Phase-Separated Binary Coupling Matrices via Barrel-Shifting: A Discrete Approach to Approximate Doubly Stochastic Transformations

D. Shirley

07/01/2026

Abstract

We present a discrete, combinatorial method for generating **binary coupling matrices** that approximate doubly stochastic behavior while minimizing simultaneous activations. Source and target distributions are represented as randomized binary matrices. A **cumulative barrel-shifting procedure** maximizes phase separation across rows and columns, and **randomized row/column shuffling** ensures low overlap and reduces fixed alignment patterns. The final output matrix is produced by a **Boolean gating operation** (e.g., AND) between the shifted and shuffled source and target matrices, enforcing per-channel constraints while minimizing concurrency.

To improve efficiency, we introduce a **pointer-and-offset implementation**, which simulates barrel-shifts and permutations without physically moving data. This approach is scalable, hardware-friendly, and allows dynamic updates when source or target distributions change.

Key Properties

The method combines:

- Discrete, interpretable binary representation.
- Phase separation via cumulative barrel-shifting and randomized shuffling.
- Boolean gating to enforce per-channel constraints.
- Pointer-and-offset implementation for memory-efficient, scalable computation.

1 Introduction

Doubly stochastic matrices, which have nonnegative entries with row and column sums equal to one, are widely used in optimal transport, neural network attention, and permutation learning. Traditional approaches rely on continuous normalization (e.g., softmax) or iterative algorithms such as Sinkhorn–Knopp [**?**].

We propose a **discrete, phase-separated construction** that enforces row and column constraints while minimizing concurrency. Unlike stochastic or continuous methods which may produce overlapping activations or require iterative computation, this technique is interpretable, low-overlap, and hardware-friendly.

2 Method

Let N denote the number of source and target channels. Construct two $N \times N$ matrices: S (source) and T (target).

(Example below: $N = 100$)

2.1 Source Matrix Construction

1. **Populate rows:** Each row i contains k_i ones (desired source activation count or normalized proportions) and $N - k_i$ zeros.

Figure 1: Source matrix after initial row population. Black = 1, White = 0.

2. **Row barrel shift:** Row i is cyclically shifted by

$$\text{offset}_i = \sum_{m=0}^{i-1} k_m$$

to maximize phase separation.

Figure 2: Source matrix after cumulative row barrel shift.

3. **Column shuffling:** Columns are randomly permuted to reduce alignment patterns.

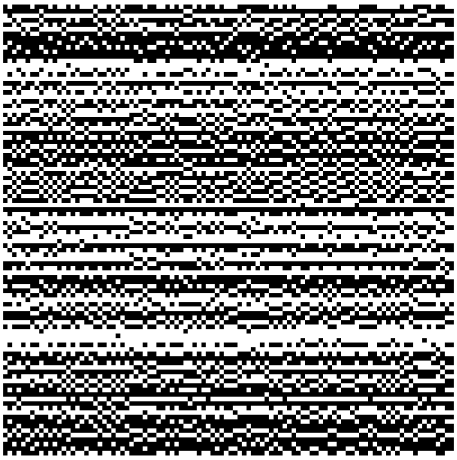

Figure 3: Source matrix after column shuffling.

2.2 Target Matrix Construction

1. **Populate columns:** Each column j contains l_j ones (desired target activation count) and $N - l_j$ zeros.

Figure 4: Target matrix after initial column population. Black = 1, White = 0.

2. **Column barrel shift:** Column j is cyclically shifted downward by

$$\text{offset}_j = \sum_{m=0}^{j-1} l_m$$

to maximize vertical phase separation.

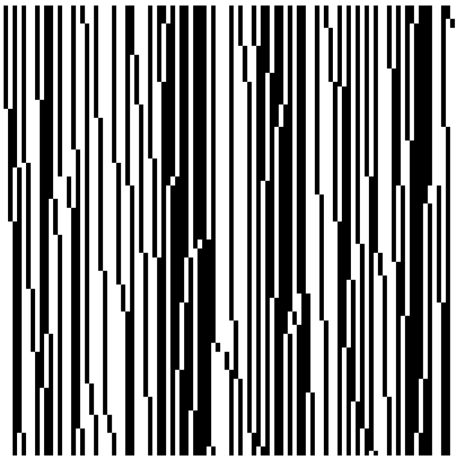

Figure 5: Target matrix after cumulative column barrel shift.

3. **Row shuffling:** Rows are randomly permuted to reduce alignment artifacts.

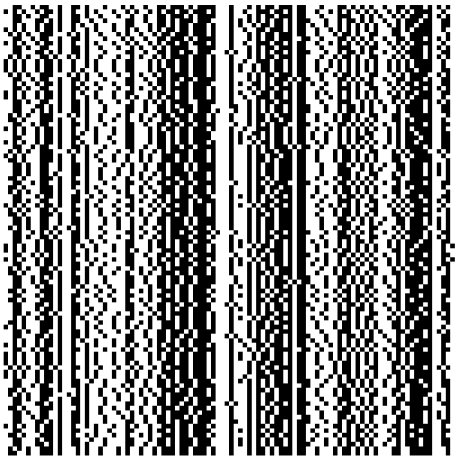

Figure 6: Target matrix after row shuffling.

