THE NATIONAL UNIVERSITY
of SINGAPORE

School of Computing
Lower Kent Ridge Road, Singapore 119260

TR10/05

An Application of Constraint Programming to Mobile Data Broadcasting□□

Stéphane BRESSAN, Wee Hyong TOK
and Steven PRESTWICH

October 2005

Technical Report

Foreword

This technical report contains a research paper, development or tutorial article, which has been submitted for publication in a journal or for consideration by the commissioning organization. The report represents the ideas of its author, and should not be taken as the official views of the School or the University. Any discussion of the content of the report should be sent to the author, at the address shown on the cover.

JAFFAR, Joxan
Dean of School

An Application of Constraint Programming to Mobile Data Broadcasting

Stéphane Bressan[1], Tok Wee Hyong[1] and Steven Prestwich[2]

1: National University of Singapore, School of Computing,
{steph,tokwh}@comp.nus.edu.sg
2: Cork Constraint Computation Centre, Department of Computer Science,
University College, Cork, Ireland, s.prestwich@cs.ucc.ie

Abstract. Given the current ubiquity of wireless networks and mobile devices, data must be disseminated to a large number of these devices in an effective and efficient manner. A promising technique for such applications is data broadcasting. We introduce a new optimisation problem derived from mobile data broadcasting applications. A single broadcaster transmits pages of data in sequence in response to requests from multiple clients, each asking for different subsets of a set of available pages. A transmitted page can be read by any number of clients, and we aim to minimise the total time taken to transmit all pages. Users impose additional constraints on how long they are willing to wait between pages, because of limited power supply in mobile devices. This problem is closely related to the multiple sequence alignment problem from bioinformatics. We present a constraint model for this problem, designed for use with randomised local search algorithms.

1 Introduction

Data broadcasting techniques for relational data have been widely studied [1–3], motivated by the need for large-scale data dissemination to a large number of clients using a wireless medium. Because of the recent proliferation of wireless networks (such as 3G and Wi-Fi) and mobile devices, there is renewed interest in the data broadcast model for delivering high-dimensional data (such as spatial data) over such media.

In mobile broadcasting applications a single broadcaster may have many clients. Simply broadcasting data to each client individually can be extremely wasteful of bandwidth and time, and does not scale up to a large number of clients. *Broadcast push* [15] allows a single broadcaster to send the same data to many clients, and is being used in commercial systems to deliver data by satellite and cable. In *broadcast pull* clients are allowed to request data, and the broadcaster decides the best strategy for delivering the data. *Broadcast disks* [5] aim to boost broadcast push performance via client caching of data and by the use of long-term cyclical broadcast schedules.

We address a related problem in which clients have distinct (acyclical) but possibly similar requirements. Each client may require a sequence of pages of data from a set of available pages, and requests this sequence in advance from the broadcaster. The broadcaster must then find a broadcast sequence such that all client requirements are satisfied (a *subscribe/publish* mechanism). To reduce broadcast time, similarities between client sequences should be exploited, for example two clients requesting the same sequence can be satisfied at the same time by broadcasting the sequence once. Identical subsequences should also be exploited. This amounts to finding a common superstring of the strings of client requests.

However, the minimal common superstring might be an extremely poor solution for some clients, and we must keep all clients reasonably happy. Moreover, mobile devices may have limited power supplies, and therefore be unable to wait for long periods. We therefore impose an additional requirement: no client should have to wait longer than a specified time between pages. To the best of our knowledge, this constraint and the advance specification of sequences make this a new problem for data broadcasting.

Clients may be mobile phones. Alternatively, the pages may be pages from a database needed to answer a query, but in a mobile broadcasting environment. A further possibility is that clients are viewers of a teletext system, and can submit the sequence of pages they want to see; the broadcast could combine all the sequences of pages into a good sequence, provided there is enough overlap among the pages of interest (this is likely: users typically want to see news, weather and so on, but possibly in different orders.)

2 A constraint model

Suppose we want to broadcast N pages $x_1 \dots x_N$ to a set of C clients, each of whom require a subset of the pages in a specified order, denoted by $y_{1j} \dots y_{m_j j}$ for client j. We want to find a optimal and feasible sequence containing those sequences (a *smallest common superstring*), possibly with repeated pages. There are also *gap* (or *offset*) *constraints*: an upper bound U on the number of pages any client must wait before receiving the next page. The problem is to minimise the superstring length, subject to these constraints. We shall refer to this as the *Gap-constrained Minimum Common Superstring* (GMCS) problem. The GMCS is closely related to the bioinformatics problem of *multiple sequence alignment* (MSA), some approach to which also model gaps [16], and we shall adapt a previous constraint-based approach to the MSA [13] to the GMCS.

