

Application of genetic algorithms in the design of fuel cycles for a PWR reactor

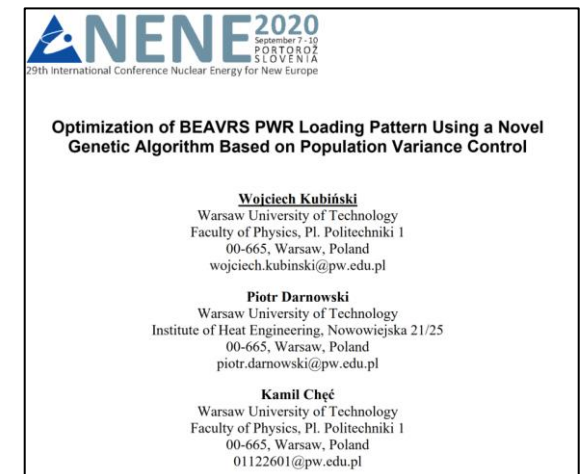
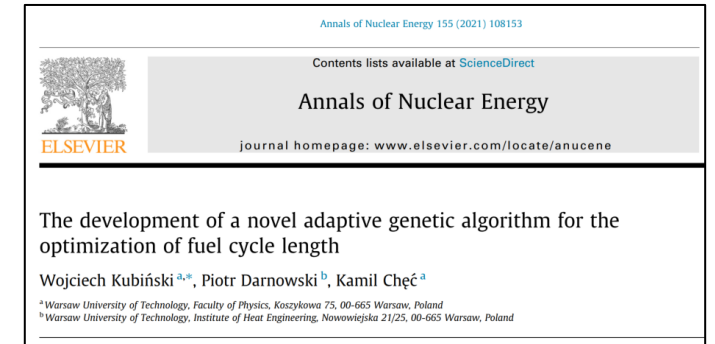
Piotr Darnowski (WUT & NCBJ)

Project leader: Wojciech Kubiński (WUT & NCBJ)

Students/Alumni: Kamil Chęć (WUT), Krzysztof Palmi (WUT), Wojciech Żurkowski (WUT), Piotr Sawicki (WUT), Patryk Bojarski (WUT),

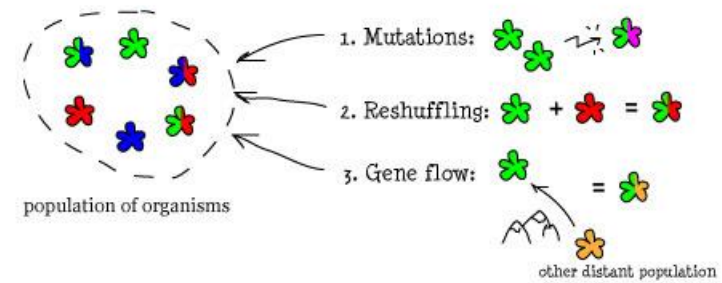
Introduction

- This presentation: activities mainly related to the genetic algorithms and their applications for PWR nuclear reactors.
- Non-funded project. Also other activities with GAs, ANN and PSO.
- Current project dedicated to application of genetic algorithms, neural networks and machine learning including hybrid approach in nuclear reactor physics.
- Recently IDUB project awarded at Faculty of Physics WUT

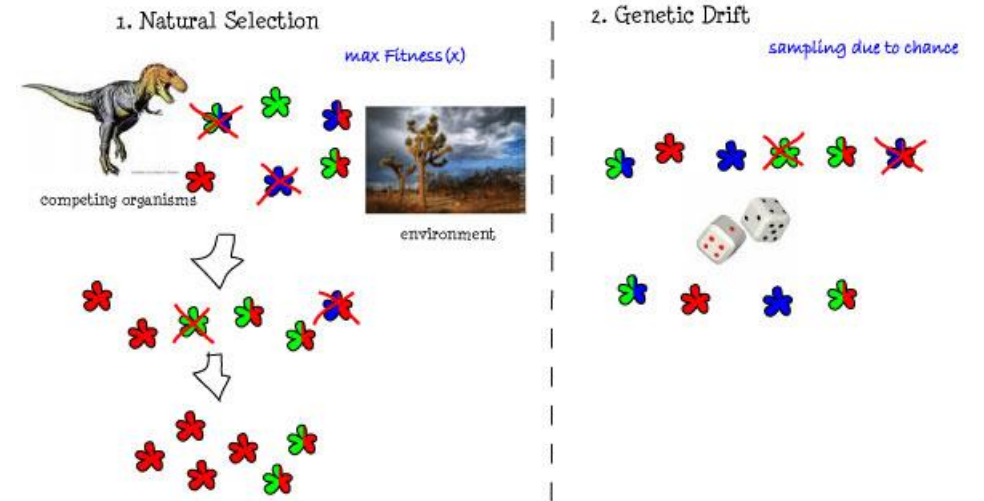


Biological Evolution

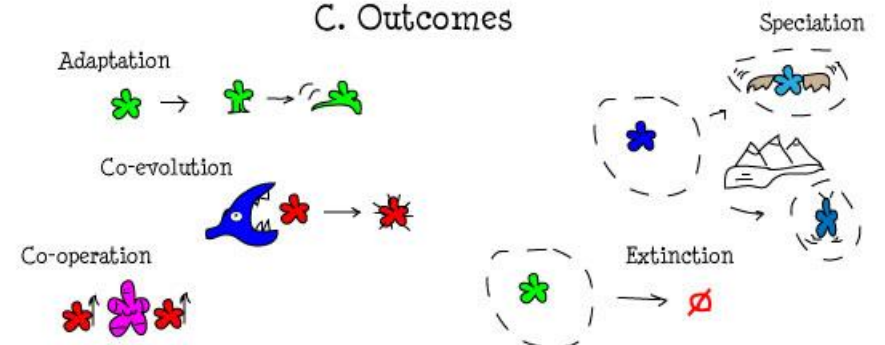
A. Mechanisms that increase genetic variation



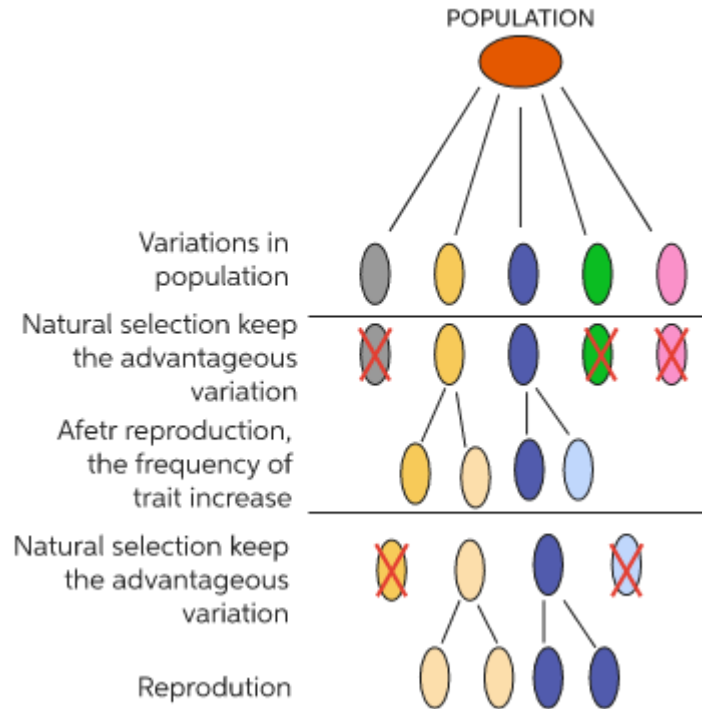
B. Mechanisms that decrease genetic variation



