An Improved Approximation Algorithm for the Partial Latin Square Extension Problem

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

November 25, 2003§

Abstract

Previous work on the Partial Latin Square Extension (PLSE) problem resulted in a 2-approximation algorithm based on the LP relaxation of a 3-dimensional assignment IP formulation. We present an $e/(e-1)$-approximation algorithm that is based on the LP relaxation of a packing IP formulation of the PLSE problem.

Keywords: Latin square, randomized rounding, approximation algorithm.

*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.

‡Corresponding author. 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.

§Acknowledgment. We would like to thank Michel Goemans for suggesting the equivalence between the packing and assignment formulations.

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 [11], 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, which turns out to be equivalent to the assignment linear programming relaxation, and show that a natural randomized rounding algorithm yields an $e/(e-1)$-approximation algorithm.

2 An alternative integer programming formulation.

The most natural integer programming formulation of our optimization problem is as follows. Given a PLS Π of order n, we introduce a 0-1 variable x_{ijk} for each cell of the array (i, j) and each color k. This gives rise to the following IP formulation:

$$\max \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} x_{ijk}$$

subject to

$$\sum_{i=1}^{n} x_{ijk} \leq 1, \quad \text{for each } j, k = 1, \ldots, n$$

$$\sum_{j=1}^{n} x_{ijk} \leq 1, \quad \text{for each } i, k = 1, \ldots, n$$

$$\sum_{k=1}^{n} x_{ijk} \leq 1, \quad \text{for each } i, j = 1, \ldots, n$$

$$x_{ijk} = 1 \quad \text{for each } i, j, k \text{ such that } \Pi_{ij} = k$$

$$x_{ijk} \in \{0, 1\} \quad \text{for each } i, j, k = 1, \ldots, n$$

This is the assignment formulation used in [11].

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:

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

subject to

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

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

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

Note that the size of the packing 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 [10], or apply more specialized LP techniques designed to give fully polynomial approximation schemes for such packing-type linear programs (e.g., [13]), or more likely in practice, to simply apply column generation techniques (e.g., [2]). Here, we recall that the *LP relaxation* of an integer program is simply the linear program obtained by relaxing the integrality constraints on the IP variables. However, it turns out that the packing LP relaxation is actually equivalent to the assignment LP relaxation. We prove this result in the next theorem using some ideas from open shop scheduling [12]. Hence, an easier way to solve the packing LP relaxation is via the assignment LP relaxation.

Theorem 1. The packing LP relaxation and the assignment LP relaxation are equivalent in the following sense: For any feasible solution to the packing LP relaxation, there is a feasible solution to the assignment LP relaxation with the same objective function value, and vice versa.

Proof. Suppose y is a feasible solution to the packing LP relaxation. Define a solution x to the assignment LP relaxation as follows: For each triple (i,j,k), where $1 \leq i,j,k \leq n$, let

$$x_{ijk} = \sum_{M\in\mathcal{M}_k:(i,j)\in M} y_{kM}. \tag{1}$$

It is easy to check that x is feasible for the assignment LP relaxation. Moreover,

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} x_{ijk} = \sum_{k=1}^{n}\sum_{i=1}^{n}\sum_{j=1}^{n} \sum_{M\in\mathcal{M}_k:(i,j)\in M} y_{kM} = \sum_{k=1}^{n}\sum_{M\in\mathcal{M}_k} |M| y_{kM}$$

Conversely, suppose that x is a feasible solution to the assignment LP relaxation. For each color k, let $X^{(k)}$ be the n by n matrix whose (i,j) entry is x_{ijk}. Moreover, let $D^{(k)} = \mathrm{diag}(d_1^{(k)},\ldots,d_n^{(k)})$

and $E^{(k)} = \mathrm{diag}(e_1^{(k)}, \ldots, e_n^{(k)})$ be diagonal matrices such that

$$d_i^{(k)} = 1 - \sum_{j=1}^{n} x_{ijk} \qquad \text{and} \qquad e_j^{(k)} = 1 - \sum_{i=1}^{n} x_{ijk}.$$

Construct the $2n$ by $2n$ matrix

$$U^{(k)} = \begin{bmatrix} X^{(k)} & D^{(k)} \\ E^{(k)} & (X^{(k)})^T \end{bmatrix} \tag{2}$$

Clearly, $U^{(k)}$ is a doubly stochastic matrix, and so by the Birkhoff-von Neumann theorem, $U^{(k)}$ is a convex combination of permutation matrices, i.e.

