparable with points in C, the strategy $S(w)$ forces a total of $\binom{w}{2} + w = \binom{w+1}{2}$ colors. □

Szemerédi's arguement remained best for over 30 years until it was improved by Bosek, Felsner, Kloch, Krawczyk, Matecki and Micek [4] to achieve a bound almost twice as good.

Theorem 2.3.8 (Bosek, Felsner, Kloch, Krawczyk, Matecki and Micek [4])**.** *For sufficiently large* w, $\mathrm{olw}(w) \geq (2 - o(1))\binom{w+1}{2}$.

In Chapter 3, we prove Theorem 2.3.8 while simultaneously proving that the resulting poset always has dimension at most 2. This improves the lower bound for a seemingly just as difficult problem of finding the on-line width for posets of bounded dimension.

2.4 On-Line Height

Before moving on, we present the dual problem of finding the on-line height. Analogously, we define the *on-line height* $\mathrm{olh}(h)$ of the class of posets of height at most h in terms of a two-player game between Alice and Bob, where Alice presents a poset of height at most h and Bob constructs an on-line antichain partition of X. First, we must justify the terminology by proving the dual of Theorem 2.2.1.

Theorem 2.4.1 (Dilworth [7])**.** *If* (X, P) *is a poset of height* h*, then there exists a partition* $X = A_1 \cup \ldots \cup A_h$*, where* A_i *is an antichain for* $i \in \{1, \ldots, h\}$.

Proof. Let (X, P) be a poset of height h. Then we prove the theorem inductively on h with the case $h = 1$ being trivial. Let $Max(X, P)$ denote the set of maximal elements in (X, P). If two elements in $Max(X, P)$ were comparable this would contradict that they are maximal. Hence, $A = Max(X, P)$ is an antichain. Moreover, $(X \setminus A, P|X \setminus A)$ is a subposet of height $h - 1$. By the induction hypothesis, $X \setminus A$ can be partitioned into antichains $A_1, \ldots, A_{h-1}$. Then $A_1, \ldots, A_{h-1}, A$ is the partition we want. □

Like the off-line version, the dual problem of finding an optimal on-line antichain partition was found to be much easier than finding an optimal on-line chain partition. In fact, the antichain problem was solved by Schmerl. The proof of the lower bound is basically identical to that of Theorem 2.3.7, so we only prove the upper bound below.

Theorem 2.4.2. *Every poset of height at most* h *can be partitioned on-line into* $\binom{h+1}{2}$ *antichains.*

Proof. Bob will maintain an antichain partition using a set of antichains $A_{(a,b)}$ indexed by pairs (a, b) of numbers $1 \leq a, b$ and $a + b \leq h + 1$. Since there are exactly $\binom{h+1}{2}$ such pairs, this suffices to prove the theorem.

When Alice presents a new point x, Beth determines the size a of the longest chain in the already presented poset that has x as its maximum element and the size b of the longest chain that has x as its minimum element. As the size of any chain in the already presented order is at most h, we get $a + b \leq h + 1$.

Now, Bob inserts x into $A_{(a,b)}$. It has to be shown that $A_{(a,b)}$ remains an antichain. Indeed, suppose that x is comparable with some y that was previously put into $A_{(a,b)}$, and say $y < x$. Membership of y in $A_{(a,b)}$ is certified by chain C of size a with maximum y. Since $C \cup \{x\}$ is a chain of size $a + 1$ with maximal element x, we have contradicted $x \in A_{(a,b)}$. In the case where $x < y$, argue with a chain of size b having y as minimum to obtain a similar contradiction. □

Corollary 2.4.3. *For every $h \geq 1$, $\mathrm{olh}(h) = \binom{h+1}{2}$.*

CHAPTER 3

ON-LINE WIDTH OF D-DIMENSIONAL POSETS

In this chapter, we focus on variants of the game where not only is the width of the poset bounded but also the dimension of the poset. Let $\mathrm{olw}(w,d)$ denote the on-line width of the class of d-dimensional posets of width at most d. The analysis of $\mathrm{olw}(w,d)$ appears to be as hard as the general problem. However, we show that the lower bound for posets of dimension $d \geq 2$ matches that of the general problem.

Theorem 3.0.1 (Biró and Curbelo (Submitted))**.** *For every pair of integers* (w,d) *such that* $2 \leq d \leq w$, $\mathrm{olw}(w,d) \leq (2-o(1))\binom{w+1}{2}$.

Let $\mathrm{olw}_R(w,d)$ denote the on-line width with representation of d-dimensional posets of width at most w. We provide the first lower bound which increases as a function of the dimension d.

Theorem 3.0.2 (Biró and Curbelo (Submitted))**.** $\mathrm{olw}_R(w,d) \geq (2-\frac{1}{d-1}-o(1))\binom{w+1}{2}$.

In Section 3.1, we provide the necessary background in the theory of dimension. In Sections 3.2 and 3.3, we define $\mathrm{olw}(w,d)$ and $\mathrm{olw}_R(w,d)$ respectively, and provide a brief history of previous work. In Section 3.4, we provide some useful notation. In Section 3.5, we define a strategy for constructing linear orders which will serve as the main building blocks for constructing the posets necessary to prove our results. Finally, in Sections 3.6 and 3.7 we prove Theorem 3.0.1 and Theorem 3.0.2 respectively.

3.1 Introduction to Dimension

When P and Q are partial orders on the same set X, we call Q an extension of P if $P \subset Q$. We call Q a linear extension of P when Q is an extension of P and (X,Q) is a chain. The set of all extensions of P is partially ordered by inclusion with P as the unique minimal element and the maximal elements being linear extensions of P. If $\{L_1,\ldots,L_t\}$ is the set of all linear extensions of P, then $\bigcap_{i=1}^{t} L_i = P$.

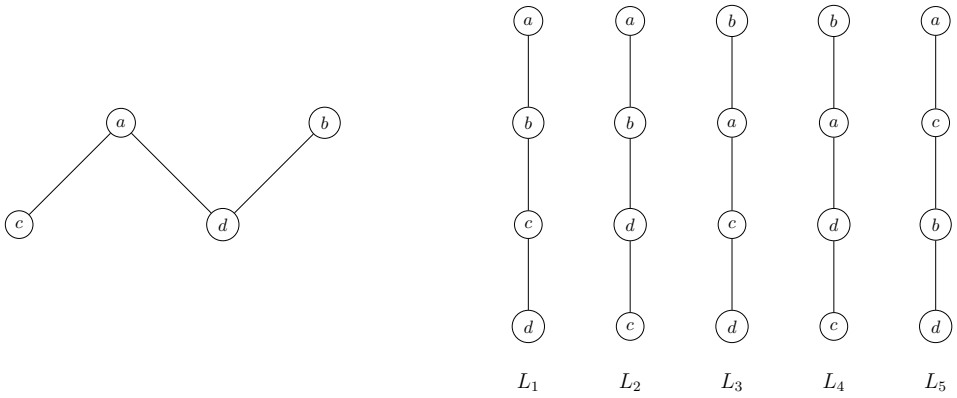


Figure 3.1: Example from Section 2.1 has 5 linear extensions.

Let (X, P) be a poset. A set $\{L_1, \ldots, L_t\}$ of linear extensions of P is called a realizer of (X, P) if it satisfies $\bigcap_{i=1}^{t} L_i = P$. Equivalently, a set $\{L_1, \ldots, L_t\}$ of linear extensions of P is called a realizer of (X, P) if it satisfies $x < y$ in P if and only if $x < y$ in L_i for $i \in \{1, \ldots, t\}$. The dimension of (X, P) is the least positive integer t for which there exists a set $\{L_1, \ldots, L_t\}$ of linear extensions of P such that $\bigcap_{i=1}^{t} L_i = P$. In other words, the dimension of a poset is the least integer t for which (X, P) has a realizer of cardinality t.

It is natural to ask if a poset can have arbitrarily large dimension. The answer to that is answered in the affirmative in the following example from Dushnik and Miller.

Example 3.1.1. *For $n \geq 3$, let $S_n = (X, P)$ be the height 2 poset with $X = \{a_1, \ldots, a_n\} \cup \{b_1, \ldots, b_n\}$, $\{a_1, \ldots, a_n\} = min(X, P)$, $\{b_1, \ldots, b_n\} = max(X, P)$, and $a_i < b_j$ in P if and only if $i \neq j$, for $i, j = 1, \ldots, n$. The poset S_n is called the standard example of an n-dimensional poset. A diagram of S_5 is given below.*

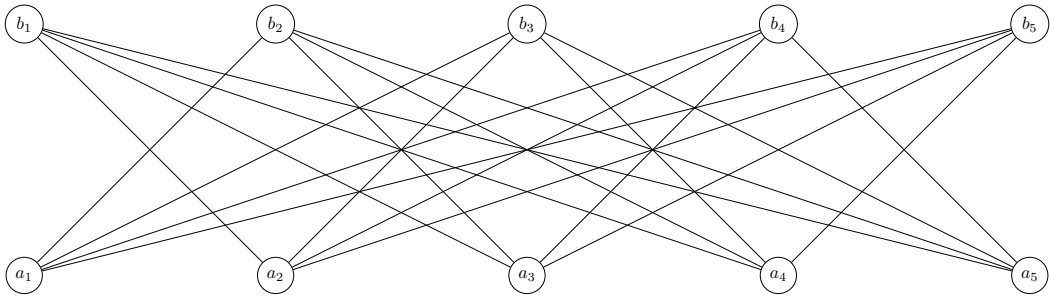


Figure 3.2: The standard example S_5 of a 5-dimensional poset.

Proof. For each $i = 1, \ldots, n$ define a linear order L_i on X by

$$L_i = [a_i, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n, b_i, a_i, b_1, \ldots, b_{i-1}, b_{i+1}, \ldots, b_n].$$

Then $\{L_1, \ldots, L_n\}$ is a realizer, so $dim(S_n) \leq n$. On the other hand, we claim that $dim(S_n) \geq n$. Define a function $f : [n] \to [t]$ as follows. For each $i \in [n]$, note that $a_i \parallel b_i$ in S_n. So we may choose $f(i)$ as some $j \in [t]$ for which $b_i < a_i$ in L_j.

We now show that f is an injection. Suppose on the contrary that there exists a pair i_1, i_2 with $1 \leq i_1 < i_2 \leq n$ and $f(i_1) = f(i_2) = j$. Then $b_{i_1} < a_{i_1}$ in L_j and $b_{i_2} < a_{i_2}$ in L_j. However, $a_{a_2} < b_{i_1}$ in P and $a_{a_1} < b_{i_2}$ in P, so $a_{i_2} < b_{i_1}$ in L_j and $a_{i_1} < b_{i_2}$ in L_j. Thus, $a_{i_2} < b_{i_1} < a_{i_1} < b_{i_2} < a_{i_2}$ in L_j which is clearly false. Since f is an injection, we conclude that $dim(X, P) = t \geq n$ as claimed. □

Our goal now is to show that dimension is well defined, and to show the connection between dimension and width. Before doing so, we prove an admittedly elementary result which turns out to be quite useful.

When (X, P) is a poset, we let $inc(X, P) = \{(x, y) \in X \times X : x \parallel y \text{ in } P\}$. An *alternating cycle* in (X, P) is a sequence $\{(x_i, y_i) : i \in [k]\}$ of ordered pairs from $inc(X, P)$ with $y_i \leq x_{i+1}$ in P for $i \in [k-1]$ and $y_k \leq x_1$ in P. The integer k is called the length of the cycle. A *strict alternating cycle* is an alternating cycle if $y_i \leq x_j$ in P if and only if $j = i+1$, or $i = k$ and $j = 1$. For a binary relation R on a set X, let $tr(R)$ denote the *transitive closure* of R. Then $tr(R) = \{(x, y) \in X \times X :$ there exists a sequence $u_1, \ldots, u_n$ so that $(u_i, u_{i+1}) \in R$ for $i \in [n-1], u_1 = x, u_n = y\}$.

Lemma 3.1.1 (Trotter and Moore [19])**.** *Let (X, P) be a poset and let $S \subseteq inc(X, P)$. Then the following statements are equivalent:*

1. *$tr(P \cup S)$ is not a partial order on X.*
2. *S contains an alternating cycle.*
3. *S contains a strict alternating cycle.*

Proof. To prove the theorem, we show that (3) $\implies$ (2), (2) $\implies$ (1), and (1) $\implies$ (3). Since every strict alternating cycle is an alternating cycle, (3) $\implies$ (2).

(2) $\implies$ (1) Let $T = tr(P \cup S)$. Suppose that $(x_1, y_1), \ldots, (x_k, y_k)$ is an alternating cycle in (X, P) of length k. Since $(y_i, x_{i+1}) \in P$ for $i \in [k-1]$, (x_i, x_{i+1}) in T for $i \in [k]$. Suppose on the contrary T is a partial order on X. Then Q is antisymmetric, so $x_1 = \ldots = x_k$. However $(x_i, y_i) \in T$ and $(y_i, x_{i+1}) \in T$ implies $x_i = y_i = x_{i+1}$ for $i \in [k]$. This contradicts $x_i \parallel y_i$ in P for $i \in [k]$. Thus (2) $\implies$ (1).

(1) $\implies$ (3) Since T is reflexive and transitive, but not a partial order, it must not be antisymmetric. Hence, we may choose a sequence $u_1, \ldots, u_n$ of points in X with minimal n of which not all

are equal and $(u_i, u_{i+1}) \in P \cup S$ for $i \in [n]$. Since P is antisymmetric and $P \cap S = \emptyset$, we may assume that $(u_1, u_2) \in S$. Then set $x_1 = u_1$ and $y_1 = u_2$. Suppose we define a pair (x_i, y_i) with $y_i = u_j$ for some $j \in \{2, \ldots, k\}$. If $(u_j, u_{j+1}) \in S$, set $x_{i+1} = y_i = u_j$ and $y_{i+1} = u_{j+1}$. If $(u_j, u_{j+1}) \in P$, then $(u_{j+1}, u_{j+2}) \in S$, otherwise we contradict minimality of n. In this case, set $x_{i+1} = u_{j+1}$ and $y_{i+1} = u_{j+2}$. It is easy to verify that this yields a strict alternating cycle. □

We may now achieve our goals for this section. First we show that every poset has a linear extension.