C. Outcomes

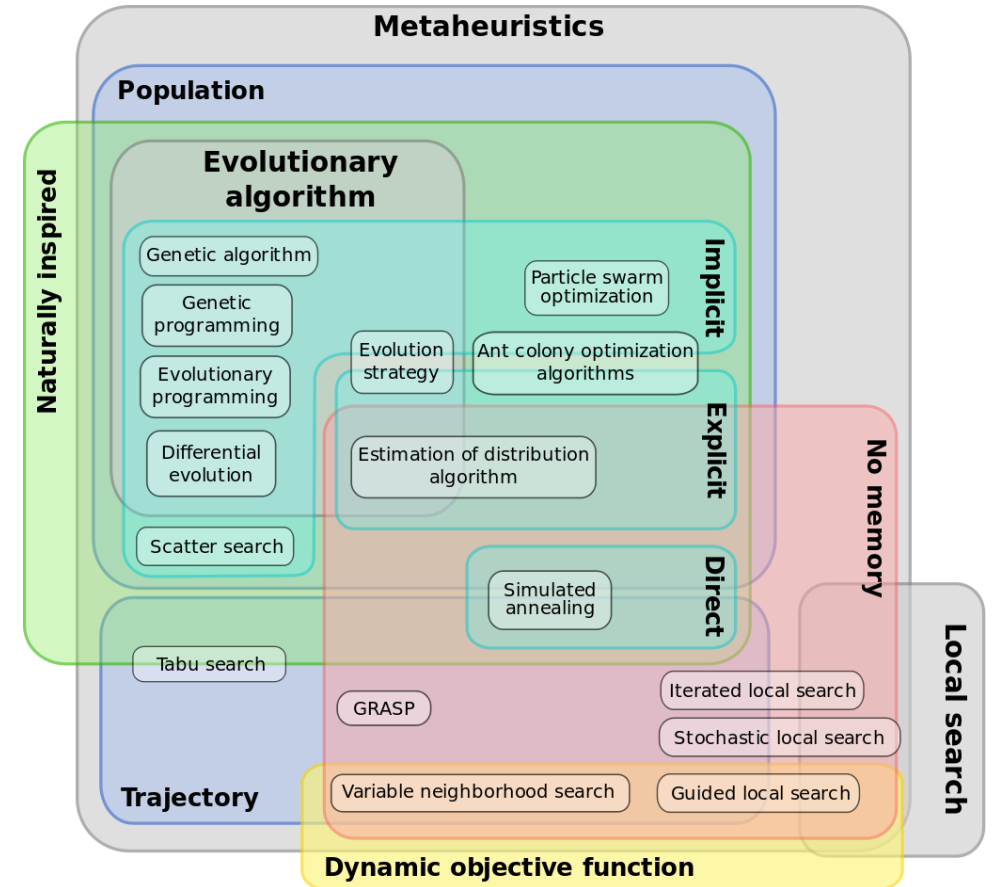


Darwin's natural selection



What is genetic algorithm?

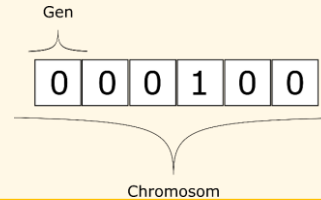
- “A genetic algorithm (GA) is a method for solving both constrained and unconstrained optimization problems based on a natural selection process that mimics biological evolution” [1].
- Genetic Algorithm (GA) is a stochastic method; metaheuristic approach based on natural selection and genetics, a type of Evolutionary Algorithm.
- One of popular biologically inspired methods.
- In practice it allows solving complex multi-dimensional global optimization or search problems.



Typical Genetic Algorithm

Typical GA includes:

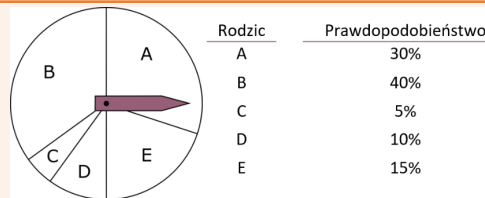
Genetic representation



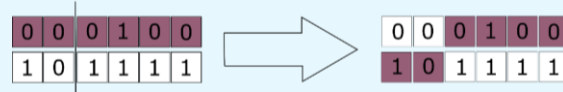
Fitness function

$$FF = f(\text{chromosome}, \text{environment}, \dots)$$

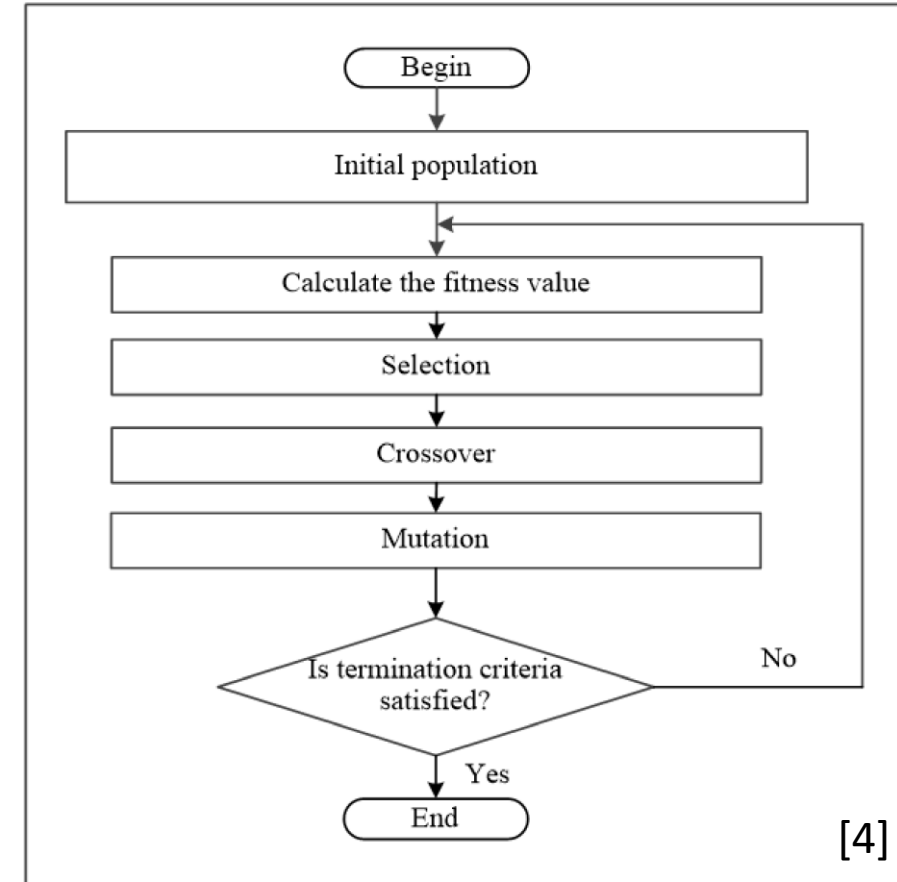
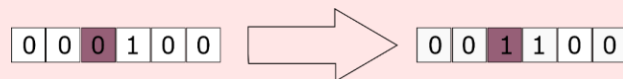
Selection



Crossover

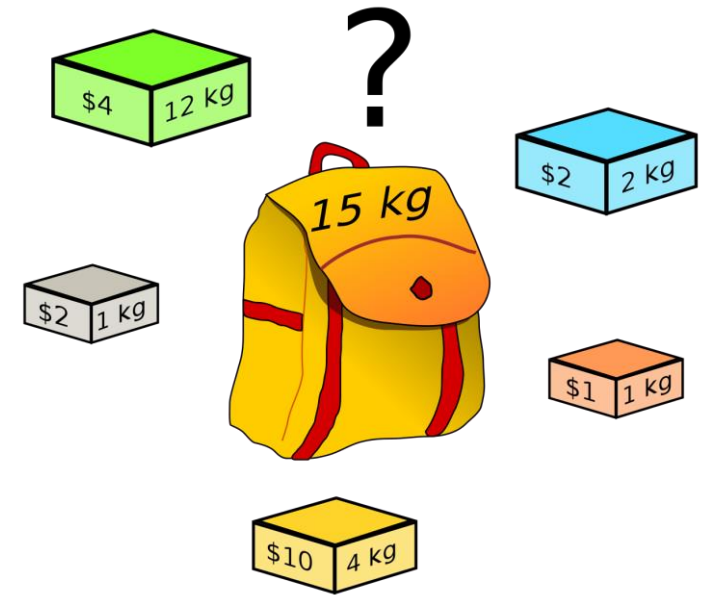


Mutation



Example “typical” problem

- Popular “simple” problem which can be solved with GAs.
- Backpack (Knapsack) problem.
- Combinatorial optimization problem.
- Backpack with limited capacity.
- Set of items, with given weight and value.
- Collect items with highest value within capacity limit.
- For small number of items, easy to solve even with brute force.
- For large number of items, it can be exponentially complex.



