Methods for Two-Party Privacy-Preserving Linear Programming

Alice Bednarz

Thesis submitted for the degree of
Doctor of Philosophy
in
Applied Mathematics
at
The University of Adelaide

Discipline of Applied Mathematics
School of Mathematical Sciences

January 22, 2012
## Contents

Abstract ...................................... vii  
Declaration .................................... ix  
Acknowledgments ................................ xi  

1 Introduction ..................................... 1  
   1.1 Cooperating with people you don’t trust ................. 1  
   1.2 Collaboration in the business world .................... 1  
       1.2.1 Benefits of collaboration .......................... 2  
       1.2.2 Barriers to collaboration ......................... 7  
       1.2.3 A solution: secure multiparty computation ......... 9  
   1.3 Summary of this thesis ............................ 10  

2 Background ..................................... 13  
   2.1 Secure multiparty computation ........................ 13  
       2.1.1 Brief history .................................... 13  
       2.1.2 Problem formulation ............................. 14  
       2.1.3 What do we mean by ‘secure’? .................... 14  
       2.1.4 Core concepts .................................. 18  
       2.1.5 Secure building blocks ........................... 20  
       2.1.6 Protocol composition ............................. 23  
       2.1.7 Application areas ............................... 24  
       2.1.8 Challenges: theory to practice .................... 25  
       2.1.9 Related secure optimisation work .................. 26  
   2.2 Linear programming ............................. 28
3 A Classification System for Privacy-Preserving LPs

3.1 Failure of canonical form 45
3.2 Defining the data partitioning 46
  3.2.1 Types of data partitioning 47
  3.2.2 Notation for data-partitioning 49
3.3 Beyond data-partitioning 50
3.4 Failure of constraint and variable conversions 51
  3.4.1 Constraint type classifications 53
  3.4.2 Variable type classifications 53
3.5 Output considerations 54
  3.5.1 Specifying the output partitioning 55
3.6 Full notation system 56
  3.6.1 Demonstration 57
  3.6.2 Extensions 57
3.7 Conclusion 59

4 Transformation Methods: Hiccups and Solutions

4.1 Background 61
  4.1.1 Transformation methods: the general idea 62
4.2 Du’s transformation method 63
  4.2.1 Why Programs A & B are not equivalent 64
4.3 Vaidya’s transformation method 65
4.4 Validity vs security 67
  4.4.1 Alice’s attack 68
4.4.2 Restoring security ........................................ 70
4.5 The generalised-permutation algorithms .......................... 74
  4.5.1 Base protocol: two-party protocol .......................... 74
  4.5.2 Sub-protocol: secure evaluation of $\hat{c}$ .................. 75
  4.5.3 Variation 1 — semi-trusted third party .................. 76
  4.5.4 Variation 2 — additively-split output .................. 77
4.6 Conclusion ........................................ 77

5 Practical Security Analysis ........................................ 79
  5.1 The acceptable security paradigm .......................... 79
  5.2 Partial exposure of private data .......................... 80
    5.2.1 Preliminary example ................................ 81
    5.2.2 Alice’s probability of revealing $k$ correct columns of $M$ 82
    5.2.3 Repeated instances of similar LPs .................... 86
  5.3 Other side issues to consider to ensure the claimed security-level .... 89
    5.3.1 Distribution of the random elements .................... 90
    5.3.2 Preliminary constraint rescalings .................... 91
    5.3.3 Features that cannot be disguised .................... 91
    5.3.4 Attacks Bob can mount against Alice .................. 92
  5.4 Conclusions ........................................ 93

6 Transformation Method for Row-Partitioned LPs .................. 95
  6.1 Background ........................................ 95
  6.2 Problem definition .................................... 96
  6.3 Solution approach .................................... 97
    6.3.1 A second layer of disguise: matrix $K$ ................ 100
    6.3.2 The transformation method .......................... 101
  6.4 Complexity ......................................... 104
  6.5 Detailed protocols ..................................... 105
8.4.1 Storing intermediate output ................................ 150
8.4.2 Composition of privacy-preserving protocols ............ 150
8.4.3 Notation ............................................. 151
8.4.4 Building blocks ....................................... 151
8.5 Privacy-preserving primal affine-scaling ..................... 152
  8.5.1 Step 1: Preliminary computations ........................ 152
  8.5.2 Step 2: Find shares of the dual variable vector ........... 155
  8.5.3 Step 3: Compute shares of the descent direction .......... 157
  8.5.4 Step 4: Find the step length .......................... 158
  8.5.5 Step 5: Find shares of the next interior point .......... 160
  8.5.6 Step 6: Check stopping conditions ...................... 160
8.6 Complexity analysis ....................................... 162
8.7 Conclusion .................................................. 163

9 An Attack on Privacy-Preserving Linear Programming ........... 165
  9.1 Introduction ............................................. 165
  9.2 Description of the attack .................................. 167
  9.3 Optimal attack ........................................... 169
    9.3.1 Optimal attack at a single vertex .................... 169
    9.3.2 Optimal assignment of constraints to vertices ....... 171
  9.4 A greedy heuristic for solving the minimisation problem .... 172
  9.5 Proof that the greedy heuristic is optimal .................. 173
  9.6 Discussion ............................................... 175
  9.7 Conclusion ................................................ 176