We map client j's sequence to row j of an *alignment matrix*, and each page i in the sequence to a matrix column. These column positions are modelled by integer variables v_{ij}. Note that gaps are not explicitly modelled: for some (i, j) it will be true that $v_{ij} \neq c$ for all i, so that entry (i, c) is empty. In the MSA additional binary variables were introduced to model the alignment of pages with a specified degree of alignment desirability (identified by dynamic programming prior to optimisation). In other MSA work, gaps are modelled and have costs [4]. However, despite the extra gap constraints the GMCS is simpler than the MSA in the sense that only identical pages may be aligned, and aligning identical pages is always desirable. In the MSA two identical pages may occur in unrelated substrings, and should therefore not be aligned; and non-identical pages may be aligned because they are related in some way (for example some bases are more likely to mutate into others).

We post constraints on the v. The order of pages in client sequences must be preserved in the alignment matrix:

$$v_{ij} < v_{i+1\,j}$$

The new gap constraints mean that no client must wait longer than U between pages:

$$v_{i+1\,j} - v_{ij} \leq U$$

Non-identical pages must not be aligned:

$$v_{ij} \neq v_{i'j'}$$

where $j < j'$ and $y_{ij} \neq y_{i'j'}$. The space complexity of this model is $O(CL)$ variables and $O(C^2L^2)$ constraints (or literals, because the constraints have constant length). The model generalises easily to the case where clients have different U. Note that if $U < C$ there might be no solution: each client may ask for a sequence with no pages in common with any other sequence.

2.1 Deriving the superstring from the matrix

From an alignment matrix we can derive a superstring as follows. Any column is either empty or contains a single page in one or more rows. First we remove all empty columns, obtaining a (possibly) smaller matrix with C columns and (say) R rows. We then construct a string of length R whose i^{th} entry is the (only) page occurring in at least one row of column i. This is guaranteed to satisfy all constraints: removing empty columns cannot cause satisfied gap constraints to become violated, client sequences to be permuted, nor non-identical pages to become aligned.

2.2 Iterative improvement

To find an optimal or near-optimal superstring, we restart the search with an alignment matrix of $R - 1$ columns, and iteratively reduce R in this way until no better solution can be found. This is not guaranteed to find an optimal superstring, but it is a pragmatic way of obtaining a near-optimal one in a reasonable time. This approach is an example of an *anytime algorithm*, used in applications in which we wish to find a good solution in a limited time (the limit may not be known in advance).

2.3 An example

As an illustrative example, suppose we have available data pages $\{a, b, c, d\}$ and three clients with requested sequences (c, b, a), (a, b, c, d) and (a, b, d). The gap constraint impose a limit of $U = 3$ pages. We do not know in advance the optimal broadcast sequence length, so we guess a length of eight. After modelling and solving the problem we might obtain the alignment matrix shown in Figure 1. Note that the three rows contain the three customer sequences, that no client ever waits for more than three pages, and that no column contains more than one distinct page. However, column 6 is empty and can be deleted, giving a new broadcast sequence of length seven. The next step is to restart the search but with a maximum broadcast sequence length of six, and so on until no further improvement can be found.

1	2	3	4	5	6	7	8
c		b	a				
	a	b				c	d
	a	b		d			

Fig. 1. An alignment matrix example

This example is illustrative only, and we expect to find good solutions to much larger problems (with hundreds of clients and pages).

2.4 Symmetry, dominance and implied constraints

This may appear to be a poor model, as it allows many symmetric and dominated solutions. For example, any solution containing an empty column is symmetrical to a similar solution in which the empty column has been shifted to another position. It is also dominated by one in which the empty column has been removed, leaving a smaller matrix. We could add constraints to remove these solutions, thus pruning the search space considerably. This may improve the performance of a complete search algorithm such as branch-and-bound or a constraint solver.

However, we intend to apply an incomplete, randomised local search algorithm to the GMCS, which often scale more effectively than complete algorithms to large problems; the related approach of genetic algorithms has also been applied to MSA problems [9]. Experiments have shown that breaking symmetry can have a very negative effect on local search performance [12, 14], and similarly for dominance reasoning [11]. Finally, a similar model for the MSA gave better results without such additional constraints; nor could we find implied constraints that noticeably improved performance [13]. We therefore have reason to believe that the above model is a good one for local search, that cannot easily be improved.