https://en.wikipedia.org/wiki/Knapsack_problem

Example: Chromosome

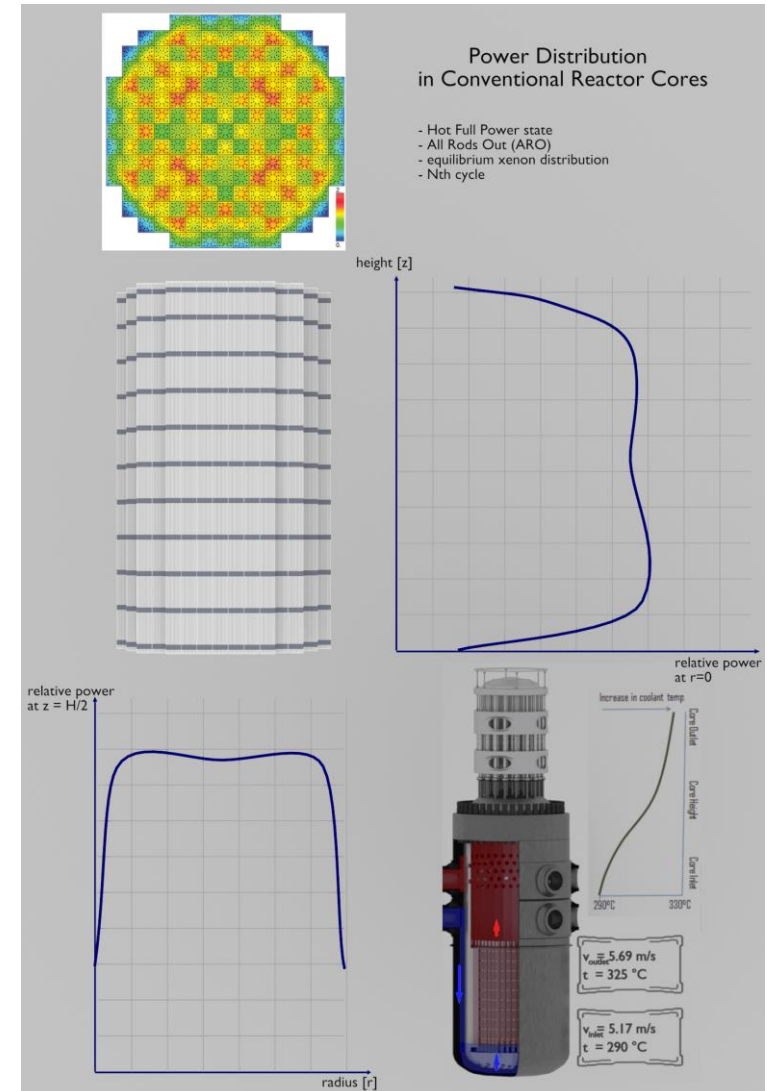
Item #	1	2	3	4	5	...
Value	1	0	1	1	0	...

Example: Fitness Function

$$FF = \begin{cases} \sum value & ,if \sum mass \leq max \\ 0.0 & ,if \sum mass > max \end{cases}$$

Nuclear reactor „backpack” problem

- Core loading pattern design – how to arrange reactor core?
- Typical PWR - ~200 FAs, ~10 types of assemblies, gigantic space of possible solutions. Assuming 20 FAs per type ($200!/10/20! \sim 1e355$ solutions).
- Main purpose - improve economics (for power reactors)
- Operational constraints – safety limits or technical limits
- Core design task or fuel cycle design – routine task since many decades. In each reactor with fuel elements.
- Typically: heuristic (trial and error, educated guess, rule of thumb), experience, special algorithms, brute force, etc.
- Optimization methods are used thanks to availability of computational power - e.x. GAs



Problem formulation and solution approach

- Main task: solve core loading pattern problem for a realistic power reactor.
- Add tasks: develop new methods, tools and insight during the process
- Constraints: safety limits, operational limits
- Realistic constraints but keep it “academic”
- Design core loading pattern for a cycle (multiple cycles – in the future)
- Use PWR Reactor BEAVRS MIT benchmark core as a reference
- Use PARCS core simulator and software written in Python

Studied Reactor Core

- BEAVRS MIT PWR Benchmark
- Core specification and benchmark data
- based on real reactor core
- Westinghouse 4-loop 3411MWth
- 1st fuel cycle with enrichments 1.6%, 2.4%, 3.1%
- FAs models developed in SCALE 6.1.2 (NEWT+TRITON)
- HZP core physics and HFP cycle simulations in PARCS 3.2

Details in:

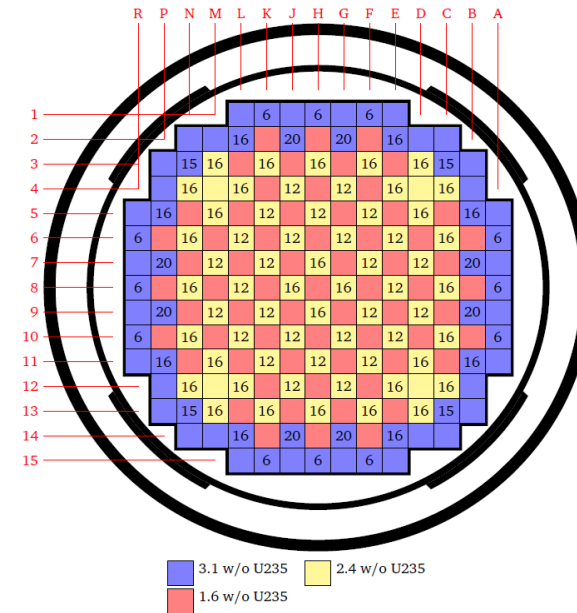
NUKLEONIKA 2019;64(3):87–96
doi: 10.2478/nuka-2019-0011

ORIGINAL PAPER

 sciendo © 2019 P. Darnowski & M. Pawluczyk. This is an open access article distributed under the Creative Commons Attribution-NonCommercial-NoDerivatives 3.0 License (CC BY-NC-ND 3.0).

**Analysis of the BEAVRS PWR benchmark
using SCALE and PARCS***

Piotr Darnowski,
Michał Pawluczyk



assembly type	enrichment	BA rods
1	1.6 %	0
2	2.4 %	0
3	2.4 %	12
4	2.4 %	16
5	3.1 %	0
6	3.1 %	6
7	3.1 %	15
8	3.1 %	16
9	3.1 %	20

