Experimental Verification of the Computational System for the Optimal Pilot Bus Selection

Michal Kopcek, Tomas Skulavik

Abstract— **The power system is generally characterized by its size, complexity and time varying, beforehand unknown electrical load. Even slight changes in the settings can produce significant economic savings, because large electric power is transferred. Based on the characteristics of the power system and the nature of the secondary voltage control, it is clear that the optimal pilot bus selection plays a key role in the quality of control and related economic impacts. This paper deals with the experimental verification of system architecture for optimal pilot bus selection of supervisory node voltage control in a large–scale power system, so that the key problem could be solved using the principle of parallelism and grid computing structure to make the computer time consumption as low as possible.**

Index Terms—**Large-scale power system, Optimal pilot bus selection, Proof of concept, System architecture verification.**

I. Introduction

THE system architecture design and proof of concept implementation of the system for optimal pilot bus selection for the need of power system supervisory control in the real time was introduced in [1] in the detail. Individual selection methods [2], [3], [4] require the calculation of objective functions. The calculation is based on simulating the operation of the power system. Hence parallel processing principles (grid computing) were used in the designed architecture. Optimal pilot bus selection is a problem with extensive state space, therefore the genetic algorithms (GA), which can also very well apply the principles of parallelism, were chosen as a base of the architecture.

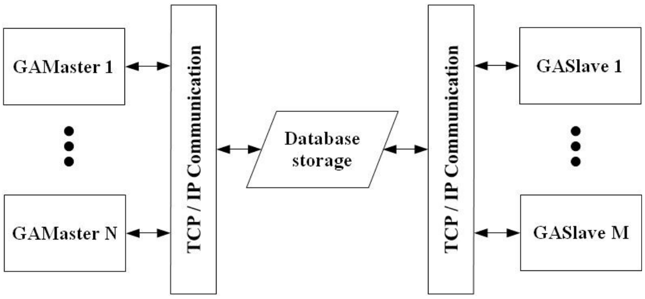

Fig. 1. Block diagram of designed system architecture [1].

Structure of the computational system captures the block diagram of the architecture shown in Fig. 1.

This contribution deals with the experimental verification of the presented proof of concept implementation of the system for optimal pilot bus selection in real time.

A steady-state model of the power system of the Slovak Republic was used during the experiments. The model at 400/200 kV comprises 82 branches and has 59 nodes, of which 16 are generators and 43 represent load. The voltage sensitivity coefficients matrix for all tests was calculated to change the voltage of +5% of nominal voltage values for all generator nodes. The symmetrical load +15% of actual value of reactive power load of each node was applied. These values are the same as in [3] and [4] in order to compare the results.

II. Size of Searched State Space

As noted before, one of the biggest problems of optimal pilot bus selection or possibly combination of two and more pilot buses, is the size of the state space, which should be searched. The size of the state space is given by the number of generator and load nodes, included in the investigated power system, and which creates a candidate pair for the pilot bus selection.

The power system model used consists of 16 generators and 43 load nodes. Therefore, the size of the state space for the search of exactly one pilot bus could be determined as follows:

$$V_P = N_G * N_Q = 16 * 43 = 688 \qquad (1)$$

where:

V_P - the state space size;

N_G - the number of generators;

N_Q - the number of loads.

It is obvious that the basic searched state space is made by 688 possible candidate pairs. The size of this space is therefore the starting parameter for determining the size of state spaces to be searched, if the optimal n-tuple of the pilot buses (for n> 1) has to be found, because it is the basic set of candidates for the pilot bus. Increasing the number of the searched pilot buses from one optimal n-tuple represents a combination of n-th class from the k-elements, without repetition. This dependence can be mathematically expressed by the following formula:

$$C_n(k) = \frac{k!}{(k-n)!.n!} \qquad (2)$$

Manuscript received June 25, 2012; revised July 23, 2012.

M. Kopcek is with the Institute of Applied Informatics, Automation and Mathematics, Slovak University of Technology in Bratislava, Faculty of Materials Science and Technology in Trnava, Trnava, 917 24, Slovak Republic (phone: +421-906-068711; e-mail: michal.kopcek@stuba.sk).

T. Skulavik is with the Institute of Applied Informatics, Automation and Mathematics, Slovak University of Technology in Bratislava, Faculty of Materials Science and Technology in Trnava, Trnava, 917 24, Slovak Republic (e-mail: tomas.skulavik@stuba.sk).

where:
$n \leq k$;
n - the size of n-tuple (number of desired pilot buses);
k - the size of candidates set (the number of candidates for one pilot bus).

The formula (2) proves, that the bigger the values of the parameters n and k are, the bigger the size of the searched state space would be. Increasing of the parameter n (the size of optimal pilot buses combination) is necessary according to the control of the investigated power system. Therefore, the only way to reduce the size of the searched state space is to decrease the parameter k using the pilot bus candidate reduction methods. The brief description of the chosen reduction methods follows.

A. Reduction by the Threshold of Sensitivity Coefficient

This method is described in detail in [4]. It can significantly reduce the size of the searched state space. It presupposes the existence of dependence between the value of the sensitivity coefficient and the value of the objective function of the searched n-tuple of pilot buses.

B. Reverse Selection Method

This method is described in detail in [3]. It can significantly reduce the size of the searched state space and it gives very good results. Its main disadvantage is the use of the global search method (for n=1) for initialization. However, it is more efficient than the reduction by the threshold of sensitivity coefficient (TSC).

