An Improved Approximation Algorithm for the Partial Latin Square Extension Problem

Carla P. Gomes * Rommel G. Regis † David B. Shmoys ‡

Abstract

The problem of completing partial latin squares arises in a number of applications, including conflict-free wavelength routing in wide-area optical networks, statistical designs, and error-correcting codes. A partial latin square is an n by n array such that each cell is either empty or contains exactly one of the colors $1, \ldots, n$, and each color occurs at most once in any row or column. In this paper, we consider the problem of finding an extension of a given partial latin square with the maximum number of colored cells. Approximation algorithms for this problem were introduced by Kumar, Russell, and Sundaram, who gave a 2-approximation algorithm for this problem that is based on a 3-dimensional assignment formulation. We introduce a packing linear programming relaxation for this problem, and show that a natural randomized rounding algorithm yields an $e/(e-1)$-approximation algorithm.

1 Problem statement.

The problem of completing partial latin squares arises in a number of applications, including conflict-free wavelength routing in wide-area optical networks [1], statistical designs, and error-correcting codes [4, 5]. A *partial latin square* (PLS) of order n is an $n \times n$ array such that each cell is either empty or contains exactly one of the "colors" $1, \ldots, n$, and each "color" occurs at most once in any row or column. A *latin square* (LS) of order n is a PLS of order n with no empty cells. A partial latin square is said to be *completable* if one can color its empty cells to obtain a latin square. More generally, one partial latin square, Π_1, is an extension of another, Π_2, if one can color (some of) Π_2's empty cells to obtain Π_1. The problem of deciding if a given PLS is completable is NP-complete [3].

In this paper, we consider the problem of finding an extension of a partial latin square with the maximum number of colored cells. Approximation algorithms for this problem were introduced by Kumar, Russell, and Sundaram [9], who gave two results for this problem: they showed that a natural greedy algorithm is a 3-approximation algorithm (where a ρ-approximation algorithm is a polynomial-time algorithm that finds a feasible solution of objective function value within a factor of ρ of the optimum); they then gave a matching-based 2-approximation algorithm that relies on the so-called assignment linear programming relaxation of the problem. We introduce a packing linear programming relaxation for this problem, and show that a natural randomized rounding algorithm yields a $e/(e-1)$-approximation algorithm.

2 A strong integer programming formulation.

The most natural integer programming formulation of our optimization problem is as follows: introduce a 0-1 variable x_{ijk} for each cell of the array (i, j) and each color k; constrain that for each cell (i, j), at most one color k is assigned; constrain that for each row i and each color k, there is at most one column of that row assigned that color; and symmetrically, for each column j and color k, there is at most one row of that column assigned that color; maximize the number of assignments made. This is the assignment formulation used in [9].

Instead, observe that each color in a PLS corresponds to a (not necessarily perfect) matching of the rows and columns of the PLS. (If a color is not pre-assigned, then it is associated with the empty matching.) For each color $k = 1, \ldots, n$, let $\mathcal{M}_k$ denote the set of all matchings of rows and columns that extend the matching associated with color k. Now for each color k and for each $M \in \mathcal{M}_k$, we introduce a binary variable y_{kM}. This gives us the following IP formulation:

$$\max \sum_{k=1}^{n} \sum_{M \in \mathcal{M}_k} |M| y_{kM}$$

*gomes@cs.cornell.edu. Department of Computer Science, Cornell University, Ithaca, NY 14853. Research partially supported by AFOSR grants F49620-01-1-0076 and F49620-01-1-0361.

†rregis@orie.cornell.edu. School of Operations Research & Industrial Engineering, Cornell University, Ithaca, NY 14853. Research partially supported by AFOSR grants F49620-01-1-0076 and F49620-01-1-0361.

‡shmoys@cs.cornell.edu. School of Operations Research & Industrial Engineering and Department of Computer Science, Cornell University, Ithaca, NY 14853. Research partially supported by NSF grant CCR-9912422.

subject to

$$\sum_{M \in \mathcal{M}_k} y_{kM} = 1, \qquad \text{for each } k = 1, \ldots, n;$$

$$\sum_{k=1}^{n} \sum_{M \in \mathcal{M}_k : (i,j) \in M} y_{kM} \leq 1, \qquad \text{for each } i, j = 1, \ldots, n;$$

$$y_{kM} \in \{0, 1\}, \qquad \text{for each } k = 1, \ldots, n, \ M \in \mathcal{M}_k.$$

It is easy to see that, for any input, the value of the LP relaxation corresponding to the packing formulation is at most the value of the assignment LP relaxation of the partial latin square extension problem; that is, the packing formulation provides a stronger upper bound for this problem. However, note that the size of this formulation is exponential in the order of the PLS. In spite of this difficulty, one can apply the ellipsoid algorithm (via its dual) to solve the packing LP relaxation in polynomial time [8], or apply more specialized LP techniques designed to give fully polynomial approximation schemes for such packing-type linear programs (e.g., [10]), or more likely in practice, to simply apply column generation techniques (e.g., [2]).