TRITON PARCS Two-step modelling

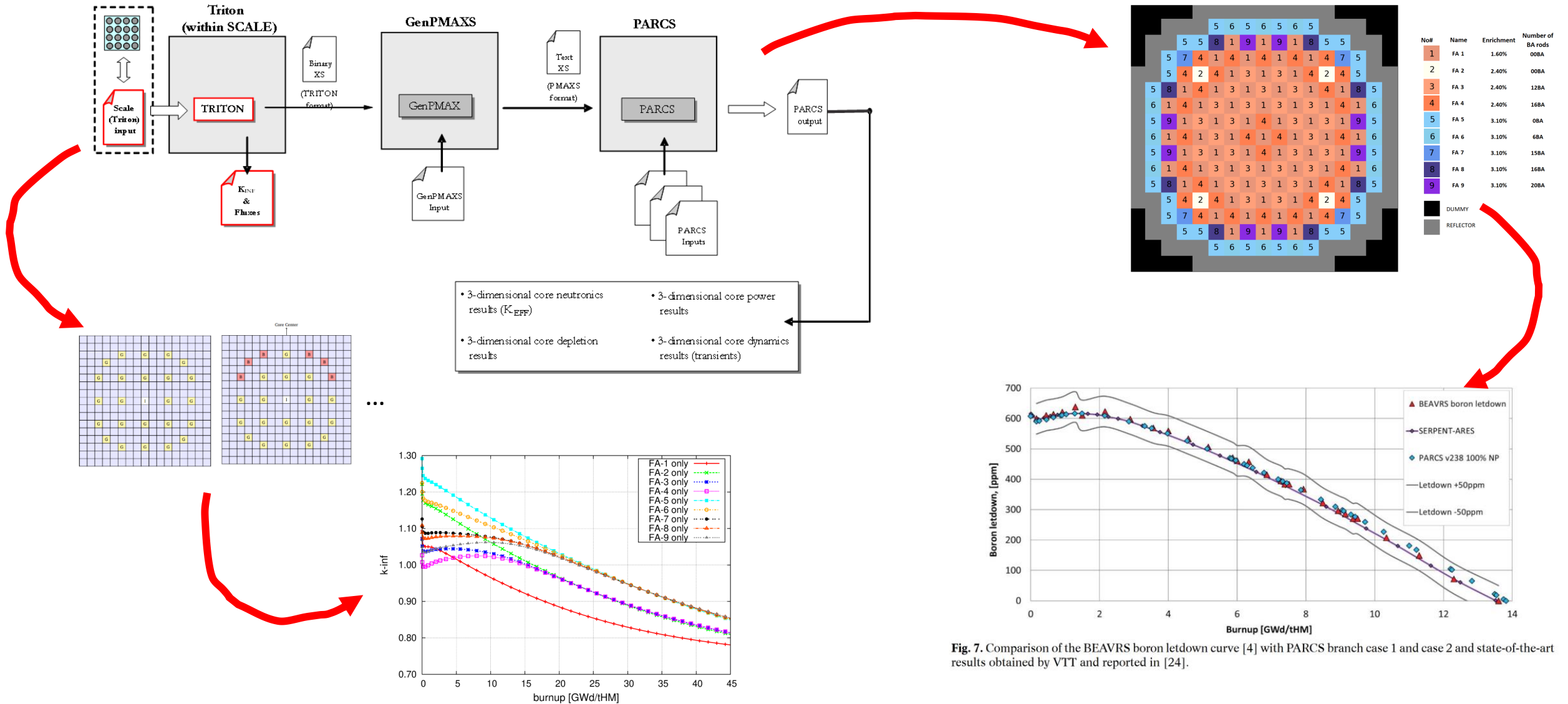
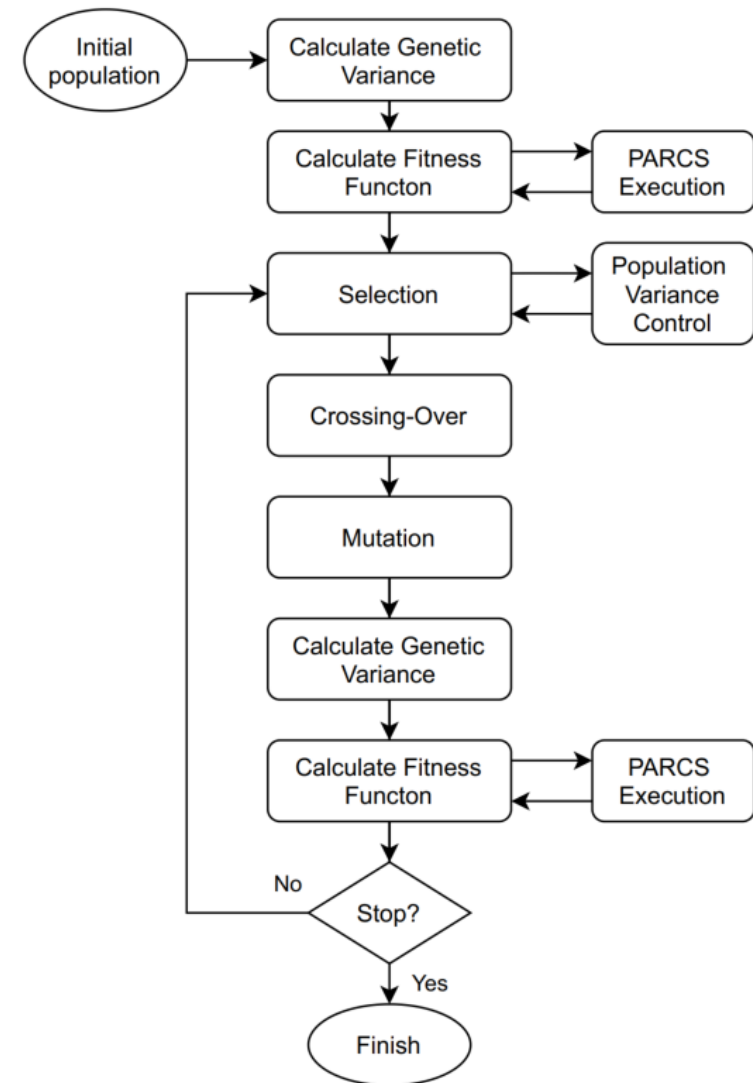


Fig. 7. Comparison of the BEAVRS boron letdown curve [4] with PARCS branch case 1 and case 2 and state-of-the-art results obtained by VTT and reported in [24].

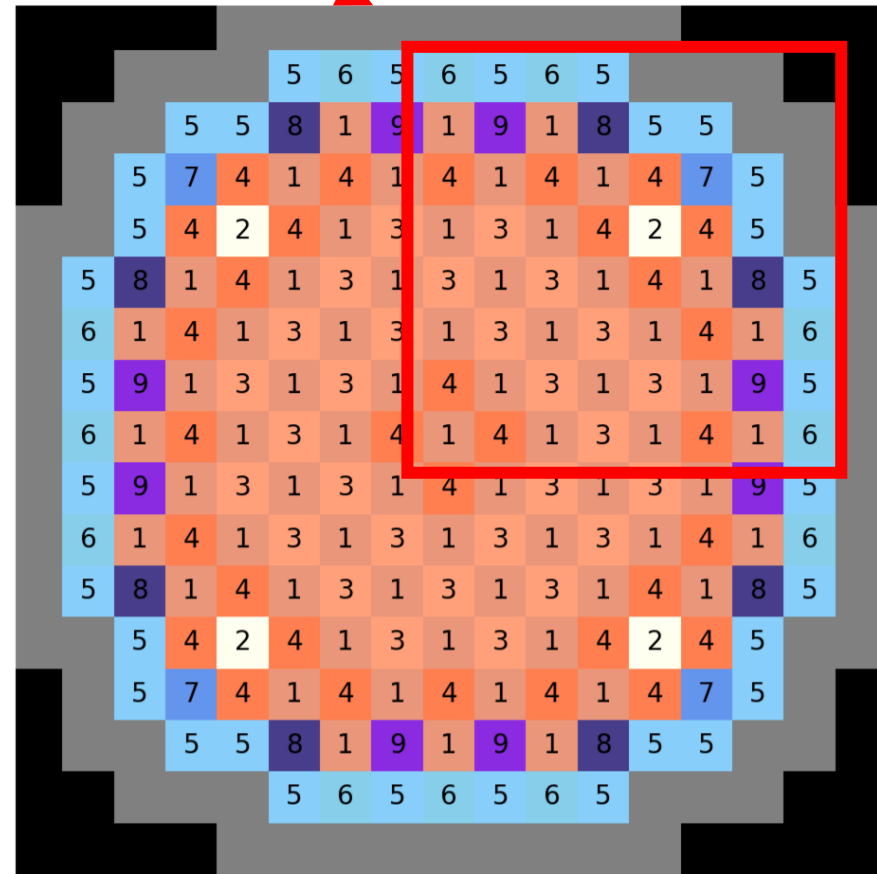
Computational framework

- PARCS coupled with in-house wrapper code
- Code with GA implementation in Python
- Typical GAs operators: crossing-over, mutation, etc.
- Additional population variance control techniques



