Radbound University

# On the Difficulty of Constructing Permutation Codes by Evolutionary Algorithms

**Luca Mariot**, Stjepan Picek, Domagoj Jakobovic, Marko Djurasevic, Alberto Leporati

luca.mariot@ru.nl

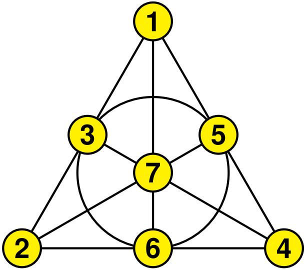
EvoAPPS 2022 – Madrid, April 21, 2022

# Combinatorial Designs

- A collection $\mathcal{A}$ of **blocks** of a finite set $X$ satisfying particular **balancedness** properties [S04]

- Example: the **Fano Plane**

$$X = \{1,2,3,4,5,6,7\}$$
$$\mathcal{A} = \{123,145,167,246,$$
$$257,347,356\}$$




- Interesting source of optimization problems for Evolutionary Algorithms (EA), to play with different representations [MMT20, KPMJL18, MPJL18, MPJL17]

# Permutation Codes

## Definition

A *Permutation Code* (or *Permutation Array*, PA) of length $n$ and distance $d$ is an $m \times n$ array such that:

- each row is a permutation of $[n] = \{1, \cdots, n\}$
- any two rows are at Hamming distance at least $d$

Example: a $PA(6,5)$

| | | | | | |
|---|---|---|---|---|---|
| 125634 | 142365 | 164523 | 236514 | 251346 | 316425 |
| 354612 | 362154 | 413562 | 426351 | 435126 | 461235 |
| 512643 | 534261 | 546132 | 623145 | 631452 | 645213 |

In other words, any two permutations in the code must *disagree* in at least $d$ positions

# Largest Possible Codes

From a coding-theoretic point of view, the main question for PA is:

**What is the largest possible size $M(n,d)$ for a $PA(n,d)$?**

| | | | |
|---|---|---|---|
| 1 | 3 | 4 | 2 |
| 4 | 2 | 1 | 3 |
| 2 | 4 | 3 | 1 |
| 3 | 1 | 2 | 4 |

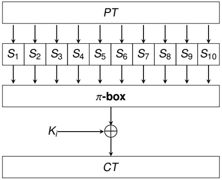
- Particular case: $n = d$, then $M(n,d) = n$ (**Latin square**)
- In general, $M(n,d)$ is bounded by the *Gilbert-Varshamov* and *Sphere-packing* bounds:

$$\frac{n!}{\sum_{k=0}^{d-1}\binom{n}{k}D_k} \leq M(n,d) \leq \frac{n!}{\sum_{k=0}^{\lfloor\frac{d-1}{2}\rfloor}\binom{n}{k}D_k} \quad ,$$

# Applications and Constructions

PA have several applications, such as:

- Error-correcting codes in **powerline communications** [H00]
- Diffusion layers in **block ciphers** [DCL00]
- Rank modulation in **flash memories** [JMSB08]




Available construction methods:

- *Algebraic constructions* (permutation polynomials, ...)
- *Heuristics* (branch and bound, iterative clique search, ...)

... What about Evolutionary Algorithms?

# Evolutionary Construction All-at-Once

- **Straightforward** approach: each individual directly represents an $m \times n$ array of permutations
- Permutation-based GA operators (CX/PMX crossover, swap mutation) are applied in a *row-wise* fashion
- **Drawback:** the search space is *really* huge! $\mathcal{S}_{n,m} = \binom{n!}{m}$

| $d \backslash n$ | | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|
| $n-2$ | $\mathcal{S}_{n,m}$ | $3.07 \cdot 10^{140}$ | $2.31 \cdot 10^{277}$ | $1.81 \cdot 10^{843}$ | $1.20 \cdot 10^{1\,658}$ | $3.83 \cdot 10^{2\,978}$ |
| | $M(n,d)$ | 120 | 77 | 336 | 504 | 720 |
| $n-1$ | $\mathcal{S}_{n,m}$ | $3.41 \cdot 10^{36}$ | $1.91 \cdot 10^{106}$ | $1.10 \cdot 10^{184}$ | $3.26 \cdot 10^{297}$ | $1.61 \cdot 10^{453}$ |
| | $M(n,d)$ | 18 | 42 | 56 | 72 | 49 |
| $n$ | $\mathcal{S}_{n,m}$ | $1.89 \cdot 10^{15}$ | $1.63 \cdot 10^{23}$ | $1.73 \cdot 10^{34}$ | $3.01 \cdot 10^{45}$ | $1.10 \cdot 10^{60}$ |
| | $M(n,d)$ | 6 | 7 | 8 | 9 | 10 |