Proposition 3.1.2 (Szpilrajn)**.** *Let (X, P) be a poset. Then P has a linear extension.*

Proof. Let (E, R) be the poset of all extensions of P ordered by inclusion. Let M be a maximal element in (E, R). If M is not a linear extension, then choose $(x, y) \in inc(X, M)$ and set $S = \{(x, y)\}$. S does not contain an alternating cycle since $|S| = 1$. So $P \subseteq M \subset tr(M \cup S)$. This contradicts M being maximal and thus M must be a linear extension of P. □

Next we show every poset has a realizer, and thus, that the concept of dimension is well defined.

Theorem 3.1.3. *Let (X, P) be a poset and let E be the set of all linear extensions of P. Then $P = \cap E$*

Proof. If (X, P) is a chain, then $E = \{P\}$ and so $P = \cap E$. So we may assume (X, P) is not a chain. For each $(x, y) \in inc(X, P)$, let $S(x, y) = \{(x, y)\}$ Then $S(x, y)$ does not contain an alternating cycle since $|S| = 1$. Let $L(x, y)$ be any linear extension of $tr(P \cup S(x, y))$. Then $P \subseteq \cap E \subseteq \cap_{(x,y) \in inc(X,P)} L(x, y) \subseteq P$. □

Theorem 2.2.1 is often referred to as Dilworth's Theorem. However, Theorem 2.2.1 was in fact presented as a lemma which was used to prove Dilworth's main theorem. In particular, Dilworth's Theorem shows that the width of a poset cannot exceed its dimension. Before we prove the theorem, we prove a lemma by Hiraguchi.

Lemma 3.1.4 (Hiraguchi [10])**.** *Let (X, P) be a poset and let $C \subseteq X$ be a chain. Then there exist linear extensions L_1 and L_2 of P so that:*

1. *$y < x$ in L_1 for every $x, y \in X$ with $x \in C$ and $x \parallel y$ in P, and*
2. *$x < y$ in L_2 for every $x, y \in X$ with $x \in C$ and $x \parallel y$ in P.*

Proof. Let $S_1 = \{(y, x) \in X \times X : x \in C, x \parallel y \text{ in } P\}$ and $S_2 = \{(x, y) \in X \times X : x \in C, x \parallel y \text{in} P\}$. Neither S_1 nor S_2 contain a strict alternating cycle. So $Q_1 = tr(P \cup S_1)$ and $Q_2 = tr(P \cup S_2)$ are partial orders on X. Take L_1 and L_2 as linear extensions of Q_1 and Q_2 respectively. □

We can now prove Dilworth's Theorem.

Theorem 3.1.5 (Dilworth [7]). *Let (X, P) be a poset. Then*

$$dim(X, P) \leq width(X, P)$$

.

Proof. Let (X, P) be a poset and let w be the width of (X, P). By Dilworth's chain partitioning theorem, X can be partitioned into w chains $C_1, \ldots, C_w$. For each chain C_i we use Lemma 3.1.4 to choose a linear extension L_i in P so that $x < c$ in L_i whenever $c \in C$ and $c \parallel x$ in P. In order to show that $\{L_i, \ldots, L_w\}$ is a realizer of P, it suffices to show that for every $(x, y) \in inc(X, P)$, there exists a $j \in [w]$ with $y < x$ in L_j. Let $(x, y) \in inc(X, P)$. Then $x \in C_j$ for some $j \in [w]$. By Lemma 3.1.4, $y < x$ in L_j. □

3.2 On-Line Width of d-Dimensional Posets

The on-line width $\text{olw}(w, d)$ of the class of d-dimensional posets of width at most w is the largest integer k for which there is a strategy that forces any on-line algorithm to use k chains to partition a poset of this class. Researchers have found the analysis of the on-line chain partition game restricted to d-dimensional posets to be as hard as the general problem and no better upper bound is known for this class (even for $d = 2$). In [4], a strategy that forces $\binom{w+1}{2}$ chains is provided. However, they include the restriction that the poset must also be contructed in an upgrowing manner. It can be shown that the poset construct using Szemeredi's argument in Theorem 2.3.7 has dimension at most 2. In Section 3.6, we prove Theorem 3.0.1 improves the lower bound to match that of the general problem.

3.3 On-Line Width of d-Dimensional Posets with Representation

The on-line width $\text{olw}_R(w, d)$ of the class of d-dimensional posets of width at most w with representation is the largest integer k for which there is a strategy that forces any algorithm to use k chains to partition a poset of this class presented via its embedding in $\mathbb{R}^d$ or equivalently, by providing on-line a realizer of size d. This variant was first analyzed by Kierstead, McNulty and Trotter [15]. They constructed an on-line algorithm which uses at most $\binom{w+1}{2}^{d-1}$ chains.

Theorem 3.3.1 (Kierstead, McNulty and Trotter [15]). *For every pair of integers (w, d) such that $1 \leq d \leq w$, $olw_R(w, d) \leq \binom{w+1}{2}^{d-1}$*

Proof. To prove the theorem, we show that Beth has a strategy which uses at most $\binom{w+1}{2}^{d-1}$ chains to partition a poset of width at most w and dimension d inductively on d. If $d = 1$, then Anna presents a chain and Beth colors every point the same color.

Let (X, P) be the presented poset of width w and dimension $d > 1$, and let $\{L_1, \ldots, L_d\}$ be the presented realizer. Define the partial order P^2 on X by $P^* = L_1 \cap \ldots \cap L_d^*$. Note that every chain in P^* is an antichain in P and so $height(X, P) \leq width(X, P) \leq w$. If Y induces an antichain in P^*, then the poset induced by Y in P is $(Y, L_1 \cap \ldots \cap L_{d-1}|_Y)$. Therefore, Y induces a subposet of (X, P) with dimension at most $d - 1$.

During the game, Beth uses Schmerl's algorithm to provide an on-line antichain partition of (X, P^*) of size at most $\binom{h+1}{2} \leq \binom{w+1}{2}$ where h is the height of (X, P^*). Each antichain A in the antichain partition induces a subposet $(A, P|_A)$ which has width at most w and $L_1|_A, \ldots, L_{d-1}|A$ is a realizer of size $d - 1$. Thus, it can be partitioned recursively into $\binom{w+1}{2}^{d-2}$ chains. Altogether Beth uses at most $\binom{w+1}{2}^{d-2}$ chains. □

Until recently, the best strategy forced only $\binom{w+1}{2}$ chains. In Section 3.7, we construct a strategy for every integer $d > 1$ and provide the first lower bound which depends on the dimension of the poset constructed. More importantly, it provides the best known lower bound for $\text{olw}_R(w, d)$

3.4 Notation

Let (X, P) be a poset. If $R \subset X$, then $D[R]$ is the union of the down-sets of each point in R. Let U and V be disjoint subsets of X. We say that $U < V$ if for any point $u \in U$ and any point $v \in V$, $u < v$. We say that U and V are completely comparable if for any point $u \in U$ and any point $v \in V$, u and v are comparable. Similarly, we say that U and V are completely incomparable if for any point $u \in U$ and any point $v \in V$, u and v are incomparable.

Suppose Anna has constructed the poset (X, P) and Beth has assigned every point $x \in X$ a color of a chain in the partition. Let x be an arbitrary point in X. We let $\phi(x)$ denote the round that x was introduced and $c(x)$ denote the color or chain to which x was assigned to. That is, if x was introduced in round i and was assigned the color j, then we say $\phi(x) = i$ and $c(x) = j$. If U is a subset of X, then we let $\|U\|$ denote the number of distinct colors in U. More specifically, $\|U\| = |\{k : c(u) = k \text{ for some } u \in U\}|$. Finally we call U a rainbow set if all points in U were assigned a different color, that is, if $\|U\| = |U|$.

3.5 Algorithms for Constructing Linear Orders

We present two algorithms $L_\alpha(k, w)$ and $L_\beta(k, w)$. On their own, each one merely constructs a chain. However, together they provide a realizer of cardinality 2 forcing $\binom{w+1}{2}$ colors. More importantly, they later serve as key building blocks for proving Theorems 3.0.1 and 3.0.2.

3.5.1 The First Linear Order Algorithm

Let w and k be positive integers such that $k \leq w$. We define the algorithm $L_\alpha(k, w)$ in two stages. For the entirety of this Chapter, we use the notation S_i to denote the set of points introduced in Stage i.

3.5.2 Stage 1

Suppose that during round $i-1$, Anna has constructed a chain L_α on the set of points $\{x_1, \ldots, x_{i-1}\}$ and every point has been assigned a color from $\{1, \ldots, c\}$ with $c < w$. If $k < w$, then in round i, Anna introduces a new point x_i and places it at the top of L_α. If $k = w$, then in round i, Anna traverses up L_α and inserts a new point x_i immediately below the first point y such that $c(y) = c(z)$ for some point $z < y$. (If no such point exists, Anna traverses all the way up L_α and the new point will be placed at the top.)

If Beth declares $c(x_i) = w$, then Anna moves onto Stage 2. Otherwise, Anna repeats Stage 1.

3.5.3 Stage 2

Suppose Stage 1 ends in round N. If $k < w$, Anna plays $L_\alpha(k, w-1)$ completely under x_1 so that $S_2 < S_1$. If $k = w$, there are two cases. If x_N is the top element in L_α, then Anna plays $L_\alpha(w-1, w-1)$ completely above x_N. Otherwise, if $x_N <: y$ in L_α, then Anna plays $L_\alpha(w-1, w-1)$ completely above x_N and completely below y so that $x_N < S_2 < y$.

3.5.4 The Second Linear Order Algorithm

Let w and k be positive integers such that $k \leq w$. We define the algorithm $L_\beta(k, w)$ in two stages.

3.5.5 Stage 1

Suppose that during round $i-1$, Anna has constructed a chain L_β on the set of points $\{x_1, \ldots, x_{i-1}\}$ and every point has been assigned a chain from $\{1, \ldots, c\}$ with $c < w$. If $k < w$, then in round

i, Anna traverses up L_β and inserts a new point x_i immediately below the first point y such that $c(y) = c(z)$ for some point $z < y$ (or at the top, if no such point exists). If $k = w$, then in round i, Anna introduces a new point x_i and places it at the top of L_β.

If Beth declares $c(x_i) = w$, then Anna moves onto Stage 2. Otherwise, Anna repeats Stage 1.

3.5.6 Stage 2

Suppose Stage 1 ends in round N. If $k < w$ there are two cases. If x_N is the top element in L_β, then Anna plays $L_\beta(k, w-1)$ completely above x_N. Otherwise, if $x_N <: y$ in L_β, then Anna plays $L_\beta(k, w-1)$ completely above x_N and completely below y so that $x_N < S_2 < y$ in L_α. If $k = w$, Anna plays $L_\beta(w-1, w-1)$ completely below x_1 so that $S_2 < S_1$.

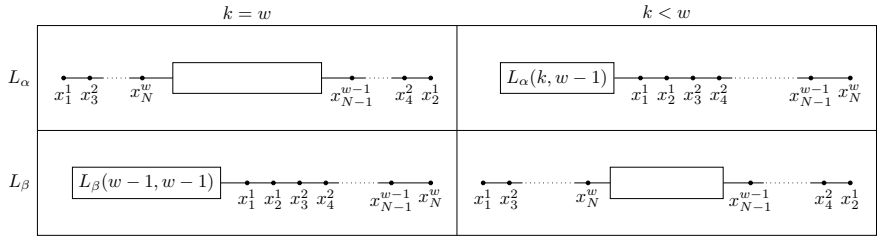


Figure 3.3: $R(k, w)$ on a greedy algorithm with colors as superscipts.

3.5.7 A Strategy for Anna

We now define a strategy $R(k, w)$ which constructs a poset $(X, L_\alpha \cap L_\beta)$ by using $L_\alpha(k, w)$ to construct L_α and $L_\beta(k, w)$ to construct L_β. The following property is elementary to check but is stated for emphasis.

Proposition 3.5.1. *Let w, k_1 and k_2 be positive integers such that $k_1 < k_2 \leq w$. The strategies $R(k_1, w)$ and $R(k_2, w)$ construct the same poset for any on-line chain partitioning algorithm.*

In the case when $k = w$, we write $R(w)$, $L_\alpha(w)$, and $L_\beta(w)$ instead of $R(w, w)$, $L_\alpha(w, w)$, and $L_\beta(w, w)$ for convenience. We prove the following theorem for completeness, and to illustrate our terminology. We use the term *rainbow chain* for a chain in the poset in which every element received different color.

Theorem 3.5.2 (Szemerédi)**.** *The strategy $R(w)$ forces $\binom{w+1}{2}$ colors on a poset (X, P) of width w.*

Proof. We argue by induction on the positive integer w. If $w = 1$, then $R(1)$ simply introduces a single point. Suppose $w > 1$. Let x, y and z be three distinct points introduced in Stage 1. Suppose

that $c(x) = c(y) = c(z)$. Without loss of generality, we may assume that $x < y < z$ in P and hence in both L_α and L_β. The algorithm $L_\beta(k)$ guarantees that $\phi(x) < \phi(y) < \phi(z)$ but the round z was introduced, it would have been inserted below y in L_α which is a contradiction. Hence, there are at most two points in S_1 assigned the same color. This implies that Stage 1 ends with Anna forcing w colors.

Suppose in round $i-1$, the point x_{i-1} was introduced and assigned the color a. Now suppose that in round i, the point x_i is introduced. If a is a new color, then clearly x_i must be placed immediately above x_{i-1} in L_α. If a is an old color, then there exists a unique point y such that $c(y) = a$. Since $\phi(y) < \phi(x_{i-1})$, $y < x_{i-1}$ in L_β and consequentially in L_α. Hence, x_i must be placed immediately below x_{i-1} in L_α. It is easy to see that if z is the last point introduced in Stage 1, then the down-set $D[z]$ of z induces a rainbow chain of size w and $S_1 \setminus D[z]$ induces an anti-chain of size at most $w-1$. Since every point introduced in Stage 2 is incomparable to $D[z]$, and by the induction hypothesis, Stage 2 forces $\binom{w}{2}$ colors on the poset $(S_2, P|_{S_2})$ of width $w-1$, the strategy $R(w)$ forces $\binom{w+1}{2}$ colors on a poset of width w. □

The previous proof highlights the rainbow chain induced by the down-set $D[z]$ of the last point z introduced in Stage 1. By the recursive nature of $R(k, w)$, we are actually always guaranteed a sequence of rainbow chains $C_1, \ldots, C_w$ satisfying the following property.