Genetic representation

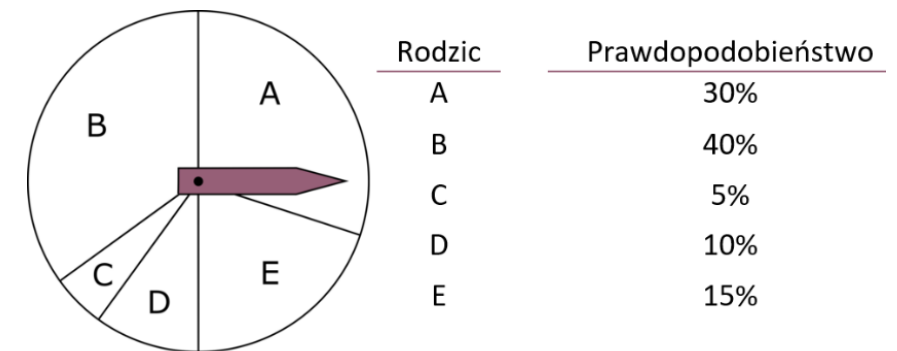
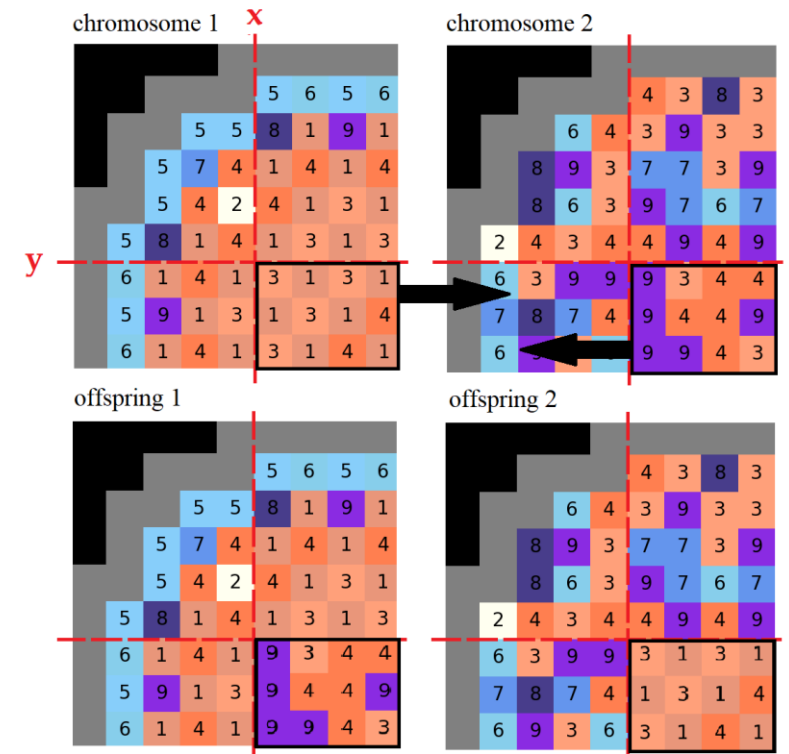
- Single chromosome
 - $\frac{1}{4}$ core symmetry
 - 8x8 matrix
 - reduced space of solutions
- Single gene – fuel assembly
 - 1-out-of-9 types



No#	Name	Enrichment	Number of BA rods
1	FA 1	1.60%	00BA
2	FA 2	2.40%	00BA
3	FA 3	2.40%	12BA
4	FA 4	2.40%	16BA
5	FA 5	3.10%	0BA
6	FA 6	3.10%	6BA
7	FA 7	3.10%	15BA
8	FA 8	3.10%	16BA
9	FA 9	3.10%	20BA
	DUMMY		
	REFLECTOR		

Genetic operators

- Selection – parent selection; Roulette method applied
- Crossover – exchange genetic data between selected parents; randomly chosen sub-matrix of 2D chromosome
- Mutation – constant value over time and population; random gene (fuel assembly)
- Fitness functions (FF) - problem specific; practically hardest part is to define them properly



Example results

- BEAVRS cycle #1 fuel only - 9 types of FAs
- Single fuel cycle with fresh fuel (first core)
- Four test problems are presented:
 - Test 1: no constraints – find core with longest cycle.
 - Test 2: maximize cycle length and limited excess reactivity ($k\text{-eff} < \text{limit}$).
 - Test 3: maximize cycle length and keep fissile materials mass constant (value, e.g. equal to BEAVRS mass).
 - Test 4: maximize cycle length and minimum PFF (power peaking factor).
- General FF:

$$FF = \left(\frac{d}{\alpha} \right)^\beta$$

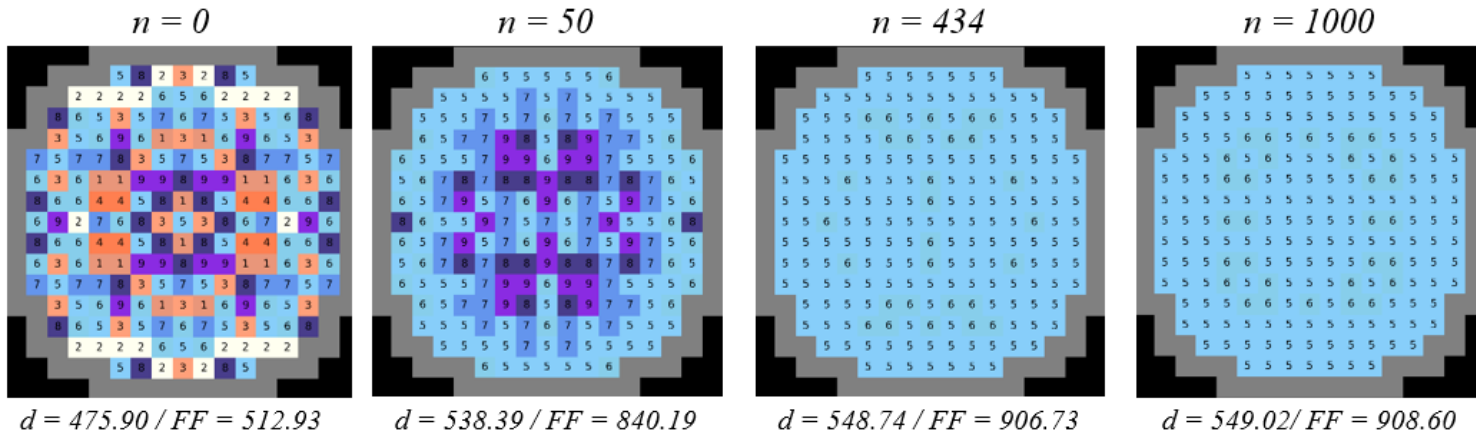
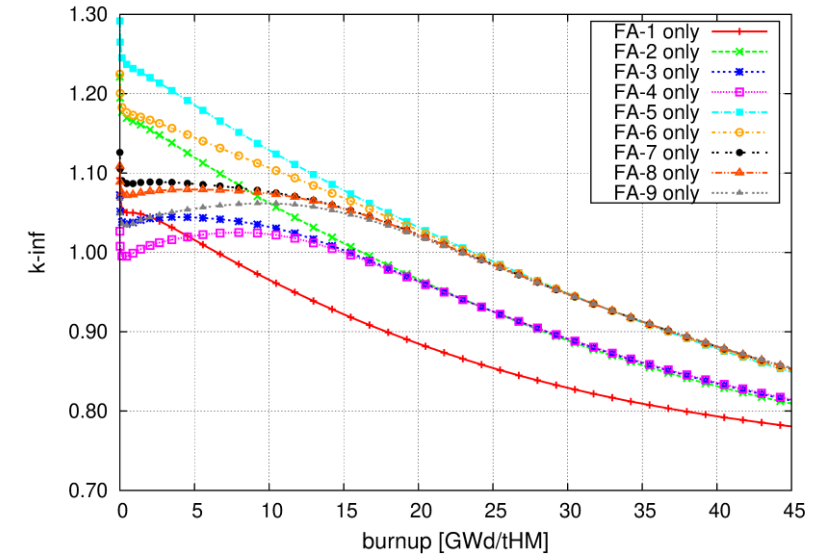
No#	Name	Enrichment	Number of BA rods
1	FA 1	1.60%	00BA
2	FA 2	2.40%	00BA
3	FA 3	2.40%	12BA
4	FA 4	2.40%	16BA
5	FA 5	3.10%	08BA
6	FA 6	3.10%	6BA
7	FA 7	3.10%	15BA
8	FA 8	3.10%	16BA
9	FA 9	3.10%	20BA