10 Conclusion and Outlook ........................................ 177

A Library of Methods ........................................... 179
  A.1 1-out-of-2 oblivious transfer algorithm .................... 179
  A.2 Selected secure building blocks ............................. 181
Imagine two companies who each manage part of a delivery network. Suppose these companies are considering the benefits of cooperation — would they be able to deliver packages faster and more cheaply if they could share each other’s networks? Answering this question would usually mean sharing private company information with each other, such as network topologies, internal capacities and budgets. Companies are reluctant to share such information, for fear of losing their competitive advantage, fear of revealing sensitive company information, or fear of breaching anti-trust laws. However, collaboration could bring mutual gain. What if it were possible to collaborate without sharing sensitive information?

Fortunately, a methodology called secure multiparty computation can alleviate these privacy concerns. Developed in the 1980s, it allows parties who don’t trust each other to compute functions involving their private data amongst themselves, whilst keeping their private information hidden. This method eliminates the need to find a trusted third party.

Linear programs are one of the most common types of optimisation problems, occurring widely in industry. For example, supply chain planning problems, delivery problems and scheduling problems can all be modelled as linear programs.

Recently, the idea of optimisation with privacy concerns has been studied in the form of privacy-preserving linear programming. The aim is to solve a linear program jointly-defined by two or more parties, who wish to solve the joint program for mutual benefit, yet are unwilling to share their data. In this thesis we address the two-party case, where each party owns a subset of the linear program’s components.

Solution methods for privacy-preserving linear programs can be grouped into two classes: cryptographic methods, and transformation methods. Cryptographic methods implement a privacy-preserving version of the Simplex method, while transformation methods ‘disguise’ the linear program using random matrices, allowing one to use any standard optimisation software to solve the linear program. The transformation methods are more efficient, but only provide ‘heuristic’ security guarantees; whereas the cryptographic methods have robust security guarantees, but suffer from slow performance.
This research investigates new variants of both methods. We correct a flaw in two of the early transformation methods, and subsequently develop two new transformation methods, each with a quantifiable worst-case security level. We also propose the first cryptographic method to be based on an interior point method.

We explain why the notion of a single ‘canonical’ form, to which all methods can be applied, does not exist for privacy-preserving linear programs. In particular, the transformation methods are extremely sensitive to the way the data is partitioned between the parties involved, and the type of constraints (inequality, equality) and variables (non-negative, free) in the linear program. To make it clear which scenarios a transformation method applies to, we develop a new notation system, in the spirit of Kendall’s queueing notation, to uniquely specify a privacy-preserving linear program. Our notation incorporates aspects which were previously overlooked.

Throughout the thesis, we explain why careful attention must always be paid to the output of a secure computation, and in the final chapter we present an attack on privacy-preserving linear programming that is entirely dependent on knowing the output (the optimal solution).

Although more work is needed to create commercially viable solution methods, privacy-preserving linear programming may one day allow companies to realise the potential benefits of collaboration, without the associated risks.
Declaration

I, Alice Bednarz, certify that this work contains no material which has been accepted for the award of any other degree or diploma in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text.

I give consent to this copy of my thesis, when deposited in the University Library, being made available for loan and photocopying, subject to the provisions of the Copyright Act 1968.

I also give permission for the digital version of my thesis to be made available on the web, via the University’s digital research repository, the Library catalogue and also through web search engines, unless permission has been granted by the University to restrict access for a period of time.

Signature........................................ Date.............
Acknowledgments

I would like to thank my supervisors, Associate Professor Matthew Roughan and Professor Nigel Bean, for their guidance, expertise, encouragement, and for always being so giving of their time. I feel very lucky to have had such wonderful supervisors. Whenever I was stuck or needed advice they always managed to help me find a solution; they were always very discerning and thorough proof-readers, and it was a pleasure to work with them.

Thanks also to everyone at the School of Mathematical Sciences for their friendship and support, especially my officemates for making the PhD a fun experience, and for all the enjoyable morning coffees!

There’s no way I could have finished this thesis without the support of my family and friends. Thanks to my sister Jana for all the little surprises, the home-cooked dinners, the interesting conversations, and for keeping me entertained with a steady supply of CDs and DVDs.

Thanks to Mum and Dad for their never-ending love and support; for all the dinners and rides to the bus station, and for putting up with me being stressed sometimes!

Thanks to Nanna and Papa for your delicious choc-bits and baking.

Thanks to the Hawkes’ for all the fun Wednesday night dinners and catch ups — they always helped take my mind off my thesis and de-stress!

A big thanks to Bernie for proof-reading some chapters for me, and for your helpful feedback and suggestions.

And finally thanks to Nathan for motivating me to keep going and keeping my spirits up — there’s no way I could have finished this if it wasn’t for your continual encouragement and pep talks — and for all your love and support, and for always making me laugh.