We say that a sequence of chains $C_1, \ldots, C_w$ has the *Rainbow Property* if it satisfies the following conditions:

1. If $x \in C_i$ and $y \in C_j$ for $i \neq j$, then $x \| y$.
2. If x and y are distinct points in $\bigcup_{i=1}^{w} C_i$, then $c(x) \neq c(y)$.
3. $|C_i| = i$ for every $1 \leq i \leq w$.
4. If $x \in \bigcup_{i=1}^{w} C_i$ and $y < x$ in P, then $y \in \bigcup_{i=1}^{w} C_i$.

In other words, $C_1, \ldots, C_w$ induce incomparable rainbow chains of size $1, \ldots, w$ respectively whose union is a rainbow set, and if $C = \bigcup_{i=1}^{w} C_i$, then $D[C] = C$.

We state and prove the following lemma for completeness.

Lemma 3.5.3. *Let w be a fixed positive integer. The strategy $R(k, w)$ constructs a poset (X, P) in such a way that X contains a sequence of chains $C_1, ..., C_w$ which has the Rainbow Property.*

Proof. It suffices to show that the strategy $R(w)$ constructs the desired poset. We argue by induction on the positive integer w. If $w = 1$, then Anna only introduces a single point x and $X = \{x\} = C_1$.

Suppose $w > 1$ and Anna plays the strategy $R(w)$ which results in the poset (X, P).

By the induction hypothesis, the set $X|_{S_2}$ contains a sequence of chains $C_1, \ldots C_{w-1}$ with the Rainbow Property. Let z denote the last point introduced in Stage 1 of $R(w)$ and let $C_w = D[z]$. We show that $C_1, \ldots, C_w$ satisfies each condition.

(1) Let x and y be distinct points such that $x \in C_i$ and $y \in C_j$ for $i \neq j$. If $i \neq w$ and $j \neq w$, then by the induction hypothesis, x and y are incomparable in P. Without loss of generality, suppose $j = w$. Since $y \leq z < x$ in L_α and $x < y$ in L_β, x and y are incomparable in P.

(2) Let x and y be distinct points in $\bigcup_{i=1}^{w} C_i$. If x and y are in distinct chains, then by (1), x and y are incomparable and hence $c(x) \neq c(y)$. Suppose x and y are points in the same chain C_i for some positive integer $i \leq w$. If $i < w$, then by the induction hypothesis, $c(x) \neq c(y)$. Suppose $i = w$. Without loss of generality, we may assume $x < y < z$ in L_α. Since z was introduced after x and y, $c(x) \neq c(y)$.

(3) From Lemma 3.2, we know that $|C_w| = w$. By the induction hypothesis, $|C_i| = i$ for $1 \leq i \leq w - 1$.

(4) Let $C = \bigcup_{i=1}^{w} C_i$. By definition, $D[C_w] = D[z] = C_w$. By the the induction hypothesis, $D[C \cap S_2] = C \cap S_2$. Since $C_w = C \cap S_1$, $D[C] = C$.

Thus, $C_1, \ldots, C_w$ has the Rainbow Property and the proof is complete. □

While the proof to the following lemma is not hard, everything up to this point was set up so that it would hold true as it is the key to proving Theorems 3.0.1 and 3.0.2.

Lemma 3.5.4. *Let (X, P) be a poset constructed by $R(k, w)$ with representation $\{L_\alpha, L_\beta\}$. Suppose $C_1, \ldots, C_w$ is the sequence of chains in X that has the Rainbow Property. If u and v are distinct points such that $u \in C_k$ and $v \in X \setminus C_k$, then $u < v$ in L_α.*

Proof. Let u and v be distinct points such that $u \in C_k$ and $v \in X \setminus C_k$. We argue by induction on the positive integer w. If $w = 1$, then $R(k, w)$ ends after introducing a single point x so that $X = \{x\} = C_1$.

Suppose $w > 1$ and $k \leq w$. If $k < w$, then in Stage 2, Anna plays $L_\alpha(k, w-1)$ completely below S_1 in L_α. By the induction hypothesis, $u < v'$ in L_α for $v' \in S_2 \setminus C_k$. Since $S_2 < S_1$ in L_α, $u < v$ in L_α. If $k = w$, then in Stage 2, Anna plays $L_\alpha(w-1, w-1)$ completely above C_w in L_α. Thus $u < v$ in L_α. □

3.6 Proof of Theorem 3.0.1

Taking inspiration from the techniques used in [4], we modify the strategy $R(k, w)$ to obtain a new strategy $S(w)$ for Anna which will force Beth to use $(2-o(1))\binom{w+1}{2}$ colors on a 2-dimensional poset (X, P) of width w.

3.6.1 The Strategy for Anna

We define the strategy $S(w)$ for Anna recursively on the positive integer w. The strategy $S(w)$ is completed in three stages. Anna constructs a realizer R of size $2w$ during the first two stages but only presents the poset (X, P) where $P = \cap R$. In Stage 3, Anna finishes the game by playing $S(w-1)$ on the remaining points in a specific way. After the game is over, we show that only two linear extensions in R are needed to realize (X, P). Let $w > N$ for some sufficiently large N.

3.6.2 Stage 1

For each positive integer $k \leq w$, Anna constructs two linear orders A_k and B_k by following the algorithms $L_\alpha(k, w)$ and $L_\beta(k, w)$ respectively.

Notice that $A_k \cap B_k = P$ for every $k \leq w$. The set S_1 contains a sequence of chains $C_1, \ldots, C_w$ with the Rainbow Property. Moreover, if $u \in C_k$ and $v \in S_1 \setminus C_k$, then $u < v$ in A_k for every $k \leq w$.

3.6.3 Stage 2

For every positive integer $k \leq w$, Anna updates A_k and B_k by applying the dual algorithms $L^*_\beta(w, w)$ and $L^*_\alpha(w, w)$ completely under S_1 in A_k and B_k respectively so that $S_2 < S_1$ in both A_k and B_k. The set S_2 contains a sequence of chains $D_1, \ldots, D_w$ with the Rainbow Property with respect to the dual P^* of P. Moreover, if $u \in D_w$ and $v \in S_2 \setminus D_w$, then $v < u$ in B_k for every $k \leq w$.

3.6.4 Stage 3

We let t denote an integer such that $\|C_t \cup D_w\| > 2w - \sqrt{2w} = (2 - o(1))w$. (A routine counting argument shows that such t always exists.) Anna plays $S(w-1)$ for the remainder of the game in such a way that $S_2 \setminus D_w < S_3 < S_1 \setminus C_t$ but S_3 and $C_t \cup D_w$ are completely incomparable in P.

3.6.5 The Result

Let C be a maximal chain containing $C_t \cup D_w$. It is easy to see that $(S_1 \cup S_2) \setminus C$ induces a poset of width $w-1$. By the induction hypothesis, $S(w-1)$ forces $(2-o(1))\binom{w}{2}$ colors on a poset of width $w-1$. Since S_3 and $X \setminus C$ are completely comparable, $X \setminus C$ induces a poset of width $w-1$, and thus, (X,P) is a poset of width w. Moreover, since S_3 and $C_t \cup D_w$ are completely incomparable, $S(w)$ forces $(2-o(1))\binom{w+1}{2}$ colors on (X,P).

We claim that (X,P) is 2-dimensional. Notice that $A_t \cap B_t = P|_{S_1 \cup S_2}$. By the induction hypothesis, the poset $(S_3, P|_{S_3})$ is 2-dimensional. Let A and B be linear extensions of $P|_{S_3}$ such that $A \cap B = P|_{S_3}$. We define a linear extension L_1 of P in such a way that $A_t \cup A \subset L_1$ and

$$S_2 < C_t < S_3 < S_1 \setminus C_t \text{ in } L_1.$$

We define a second linear extension L_2 of P in such a way that $B_t \cup B \subset L_2$ and

$$S_2 \setminus D_w < S_3 < D_w < S_1 \text{ in } L_2.$$

Thus $R = \{L_1, L_2\}$ is a realizer of P of cardinality 2. This completes the proof.

3.7 Proof of Theorem 3.0.2

In this variant of the game, Anna does not have the luxury of hiding the realizer from Beth. Each round, Anna must present a poset (X,P) with representation in the form of a realizer R of size d. Hence, we must be more selective when constructing the linear extensions. We modify the strategy $R(k,w)$ again to obtain a new strategy $S(d,w)$ for Anna which will force Beth to use $(2-\frac{1}{d-1}-o(1))\binom{w+1}{2}$ colors on an d-dimensional poset (X,P) of width w presented with representation.

3.7.1 The Strategy for Anna

We fix the positive integer d and define the strategy $S(d,w)$ for Anna recursively on the positive integer w. Anna constructs a poset (X,P) by presenting a realizer R of size d. Let d and w be positive integers. Anna constructs R by constructing d linear extensions $L_{w-d+2}, \ldots, L_w, L_{w+1}$. In order to handle the case when $w < d-1$, we extend the algorithm $L_\alpha(k,w)$ to be defined for $k < 1$ as follows: If $k < 1$, then $L_\alpha(k,w) = L_\alpha(w,w)$. The strategy $S(d,w)$ is completed in three stages.

3.7.2 Stage 1

For each integer i such that $w-d+2 \leq i \leq w$, Anna constructs the linear extension L_i by following the algorithm $L_\alpha(i,w)$. Anna simultaneously constructs L_{w+1} by following the algorithm $L_\beta(w,w)$.

Notice that $L_w \cap L_{w+1} = P|_{S_1}$. The set S_1 contains a sequence of chains $C_1,\ldots,C_w$ with the Rainbow Property with respect to $P|_{S_1}$. Moreover, if $u \in C_i$ and $v \in S_1 \setminus C_i$, then $u < v$ in L_i for $w-d+2 \leq i \leq w$.

3.7.3 Stage 2

For each integer i such that $w-d+2 \leq i \leq w$, Anna updates L_i by following the dual algorithm $L^*_\beta(w,w)$ completely under S_1 in L_i. Anna simultaneously updates L_{w+1} by following the dual algorithm $L^*_\alpha(w,w)$ completely under S_1 in L_{w+1}.

The dual S_2 contains a sequence of chains $D_1,\ldots,D_w$ with the Rainbow Property with respect to the dual P^* of P. Moreover, if $u \in D_w$ and $v \in S_2 \setminus D_w$, then $v < u$ in L_{w+1}.

3.7.4 Stage 3

We let t denote an integer such that $w-d+2 \leq t \leq w$ and $\|C_t \cup D_w\| \geq 2w - \frac{w}{d-1} - \frac{d-2}{2}$. (The existence of such t will be shown later.) For each integer i such that $w-d+2 \leq i \leq w+1$, Anna plays the points S_3 of Stage 3 in L_i in such a way that the following inequalities hold:

1. $S_2 < C_t < S_3 < S_1 \setminus C_t$ in L_t
2. $S_2 \setminus D_w < S_3 < D_w < S_1$ in L_{w+1}
3. $S_2 < S_3 < S_1$ in L_i for $i \notin \{t, w+1\}$.

It remains to be defined the linear extension on S_3 that Anna uses in L_i. Anna will play $L_\alpha(i-1,w-1)$ for each integer i such that $w = d+2 \leq i < w$, and $L_\beta(w-1,w-1)$ for $i = w+1$.

3.7.5 The Result

Let C be a maximal chain containing $C_t \cup D_w$. It is easy to see that $(S_1 \cup S_2) \setminus C$ induces a poset of width $w-1$. By the induction hypothesis, S_3 induces a poset of width $w-1$. Since $S_2 \setminus D_w < S_3 < S_1 \setminus C_t$ in every linear extension, S_3 and $X \setminus C$ are completely comparable. Since S_3 and $X \setminus C$ are completely comparable, $X \setminus C$ induces a poset of width $w-1$, and thus, (X,P) is a poset of width w.

Since $D_w < C_t < S_3$ in L_t and $S_3 < D_w < C_t$ in L_{w+1}, S_3 and $C_t \cup D_w$ are completely incomparable. Thus, if there exists a t as defined in Stage 3, then $S(d, w)$ forces at least

$$\sum_{i=1}^{w} \left(2i - \frac{i}{d-1} - \frac{d-2}{2}\right) = \left(2 - \frac{1}{d-1} - o(1)\right) \binom{w+1}{2}$$

colors on a poset (X, P) of width at most w. Therefore, all that is left to show is that such a t exists.

Let $C' = \bigcup_{i=w-d+2}^{w} C_i$. Each color from D_w may only be used once in C' and $|C'| = w(d-1) - \frac{1}{2}(d-1)(d-2)$. If we let C'' denote the set of points not colored with colors from D_w, then $|C''| \geq w(d-1) - \frac{1}{2}(d-1)(d-2) - w$. On average each chain has $w - \frac{1}{2}(d-2) - \frac{w}{d-1}$ colors distinct from those in D_w. Thus there must exist an integer t such that $w - d + 2 \leq t \leq w$ and $\|C_t \cup D_w\| \geq 2w - \frac{w}{d-1} - \frac{d-2}{2}$.

CHAPTER 4

ON-LINE WIDTH OF SEMI-ORDERS

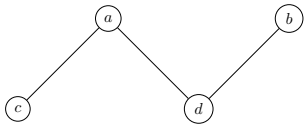
In this chapter, we focus on the on-line width $\text{olws}_R(w)$ of the class of semi-orders presented in the form of an interval representation and improve upon previously known bounds by proving the following result.