Test 1 – max cycle

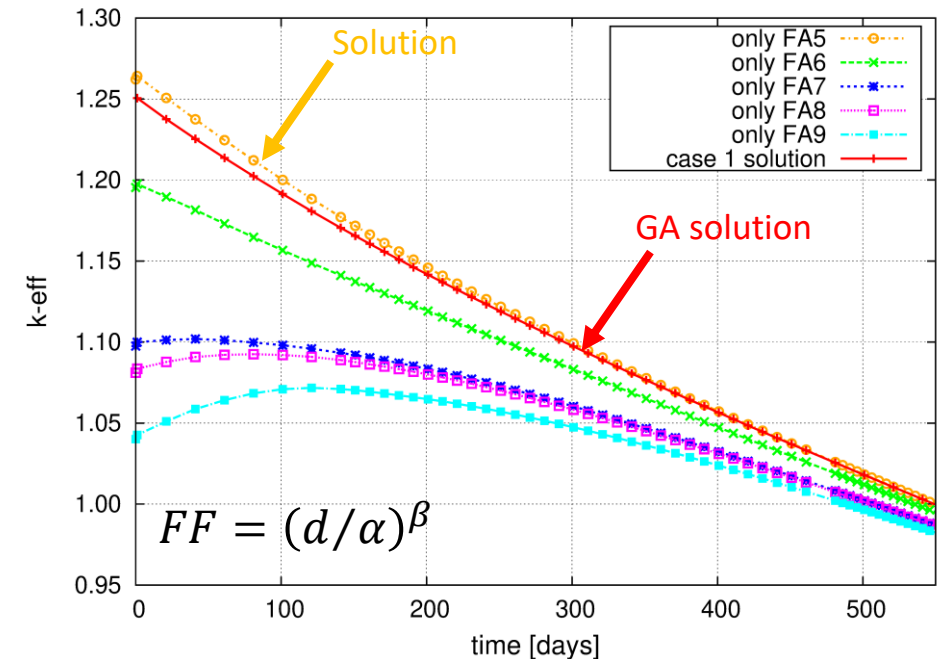
- No constraints
- Problem with intuitive solution (“validation”).
- Core with single assembly type, with highest enrichment 3.1 wt% (FA5) and no BAs.
- 1000 generations, 100 specimen per generation
- ~0.06% difference to solution, 549.02 vs 549.36 days

No#	Name	Enrichment	Number of BA rods
1	FA 1	1.60%	00BA
2	FA 2	2.40%	00BA
3	FA 3	2.40%	12BA
4	FA 4	2.40%	16BA
5	FA 5	3.10%	0BA
6	FA 6	3.10%	6BA
7	FA 7	3.10%	15BA
8	FA 8	3.10%	16BA
9	FA 9	3.10%	20BA

TRITON results for each assembly



Results for cores with one FA type - PARCS



Test 1 – FF evolution

- Studying outcomes we investigate behavior of FF in subsequent generations.
- Other effects, e.g. mutation impact, stochastic bias, population size, etc.

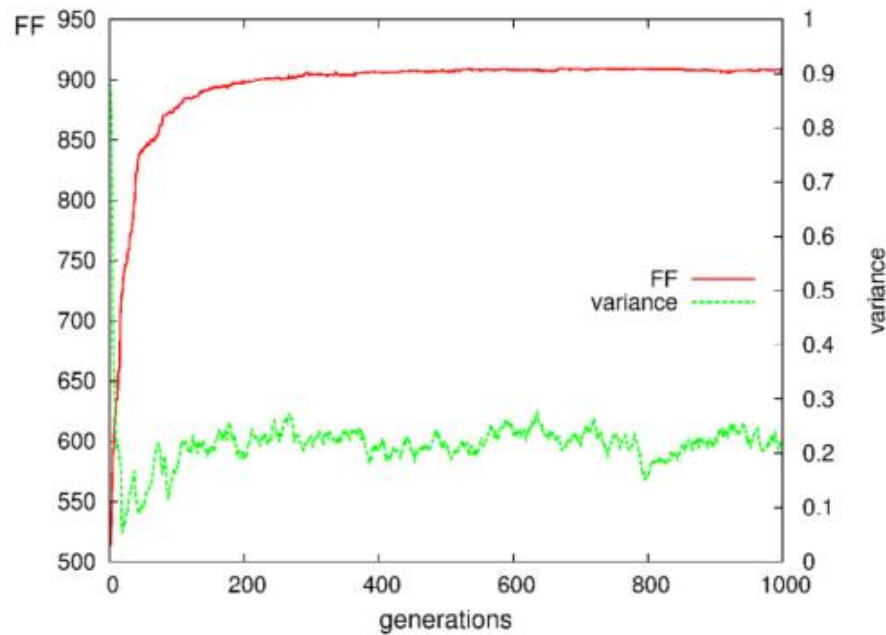


Fig. 3.5. Case 1 Fitness function and genetic variance for $N = 100$, $n = 1000$, $p_m = 1\%$, $\sigma_{\min} = 1.0$.

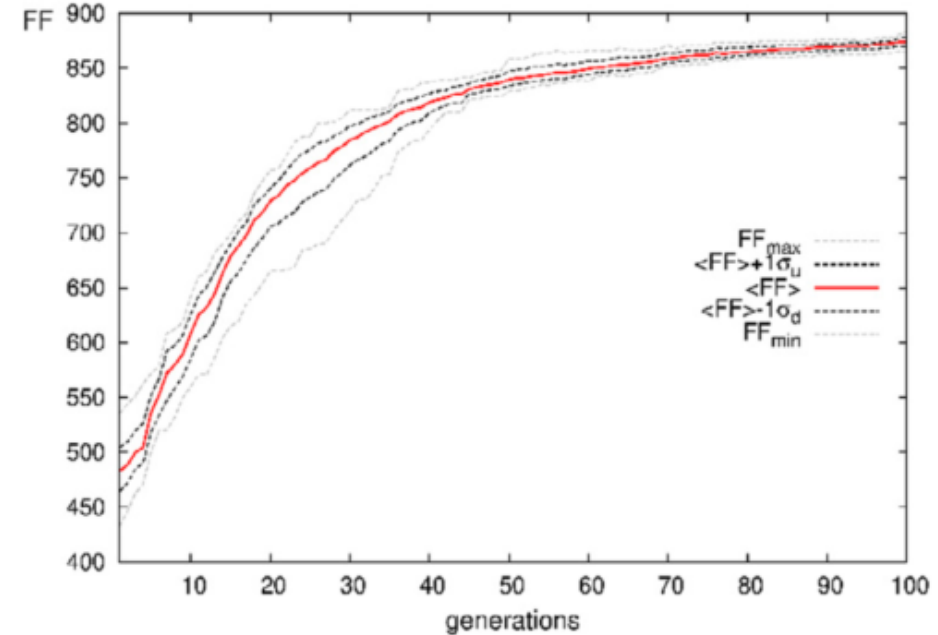


Fig. 3.7. Results for stochastic bias tests for the ensemble of 8 realizations of Case 1. |
Maximal value, minimal value and mean value for realizations as a function of generations with 1 σ uncertainty band for Fitness Function (left) and genetic variance (right).