# Iterative Evolutionary Construction

**Idea:** incrementally optimize a single permutation at a time

1. Start from a random permutation of $[n]$ and add it to the PA
2. Apply a permutation GA to search for a new permutation
3. When a permutation at distance $\geq d$ from all previous ones is found, add it to the PA
4. If the fitness budget has not expired, go back to 2. Otherwise, return the PA constructed so far

**Advantage:** search space size is much smaller

**Disadvantage:** greedy approach

# Fitness Functions (1/2)

We experimented with four fitness functions:

- **Fitness 1:** *maximize* sum of distances, only if they are $\geq d$

$$fit_1(p) = \sum_{p_i \in P} \delta_i \cdot d_H(p, p_i), \text{ where } \delta_i = \begin{cases} 1, & \text{if } d_H(p, p_i) \geq d, \\ 0, & \text{otherwise} \end{cases}$$

- **Fitness 2:** *maximize* sum of discounted distances

$$fit_2(p) = \sum_{p_i \in P} \delta'_i \cdot d_H(p, p_i), \text{ where } \delta'_i = \begin{cases} 1, & \text{if } d_H(p, p_i) \geq d, \\ 2^{d_H(p, p_i) - d}, & \text{otherwise} \end{cases}$$

# Fitness Functions (2/2)

We experimented with four fitness functions:

- **Fitness 3:** *maximize* minimum distance:

$$fit_3(p) = \min_{p_i \in P} \{d_H(p, p_i)\}$$

- **Fitness 4:** *minimize* number of pairs at Hamming distance $< d$

$$fit_4(p) = \left| \{(p, p_i) : p_i \in P,\ d_H(p, p_i) < d\} \right|$$

Each fitness considers only the pairs formed by the *current* permutation and all the previous ones in the PA

# Experimental Setting (1/2)

**Random reset:**

- If GA does not find a good permutation within a certain fitness budget, some previous permutations are randomly deleted
- Removed permutations decrease through *cooling policy*
- Comparison with *random search* (RS) as inner algorithm

**Combinations tested:**

- **EA1:** EA without random reset
- **EA2:** EA with random reset
- **RS1:** RS without random reset
- **RS2:** RS with random reset

# Experimental settings (2/2)

**Common Parameters**:

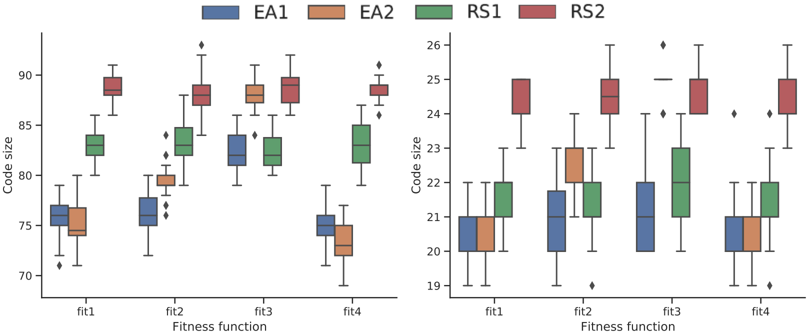
- Problem instances: $6 \leq n \leq 10$, $d = n, n-1, n-2$
- Termination condition: $10^7$ fitness evaluations
- Each experiment is repeated over 30 independent runs

**GA Parameters**:

- Selection operator: steady-state with 3-tournament operator
- Population size: 1000 individuals
- Mutation probabilities: $p_m = 0.3$

| $d \backslash n$ | | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|
| $n-2$ | $\mathcal{S}_{n,m}$ | $3.07 \cdot 10^{140}$ | $2.31 \cdot 10^{277}$ | $1.81 \cdot 10^{843}$ | $1.20 \cdot 10^{1\,658}$ | $3.83 \cdot 10^{2\,978}$ |
| | $M(n,d)$ | 120 | 77 | 336 | 504 | 720 |
| $n-1$ | $\mathcal{S}_{n,m}$ | $3.41 \cdot 10^{36}$ | $1.91 \cdot 10^{106}$ | $1.10 \cdot 10^{184}$ | $3.26 \cdot 10^{297}$ | $1.61 \cdot 10^{453}$ |
| | $M(n,d)$ | 18 | 42 | 56 | 72 | 49 |
| $n$ | $\mathcal{S}_{n,m}$ | $1.89 \cdot 10^{15}$ | $1.63 \cdot 10^{23}$ | $1.73 \cdot 10^{34}$ | $3.01 \cdot 10^{45}$ | $1.10 \cdot 10^{60}$ |
| | $M(n,d)$ | 6 | 7 | 8 | 9 | 10 |

