The Complexity of Embedding Orders into Small Products of Chains

Olivier Raynaud · Eric Thierry

Received: 30 August 2008 / Accepted: 28 October 2009 / Published online: 13 November 2009
© Springer Science + Business Media B.V. 2009

Abstract Embedding a partially ordered set into a product of chains is a classical way to encode it. Such encodings have been used in various fields such as object oriented programming or distributed computing. The embedding associates with each element a sequence of integers which is used to perform comparisons between elements. A critical measure is the space required by the encoding, and several authors have investigated ways to minimize it, which comes to embedding partially ordered sets into small products of chains. The minimum size of such an encoding has been called the encoding dimension (Habib et al. 1995), and the string dimension (Garg and Skawratananand 2001) for a slightly different definition of embeddings. This paper investigates some new properties of the encoding dimension. We clear up the links with the string dimension and we answer the computational complexity questions raised in Garg and Skawratananand (2001) and Habib et al. (1995): both these parameters are \( \mathcal{NP} \)-hard to compute.

Keywords Partially ordered sets · Encodings · Optimization

1 Introduction

Partially ordered sets (orders for short) occur in numerous fields of computer science, like distributed computing, programming languages, databases or knowledge representation. Such applications have raised the need for storing and handling them
efficiently. Many ways of encoding partially ordered sets have been proposed in the literature. Depending on the purposes, several criteria are commonly considered to guide the choice of the most appropriate encoding. One may cite the compromise between speeding up operations and saving space, the choice between dynamic or static data structures with regard to possible modifications of the order, the complexity of generating the encoding from usual data structures (like matrices or lists of successors), the restrictions on the data structures imposed by hardware and software (e.g. storing the order in a database which can be then accessed only by means of SQL requests). Performing fast comparisons between elements while saving space is the most usual issue.

Here is a non-exhaustive list of approaches that have been studied: numbering the elements in order to compress their lists of successors [1, 39], partitioning the order into nice subsets like antichains [10, 16, 45] or chains [6, 16, 30, 34, 36, 40], mixing numbering and partitioning [23, 48], seeing the order as the inclusion order on some geometrical shapes [2, 18], describing the order as the union of nice orders on the same set of elements [8, 46], describing the order by combinations of boolean formulas on integer tuples [11, 19–21], focusing on lattice operations [4, 42].

Another classical scheme consists in embedding the order into another one which is known to have a nice representation. More formally, let $P = (X, \leq_P)$ be an order where $\leq_P$ is an order relation (i.e. reflexive, antisymmetric and transitive) on a ground set $X$, and $Q = (Y, \leq_Q)$ another order. An order embedding (embedding for short) of $P$ into $Q$ is a mapping $\varphi$ from $X$ into $Y$ such that for all $x, y \in X$, $x \leq_P y$ if and only if $\varphi(x) \leq_Q \varphi(y)$. We will denote the existence of such an embedding of $P$ into $Q$ by $P \hookrightarrow Q$. By requiring that $Q$ should belong to a particular class of orders, different interesting classes of embeddings can be defined.

This article focuses on finite orders and investigates a class of embeddings which have been highlighted by Habib et al. [24], namely embeddings of orders into products of chains. Let $n \geq 1$ be an integer, a chain of size $n$ is a total order with $n$ elements. Up to an isomorphism, it can be represented by the order $(0 < 1 < 2 < \cdots < n - 1)$ which is denoted $[n]$. Then, let $n_1, n_2, \ldots, n_d \geq 1$ be $d \geq 1$ integers, we denote by $[n_1] \times [n_2] \times \cdots \times [n_d]$ the product of the $d$ chains where the elements are the corresponding $d$-uples $\{(x_1, x_2, \ldots, x_d) \mid \forall 1 \leq i \leq d, 0 \leq x_i \leq n_{i} - 1\}$ and where $(x_1, x_2, \ldots, x_d) \leq (y_1, y_2, \ldots, y_d)$ if and only if $\forall 1 \leq i \leq d, x_i \leq y_i$.

Any embedding $\varphi$ of $P$ into some product of chains $[n_1] \times [n_2] \times \cdots \times [n_d], d \geq 1$ provides a simple way to encode $P$: each element $x \in X$ is labelled by its image $\varphi(x)$. Among the advantages, this information can be stored locally and no spare data structure is needed to perform the comparisons in $P$. Using pairwise comparisons of integers is simple enough to be implemented in various contexts. Up to small adjustments depending on the precise way the $d$-uples will be stored, the size of the label associated with each element is $\sum_{i=1}^{d} \log_2(n_i)$ bits. The comparison between two elements requires $d$ comparisons of integers, that is $O(d)$ time which can be shortened if they are parallelized. Figure 1 shows the embeddings of two orders into some products of chains.

As a matter of fact, such embeddings have been intensively studied when some conditions are imposed on $d$ and the $n_i$'s. The first important results concern the existence of those embeddings for any order. From [14, 38], it is known that any order with $n$ elements can be embedded into some product of finite chains (of size $n$) and the smallest number of chains for which it works is called the dimension of $P$. 

\[ \sum \]