



Constructing concept lattices and compressed pseudo-lattices

by

F.J. (Dean) van der Merwe

Submitted in fulfilment of the requirements for the degree
M.Sc. Computer Science
in the Faculty Engineering, Built Environment
and Information Technology

University of Pretoria

Pretoria, South Africa

May 2003

Constructing concept lattices and compressed pseudo-lattices

By F.J. (Dean) van der Merwe
Supervisor: Prof. D.G. Kourie
Department of Computer Science
University of Pretoria
M.Sc. Computer Science

Summary

Concept lattices are well-defined mathematical structures used in a variety of areas of application. Concept lattices are exponential in size in the worst case and therefore constructing concept lattices may be time consuming. This dissertation approaches the efficiency problems associated with constructing concept lattices from two points of view and proposes complementary solutions for each. Firstly, a new lattice construction algorithm, called AddAtom, unrelated to well known and published algorithms is proposed. Secondly, a data structure called a compressed pseudo-lattice is proposed. Compressed pseudo-lattices involve the construction of sublattices instead of the complete lattice of all concepts.

The AddAtom algorithm efficiently constructs lattices and is proposed as a general purpose lattice construction algorithm. The algorithm is an incremental algorithm and has a theoretic complexity bound of $O(|L| \cdot |O|^2 \cdot \max(|O'|))$ when constructing the concept lattice, L of a context with a set of objects O each of which can possess a maximum of $\max(|O'|)$ attributes from a set of attributes A . In experimental comparisons AddAtom outperformed other published algorithms in a range of contexts. In randomly generated contexts that had very high or very low densities its performance was the second best. It was however the best performing incremental lattice construction algorithm.

The notion of a compressed pseudo-lattice is introduced and suggested as a data structure in areas of application related to Formal Concept Analysis (FCA). It is closely related to the line diagram of a lattice and its use as a computational tool in applications such as machine learning, information retrieval and knowledge discovery in databases is discussed. The data structure, essentially a bipartite graph that incorporates an embedded sublattice, combines desirable features of concept lattices in a structure that allows for a flexible mechanism of scaling the size of the embedded sublattice, using defined operations that compress and expand it by removing or adding atoms and coatoms. A compressed pseudo-lattice essentially represents a complete sublattice from which a number of atoms and/or coatoms have been removed. Additionally the relation of the sublattice to the context from which it was derived is preserved. An application-dependent compression strategy or criterion is required to guide this process. The intent- and extent representative operations of a lattice are defined as substitutes for the infimum and supremum operations in compressed pseudo-lattices. It is argued that the removal of concepts from a concept lattice may hold advantages over traditional approaches.



Acknowledgements

My sincere thanks to:

- My family for their support and encouragement.
- Deon Oosthuizen for introducing me to the subject during many enthusiastic discussions and his support and assistance during his tenure as an associate professor at the University of Pretoria during which he also acted as supervisor.
- Prof. Derrick Kourie for his assistance, patience and invaluable advice as supervisor.
- Sergei Obiedkov for his collaborative assistance and for making available the data of his experimental comparisons of AddAtom and other lattice construction algorithms as well as helpful comments and suggestions.



Table of contents

Acknowledgements	3
Table of contents	4
Chapter 1: Introduction	6
Chapter 2: Order theoretic and FCA lattice definitions and notation.....	9
2.1 List and set notation.....	9
2.2 Order theoretic lattice definitions	9
2.3 FCA definitions	11
2.4 Why EA-lattices?	13
2.5 EA-lattice definition	15
2.6 Boolean lattices	18
2.7 Augmented lattices	19
2.8 Different views of a lattice	19
2.8.1 Set-theoretic view of lattices.....	19
2.8.2 Graph-theoretic view of lattices	20
2.9 Intent and extent operations	21
2.10 Summary	23
Chapter 3: Lattice construction	24
3.1 Algorithmic lattice construction.....	24
3.2 Incremental vs. batch lattice construction algorithms.....	24
3.3 Constructing the line diagram.....	24
3.4 An inefficient batch lattice construction algorithm	25
Chapter 4: The AddAtom lattice construction algorithm.....	28
4.1 Informal description.....	28
4.2 Intent- and extent representative operations and lattice construction.....	33
4.3 Definition of the AddAtom algorithm.....	36
4.4 AddAtom example.....	39
4.5 An algorithm for AIR and EIR	41
4.6 Efficient AddAtom algorithm	45
4.7 Discussion	48
Chapter 5: AddAtom algorithmic performance	49
5.1 A survey of concept lattice construction algorithms.....	50
5.2 A Theoretical performance bound for AddAtom	52
5.2.1 Notation	52
5.2.2 Concept lattice size formulae	53
5.2.3 Complexity of set operations	57
5.2.4 AddAtom theoretical performance	58
5.3 Empirical performance: pilot study	61
5.4 Empirical performance: wide study.....	69
5.5 Conclusions.....	76
Chapter 6: Compressed pseudo-lattices	77
6.1 A bipartite database and query operation.....	77
6.2 A concept lattice database and query operation	79
6.3 An adapted sublattice database and query operation	79
6.4 Removing concepts from a lattice.....	80
6.5 The use of the intent- and extent representative operations	83
6.6 The CompressLattice operation.....	84
6.7 Definition and properties of compressed pseudo-lattices	85
6.8 The ExpandLattice operation.....	88
6.9 Interpretation of compressed pseudo-lattices	89
6.10 Why compressed pseudo-lattices?	91
6.11 Compression strategies and criteria.....	94
6.12 Implementation and discussion of preliminary results.....	96
Chapter 7: AddAtom implementation.....	98



7.1 Evolution of code.....	98
7.2 Set class	99
7.3 Lattice class	99
7.4 Compressed pseudo-lattice implementation	101
7.5 Implementation issues.....	102
7.5.1 Time	102
7.5.2 Space / memory	102
7.5.3 Object-oriented implementation	103
7.5.4 UNIX and Windows.....	103
7.5.5 Testing.....	104
7.5.6 User interface	105
7.5.7 Continued advances in hardware.....	105
7.6 Comparison with other lattice construction algorithms	105
Chapter 8: Summary and future work.....	106
8.1 Positioning and related research	106
8.2 Further work.....	108
References.....	110
Appendix A: Further optimised AddAtom algorithm.....	114
Appendix B: AddAtom algorithmic complexity bounds.....	118