$$U^{(k)} = \lambda_1^{(k)} U_1^{(k)} + \ldots + \lambda_{m_k}^{(k)} U_{m_k}^{(k)}, \tag{3}$$

for some permutation matrices $U_1^{(k)}, \ldots, U_{m_k}^{(k)}$ and some strictly positive $\lambda_1^{(k)}, \ldots, \lambda_{m_k}^{(k)}$ satisfying $\sum_{h=1}^{m_k} \lambda_h^{(k)} = 1$. Let $Y_1^{(k)}, \ldots, Y_{m_k}^{(k)}$ be the upper left n by n submatrices of $U_1^{(k)}, \ldots, U_{m_k}^{(k)}$, respectively. Note that each $Y_h^{(k)}$ corresponds to a (not necessarily perfect) matching $M_h^{(k)} \in \mathcal{M}_k$. Now define a solution y to the packing LP relaxation as follows: For each color k and for each matching $M \in \mathcal{M}_k$, let

$$y_{kM} = \begin{cases} \lambda_h^{(k)} & \text{if } M = M_h^{(k)} \text{ for some } h = 1, \ldots, m_k \\ 0 & \text{otherwise} \end{cases}$$

It is easy to check that y is a feasible solution to the packing LP relaxation. Moreover, y satisfies (1), and hence, y has the same objective function value as the feasible solution x in the assignment LP relaxation. $\square$

Observe that Theorem 1 can also be stated as follows: The map from the set of feasible solutions of the packing LP relaxation to the set of feasible solutions of the assignment LP relaxation defined by (1) is surjective. However, the decomposition of $U^{(k)}$ in (3) is not necessarily unique, and so, the map is not necessarily a one-to-one correspondence.

Now suppose we are given an optimal solution to the assignment LP relaxation. In order to obtain an optimal solution to the packing LP relaxation, we need an algorithm that can decompose each doubly stochastic matrix $U^{(k)}$, $k = 1, \dots, n$, in (2) into a convex combination of permutation matrices. The results of Gonzalez and Sahni [9] on open shop scheduling show that this can be accomplished in $O(n^4)$ time. We summarize their ideas into the following algorithm.

Input: `An` n `by` n `doubly stochastic matrix` U.

Output: `A decomposition` $U = \sum_{h=1}^{m} \lambda_h U_h$ `such that` $U_1, \dots, U_m$ `are permutation matrices,` $\lambda_h > 0 \ \forall h$ `and` $\sum_{h=1}^{m} \lambda_h = 1$.

(1) Let $\mathcal{R} = \{R_1, \dots, R_n\}$ be the set of rows of U and let $\mathcal{C} = \{C_1, \dots, C_n\}$ be the set of columns of U.

(2) Set $h \longleftarrow 1$.

(3) While $U \neq 0$

- (3a) Construct a bipartite graph $\mathcal{G}_h$ whose vertex set is $\mathcal{R} \cup \mathcal{C}$ and whose edge set is $\mathcal{E}_h = \{(R_i, C_j) : U_{ij} \neq 0\}$.
- (3b) Determine a perfect matching $\mathcal{I}_h$ in $\mathcal{G}_h$. (It can be shown that a perfect matching always exists provided U is not yet the zero matrix.)
- (3c) Let U_h be the permutation matrix that corresponds to $\mathcal{I}_h$ and let

$$\lambda_h = \min_{(R_i, C_j) \in \mathcal{I}_h} U_{ij}.$$

- (3d) For each edge $(R_i, C_j) \in \mathcal{I}_h$, $U_{ij} \longleftarrow U_{ij} - \lambda_h$.
- (3e) Reset $h \longleftarrow h + 1$.

end.

Gonzalez and Sahni [9] used Hall's theorem on bipartite matching to show that a perfect matching will always be found in each iteration provided U is not yet the zero matrix. In step (3b), an initial perfect matching can be found in in $O(n^{5/2}/\log n)$ time by the algorithm of Feder and Motwani [6]. For each $h \geq 2$, a perfect matching $\mathcal{I}_h$ in $\mathcal{G}_h$ can be obtained by simply finding augmenting paths, using as initial matching the perfect matching $\mathcal{I}_{h-1}$ with some edges removed by the previous step (3d). Note that each augmenting path can be found in $O(n^2)$ time by breadth-first search. Moreover, the number of entries of U that become 0 (or the number of edges removed in a perfect matching) in each iteration is also the number of augmenting paths that are needed to find a perfect matching in the next iteration. Since a total of $O(n^2)$ augmenting paths are needed for the algorithm to terminate, it follows that the running time of the above algorithm is $O(n^4)$. Of course, this running time is dominated by the time needed to find an optimal solution to the assignment LP relaxation.