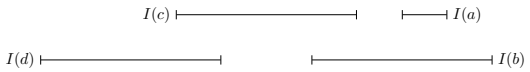
Theorem 4.0.1. *For every $w \geq 2$, $\text{olws}_R(w) \geq \lceil \frac{3}{2}w \rceil$.*

In Section 4.1, we provide the necessary background on interval orders and semi-orders. In Section 4.2, 4.3, 4.4 and 4.5, we define the on-line width of interval orders, interval orders with representation, semi-orders and semi-orders with represention respectively, and provide a brief summary of previous results for each. Finally in Section 4.6, we prove Theorem 4.0.1.

4.1 Introduction to Interval Orders and Semi-orders

A poset (X, P) is an interval order if there is a function I which assigns to each element $x \in X$ a closed interval $I(x) = [l_x, r_x]$ on the real line so that for all $x_1, x_2 \in X$ we have $x_1 < x_2$ if and only if $r_{x_1} < l_{x_2}$. We call I an interval representation of (X, P). Of course, the interval representation is not unique.



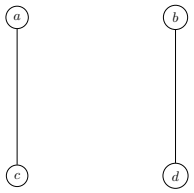


Interval orders have a characterization in terms of forbidden subposets. The following theorem was first given explicitly by Fishburn.

Theorem 4.1.1. *A poset (X, P) is an interval order if and only if it does not contain the subposet $\mathbf{2} + \mathbf{2}$ below.*

This immediately implies the following.

Corollary 4.1.2. *If $n \geq 2$, then S_n is not an interval order.*

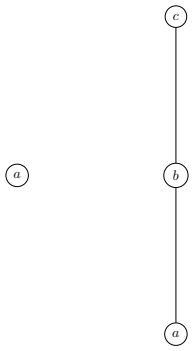


While the theory of interval orders is rich, we do not need more than this for our studies.

An interval order (X, P) is a semi-order if there is an interval representation I assigning to each element $x \in X$ a closed unit-length interval $I(x) = [r_x - 1, r_x]$ on the real number line so that for all $x_1, x_2 \in X$ we have $x_1 < x_2$ if and only if $r_{x_1} < r_{x_2} - 1$.

Similarly, semi-orders have a characterization in terms of forbidden subposets. Clearly, semi-orders cannot contain a $\mathbf{2} + \mathbf{2}$.

Theorem 4.1.3. *An interval order (X, P) is a semi-order if and only if it does not contain the subposet $3 + 1$ below.*



It is important to point out that the restriction on the intervals in the representation can be weakened and still define the same class of semi-orders.

Proposition 4.1.4. *An interval order (X, P) is a semi-order if there is an interval representation I assigning to each element $x \in X$ a closed interval $I(x) = [l_x, r_x]$ on the real number line so that for all $x_1, x_2 \in X$ we have $x_1 < x_2$ if and only if $r_{x_1} < r_{x_2} - 1$, and neither $I(x_1)$ nor $I(x_2)$ is contained in the interior of the other. We call the representation a* proper representation *of (X, P) and refer to the intervals as proper intervals.*

4.2 On-Line Width of Interval Orders

The on-line width olwi(w) of the class of interval orders of width at most w is the largest integer k for which there exists a strategy that forces any algorithm to use k chains to partition an interval order of width w. This variant was solved by Kierstead and Trotter in the early 80's.

Theorem 4.2.1 (Kierstead and Trotter [16])**.** *For every* $w \geq 1$, $\text{olwi}(w) = 3w - 2$.

Proof. We argue by induction on the width w of the poset (X, P). If $w = 1$, then the result is trivial. Assume that there exists a strategy $S(w)$ that forces $3w - 2$ chains for $w \leq k$ and suppose (X, P) is a poset of width $w = k + 1$. The strategy consists of two stages.

Stage 1

We begin by playing the strategy $S(k)$ enough times in such a way that each subsequent $S(k)$ is completely above the previous one. We stop once we have 4 sets of points S_1, S_2, S_3, S_4 with $S_1 < S_2 < S_3 < S_4$ with each having forced the same $3k - 2$ chains. We can guarantee this by the pigeonhole principle.

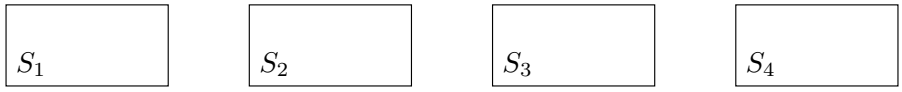


Figure 4.1: Stage 1 ends with subposets S_1, S_2, S_3, S_4 consisting of the same $3k - 2$ colors. The subposets are depicted left to right instead of bottom to top as if it were an interval representation.

Stage 2

Suppose Stage 1 ends in round $t - 1$. In round t, Anna introduces a point x_t so that $a \parallel S_1$ and $x_t < S_2$. Since x_t is incomparable with every point in S_1, without loss of generality, x_t must be assigned a new color $3k - 1$. Anna then introduces a point x_{t+1} so that $x_{t+1} \parallel S_4$ and $x_{t+1} > S_3$. At this point there are two possibilities. Beth can either assign x_{t+1} a new color $3k$ or Beth can assign x_{t+1} the color $3k - 1$.

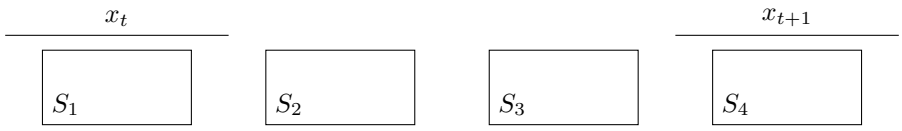


Figure 4.2: Here is what an interval representation may look like after introducing points x_t and x_{t_1} during Stage 2.

If Beth assigns x_{t+1} a new color, then we introduce a point x_{t+2} so that $S_1 < x_{t+2} < S_4$ and $x_{t+2} \parallel (\{x_t, x_{t+1}\} \cup S_2 \cup S_3)$. Beth cannot assign x_{t+2} a color from $[k]$ since $x_{t+2} \parallel S_2$. Beth also cannot assign x_{t+2} the color $3k - 1$ or the color $3k$ since $x_{k+2} \parallel \{x_t, x_{t+1}\}$. Thus Beth must assign x_{t+2} a new color $3k + 1$.

If Beth assigns x_{t+1} the color $3k - 1$, then Anna introduces two more points x_{t+2} and x_{t+3} so that

$$S_1 < x_{t+2} < S_3$$

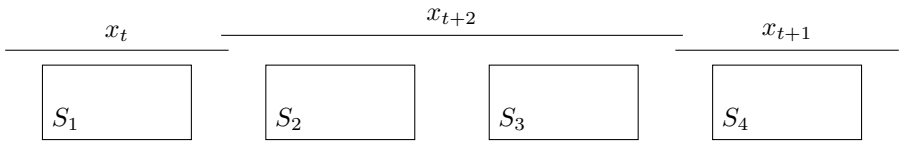


Figure 4.3: The end game if Beth assigns x_{t+1} a new color.

$$x_{t+2} \parallel < x_t \cup S_2$$

$$S_2 < x_{t+3} < S_4$$

$$x_{t+3} \parallel \{x_{t+2}, x_{t+1}\} \cup S_3.$$

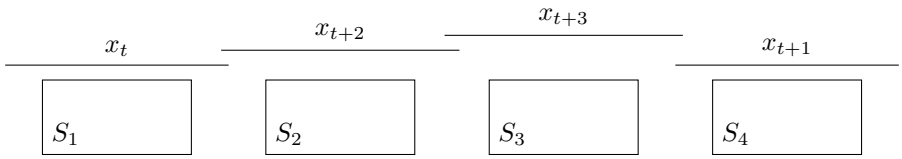


Figure 4.4: The end game if Beth assigns x_{t+1} the color $3k-1$.

Beth is forced to assign x_{t+2} a new color $3k$. Finally Beth is forced to assign x_{t+3} a new color $3k+1$. Since $3w-2 = 3(k+1)-2 = 3k+1$, this concludes the proof of the lower bound.

Proof of upper bound. In order to prove the upper bound, we present the following algorithm $A(w)$ and argue inductively that it needs at most $3w-2$ chains. If $w = 1$, Beth colors every point with 1 color.

Beth maintains a partition of the elements of the ground set X into two sets U and V so that U induces a poset of width at most $w-1$. When Anna presents a new point x, Beth adds x to U if adding x to U does not increase the width of U to w, otherwise, x is added to V. Beth partitions U recursively by using the algorithm $A(w-1)$. By the induction hypothesis, at most $3w-5$ colors are used in U. Hence, it suffices to show that at most 3 colors are used in V.

Let $v' \in V$. Then v' is part of an antichain $v', u_1, \ldots, u_{w-1}$ where $u_i \in U$ otherwise v would have been added to U instead of V. Now fix an interval representation I of (X, P). We know that $I(u_1) \cap \ldots \cap I(u_{w-1}) \neq \emptyset$. Let $r \in \mathbb{R}$ be any real number in the intersection. Notice that no other interval $I(v)$ can contain r, otherwise (X, P) would contain an antichain of size $w+1$. Hence, no two intervals in $\{I(v) : v \in V\}$ contain each other. Moreover, no interval $I(v)$ with $v \in V$ is contained in the union of all other intervals in $\{I(v) : v \in V\}$. Then, the co-compararability graph of the poset induced on V is a subgraph of a path. Thus, Beth can use any greedy algorithm such as First Fit to color points in V. In total, Beth uses $3w-5+3 = 3w-2$ colors to partition (X, P).

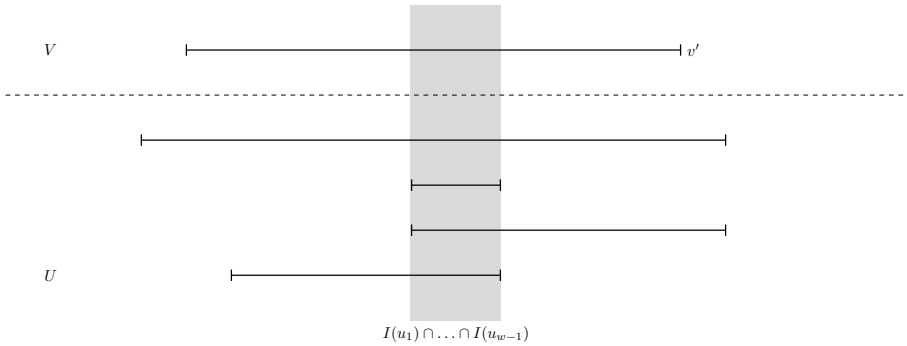


Figure 4.5: Interval representation of (X, P) partitioned into U and V.

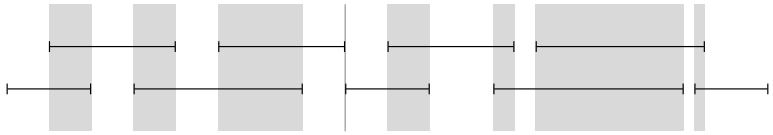


Figure 4.6: The co-compararability graph of the poset induced on V is a subgraph of a path.

□

4.3 On-Line Width of Interval Orders with Representation

The on-line width $\text{olwi}_R(w)$ of the class of interval orders of width at most w with representation is the largest integer k for which there exists a strategy that forces any algorithm to use k chains to partition an interval order of width w presented as intervals. This means that instead of presenting the elements of the interval order as points, the elements are presented as intervals. These intervals provide an interval representation for a unique poset (X, P). This variant of the problem was solved by Chrobak and Ślusarek.

Theorem 4.3.1 (Chrobak and Ślusarek [5])**.** *For every* $w \geq 1$, $\text{olwi}_R(w) = 3w - 2$.

Proof. Presenting the interval order (X, P) as intervals instead of points only makes Beth stronger. Therefore, the upper bound of $3w - 2$ comes directly from the variant without representation.

For the lower bound, the exact strategy from the variant without representation works. Only difference is that the elements are presented as intervals instead of points.

□

After the last result, one may wonder if presented an interval order as intervals instead of points makes any difference. While in the general case, it did not, restricting the lengths of the intervals seems to potentially change the result depending on if the elements are presented as intervals or not.

4.4 On-Line Width of Semi-Orders

The on-line width olws(w) of the class of semi-orders of width at most w is the largest integer k for which there exists a strategy that forces any algorithm to use k chains to partition a semi-order of width w. Chrobak and Ślusarek showed that first-fit needs at most $2w-1$ chains, and over 20 years later, Bosek, Felsner, Kloch, Krawczyk, Matecki and Micek showed that any on-line algorithm can be forced to use $2w-1$ chains.

Theorem 4.4.1 (Chrobak and Ślusarek [5], Bosek et al. [4])**.** *For every $w \geq 1$,* olws$(w) = 2w-1$.

Proof. In order to prove the upper bound, we show that First Fit uses at most $2w-1$ chains to partition any poset of width at most w. To see that, simply consider an element x introduced by Anna. Now let I be any interval representation of (X, P). Since no other interval can be contained in the interior of $I(x)$, any interval intersecting $I(x)$ must contain one of the endpoints of $I(x)$. Hence, $I(x)$ can intersect at most $2(w-1)$ intervals. This concludes the proof for the upper bound.

In order to prove the lower bound we construct a strategy $S(w)$ for Anna which forces Beth to use $2w-1$ chains on a poset of width w. We begin by introducing an antichain $A = a_1, \ldots, a_w$ and a second antichain $B = b_1, \ldots, b_w$ so that $a_i < b_j$ for all $i \in [w]$ and $j \in [w]$. Let $B' = b_{i_1}, \ldots, b_{i_k}$ with $2 \leq k \leq w$ denote the elements in B which were assigned chains already used in A. Notice that if $k = 0$ or $k = 1$, then we are done. Hence we may assume $k \geq 2$. Now let $A' = a_{i_1}, \ldots, a_{i_k}$ denote the elements in A such that a_{i_j} and b_{i_j} were assigned the same chain for $j \in [k]$.

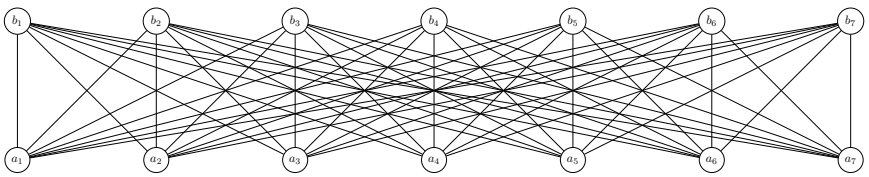


Figure 4.7: Subposet induced by antichains A and B in $S(7)$.