3 Applying randomized rounding.

We now describe an improved approximation algorithm. Let y^* be an optimal solution to the LP relaxation of the above packing formulation. For each color k, we interpret the values $\{y^*_{kM}\}_{M \in \mathcal{M}_k}$ as probabilities and select *exactly one* matching for color k according to these probabilities. Note that this procedure could result in matchings that overlap. In that case, we proceed as follows: For each cell where an overlap occurs, we arbitrarily select a color from among the colors involved in the overlap. It is easy to see that the result is an extension of the original PLS.

THEOREM 3.1. *Randomized rounding yields a $e/(e-1)$-approximation algorithm for the partial latin square extension problem.*

Proof. Let y^*_{kM} be an optimal solution to the LP relaxation of the packing formulation, and for each pair (i, j) and for each color k, define $x^*_{ijk} = \sum_{M \in \mathcal{M}_k : (i,j) \in M} y^*_{kM}$, and set $z^*_{ij} = \sum_{k=1}^{n} x^*_{ijk}$. In a manner similar to the approach used by Goemans & Williamson [6] for the maximum satisfiability problem, the probability that cell (i, j) is left uncolored is exactly the product of the values $(1 - x^*_{ijk})$, $k = 1, \ldots, n$, which can be upper bounded by $(1 - z^*_{ij}/n)^n$. By appealing to the concavity of the function $f(x) = 1 - (1 - (x/n))^n$, we see that this implies that cell (i, j) is colored with probability at least $(1 - (1 - 1/n)^n) z^*_{ij}$. Hence, we find that the *expected number* of cells that are colored in the solution found by the algorithm is

$$\sum_{i=1}^{n} \sum_{j=1}^{n} \Pr[\text{cell } (i,j) \text{ is colored}] \geq \frac{e-1}{e} \sum_{i=1}^{n} \sum_{j=1}^{n} z^*_{ij}$$

$$= \frac{e-1}{e} \sum_{k=1}^{n} \sum_{M \in \mathcal{M}_k} |M| y^*_{kM} = \frac{e-1}{e} LP_{opt} \geq \frac{e-1}{e} IP_{opt}.$$

It is straightforward to derandomize the algorithm by the method of conditional expectations. Our analysis of the packing LP formulation is, most likely, not tight. It would be very interesting to show improved bounds based on this formulation, possibly by means of a primal-dual approach. Finally, it is interesting to note that a randomized selection rule (based on the assignment LP relaxation) has recently been incorporated within the inner workings of a constraint programming approach with striking success in speeding up such an enumerative approach to solving the decision version of this problem [7].

References

[1] R. A. Barry and P. A. Hublet. Latin routers, design and implementation. *IEEE/OSA J. Lightwave Tech.*, 11:891–899, 1993.

[2] V. Chvatal. *Linear Programming.* W. H. Freeman Company, San Francisco, 1983.

[3] C. J. Colbourn. The complexity of completing partial latin squares. *Discrete Appl. Math.*, 8:25–30, 1984.

[4] J. Denes and A. D. Keedwell. *Latin Squares and Their Applications.* Academic Press, Inc. NY, 1974.

[5] J. Denes and A. D. Keedwell. *Latin Squares: New Developments in the Theory and Applications. Annals of Discrete Math* 46, 1991.

[6] M. X. Goemans and D. P. Williamson. New 3/4-Approximation Algorithms for MAX SAT. *SIAM J. Discrete Math.*, 7:656–666, 1994.

[7] C. P. Gomes and D. B. Shmoys. The Promise of LP to Boost CSP Techniques for Combinatorial Problems. In *Proceedings of the 4th International Workshop on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems* (CP-AI-OR'02), pages 291-305, 2002.

[8] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms & Combinatorial Optimization* Springer-Verlag, New York, 1993.

[9] S. R. Kumar, A. Russell, and R. Sundaram. Approximating Latin Square Extensions. *Algorithmica*, 24:128-138, 1999.

[10] S. A. Plotkin, D. B. Shmoys, and É. Tardos, Approximation algorithms for fractional packing and covering problems. *Math. Oper. Res.* 20:257–301, 1995.