3 Applying randomized rounding.

We now describe an improved approximation algorithm. Let y^* be an optimal solution to the packing LP relaxation. 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 2. Randomized rounding yields an $e/(e-1)$-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 $z^*_{ij} = \sum_{k=1}^{n} x^*_{ijk}$. We proceed in a manner similar to the approach used by Goemans & Williamson [7] for the maximum satisfiability problem. Note that

$$\Pr[\text{cell } (i,j) \text{ is left uncolored}] \;=\; \prod_{k=1}^{n}(1-x^*_{ijk}) \;\le\; \left(\frac{n-\sum_{k=1}^{n} x^*_{ijk}}{n}\right)^n \;=\; \left(1-\frac{z^*_{ij}}{n}\right)^n,$$

where the previous inequality follows from the arithmetic mean – geometric mean inequality. Hence,

$$\Pr[\text{cell } (i,j) \text{ is colored}] \;\ge\; 1-\left(1-\frac{z^*_{ij}}{n}\right)^n.$$

Next, observe that $f(x) = 1-(1-(x/n))^n$ is a concave function with $f(0)=0$ and $f(1) = 1-(1-(1/n))^n$, and so it follows that

$$f(x) \;\ge\; \left[1-\left(1-\frac{1}{n}\right)^n\right] x, \quad \forall x \in [0,1].$$

In particular, we obtain

$$\Pr[\text{cell } (i,j) \text{ is colored}] \;\ge\; \left[1-\left(1-\frac{1}{n}\right)^n\right] z^*_{ij} \;\ge\; \left(1-\frac{1}{e}\right) z^*_{ij}.$$

Hence, we find that the *expected number* of cells that are colored in the solution found by the algorithm is

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\Pr[\text{cell } (i,j) \text{ is colored}] \;\ge\; \frac{e-1}{e}\sum_{i=1}^{n}\sum_{j=1}^{n} z^*_{ij} \;=\; \frac{e-1}{e}\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{\substack{M\in\mathcal{M}_k\\(i,j)\in M}} y^*_{kM}$$

$$= \;\frac{e-1}{e}\sum_{k=1}^{n}\sum_{M\in\mathcal{M}_k}|M|y^*_{kM} \;=\; \frac{e-1}{e}LP_{opt} \;\ge\; \frac{e-1}{e}IP_{opt}. \;\Box \qquad (4)$$

4 Derandomizing the algorithm.

It is straightforward to derandomize the above algorithm by the method of conditional expectations. Let C be the random variable representing the number of colored cells in a feasible solution to PLSE.

Furthermore, for each color k, let Λ_k be the random element representing the matching selected for color k in a feasible solution to PLSE. A derandomized version of the algorithm is as follows:

Input: `An optimal solution` y^* `to the packing LP relaxation.`

Output: `A matching` M_k `for each color` k.

(1) `for` $k = 1$ `to` n

(1a) `define` $\mathcal{M}_k^* = \{M \in \mathcal{M}_k : y_{kM}^* > 0\}$

(1b) `compute` $C_k(M) = E[C|\Lambda_1 = M_1, \ldots, \Lambda_{k-1} = M_{k-1}, \Lambda_k = M]$ `for each` $M \in \mathcal{M}_k^*$

(1c) `select` $M_k = \texttt{argmax}_{M \in \mathcal{M}_k^*} \; C_k(M)$

`end`

Note that we may assume without loss of generality that the matching selected for color k does not overlap with the matchings $M_1, \ldots, M_{k-1}$. Moreover, the objective value of the solution obtained by this algorithm (i.e. the actual number of colored cells, including pre-assignments) is given by $E[C|\Lambda_1 = M_1, \ldots, \Lambda_n = M_n]$.

The conditional expectations in the algorithm are computed as follows:

$$C_k(M) \;=\; \text{Colored}(M_1, \ldots, M_{k-1}, M) \quad + \sum_{\substack{(i,j) \notin \cup_{\ell=1}^{k-1} M_\ell \\ (i,j) \notin M}} \left(1 - \prod_{r=k+1}^{n} (1 - x_{ijr}^*)\right) \tag{5}$$

where the first term is the number of cells colored by $M_1, \ldots, M_{k-1}, M$ (including pre-assignments) and the expression inside the sum is the conditional probability that cell (i,j) is colored by one of the colors $k+1, \ldots, n$ given that $\Lambda_1 = M_1, \ldots, \Lambda_{k-1} = M_{k-1}, \Lambda_k = M$.