Test 2 - max cycle and limited k-eff

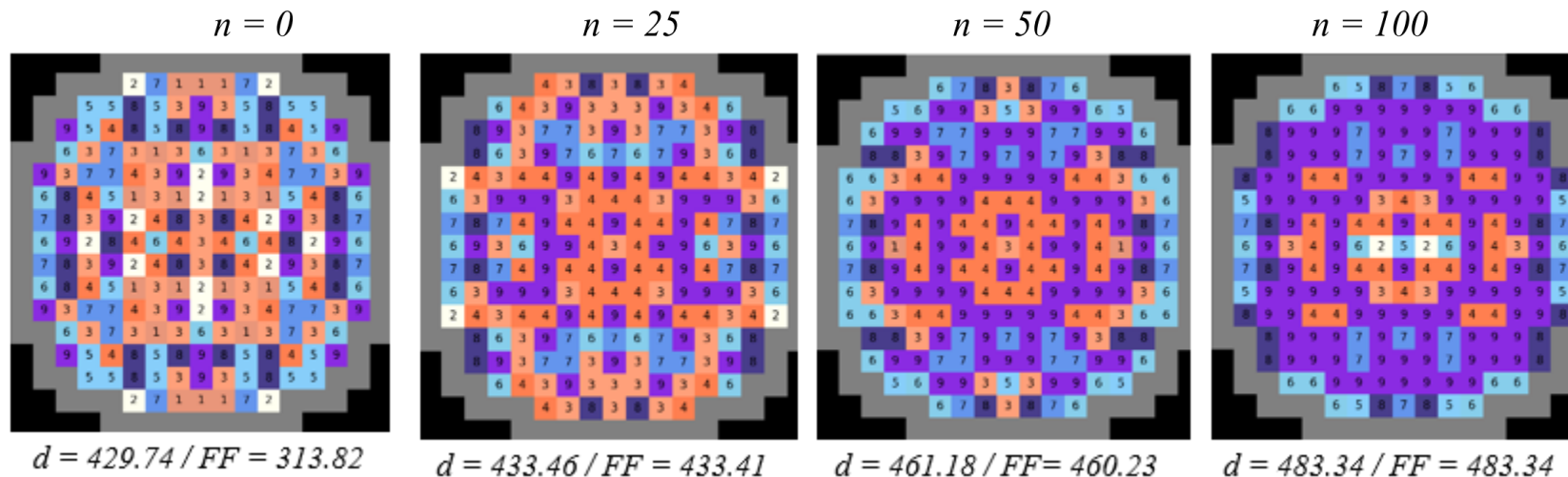
- k-eff (excess reactivity) < 1.07 (corresponds to CRW limit ~ 7500 pcm)
- Not known solution
- 100 generations and 100 specimens
- Solution with k-eff = 1.07

Chromosome max k-eff

Target k-eff=1.07

$$FF = d^\beta \cdot \left[\alpha_0 \cdot \left(1 + \frac{k_1 - \max(k)}{k_2 - k_1} \right)^\gamma \right]^{-\beta}$$

max possible k-eff=1.22



Test 3 – max cycle and fissile mass constant

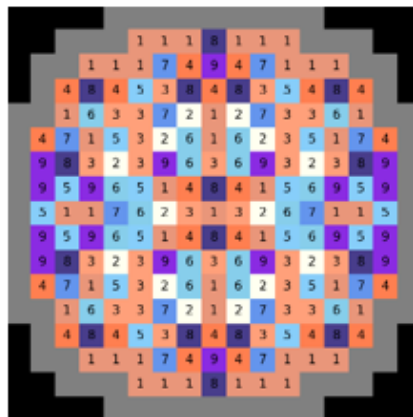
- Keep fissile mass close to BEAVRS mass
- Not known solution
- 100 generations and 100 specimens
- Longer cycle, less BAs but k-eff much larger – not perfect

Chromosome mass BEAVRS mass

$$FF = d^\beta \cdot \left(\alpha_0 \cdot \left[1 + \left(\frac{E - E_0}{E_0} \right)^2 \right]^\gamma \right)^{-\beta}$$

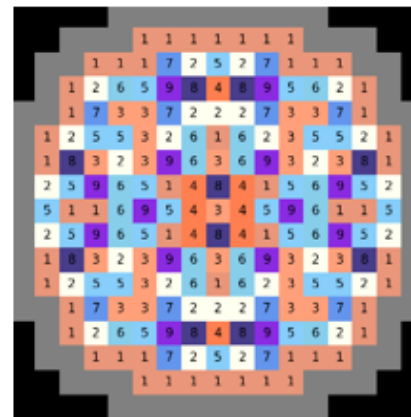
Case	Cycle length [days]	Fissile material mass indicator	Total number of BA rods	Max keff
3	426.71	476	288	1.204
BEAVRS	333.65	456	1268	1.080

$n = 0$



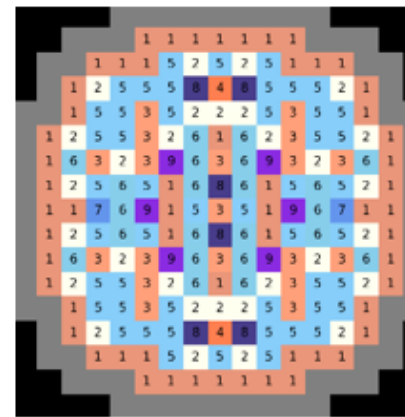
$d = 391.93 / FF = 367.34$

$n = 25$



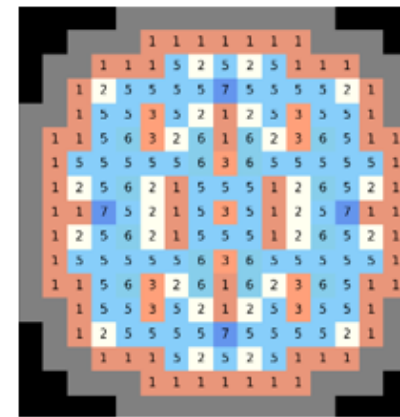
$d = 411.79 / FF = 399.86$

$n = 50$



$d = 422.19 / FF = 410.27$

$n = 100$



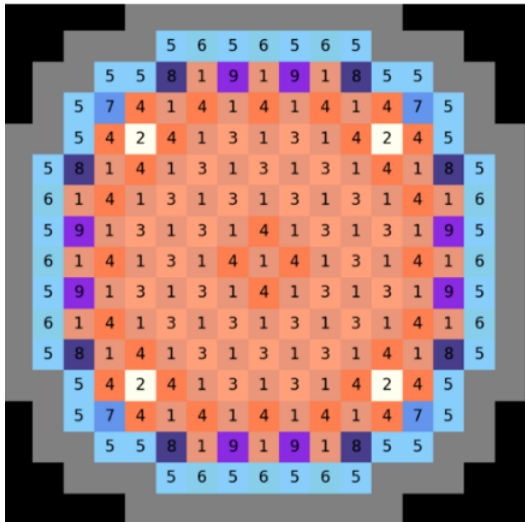
$d = 426.71 / FF = 414.10$

Test 4 – max cycle and min PPF

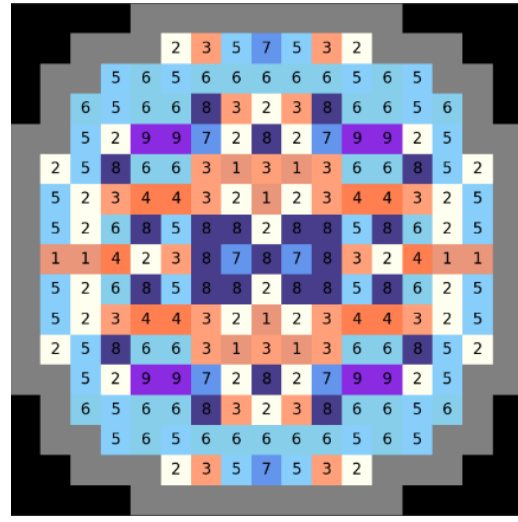
- Maximize cycle length and minimize power peaking
- 500 generations with 100 specimens

$$PPF = P_{xyz} = \frac{\text{max heat flux in the core}}{\text{average heat flux in the core}}$$

$$P_{xyz} = P_{xy}P_z = \frac{\text{average heat flux of the hot channel}}{\text{average heat flux of all channels}} \times \frac{\text{max heat flux of the hot channel}}{\text{average heat flux of the hot channel}}$$



