2. Calculating Dimension

After deciding that one of our main goals in this thesis would be to determine the dimension of products of orders, it became immediately apparent that we needed a good way to calculate the dimension in order to test hypotheses and verify examples. Yannakakis [30] showed that determining the order dimension of a partial order is NP-Hard using a reduction from the graph coloring problem, hence finding an efficient algorithm is difficult. Furthermore, there are few publicly available tools to calculate poset dimension. The SageMath software system [24] does provide an implementation, but this is often far too slow, even for comparatively small examples.

There have also been few published attempts at finding empirically fast algorithms. Koppen [19] as well as Yáñez and Montero [29] have described potential algorithms, but they neither seemed easy to implement, nor was it likely that they would greatly outperform our methods. We have primarily considered two different approaches to this problem. Both of them involve reductions to NP-Complete problems and then using modern solving tools to calculate a solution.

2.1. Selecting Pairs with SAT

There is a sensible way to view the decision problem, whether an order admits a realizer of size at most n, as an instance of SAT. This is best derived from the technical definition of a poset. A total order is a reflexive, antisymmetric and transitive relation requiring that for each pair exactly one of its orientations is chosen. To transfer this over to a SAT instance, we can ignore the pairs obtained from reflexivity, as they are always contained. Antisymmetry and the need to choose exactly one orientation for each pair can be encoded by the variables themselves, as we have to make exactly one of two choices for each variable in any assignment. Transitivity is then encoded by the clauses of the SAT formula. Since transitivity constraints consist of simple implications on at most three literals, this is an encoding that SAT is well-suited to.

The concrete formulation we came up with is as follows. Let $P = (X, \leq)$ be a poset. Define a variable x_{ab} for each unordered incomparable pair $\{a, b\} \subseteq X, a \mid\mid b$, where the order of (a, b) is fixed arbitrarily. Let V be the set of these variables. For each ordered pair (a, b) with $a \neq b$ declare a corresponding literal

$$l(a,b) := \begin{cases} x_{ab} & \text{if } x_{ab} \in V \\ \neg x_{ba} & \text{if } x_{ba} \in V \end{cases} .$$

This describes the aforementioned choice for each pair. In order to make sure that these choices obey transitivity constraints, add clauses

$$C_{abc} := \neg l(a,b) \vee \neg l(b,c) \vee l(a,c) \text{ for all } \{a,b,c\} \subseteq X.$$

Solving just this SAT formula would return some permutation of the elements of X. To find a linear extension of P now just fix the literals $l(a,b)$ as 1 for all (a,b) with $a \leq b$. Fixing one such literal can be accomplished by adding an additional clause with just $l(a,b)$. Alternatively we can also perform some preprocessing manually. Essentially, one ignores a clause C if $l(a,b) \in C$ and remove $\neg l(a,b)$ from C if $\neg l(a,b) \in C$. Let $\widehat{C}_{abc}$ be the clause after applying this reduction. Then we set

$$L := \bigwedge_{a,b,c \in X} \widehat{C}_{abc}.$$

Now any satisfying assignment of L will correspond to a linear extension of P. Finding a realizer of size k now requires little additional work. Let L_i be an independent copy of L for $1 \leq i \leq k$ and denote its literals by $l(a,b,i)$. Introducing clauses

$$C_{ab} = l(a,b,1) \vee \cdots \vee l(a,b,k)$$

for each $(a,b) \in \operatorname{inc}(P)$ will ensure that each incomparable pair is covered by the realizer at least once. Thus, the final formula is given by

$$S_k := \bigwedge_{i=1}^{k} L_i \wedge \bigwedge_{(a,b) \in \operatorname{inc}(P)} C_{ab}. \tag{2.1}$$

If the formula is satisfiable, recovering the realizer from a satisfying assignment is not hard. For fixed i, the comparable pairs of P form a total order with those a,b where $l(a,b,i)$ was set to 1. Calculating a topological order of the directed comparability graph of P with the added (a,b) will then give the desired result. Finally, determining the order dimension now just involves finding the minimal k such that S_k is satisfiable.

When implementing this approach, it is sensible to apply the SAT formula to a subposet of P with the same dimension. The natural choice for this is the poset induced by $\{x \mid (x,y) \in \operatorname{crit}(P)\} \cup \{y \mid (x,y) \in \operatorname{crit}(P)\}$. This must have the same dimension as P according to Fact 1.3. We did implement this, but did generally not see significant performance gains. It seems probable that the SAT solver can learn which pairs are being forced by other pairs from the given clauses and thus determine that mostly only the critical pairs are important.

2.2. Digraph Coloring

In SageMath, order dimension is calculated by determining a coloring of the hypergraph of incomparable pairs. As described by Felsner and Trotter [11], it is defined by $G = (\operatorname{inc}(P), E)$, with E defined as the set of $S \subseteq \operatorname{inc}(P)$ that contain an alternating cycle but where all subsets of S do not contain an alternating cycle. In general, this hypergraph