Now, we introduce an antichain $c_1, \ldots, c_{k-1}$ so that for each $h \in [k-1]$ (1) $a_{i_j} \leq c_h$ for $j \in [h]$, and

(2) $c_h \leq b_{i_j}$ fo $h+1 \leq j \leq k-1$.

It is easy to verify that the width of the poset is w, the poset does not contain a $2+2$ nor a $3+1$, and in total, Beth must use $2(w-k)+k+(k-1) = 2w-1$ chains.

□

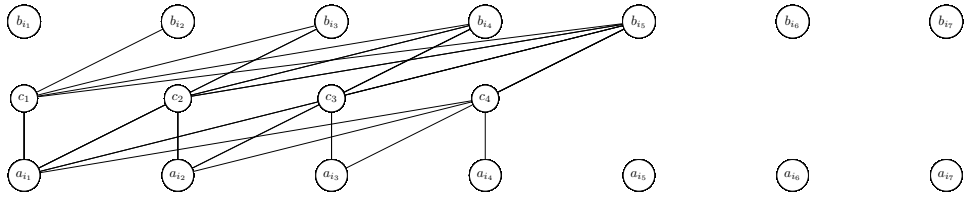


Figure 4.8: Antichain C with all of its comparabilities in $S(7)$ with 5 chains in common between A and B.

4.5 On-Line Width of Semi-Orders with Representation

The on-line width $\text{olws}_R(w)$ of the class of semi-orders of width at most w with representation is the largest integer k for which there exists a strategy that forces any algorithm to use k chains to partition a semi-order of width w presented as unit-intervals. The semi-order is presented in the form of intervals instead of points, however, the intervals in this variant must all have length 1.

This problem was first considered by Chrobak and Ślusarek [5]. They showed that first-fit needs at most $2w-1$ chains to partition a semi-order with representation. They also showed that any greedy algorithm can be forced to use $2w-1$ chains. However, it remained unkown whether a more optimal algorithm exists. Epstein and Levy [8] constructed a strategy which forces $3k$ chains on a semi-order of width $2k$ presented with representation for any positive integer k. This provides the following previously best known bounds.

$$\lfloor \frac{3}{2} w \rfloor \leq \text{olws}_R(w) \leq 2w-1$$

While progress continues to be made for the general problem, as well as other variants, no improvements have been made to these bounds for almost 20 years. In this chapter, we improve the lower bound slightly by presenting a strategy which forces $3k+2$ chains on a semi-order of width $2k+1$ presented with representation for any positive integer k.

Remark. Recall that there are in fact two choices of representation for the class of semi-orders. In particular, we could have gone with a proper representation instead of the unit-interval representation. Let $\text{olws}_{R_P}(w)$ denote the on-line width of the class of semi-orders with proper representation. It may be the case that $\text{olws}_R(w) \neq \text{olws}_{R_P}(w)$. Moreover, since any set of unit-intervals are proper and a proper representation may contain intervals of different lengths, we only know that

$$\text{olws}_R(w) \leq \text{olws}_{R_P}(w) \leq \text{olws}(w) = 2w-1.$$

Therefore, our result is stronger than proving the same result for proper representations.

4.6 Proof of Theorem

Since in this variant, we introduce the elements of the poset (X, P) as unit-intervals, we may define each element by a real number r_i. More specifically, we define each element introduced by the right endpoint of the interval in the representation so that if we introduce the element x_i as the unit-interval $[r_i - 1, r_i]$, we simply define x_i by $x_i = r_i$. Assume that $w = 2k+1$ for some positive integer k. The strategy consists of 5 stages.

4.6.1 Stage 1

We begin by introducing a stack of intervals $x_1, \ldots, x_k$ so that $x_i = 0$ for $i \in \{1, \ldots, k\}$. Notice that the intervals in Stage 1 form an antichain, and hence, must each be assigned a distinct chain. Let A denote the set of chains $\{a_1, \ldots, a_k\}$ used in Stage 1.

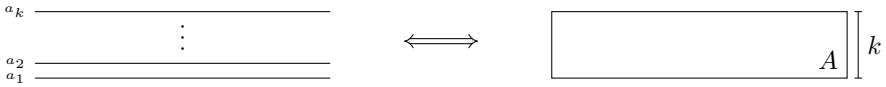


Figure 4.9: Stage 1: forcing the first k chains.

4.6.2 Stage 2

Initialize $l_2 = 1$ and $h_2 = 2$. In round i, we introduce the interval x_i so that $x_i = (l_2 + h_2)/2$. Suppose that the algorithm assigns the interval to chain j. If $j \in A$, then we update h_2 so that $h_2 = x_i$. Otherwise, if $j \notin A$, then we update l_2 so that $l_2 = x_i$. Let B denote the set of new chains used in Stage 2. If $|B| = k + 1$, we move onto Stage 3. Otherwise, if $|B| < k + 1$, then we repeat Stage 2 in round $i + 1$.

Since $1 < x_i < 2$ for every interval x_i introduced in Stage 2, the intervals presented in Stage 2 form an antichain of size at most w. Therefore, every interval is assigned to a different chain by the algorithm of which at most k are in A. Hence, Stage 2 ends forcing $k + 1$ new chains.

4.6.3 Stage 3

Initialize $l_3 = l_2 - 3$ and $h_3 = h_2 - 3$. In round i, we introduce a new interval x_i so that $x_i = (l_3 + h_3)/2$. Suppose that the algorithm assigns the interval to chain j. If $j \in B$, then we update h_3 so that

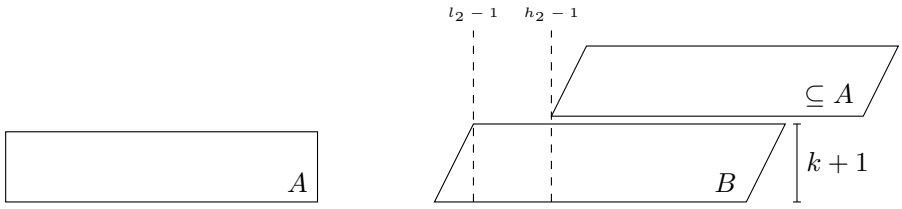


Figure 4.10: Stage 2: forcing $k+1$ new chains.

$h_3 = x_i$ and move onto Stage 4. Otherwise, if $j \notin B$, then we update l_3 so that $l_3 = x_i$ and we repeat Stage 3 in round $i+1$

Since $-2 < x_i < -1$ for every interval x_i introduced in Stage 3, the intervals presented in Stage 3 form an antichain of size at most w. Therefore, every interval is assigned to a different chain by the algorithm of which at most k are in A. If $k+1$ intervals are assigned entirely new chains, then the proof is complete. Hence, we may assume that Stage 3 ends with the algorithm assigning an interval x_B to a chain $b \in B$. Note that in this case, $x_B = h_3$

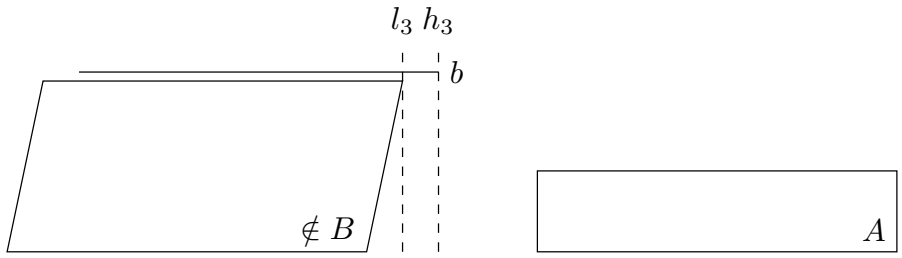


Figure 4.11: Stage 3: forcing a chain $b \in B$ on x_B.

4.6.4 Stage 4

Initialize $l_4 = l_3+1$ and $h_4 = h_3+1$. In round i, we introduce a new interval x_i so that $x_i = (l_4+h_4)/2$. Suppose that the algorithm assigns the interval to chain j. Since $-1 < x_i < x_b + 1 < 0$, $j \notin A$ and $j \neq b$. We update l_4 so that $l_4 = x_i$. If $j \notin B$, then we move onto Stage 5. Otherwise, if $j \in B$, we repeat Stage 4 in round $i+1$.

The intervals introduced in Stage 4 form an antichain of size at most $k+1$. Therefore, every interval is assigned to a distinct chain of which no chain is in $A \cup \{b\}$ and at most k chains are in $B \setminus \{b\}$. Hence, Stage 4 ends with the algorithm assigning an interval x_C to an entirely new chain c.

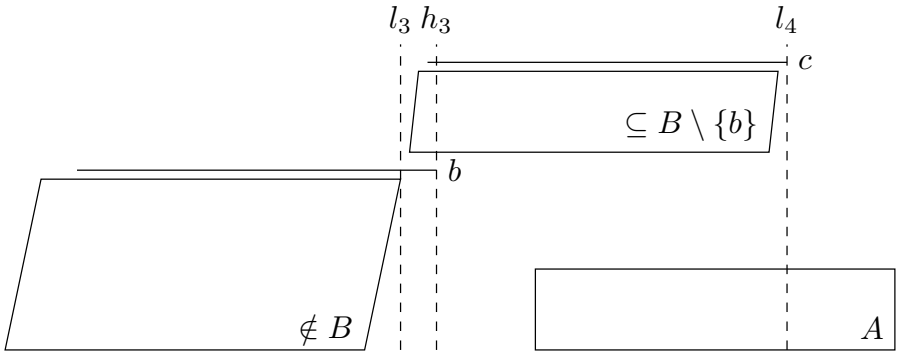


Figure 4.12: Stage 4: forcing a new chain c on x_C.

4.6.5 Stage 5

Finally, for each $i \in \{c+1, \ldots, c+k\}$, we introduce an interval x_i so that $x_i = x_C + 1$. The intervals introduced in Stage 5 form an antichain of size k of which each interval cannot be assigned to any chain in $A \cup B \cup \{c\}$. All that is left to show is that we have not exceeded the width w. Let x be any interval introduced in Stage 5. It is trivial to check that the only interval from Stages 3 and Stage 4 which is incomparable to x_i is x_C. Moreover, solving for the following:

$$l_2 - 3 < x_B < h_2 - 3$$

$$l_2 - 2 < x_C < x_B + 1$$

$$x_i = x_C + 1$$

we get that $l_2 - 1 < x_i < h_2 - 1$ which implies that the only intervals from Stage 2 that are incomparable to x_i are exactly the $k+1$ intervals which were assigned to chains from B. Let D denote the set of new chains forced in Stage 5. Thus, the total number of chains forced on this poset of width w is

$$|A| + |B| + |\{c\}| + |D| = k + (k+1) + 1 + k = 3k + 2.$$

This concludes the proof.

4.6.6 Remarks

We proved that if $w = 2k+1$, then our strategy will force any on-line algorithm to use $3k+2$ chains for any positive integer k. Moreover, we know that any greedy algorithm uses at most $2w-1$ chains. Thus, we get the answer to the previously open problem of finding the on-line width of the class of

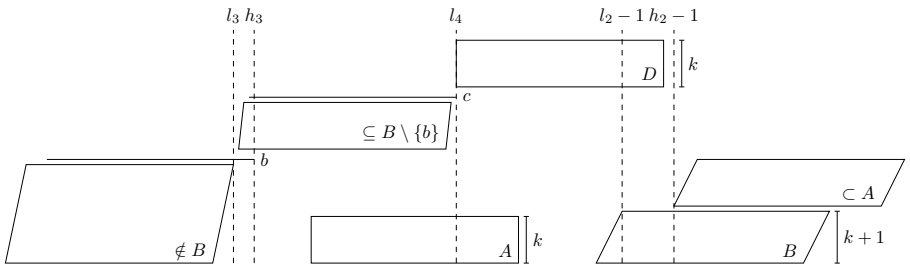


Figure 4.13: Stage 5: forcing the last k chains.

semi-orders of width 3 with representation.

Corollary 4.6.1. $\text{olws}_R(3) = 5$.

CHAPTER 5

ON-LINE WIDTH OF L-INTERVAL ORDERS

In this chapter, we consider classes of posets strictly between the class of semi-orders and the class of interval orders. We refer to posets in these classes as L-Interval orders. For $T \in \mathbb{R}^+$, let $\text{olwi}_R(L, w)$ denote the on-line width of L-Interval Orders of width at most w with representation. Then we provide a non-trivial bound for $\text{olwi}_R(\{1, 1+\epsilon\}, w)$.

Theorem 5.0.1 (Biró and Curbelo (Unpublished))**.** *For every $w \geq 1$, $\text{olwi}_R(\{1, 1+\epsilon\}, w) \geq \lfloor \frac{5}{3} w \rfloor$ for every $\epsilon > 0$.*

In Section 5.1, we define L-Interval orders and show that for almost all R, the class of L-Interval orders is strictly between that of semi-orders and interval orders. In Section 5.2, we define $\text{olwi}(L, w)$ and $\text{olwi}_R(L, w)$. Finally, in Section 5.3, we prove Theorem 5.0.1.

5.1 Introduction to L-Interval Orders

For any subset $L \subseteq \mathbb{R}^+$ of real numbers greater than 0, we define an L-Interval order as follows. We call an interval order (X, P) an L-Interval order if there is an interval representation I assigning to each $x \in X$, a closed interval $I(x) = [l_x, r_x]$ with $leqr_x - l_x \in R$ so that for all $x_1, x_2 \in X$ we have $x_1 < x_2$ if and only if $r_{x_1} < l_{x_2}$. We call I an L-Interval representation of (X, P). Clearly, if $|L| = 1$, then the class of L-Interval orders is the same as the class of semi-orders, and if $R = \mathbb{R}^+$, then the class of L-Interval orders is the same as the class of interval orders. Hence, we are interested in cases when

1. $|L| > 1$, and
2. $1 \leq l \leq m$ for every $l \in L$ for some upper bound m.

So we will assume such from this point on.