2.3 Output Matrix

The output matrix O is computed via Boolean gating:

$$O_{i,j} = S_{i,j} \wedge T_{i,j}.$$

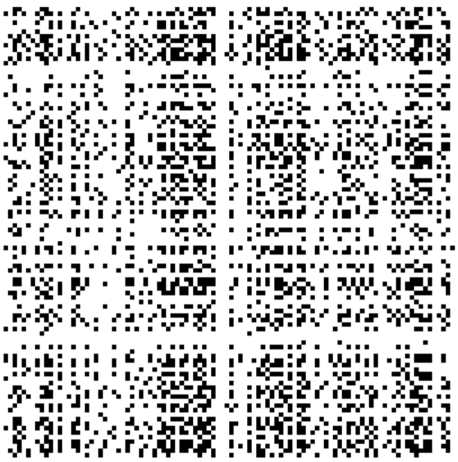

Figure 7: Output matrix after Boolean AND gating between the final source and target matrices.

This produces a binary coupling matrix with row and column sums approximating the source and target activation counts while minimizing overlap.

3 Optimization: Pointer-and-Offset Implementation

While barrel-shifting and random shuffling provide phase separation, physically moving rows, columns, or matrix elements can be computationally expensive for large matrices. To address this, we introduce a pointer-and-offset implementation that simulates the same operations efficiently:

1. **Logical offsets:** Instead of physically rotating rows or columns, maintain *logical offsets* for each row and column. Shifted elements are accessed via modulo indexing:

   $$S[i,j] \mapsto S[i,\,(j+\text{offset}_i) \bmod N]\,, \quad T[i,j] \mapsto T[(i+\text{offset}_j) \bmod N,\,j]\,.$$

   This simulates cumulative barrel-shifting without moving data in memory.

2. **Random permutations via pointers:** Rather than shuffling rows and columns explicitly, maintain permutation arrays:

   $$\text{row\_perm}, \text{col\_perm} \in \{0, \ldots, N-1\}.$$

   Elements of the source matrix are accessed as:

   $$S[\text{row\_perm}[i],\,(\text{col\_perm}[j]+\text{offset}_i) \bmod N]\,.$$

3. **Boolean gating:** The output matrix computed elementwise using a Boolean AND:

   $$O[i,j] = S[\text{row\_perm}[i],\,(\text{col\_perm}[j]+\text{offset}_i) \bmod N] \wedge T[(\text{row\_perm}[i]+\text{offset}_j) \bmod N,\,\text{col\_perm}$$

This approach allows dynamic updates, is vectorizable, and avoids memory-intensive shifts.

4 Properties

- **Discrete representation:** Binary entries are interpretable and hardware-friendly.
- **Phase separation:** Barrel-shifting and shuffling reduce concurrency.
- **Approximate stochasticity:** Row and column sums match desired percentages.
- **Flexible gating:** Supports AND, OR, XOR operations.
- **Efficient computation:** Pointer-and-offset avoids moving large matrices.

5 Potential Applications

The proposed method is applicable wherever discrete resources must be allocated with minimal overlap:

Neural Network Routing / Mixture-of-Experts: Deterministic, low-overlap routing improves efficiency and preserves per-expert activation levels.

Attention Mechanisms: Provides a discrete alternative to softmax attention with controlled overlaps; phase separation prevents collapse.

Energy-Constrained / Neuromorphic Hardware: Limits concurrency and reduces peak power; suitable for spiking neurons, FPGA, and IoT devices.

Combinatorial Scheduling and Resource Allocation: Fair allocation of actuators, channels, or tasks; phase separation reduces conflicts.

Approximate Optimal Transport / Permutation Matrices: Produces low-concurrency discrete transport plans for permutation learning or matching problems.

Temporal or Spiking Signal Representations: Can extend to temporal sequences for low-concurrency spiking patterns, event-based sensing, and low-latency signal processing.

6 Discussion

The method enforces hard per-channel constraints while minimizing concurrency. Unlike softmax or Sinkhorn-based approaches, it is deterministic, interpretable, and scalable. Future work may explore differentiable relaxations and rectangular matrices for neural network integration.

7 Conclusion

This work introduced a discrete barrel-shifting and shuffling construction for phase-separated binary coupling matrices, together with a pointer-and-offset implementation that avoids explicit data movement. The method enforces hard per-channel constraints while significantly reducing concurrent activations, making it well suited to large-scale, energy-constrained, or hardware-oriented settings. Future work may explore differentiable relaxations, rectangular couplings, and integration into neural architectures requiring sparse or structured routing.

8 Related Work and Novelty

The proposed method draws inspiration from several existing research areas, including doubly stochastic matrix constructions, combinatorial scheduling, and sparse neural routing. However, unlike continuous or probabilistic approaches such as softmax attention or Sinkhorn normalization, our method operates entirely in the discrete domain and enforces hard structural constraints.

Its novelty lies in the specific combination of:

- binary matrix representations,
- cumulative barrel-shifting to enforce phase separation,
- randomized row and column permutations to prevent alignment artifacts,
- Boolean-gated intersections to control concurrency, and
- a pointer-and-offset implementation enabling scalable, memory-efficient computation.

To our knowledge, this combination has not previously appeared in literature and provides a deterministic, low-overlap alternative to continuous coupling mechanisms.