3 Future work

This is a work in progress and we currently have no experimental data. In the near future we will create a set of randomly-generate benchmarks with different characteristics. Obvious parameters that should be controlled are the number of clients, the number of data pages, and the degree of similarity between client requirements. Later on we hope to obtain realistic problem instances to use as benchmarks. We could also use MSA instances (for example from [8]), though of course these are not derived from data broadcasting applications. We may then enter the problem on CSPLib [6], an online library of problems, to stimulate research into this area. We will then experiment with randomised local search algorithms, to determine good heuristics for these problems, and possibly compare the results with existing algorithms for related problems. In particular, we would like to check for the existence of good polynomial approximation algorithms, perhaps using algorithms for the related MSA problem as a starting point [7, 10, 16]. The type of algorithm we intend to apply is a hybrid of local search and constraint propagation, specifically *bounds propagation* on linear inequalities and disequalities. In previous work on the MSA [13] we used a simpler form of propagation on linear inequalities over binary variables, but we are currently extending this to bounds propagation, integer variables and disequalities (which occur in our new model) [17]. Bounds propagation has the advantage that only domain end-points need to be maintained, thus reducing the space complexity of the runtime memory required by the algorithm.

References

1. S. Acharya, R. Alonso, M. Franklin, S. Zdonik. Broadcast Disks: Data Management for Asymmetric Communication Environments. *ACM SIGMOD International Conference on Management of Data*, ACM Press, 1995, pp. 199–210.
2. S. Acharya, M. Franklin, S. Zdonik. Dissemination-Based Data Delivery Using Broadcast Disks. *Personal Communications* 2(6), December 1995.
3. D. Aksoy, M. J. Franklin. Scheduling for Large-Scale On-Demand Data Broadcasting. *INFOCOM* (2), 1998, pp. 651–659.
4. E. Althaus, A. Caprara, H.-P. Lenhof, K. Reinert. Multiple Sequence Alignment With Arbitrary Gap Costs: Computing an Optimal Solution Using Polyhedral Combinatorics. *Bioinformatics* Suppl 2:S4–S16.
5. M. Franklin, S. Zdonik. A Framework for Scalable Dissemination-Based Systems. *International Conference on Object Oriented Programming Language Systems*, 1997.
6. I. P. Gent, T. Walsh. CSPLIB: A Benchmark Library for Constraints. *Fifth International Conference on Principles and Practice of Constraint Programming, Lecture Notes in Computer Science* vol. 1713, Springer-Verlag, 1999, pp. 480–481.
7. J. D. Kececioglu. Exact and Approximation Algorithms for DNA Sequence Reconstruction. PhD thesis, University of Arizona, 1991.
8. K. Mizuguchi, C. M. Deane, T. L. Blundell, J. P. Overington. HOMSTRAD: A Database of Protein Structure Alignments for Homologous Families. *Protein Science* vol. 7, 1998, pp. 2469–2471.

9. C. Notredame, D. G. Higgins. SAGA: Sequence Alignment by Genetic Algorithm. *Nucleic Acids Research* vol. 2, 1996, pp. 1515–1524.
10. C. Notredame, D. G. Higgins, J. Heringa. T-COFFEE: A Novel Method for Fast and Accurate Multiple Sequence Alignment. *Journal of Molecular Biology* vol. 302, 2000, pp. 205–217.
11. S. Prestwich. Increasing Solution Density by Dominated Relaxation. *Fourth International Workshop on Modelling and Reformulating Constraint Satisfaction Problems, CP'05*, Sitges, Spain, 2005, pp. 1–13.
12. S. D. Prestwich. Negative Effects of Modeling Techniques on Search Performance. *Annals of Operations Research* vol. 118, Kluwer Academic Publishers, 2003, pp. 137–150.
13. S. D. Prestwich, D. Higgins, O. O'Sullivan. A SAT-Based Approach to Multiple Sequence Alignment. Poster, *Ninth International Conference on Principles and Practice of Constraint Programming*, Kinsale, Ireland, 2003, pp. 940–944.
14. S. D. Prestwich, A. Roli. Symmetry Breaking and Local Search Spaces. Second International Conference on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, Lecture Notes in Computer Science vol. 3524, Springer, 2005, pp. 273-287.
15. K. Stathatos, N. Roussopoulos, J. S. Baras. Adaptive Data Broadcast in Hybrid Networks. *Twenty-third International Conference on Very Large Databases*, 1997.
16. J. D. Thompson, D. G. Higgins, T. J. Gibson. CLUSTAL W: Improving the Sensitivity of Progressive Multiple Sequence Alignment Through Sequence Weighting, Position-Specific Gap Penalties and Weight Matrix Choice. *Nucleic Acids Research* vol. 22, 1994, pp. 4673–80.
17. S. Verachi, S. Prestwich. Bounds-Consistent Local Search. *Doctoral program, Eleventh International Conference on Principles and Practice of Constraint Programming*, Sitges, Spain, 2005.