Proposition 5.1.1. *Let $\mathcal{I}$ be the class of interval orders, $\mathcal{S}$ be the class of semi-orders, and let $\mathcal{L}$ be the class of L-Interval orders where $L \subseteq \mathbb{R}^+$ satisfying the following conditions.*

1. $|L| > 1$.

2. $1 \leq l \leq m$ *for every* $l \in L$ *for some upper bound* m.

Then,

$$\mathcal{S} \subset \mathcal{R} \subset \mathcal{I}.$$

Proof. By definition, $\mathcal{S} \subseteq \mathcal{L} \subseteq \mathcal{I}$. Hence it suffices to show that for any L satisfying the conditions, there exists an interval order that is not an L-Interval order, and an L-Interval order that is not a semi-order. Let L be a subset of real numbers satisfying conditions 1 and 2.

In order to prove the latter, let $m = inf(L)$ and $M = sup(L)$. Then the poset $\mathbf{1} + \mathbf{3}$ has L-Interval representation

$$\{[0, M], [(M - m)/2, (M + m)/2], [-M, 0], [M, 2M]\}$$

and thus is an L-Interval order but not a semiorder.

Now we constuct an interval order which cannot have an L-Interval representation. As before, let $M = sup(L)$. Now let $\mathbf{1} + \mathbf{M'}$ denote the poset consisting of a chain M' of size $M + 2$ and an isolated point. Then, the poset $\mathbf{1} + \mathbf{M'}$ is an interval order since it does not contain a $\mathbf{2} + \mathbf{2}$.

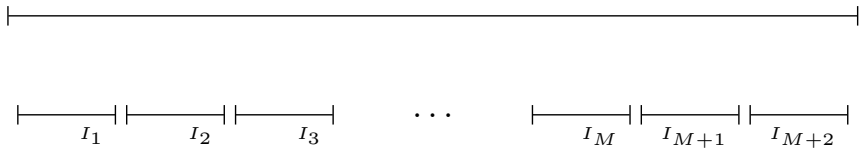


Figure 5.1: An interval representation of $\mathbf{1} + \mathbf{M'}$.

However, $\mathbf{1} + \mathbf{M'}$ is not an L-Interval order since one interval of length at most M would have to intersect $M + 2$ intervals of length at least 1 which is impossible. This concludes the proof.

□

We now know that we are dealing with entirely new classes of posets. Not much is known about these classes of posets. However, we study their on-line widths with representation which means no information on the class is needed.

5.2 On-Line Width of L-Interval Orders

Let $L \subseteq \mathbb{R}^+$. Then the on-line width $\text{olwi}_R(L, w)$ of the class of L-interval orders of width w with representation is the largest integer k for which there exists a strategy that forces any algorithm to

use k chains to partition a L-interval order of width w presented as intervals with length in L. We show that allowing intervals of one more length, even with the lengths arbitrarily close, we achieve a significantly better result.

Remark. After writing up the proof of our results, we found a recent paper by Chybowska-Sokól, Gutowski, Junosza-Szaniawski, Mikos, and Polak [6] where they study this in a graph-theoretic setting. Admittingly, their paper goes more in depth on the subject and they even prove the following.

Theorem 5.2.1 (Chybowska-Sokól, et al. [6])**.** $\mathrm{olwi}_R([1, 1+\epsilon], w) \geq \lfloor \frac{5}{3} w \rfloor$ *for every* $\epsilon > 0$.

However, it is important to note that what we prove is in fact stronger, as we only allow intervals of exactly two lengths as opposed to allowing intervals of any length between 1 and $1+\epsilon$. For that reason, we still present our proof.

5.3 Proof of Theorem

Since in this variant, we introduce the elements of the poset (X, P) as unit-intervals and intervals of length $1+\epsilon$, we may define each element by a real number r_i. More specifically, we define each element introduced by the right endpoint of the interval in the representation so that if we introduce the element x_i as the unit-interval $[r_i - 1, r_i]$, we simply define x_i by $x_i = r_i$. Moreover, we superscript the number by ϵ for intervals of length $1+\epsilon$, i.e., defining x_i by $x_i = r_i^\epsilon$ means we are introducing the interval $[r_i - 1 - \epsilon, r_i]$. Assume that $k = \lfloor w/3 \rfloor$ where w is the width of the poset (X, P). The strategy consists of 4 stages.

5.3.1 Stage 1

We begin by introducing a stack of intervals $x_1, \ldots, x_k$ so that $x_i = 0$ for $i \in \{1, \ldots, k\}$. Notice that the intervals in Stage 1 form an antichain, and hence, must each be assigned a distinct chain. Let A denote the set of chains $\{a_1, \ldots, a_k\}$ used in Stage 1.

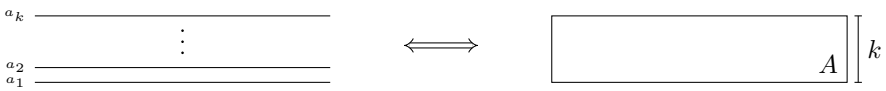


Figure 5.2: Stage 1: forcing the first k chains.

5.3.2 Stage 2

Initialize $l_2 = 1$ and $h_2 = 1 + \epsilon/2$. In round i, we introduce the interval x_i so that $x_i = (l_2 + h_2)/2$. Suppose that the algorithm assigns the interval to chain j. If $j \in A$, then we update h_2 so that $h_2 = x_i$. Otherwise, if $j \notin A$, then we update l_2 so that $l_2 = x_i$. Let B denote the set of new chains used in Stage 2. If $|B| = k$, we move onto Stage 3. Otherwise, if $|B| < k$, then we repeat Stage 2 in round $i + 1$.

Since $1 < x_i < 2$ for every interval x_i introduced in Stage 2, the intervals presented in Stage 2 form an antichain of size at most w. Therefore, every interval is assigned to a different chain by the algorithm of which at most k are in A. Hence, Stage 2 ends forcing k new chains.

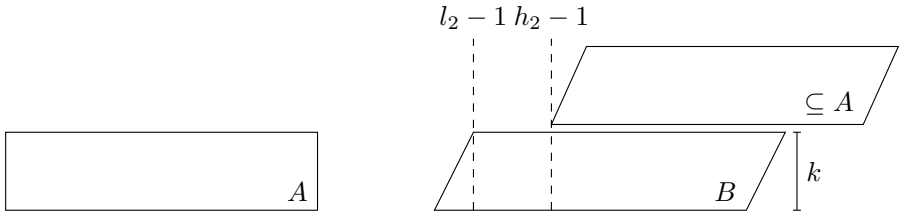


Figure 5.3: Stage 2: forcing k new chains.

5.3.3 Stage 3

Initialize $l_3 = l_2 - 2 - \epsilon$ and $h_3 = h_2 - 2 - \epsilon$. In round i, we introduce the interval x_i so that $x_i = (l_2 + h_2)/2$. Suppose that the algorithm assigns the interval to chain j. If $j \in A \cup B$, then we update l_3 so that $l_3 = x_i$. Otherwise, if $j \notin A \cup B$, then we update h_3 so that $h_3 = x_i$. Let C denote the set of new chains used in Stage 2. If $|C| = k$, we move onto Stage 4. Otherwise, if $|C| < k$, then we repeat Stage 3 in round $i + 1$.

Since $l_3 < x_i < h_3$ for every interval x_i introduced in Stage 3, the intervals presented in Stage 3 form an antichain of size at most w. Therefore, every interval is assigned to a different chain by the algorithm of which at most $2k$ are in $A \cup B$. Hence, Stage 3 ends forcing k new chains.

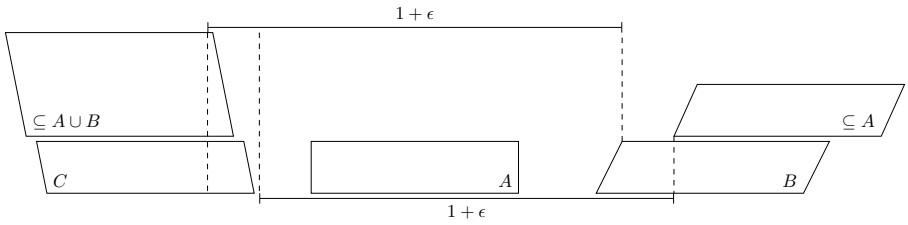


Figure 5.4: Stage 3: forcing k more chains.

5.3.4 Stage 4

Finally, suppose Stage 3 ends in round c. for each $i \in \{c+1, \ldots, c+w-k\}$, we introduce an interval x_i so that $x_i = (x_c + 1 + \epsilon)^\epsilon$. The intervals introduced in Stage 4 form an antichain of size $w-k$ of which each interval cannot be assigned to any chain in $A \cup B \cup C$. All that is left to show is that we have not exceeded the width w. Let x be any interval introduced in Stage 4. Solving for the following:

$$l_2 - 2 - \epsilon < x_c < h_2 - 2 - \epsilon$$

$$x = x_c + 1 + \epsilon$$

we get that $l_2 - 1 < x < h_2 - 1$ which implies that the only intervals from Stage 2 that are incomparable to x are exactly the $w-k$ intervals which were assigned to chains in B. Let D denote the set of new chains forced in Stage 4. Thus, the total number of chains forced on this poset of width w is

$$|A| + |B| + |C| + |D| = k + k + k + (w-k) = w + 2k \geq \lfloor \frac{5}{3} w \rfloor.$$

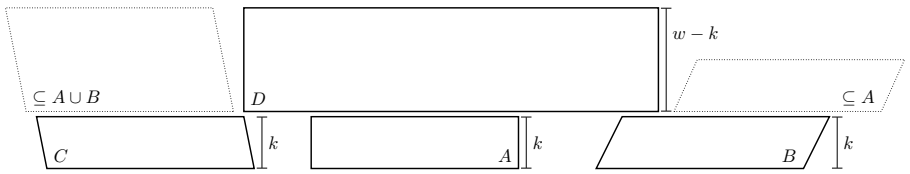


Figure 5.5: Stage 4: forcing the last $w-k$ chains.

This concludes the proof.

CHAPTER 6

GREEDY WIDTH

In this Chapter, instead of restricting Anna, we replace Beth with an easier opponent who we call Bertha. Bertha is greedy which means that Bertha never uses a new color unless she absolutely has to. Then, we refer to the maximum number of colors that Anna can force Bertha to use on a class of posets as the greedy width. One very popular example of a greedy algorithm is First Fit.

In Section 6.1, we consider the greedy width $\text{gw}(w)$ of the class of posets and show that the value is unbounded. In Section 6.2, we analyze the greedy width $\text{gws}(w)$ of the class of semi-orders of width at most w which was solved by Chrobak and Ślusarek. In Section 6.3, we consider the greedy width $\text{gwi}(w)$ of the class of interval orders of width at most w and give a brief summary of the rich history of research on First Fit along with its implication for $\text{gwi}(w)$.

In Section 6.4, we define a new class $\mathcal{A}$ of posets called almost-semiorders which is strictly between that of semi-orders and interval orders. Finally, in Section 6.5, we prove that the greedy width $\text{gwa}(w)$ of posets in $\mathcal{A}$ of width at most w is strictly between the values for that of semi-orders and interval orders.

Theorem 6.0.1 (Biró and Curbelo (Unpublished))**.** *For every $w \geq 1$, $2w \leq \text{gwa}(w) \leq 3w - 3$.*

6.1 Greedy Width

We are also interested in knowing the performance of specific algorithms. Probably the simplest and most popular on-line partitioning algorithm is First Fit. First Fit maintains a chain partition $X = C_1 \cup \ldots \cup C_k$ and when a new element x is introduced, places the element in the chain C_i with the smallest i such that $C_i \cup \{x\}$ induces a chain in (X, P). If no such chain C_i exists, then First Fit introduces a new chain C_{k+1} to the partition and adds x to it. The performance of First Fit is similarly measured with respect to the width w of the poset. Problems on finding the efficiency of First Fit have a very rich history. However, we would like to allow slightly more freedom to the algorithm than that. We consider all greedy algorithms. Greedy algorithms, like First Fit, only

introduce a new chain to the partition if and only if a new element is presented which does not induce a chain with any of the chains in the current partition.

The greedy width $\mathrm{gw}(w)$ of the class of posets of width at most w is the largest integer k for which there exists a strategy that forces any greedy algorithm to use k chains to partition a poset of width w. If no such k exists, we say that the greedy width is infinite and write $\mathrm{gw}(w) = \infty$. It has been shown that that there exists a strategy that forces First Fit to use an arbitrarily large number of chains to partition of poset of width 2. However, a different greedy algorithm could partition the same poset using just 2 chains constructed with that strategy. We present a new strategy constructing the same poset of width 2 which forces any greedy algorithm to use an arbitrarily large number of chains.

Lemma 6.1.1. *For every $n \in \mathbb{N}$, there is a strategy which forces any greedy algorithm to use n chains to partition a poset of width 2.*

Proof. We construct a poset (X, P) so that $X = A \cup B$, $A \cap B = \emptyset$ and A and B induce chains in P. By Dilworth's chain partitioning theorem, the result will be a poset of width at most 2. The strategy $S(n)$ constructs the poset (X, P) in n stages.

We begin in Stage 1 by presenting a chain $C_1 = x_1^1, \ldots, x_{n+1}^1$. Clearly, any greedy algorithm would color C_1 using only the color 1.

In Stage k for $k \in \{2, \ldots, n-1\}$, we construct a chain $C_k = x_1^k, \ldots, x_{n-k+2}^k$ in increasing order so that for each $i \in \{1, \ldots, n-k+2\}$ so that the following conditions are satisfied.

1. $x_{i+k-3}^1 < x_i^k < x_{i+1}^{k-1}$.
2. $x_i^k \parallel x_{i+j-1}^{k-j}$ for $j \in \{1, \ldots, k-1\}$.

Furthermore, x_i^k is added to A if k and i are both odd or k and i are both even, otherwise x_i^k is added to B.

Finally in stage n, we introduce single element, x_1^n so that $x_1^n \parallel x_j^l$ if $j + l = n$, otherwise $x_j^l < x_1^n$. We claim that every point introduced in Stage k for $k \in [n]$ is assigned the color k. We prove our claim inductively on k knowing that the claim holds for $k = 1$.