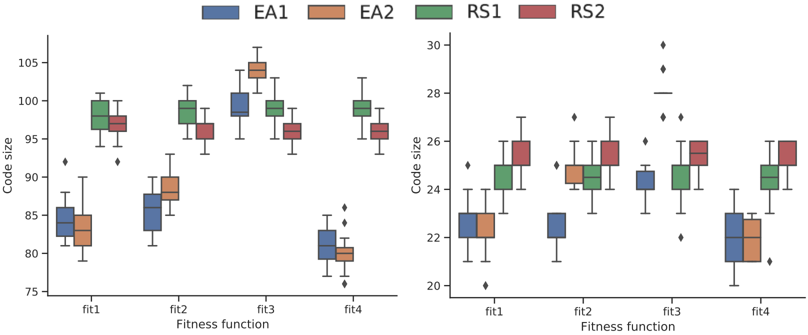
# Results for $n = 9$




(a) $9, 7$ (b) $9, 8$

Figure: Largest code size achieved by all methods across the problem instances with $n = 9$ and $d = n - 2, n - 1$.

# Results for $n = 10$




(a) $10,8$

(b) $10,9$

Figure: Largest code size achieved by all methods across the problem instances with $n = 10$ and $d = n-2, n-1$.

# Findings & Discussion

**Main findings:**

- For $n = d$, EA and RS *always reach* the maximum size $m = n$
- Both EA and RS scale badly as $n$ grows (maximum sizes obtained far from known upper bounds on $M(n,d)$
- EA and RS behave similarly, except on $n = 10$
- Surprisingly, $fit_3$ is the best performing fitness

**Possible explanations:**

- Relatively small size of the *local* search space
- Exceptional difficulty for EA to find a good solution might be related to the graph-theoretic interpretation of the problem

# Conclusions and Future Works

**Conclusions:**

- We applied permutation-based GA to construct permutation codes in an incremental way
- Results show that this problem is exceptionally difficult for EA, and in most instance it behaves as RS

**Future work:**

- Experiment with larger instances
- Exploit the MAX-CLIQUE interpretation of the problem [MBS16]

# References


[DCL00] De la Torre, D., Colbourn, C., Ling, A.: An application of permutation arrays to block ciphers. Congressus Numerantium pp. 5–8 (2000)

[H00] Han Vinck, A.: Coded modulation for powerline communications. AEU Int. J. Eletron. Commun. 54(1), 45–49 (2000)

[JMSB08] Jiang, A., Mateescu, R., Schwartz, M., Bruck, J.: Rank modulation for flash memories. In: Kschischang, F.R., Yang, E. (eds.): Proceedings of ISIT 2008, pp. 1731–1735 (2008)

[KPMJL18] Knezevic, K., Picek, S., Mariot, L., Jakobovic, D., Leporati, A.: The design of (almost) disjunct matrices by evolutionary algorithms. In: Fagan, D., Mart'in-Vide C., O'Neill, M., Vega-Rodríguez, M.A. (eds.): TPNC 2018. LNCS vol. 11324, pp. 152–163. Springer (2018)

[MMT20] Manzoni, L., Mariot, L., Tuba, E.: Balanced crossover operators in Genetic Algorithms. Swarm Evol. Comput. 54: 100646 (2020)

[M18] Mariot, L., Picek, S., Jakobovic, D., Leporati, A.: Evolutionary Search of Binary Orthogonal Arrays. In: Auger, A., Fonseca, C.M., Lourenço, N., Machado, P., Paquete, L., Whitley, D. (eds.): PPSN 2018 (I). LNCS vol. 11101, pp. 121–133. Springer (2018)

[MPJL17] Mariot, L., Picek, S., Jakobovic, D., Leporati, A.: Evolutionary Algorithms for the Design of Orthogonal Latin Squares based on Cellular Automata. In: Proceedings of GECCO'17, pp. 306–313 (2017)

[MBS16] Montemanni, R., Barta, J., Smith, D.H.: Graph colouring and branch and bound approaches for permutation code algorithms. In: Rocha, À., Correia, A.M.R., Adeli, H., Reis, L.P., Teixeira, M.M. (eds.): WorldCIST'16. AISC, vol. 444, pp. 223–232 (2016)

[S04] Stinson, D. R.: Combinatorial designs. Springer (2004)