Observe that for $k = 1, \ldots, n$, we have

$$E[C|\Lambda_1 = M_1, \ldots, \Lambda_{k-1} = M_{k-1}] \;=\; \sum_{M \in \mathcal{M}_k} \Pr[\Lambda_k = M] \; E[C|\Lambda_1 = M_1, \ldots, \Lambda_{k-1} = M_{k-1}, \Lambda_k = M]$$

$$\leq \sum_{M\in\mathcal{M}_k} \Pr[\Lambda_k = M] \left(\max_{M\in\mathcal{M}_k} E[C|\Lambda_1 = M_1, \ldots, \Lambda_{k-1} = M_{k-1}, \Lambda_k = M]\right)$$

$$= E[C|\Lambda_1 = M_1, \ldots, \Lambda_{k-1} = M_{k-1}, \Lambda_k = M_k]$$

Therefore,

$$E[C|\Lambda_1 = M_1, \ldots, \Lambda_n = M_n] \geq E[C] \geq \frac{e-1}{e} IP_{opt}.$$

5 A stronger objective function for PLSE.

From the perspective of computing an optimal solution, a problem that is equivalent to the one considered here is to find an extension for the given partial latin square that maximizes the number of *additional* cells that are colored. However, the two problems are not equivalent from the perspective of approximation; a ρ-approximation algorithm for this augmentation version is also a ρ-approximation algorithm for the original version of the problem, but not necessarily vice versa. The earlier approximation guarantees by Kumar, Russell, and Sundaram [11] were stated in terms of this stronger augmentation objective function. However, since our analysis merely computes the expectation cell by cell, it is easy to see that our algorithm has precisely the same performance with respect to the augmentation objective function. Hence, our result is an improvement over the earlier known results.

To verify that the analysis holds for the augmentation version, let M_k^o be the pre-assigned partial matching for color k, for each $k = 1, \ldots, n$. The augmentation objective is given by

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

Let B be the set of empty cells in the given partial latin square. Theorem 2 still holds for the augmentation objective since the proof is the same except that equation (4) needs to be changed to the following:

$$\sum_{(i,j)\in B} \Pr[\text{cell } (i,j) \text{ is colored}] \;\geq\; \frac{e-1}{e} \sum_{(i,j)\in B} z^*_{ij} \;=\; \frac{e-1}{e} \sum_{(i,j)\in B} \sum_{k=1}^{n} \sum_{\substack{M\in\mathcal{M}_k \\ (i,j)\in M}} y^*_{kM}$$

$$= \frac{e-1}{e} \sum_{k=1}^{n} \sum_{M\in\mathcal{M}_k} \sum_{(i,j)\in M\cup B} y^*_{kM} = \frac{e-1}{e} \sum_{k=1}^{n} \sum_{M\in\mathcal{M}_k} (|M| - |M^o_k|) y^*_{kM} \;=\; \frac{e-1}{e} LP_{opt} \;\geq\; \frac{e-1}{e} IP_{opt}.$$

For the derandomized version, C now represents the number of additional cells colored. Moreover, in equation (5), the first term is now the number of additional cells colored by $M_1, \ldots, M_{k-1}, M$, and the summation in the second term should be taken over all $(i,j) \in (B \setminus M) \setminus \cup_{\ell=1}^{k-1} M_\ell$.

6 Future work.

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 [8].

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] T. Feder and R. Motwani. Clique partitions, graph compression and speeding-up algorithms. In *Proceedings of the 23rd Annual ACM Symposium on the Theory of Computing*, pages 122–133, 1991.

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

[8] 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.

[9] T. Gonzalez and S. Sahni. Open shop scheduling to minimize finish time. *J. ACM*, 23(4): 665–679, 1976.

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

[11] S. R. Kumar, A. Russell, and R. Sundaram. Approximating Latin square extensions. *Algorithmica*, 24: 128–138, 1999.

[12] E. L. Lawler and J. Labetoulle. On Preemptive scheduling of unrelated parallel processors by linear programming. *J. ACM*, 25(4): 612–619, 1978.

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