Assume that the claim holds for $k \in [m-1]$ and suppose we introduce the point x_i^m in Stage m. By the second condition, we know that if the second condition is met that a new color would have to be used. It is easy to verify that condition 2 is still satisfied after taking the transitive closure.

In order to show that (X, P) has width 2, it suffices to show that A and B are chains. Suppose A and B have been maintained as chains in round $r-1$. Now suppose in round r we introduce the

point x_i^k between consecutive points x_{i+k-3}^1 and x_{i+1}^{k-1}. Suppose k is odd and i is odd. Then x_i^k is added to A. Since $i+k-3$ is odd and 1 is odd, $x_{i+k-3}^1 \in A$. Since $i+1$ is even and $k-1$ is even, $x_{i+1}^{k-1} \in A$. Now suppose k is even and i is even. Then x_i^k is added to A. Since $i+k-3$ is odd and 1 is odd, $x_{i+k-3}^1 \in A$. Since $i+1$ is odd and $k-1$ is odd, $x_{i+1}^{k-1} \in A$. Now suppose k is even and i is odd. Then x_i^k is added to B. Since $i+k-3$ is even and 1 is odd, $x_{i+k-3}^1 \in B$. Since $i+1$ is even and $k-1$ is odd, $x_{i+1}^{k-1} \in B$. Lastly, suppose k is odd and i is even. Then x_i^k is added to B. Since $i+k-3$ is even and 1 is odd, $x_{i+k-3}^1 \in B$. Since $i+1$ is odd and $k-1$ is even, $x_{i+1}^{k-1} \in B$. This concludes the proof.

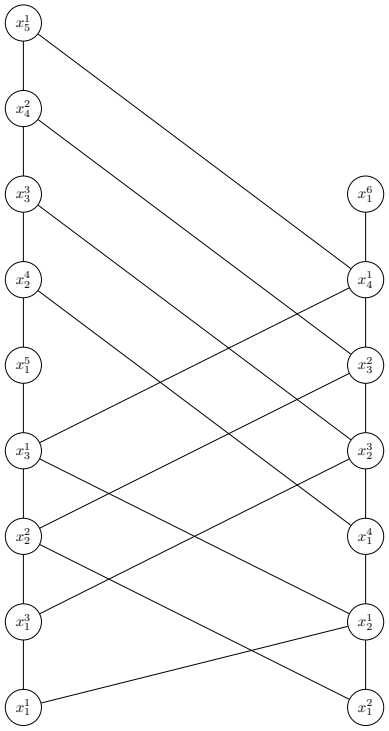


Figure 6.1: Strategy $S(6)$ forces any greedy algorithm to use 6 chains on a poset of width 2.

□

Theorem 6.1.2. *For every $w \geq 2$, $\mathrm{gw}(w) = \infty$.*

Proof. The result is a consequence of the previous Lemma. □

In the previous result, we were able to prove to prove a stronger result by simply changing the order by which the elements were introduced. One may wonder if this could always be done. We answer this question in the affirmative below.

Theorem 6.1.3. *Let $S(w)$ be a strategy which forces First Fit to use N chains to partition a poset (X, P) of width w. Then there exists a strategy $S'(w)$ which forces any greedy algorithm to use N chains to partition the same poset (X, P) of width w.*

Proof. Suppose there exists a strategy $S(w)$ which forces First Fit to use N chains to partition a poset (X, P) of width w. For every $k \in [N]$, let $x_1^k, \ldots, x_{n_k}^k$ denote the elements that First Fit assigned to chain k. Now define the strategy $S'(w)$ so that it constructs (X, P) in the following order.

$$x_1^1, \ldots, x_{n_1}^1, x_1^2, \ldots, x_{n_2}^2, \ldots, x_1^N, \ldots, x_{n_N}^N$$

We prove inductively on the round r that any greedy algorithm assigns x_i^k the color k for every $k \in [N]$ and every $i \in [n_k]$. In round 1, x_1^1 gets assigned the first color 1.

Assume that in round $r-1$, it holds that all points have been assigned the correct color. Now suppose in round r, the point x_i^k is introduced. Suppose on the contrary that x_i^k gets assigned a color other than k. There are at most k colors used in the partition at this point, and so x_i^k either is assigned a color in $[k-1]$ or a new color $k+1$. However, it cannot be assigned $k+1$ since we know that $x_1^k, \ldots, x_{i-1}^k$ are exactly the points colored k and $x_1^k, \ldots, x_{i-1}^k, x_i^k$ forms a chain. So we may assume that x_i^k was assigned a color in $c \in [k-1]$. This would imply that First Fit assigned x_i^k to chain k when it could have assigned it to chain c with $c < k$ which is a contradiction. Therefore, x_i^k must be assigned the color k. □

6.2 Greedy Width of Semi-Orders

Naturally, we would like to know how greedy algorithms perform for more restricted classes of posets as well. The greedy width $\text{gws}(w)$ of the class of semi-orders of width at most w is the largest integer k for which there exists a strategy that forces any greedy algorithm to use k chains to partition a poset of width w. Chrobak and Ślusarek completely solved this variant over 30 years ago.

Theorem 6.2.1 (Chrobak and Ślusarek [5])**.** *For every $w \geq 1$, $\text{gws}(w) = 2w - 1$*

Proof. We know from Chapter 3 that First Fit needs at most $2w-1$ chains to partition any poset of width at most w. Hence, it suffices to show that we can force First Fit to use $2w-1$ colors.

As in Chapter 3, we introduce intervals by their right endpoint. The strategy $S(w)$ involves constructing 3 stacks of intervals A, B and C.

To construct A, for $i \in [w]$, we introduce an interval $a_i = 1 + (i-1)/2w$. This forms an antichain of size w and therefore, a_i gets assigned the color i by First Fit.

To construct B, for $i \in [w]$, we introduce an interval $b_i = 3 + i/2w$. This forms an antichain of size w and therefore, b_i gets assigned the color i by First Fit.

Lastly, to construct C, for $i \in [w]$, we introduce an interval $c_i = 2+i/2w$. This forms an antichain of size w. Moreover, c_i intersects a_j if and only if $j > i$, and c_i intersects b_j if and only if $j \leq i$ for all $i \in [w-1]$. Therefore, every interval in C gets assigned a new color. In particular, c_i gets assigned the color $w + i$ for $i \in [w-1]$. Thus, in totol First Fit is forced to use $w + (w-1) = 2w - 1$ colors to partition a semi-order of width w.

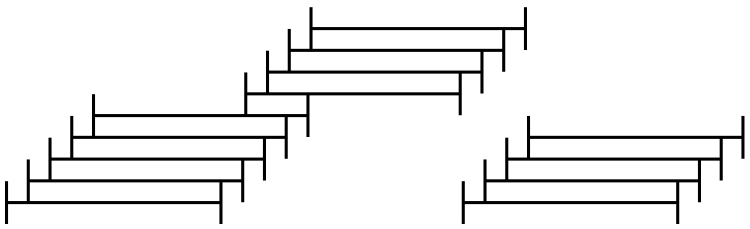


Figure 6.2: The Strategy $S(5)$.

This concludes the proof. □

6.3 Greedy Width of Interval Orders

The greedy width gwi(w) of the class of interval orders of width at most w is the largest integer k for which there exists a strategy that forces any greedy algorithm to use k chains to partition a poset of width w. Theorem 6.1.3 says that we simply have to prove that we can force First Fit to use k colors in order to prove gwi(k) $\geq k$. Luckily for us, there is a rich history of finding the performance of first coloring of interval graphs. Let $FF(w)$ be the largest integer for which First Fit can be forced to use t chains on an interval order with width at most w. In 1976, Woodall [20] showed that $FF(w) = O(w \log w)$. In 1988, Kierstead [11] showed that $FF(w) \leq 40w$, and Kierstead and Qin [12] improved this in 1995 to $FF(w) \leq 25.8w$. From below, it follows from the work of Kierstead and Trotter [16] on on-line coloring of interval graphs that so that $FF(w) \geq 3w - 2$. In 1990, Chrobak and Ślusarek [5] proved that $FF(w) \geq 4w - 9$ when $w \geq 4$, and they later improved the lower bound to $4.4w - C$, where C is a large constant. In 2003, Pemmaraju, Raman and Varadarajan [18] made a major breakthrough by showing that $FF(w) \leq 10w$ and commented that their upper bound might be improved but that the technique wouldn't yield a result better than $FF(w)/w \leq 8$, when w is large. Later in 2003, their predictions were confirmed, and their technique was refined by Brightwell, Kierstead and Trotter (unpublished) to obtain an upper bound of $8w$ on $FF(w)$. In 2004, Narayansamy and Babu [17] found an even cleaner argument for this bound that actually yields the slightly stronger result: $FF(w) \leq 8w - 3$. Howard has recently

pointed out that one can actually show that $FF(w) \leq 8w - 4$. Later in 2004, Kierstead and Trotter gave a computer proof that $FF(w) >= 4.99w - C$. This technique was subsequently refined to show that $FF(w) >= 4.99999w - C$. Kierstead, Smith and Trotter showed that for every $e > 0$, $FF(w) > (5 - e)w$, when w is sufficiently large. Again, all of the bounds above on $FF(w)$ also hold for gwi(w) by Theorem 6.1.3 and by the fact that First Fit is a greedy algorithm.

Theorem 6.3.1 (Narayansamy and Babu [17], Kierstead, Smith and Trotter)**.** *For every $w \geq 1$, $8w - 4 \leq \text{gwi}(w) \leq 5w - \epsilon$ for every $\epsilon > 0$.*

6.4 Almost-Semiorders

We consider a class that is "almost" a semi-order. In particular, we consider the class of posets that have an interval representation where the intervals all have unit-length. Though unlike semi-orders, the intervals can be either closed or open. Ingeniously, we refer to posets in this class as almost-semiorders. A nearly identical argument as the one we use in Proposition 5.1.1 proves that this class of posets is strickly between that of semi-orders and that of interval orders.

Proposition 6.4.1. *Let $\mathcal{S}$ denote the class of semi-orders, $\mathcal{A}$ denote the class of almost-semiorders and $\mathcal{I}$ denote the class of interval orders. Then $\mathcal{S} \subset \mathcal{A} \subset \mathcal{I}$.*

Proof. Clearly, $\mathcal{S} \subset \mathcal{A}$. The fact that $\mathcal{A} \subset \mathcal{I}$ is not immediate considering that open intervals are not allowed in the interval representation of interval orders. However, by Theorem 4.1.1, it suffices to see that an almost-semiorder does not contain a $\mathbf{2} + \mathbf{2}$.

To show that $\mathcal{S} \neq \mathcal{A}$, it suffices to see that the poset $\mathbf{3} + \mathbf{1}$ is an almost-semiorder.

Figure 6.3: The almost-semiorder $\mathbf{3} + \mathbf{1}$ is not a semi-order.

To show that $\mathcal{A} \subset \mathcal{I}$, consider the interval order below.

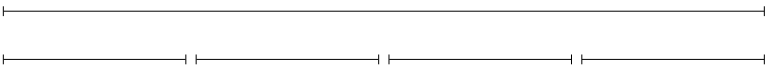


Figure 6.4: An interval order that is not an almost-semiorder.

For obvious reasons, we may refer to this interval order as $\mathbf{4} + \mathbf{1}$. It is clear that $\mathcal{A}$ does not contain $\mathbf{4} + \mathbf{1}$ since Figure 6.4, shows the only way an interval can intersect three disjoint intervals in an almost-semiorder representation. This concludes the proof.

□

6.5 Greedy Width of Almost-Semiorders

The greedy width $\text{gwa}(w)$ of the class of almost-semiorders of width at most w is the largest integer k for which there exists a strategy that forces any greedy algorithm to use k chains to partition an almost-semiorder of width w. Theorems 6.2.1 and 6.3.1 imply that $2w-1 \leq \text{gwa}(w) \geq 8w-4$. We show that neither of these bounds are tight. In fact, we show that

$$\text{gws}(w) < \text{gwa}(w) < \text{gwi}(w).$$

6.5.1 Proof of Theorem 6.0.1 (Lower Bound)

Since in this variant, we introduce the elements of the poset (X, P) as unit-intervals of which are either open or closed, we may define each element by a real number r_i. More specifically, we define each element introduced by the right endpoint of the interval in the representation so that if we introduce the element x_i as the closed interval $[r_i - 1, r_i]$, we simply define x_i by $x_i = [r_i]$ and if we introduce the element x_i as the open interval $(r_i - 1, r_i)$, we simply define x_i by $x_i = (r_i)$. Let $k = \lfloor w/2 \rfloor$. The strategy consists of 4 stages. By Theorem 6.1.3, we may assume that Bertha uses First Fit to partition the poset. Then in each stage, First Fit will use k new colors.

6.5.2 Stage 1

In Stage 1, Anna constructs 4 disjoint stacks A_1, A_2, A_3, A_4 of k intervals each as follows. In order to construct A_i, Anna presents intervals $a_1^i, \ldots, a_k^i$ so that $a_j^i = i^0$ for $j \in [k]$. Hence, Bertha assigns the color j to interval a_j^i for every $j \in [k]$ and $i \in \{1, 2, 3, 4\}$.

Suppose $w = 2k$ for some positive integer k. We begin by introducing a stack of intervals $x_1, \ldots, x_k$ so that $x_i = 0^O$ for $i \in \{1, \ldots, k\}$. Notice that the intervals in Stage 1 form an antichain, and hence, must each be assigned a distinct chain. Let A denote the set of chains $\{a_1, \ldots, a_k\}$ used in Stage 1.

6.5.3 Stage 2

In stage 2, Anna introduces 2 disjoint stacks of intervals of intervals B_1, B_2 of k intervals each. To construct B_1, Anna introduces intervals $b_1^1, \ldots, b_k^1$ so that $b_i^1 = [1]$ for $i \in [k]$. To construct B_2, Anna introduces intervals $b_1^2, \ldots, b_k^2$ so that $b_i^2 = [4]$. Since every interval presented in Stage 2 intersects a stack from Stage 1, Bertha must assign the color $k + i$ to interval a_i^j for $i \in [k]$ and $j \in \{1, 2\}$.