BEAVRS

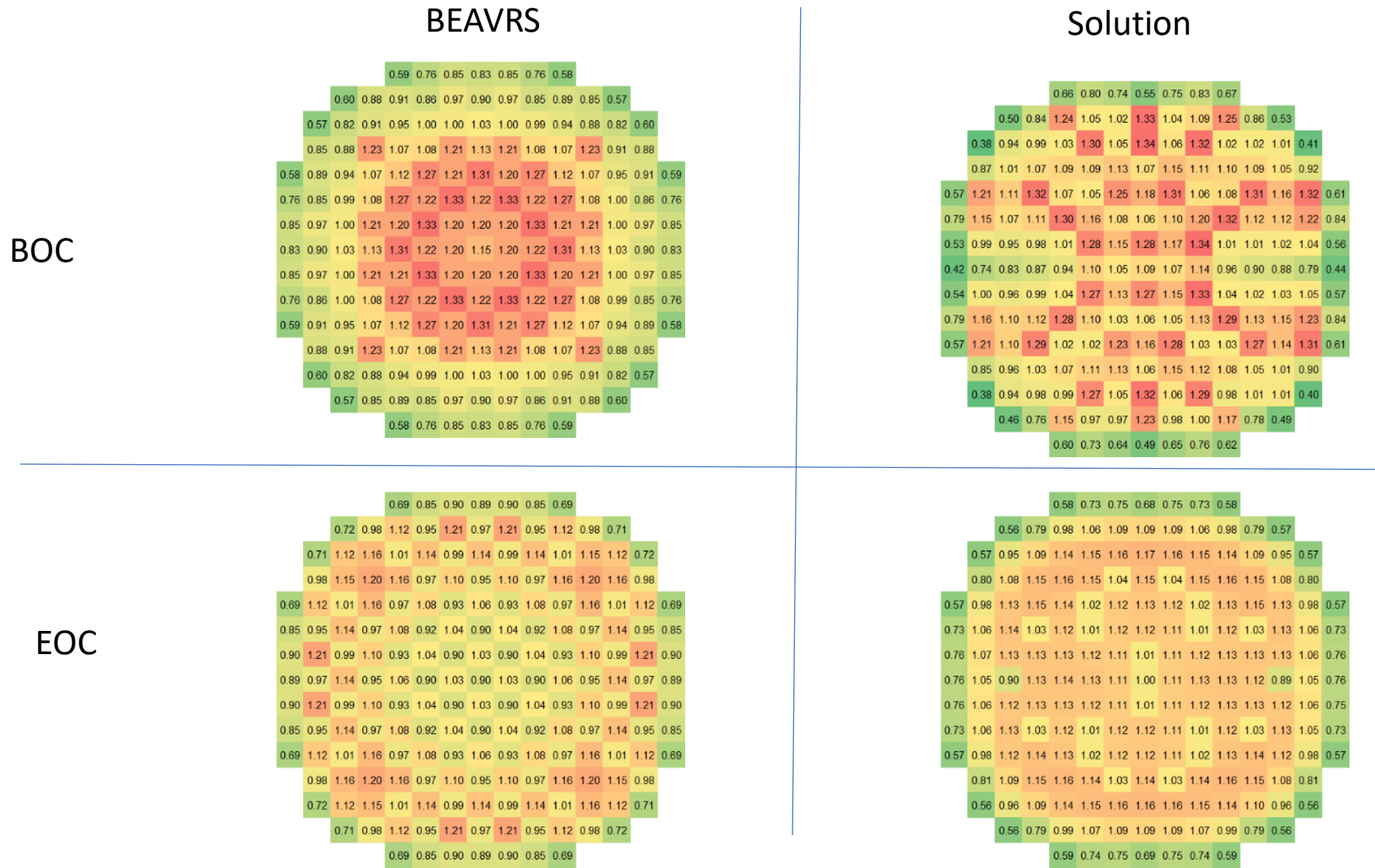


Solution

$$FF = d/PPF$$

Param.	Average enrichment [%]	No. of BA rods	Initial k_{eff}	PPF _{max}	Cycle length [days]
BEAVRS	2.36	1268	1.08	1.88	333.6
Case 2	2.78	1356	1.13	1.89	512.5

Test 4 – Radial Power Distr. Maps



	BEAVRS	Solution
$P_{xy_{max}}$ BOC	1.33	1.34
$P_{xy_{max}}$ EOC	1.21	1.17

Summary and Conclusions

- GA were introduced.
- GA are efficient tools to optimize in nuclear engineering – examples for core loading pattern selection presented.
- Can be useful in design optimization.
- GAs are easy to implement.
- Populations are large so application of computationally expensive tools is limited.
- FF development can be challenging.
- Efforts are ongoing.

References

- [1] <https://www.sciencedirect.com/topics/engineering/genetic-algorithm>
- [2] MATLAB, What is genetic algorithm?, 2015. https://www.youtube.com/watch?v=1i8muvzZkPw&list=PLkOyLScCF4hWlf27ZiMZmI525AKRS-Mp&index=12&ab_channel=MATLAB
- [3] Kie Codes, Genetic Algorithms Explained by Example, 2020. https://www.youtube.com/watch?v=uQj5UNhCPuo&t=459s&ab_channel=KieCodes
- [4] Albadr, M., et. al. Genetic Algorithm Based on Natural Selection Theory for Optimization Problems <https://www.mdpi.com/2073-8994/12/11/1758>
- [5] K. Chec, Praca inzynierska, "Wykorzystanie algorytmów genetycznych w zarządzaniu cyklem paliwowym reaktora PWR (In-Core Fuel Management)"
- [6] E. Israeli and E. Gilad, "Novel genetic algorithms for loading pattern optimization using state-of-the-art operators and a simple test case," J. Nucl. Eng. Radiat. Sci., vol. 3, no. 3, pp. 1–10, 2017, doi: 10.1115/1.4035883.
- [7] J. H. (John H. Holland, Adaptation in natural and artificial systems : an introductory analysis with applications to biology, control, and artificial intelligence. MIT Press, 1992.
- [8] MIT CRPG, "BEAVRS - Benchmark for Evaluation Reactor Validation of And Simulations Rev. 2.0.2.," 2018.
- [9] P. Darnowski and M. Pawluczyk, "Analysis of the BEAVRS PWR benchmark using SCALE and PARCS," Nukleonika, 2019, doi: 10.2478/nuka-2019-0011.
- [10] W. Kubiński, P. Darnowski, and K. Chęć, "The development of a novel adaptive genetic algorithm for the optimization of fuel cycle length," Ann. Nucl. Energy, vol. 155, 2021, doi: 10.1016/j.anucene.2021.108153.
- [11] W. Kubinski, P. Darnowski, K. Chec, "Optimization of the loading pattern of the PWR core using genetic algorithms and multi-purpose fitness function", Nukleonika, 2021;66(4):147-151
- [12] W. Żurkowski, P. Sawicki, W. Kubiński , P. Darnowski, "Application of genetic algorithms in optimization of SFR nuclear reactor design", Nukleonika, 2021;66(4):139-145
- [13] W. Kubinski, P. Bojarski, P. Darnowski, Application of Artificial Neural Network and Particle Swarm Optimization in determining selected parameters of the nuclear reactor core, 2021 Conference: ENYGFAT: Tarragona, Spain
- [14] W. Kubinski, P. Darnowski, K. Chec, Optimization of BEAVRS PWR Loading Pattern Using a Nove Genetic Algorithm Based on Population Variance Control, NENE-2020