6.5.4 Stage 3

In Stage 3, Anna introduces a stack C of k intervals. To construct C, Anna introduces intervals $c_1, \ldots, c_k$ so that $c_i = [3]$ for $i \in [k]$. For every $i \in [k]$ and $j \in [k]$, $c_i \cap a_j^3 = (2, 3)$ and $c_i \cap b_j^2 = \{3\}$. Hence, Bertha must assign the color $2k + i$ to c_i for every $i \in [k]$.

6.5.5 Stage 4

Finally, in Stage 4, Anna introduces a stack D of k intervals. To construct D, Anna introduces intervals $d_1, \ldots, d_k$ so that $d_i = [2]$ for $i \in [k]$. For every $i \in [k]$ and $j \in [k]$,

$$c_i \cap a_j^2 = (1, 2),$$

$$c_i \cap b_j^1 = \{1\},$$

and

$$c_i \cap c_j = \{2\}.$$

Hence, Bertha must assign the color $3k + i$ to c_i for every $i \in [k]$. Thus Bertha is forces to use $4k = 2w$ colors on an almost-semiorder of width w.

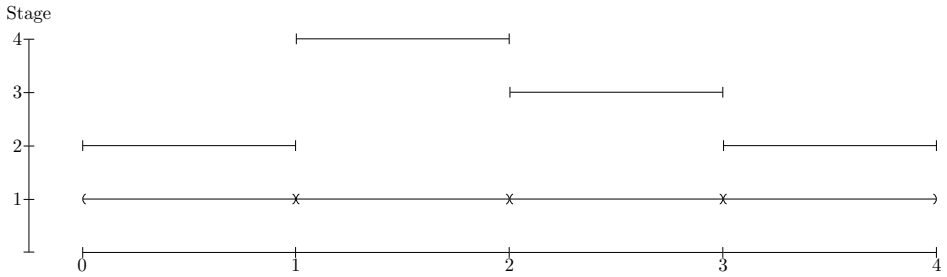


Figure 6.5: Strategy $S(2)$ forcing 4 colors.

6.5.6 Proof of Theorem 6.0.1 (Upper Bound)

First we prove that $\text{gwa}(w) \leq 3w-2$. To show this, consider the greedy algorithm First Fit. Suppose First Fit has successfully partitioned the almost semiorder (X, P) up to round $i-1$. Now suppose in round i, Anna introduces a new point x. Consider any interval representation I of the poset (X, P). Then the only way that First Fit assigns x the color $3w-1$ is if $I(x)$ intersects $3w-2$ other intervals of all different colors. However, it is easy to see that $I(x)$ can intersect at most $3w-3$. In particular

if $I(x)$ intersects $3w-2$ other intervals, then it must be in the way shown below.

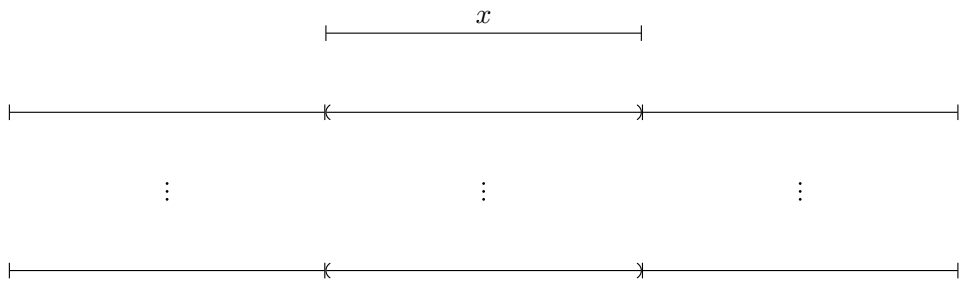


Figure 6.6: Every interval intersects at most $3w-3$ other intervals.

This proves that $\text{gwa}(w) \leq 3w-2$. It is unlikely that First Fit needs this many colors. In fact, consider the scenario where x intersects $3w-3$ other intervals of all colors in $[3w-3]$. Let y be the point that was assigned the color $3w-3$. Then $I(y)$ must intersect $3w-4$ intervals but as we can see in the figure above, y intersects $w-2$ other intervals already and could at most intersect another 2. Hence, $I(y)$ intersects no more than w other intervals. This improves the upper bound to $3w-3$.

It is easy to see that with some work, this argument can be extended to improve the upper bound even more. In particular, $\text{gwa}(w) \leq 3w - f(w)$ for some function f which increases with w.

REFERENCES

[1] Bartł omiej Bosek and Tomasz Krawczyk. "A subexponential upper bound for the on-line chain partitioning problem". In: *Combinatorica* 35.1 (2015), pp. 1–38. ISSN: 0209-9683. DOI: 10.1007/s00493-014-2908-7. URL: https://doi-org.echo.louisville.edu/10.1007/s00493-014-2908-7.

[2] Bartł omiej Bosek et al. "An easy subexponential bound for online chain partitioning". In: *Electron. J. Combin.* 25.2 (2018), Paper No. 2.28, 23. DOI: 10.37236/7231. URL: https://doi-org.echo.louisville.edu/10.37236/7231.

[3] Bartłomiej Bosek and Tomasz Krawczyk. "On-line partitioning of width w posets into $w^{O(\log \log w)}$ chains". In: *European J. Combin.* 91 (2021), Paper No. 103202, 17. ISSN: 0195-6698. DOI: 10.1016/j.ejc.2020.103202. URL: https://doi.org/10.1016/j.ejc.2020.103202.

[4] Bartłomiej Bosek et al. "On-line chain partitions of orders: a survey". In: *Order* 29.1 (2012), pp. 49–73. ISSN: 0167-8094. DOI: 10.1007/s11083-011-9197-1. URL: https://doi.org/10.1007/s11083-011-9197-1.

[5] Marek Chrobak and Maciej Ślusarek. "On some packing problem related to dynamic storage allocation". In: *RAIRO Inform. Théor. Appl.* 22.4 (1988), pp. 487–499. ISSN: 0296-1598.

[6] Joanna Chybowska-Sokół et al. "Online coloring of short intervals". In: *Approximation, randomization, and combinatorial optimization. Algorithms and techniques*. Vol. 176. LIPIcs. Leibniz Int. Proc. Inform. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2020, Art. No. 52, 18.

[7] R. P. Dilworth. "A decomposition theorem for partially ordered sets". In: *Ann. of Math. (2)* 51 (1950), pp. 161–166. ISSN: 0003-486X. DOI: 10.2307/1969503. URL: https://doi-org.echo.louisville.edu/10.2307/1969503.

[8] Leah Epstein and Meital Levy. "Online interval coloring and variants". In: *Automata, languages and programming*. Vol. 3580. Lecture Notes in Comput. Sci. Springer, Berlin, 2005, pp. 602–613. DOI: 10.1007/11523468_49. URL: https://doi.org/10.1007/11523468_49.

[9] Stefan Felsner. "On-line chain partitions of orders". In: *Theoret. Comput. Sci.* 175.2 (1997). Orders, algorithms and applications (Lyon, 1994), pp. 283–292. ISSN: 0304-3975. DOI: 10.1016/S0304-3975(96)00204-6. URL: https://doi.org/10.1016/S0304-3975(96)00204-6.

[10] Tosio Hiraguti. "On the dimension of orders". In: *Sci. Rep. Kanazawa Univ.* 4.1 (1955), pp. 1–20. ISSN: 0022-8338.

[11] H. A. Kierstead. "The linearity of first-fit coloring of interval graphs". In: *SIAM J. Discrete Math.* 1.4 (1988), pp. 526–530. ISSN: 0895-4801. DOI: 10.1137/0401048. URL: https://doi-org.echo.louisville.edu/10.1137/0401048.

[12] H. A. Kierstead and Jun Qin. "Coloring interval graphs with First-Fit". In: vol. 144. 1-3. Combinatorics of ordered sets (Oberwolfach, 1991). 1995, pp. 47–57. DOI: 10.1016/0012-365X(94)00285-Q. URL: https://doi-org.echo.louisville.edu/10.1016/0012-365X(94)00285-Q.

[13] Henry A. Kierstead. "An effective version of Dilworth's theorem". In: *Trans. Amer. Math. Soc.* 268.1 (1981), pp. 63–77. ISSN: 0002-9947. DOI: 10.2307/1998337. URL: https://doi.org/10.2307/1998337.

[14] Henry A. Kierstead. "Recursive ordered sets". In: *Combinatorics and ordered sets (Arcata, Calif., 1985)*. Vol. 57. Contemp. Math. Amer. Math. Soc., Providence, RI, 1986, pp. 75–102. DOI: 10.1090/conm/057/856233. URL: https://doi.org/10.1090/conm/057/856233.

[15] Henry A. Kierstead, George F. McNulty, and William T. Trotter Jr. "A theory of recursive dimension for ordered sets". In: *Order* 1.1 (1984), pp. 67–82. ISSN: 0167-8094. DOI: 10.1007/BF00396274. URL: https://doi.org/10.1007/BF00396274.

[16] Henry A. Kierstead and William T. Trotter Jr. "An extremal problem in recursive combinatorics". In: *Congr. Numer.* 33 (1981), pp. 143–153. ISSN: 0384-9864.

[17] N. S. Narayanaswamy and R. Subhash Babu. "A note on first-fit coloring of interval graphs". In: *Order* 25.1 (2008), pp. 49–53. ISSN: 0167-8094. DOI: 10.1007/s11083-008-9076-6. URL: https://doi-org.echo.louisville.edu/10.1007/s11083-008-9076-6.

[18] Sriram V. Pemmaraju, Rajiv Raman, and Kasturi Varadarajan. "Buffer minimization using max-coloring". In: *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. ACM, New York, 2004, pp. 562–571.

[19] William T. Trotter Jr. and John I. Moore Jr. "The dimension of planar posets". In: *J. Combinatorial Theory Ser. B* 22.1 (1977), pp. 54–67. ISSN: 0095-8956. DOI: 10.1016/0095-8956(77)90048-x. URL: https://doi-org.echo.louisville.edu/10.1016/0095-8956(77)90048-x.

[20] D. R. Woodall. "Applications of polymatroids and linear programming to transversals and graphs". In: *Combinatorics (Proc. British Combinatorial Conf., Univ. Coll. Wales, Aberystwyth, 1973)*. London Math. Soc. Lecture Note Ser., No. 13. Cambridge Univ. Press, London, 1974, pp. 195–200.

CURRICULUM VITAE

Israel R. Curbelo

	Department of Mathematics	(917)385-6703
	University of Louisville	israel.curbelo@louisville.edu
	Louisville, Kentucky 40292 USA	github.com/OnlineDimension

EDUCATION

2018 – 2022	Ph.D. in Applied Mathematics, *University of Louisville*
2021 – 2022	Graduate Certificate in Data Science, *University of Louisville*
2016 – 2018	Master of Arts in Mathematics, *University of Louisville*
2012 – 2016	Bachelor of Arts in Mathematics, *The College of New Jersey*

PUBLICATIONS

Csaba Biró and Israel R. Curbelo, *On-line partioning of d-dimensional posets*, Submitted. arXiv:2111.04802

Csaba Biró and Israel R. Curbelo, *On-line partioning of semi-orders with unit-interval representation,* Submitted. arXiv:2111.04790

Csaba Biró and Israel R. Curbelo, *Weak independence of events and the converse of the Borel–Cantelli lemma,* to appear in Expositiones Mathematicae. arXiv:2004.11324

TALKS

Feb. 2021	*On-Line Chain Partition Game*, AMS Student Chapter, University of Louisville, Louisville, Kentucky.
Mar. 2020	*On-line Dimension of Posets*, AMS Student Chapter, University of Louisville, Louisville, Kentucky.
Mar. 2019	*Markov Chains on General State Spaces*, AMS Student Chapter, University of Louisville, Louiville, Kentucky.
Feb. 2018	*Space-Filling Curves 4 Dummies*, AMS Student Chapter, University of Louisville, Louisville, Kentucky.
May 2016	*Space-Filling Curves*, Celebration of Student Achievement, The College of New Jersey, Ewing, New Jersey.

RESEARCH EXPERIENCE	2020 – 2022	Borel–Cantelli extensions, and on-line algorithms. *with Csaba Biró, University of Louisville.*
	2020	Sperner's lemma, Brouwer fixed point theorem, KKM lemma. *with Jenő Lehel, University of Louisville.*
	2018 – 2019	Markov chains and change–point asymptotics. *with Ryan Gill, University of Louisville.*
	2018	Ergodic theory & rank 1 transformations . *with Aaron Hill, University of Louisville.*
	2015 – 2016	Conditions for space-filling curves. *with Judit Kardos, The College of New Jersey.*

TEACHING EXPERIENCE	2021	Fall	Teaching Assistant	College Algebra (2)
		Summer	Instructor	Calculus I
		Spring	Teaching Assistant	College Algebra
		Spring	Teaching Assistant	Elements of Calculus
		Spring	Teaching Assistant	College Algebra
	2020	Fall	Instructor	Precalculus
		Summer	Instructor	College Algebra
		Spring	Teaching Assistant	Elements of Calculus
	2019	Fall	Teaching Assistant	Elementary Statistics (2)
		Fall	Teaching Assistant	Elements of Calculus
		Summer	Instructor	Precalculus
	2018	Fall	Teaching Assistant	Elementary Statistics (2)
		Fall	Teaching Assistant	Elements of Calculus
		Summer	Instructor	Quantitative Reasoning
		Spring	Teaching Assistant	College Algebra (2)
	2017	Fall	Teaching Assistant	College Algebra (3)
		Spring	Teaching Assistant	Elements of Calculus
	2016	Fall	Teaching Assistant	College Algebra (2)
		Spring	Teaching Assistant	Real Analysis II
	2015	Fall	Teaching Assistant	Real Analysis
		Spring	Teaching Assistant	Real Analysis

WORK EXPERIENCE	2022 –	Assistant Professor Kean University, Union Township, New Jersey
	2016 – 2022	Graduate Teaching & Research Assistant University of Louisville, Louisville, Kentucky
	2020 – 2021	President of the Graduate Student Council University of Louisville, Louisville, Kentucky
	2015 – 2016	Undergraduate Teaching Assistant The College of New Jersey, Ewing, New Jersey