For better understanding of the state space size and the effectiveness of selected methods the comparison of their results with unreduced space is shown in Fig. 2. In order to view all data in a common graph, the vertical axis representing the size of the state space uses the logarithmic scale.

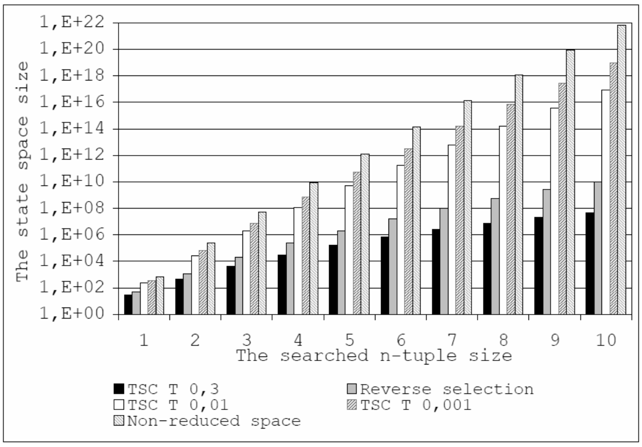

Fig. 2. The size of the searched state space for selected reduction methods. [2].

III. Parallel Computation of Fitness Function

The chromosomes fitness function is evaluated in GA Slave modules (Fig. 1). The computational system enables the deployment of multiple GA Slave modules to solve separate parts of one defined task. The contribution of this approach is proved by the following tests.

A. Test No 1. - Global Search of One Pilot Bus

The global search of the whole state space was performed in this experiment in two different computational system configurations. These configurations vary in the number of used GA Slave modules. The size of the searched state space was 688 combinations. The results are shown in Table 1.

The obtained results confirm the suitability of using the parallel computation. The computation time of the task was significantly reduced thanks to the use of multiple GA Slave modules. These results also prove the ability to improve the performance of the reverse selection method, because as mentioned before this method uses global search for initialization.

TABLE I
Results of Global Search

		Conf. 1	**Conf. 2**
Number of modules	GA Master	1	1
	GA Slave	1	8
Average time of one chromosome fitness evaluation [s]		2.04	1.77
Time consumption of the whole task [h:m:s]		**1:50:06**	**0:16:44**

B. Test No 2. - Searching a Triplet of Pilot Buses Using GA

The genetic algorithms with parallel computation of fitness function were used during this experiment in three different configurations. The size of the basic state space was 688 combinations and an optimal triplet of pilot buses was searched. The GA are able to find optimal solution even in an extensive state space, however they are essentially slow. This disadvantage could be efficiently removed using parallelism. The results shown in Table 2 prove this fact.

Increasing the number of GA Slave modules used for the computation of one task led in a significant reduction of computing time as seen from the results. This test confirms the results obtained in Test no. 1 and again clearly demonstrates the correctness of the hypothesis, that the use of the parallel approach to calculate the fitness function of chromosomes provides notable reduction in computing time and hence the overall time for solving the defined task.

TABLE II
Results of Triplet of Pilot Buses Search Using GA

		C. 1	**C. 2**	**C. 3**
Number of modules	GA Master	1	1	1
	GA Slave	1	4	8
Time consumption of the task [h:m:s]		**2:58:14**	**0:44:01**	**0:34:29**

IV. Performance of Genetic Algorithms

The purpose of this experiment is to assess the main features of the implemented genetic algorithms, particularly, the computational complexity and rate of convergence of the algorithm to the optimal solution.

The computational complexity of classical genetic algorithms can be determined quite easily as the product of the population size and number of generations.

This calculation is not applicable to the designed architecture, because with the procedures implemented on the database server, the fitness function is evaluated just once for just one chromosome. Therefore, it is not possible to determine the number of runs of the fitness value calculation for one population.

For these reasons the computational complexity of each experiment is determined by using the average processing time of one population in order to gain a rough idea of the processing rate of defined task, and thus the speed of the implemented system.

Determining the number of generations needed to reach the global minimum of the fitness function is also complicated. This number can be determined experimentally based on the speed of convergence of genetic algorithm to the global optimum. The speed of convergence is strongly dependent on many factors. The structure of the algorithm has significant influence, which means not only the implemented functions, but also a way of sorting of these functions. Other factors that significantly affect the rate of convergence are:

- population size,
- number of best chromosomes passing into the new population,
- number of original individuals passing into the new population,
- method of the original individuals selection,
- size and the method of a working group of chromosomes selection on which the genetic operations are carried out,
- probability of crossover, mutation and migration.

The genetic algorithms convergence speed was investigated for different settings of the above-mentioned parameters during the experiments. The structure of the algorithm used was implemented according to [2] and multiobjective function was used to evaluate the fitness function. The populations were generated randomly without initialization. Several experiments with different configurations were done while different n-tuples of optimal pilot buses were searched. Given the amount of obtained data, only the representative result was described. Similar results were achieved in other experiments, too.

The result of genetic algorithm convergence for searching the optimal sextuplet of pilot buses (n=6) is shown in Fig. 3. Each curve corresponds with one search carried out using the genetic algorithms. The main aim of the experiment was to test the system's ability to find an optimal solution and determine the speed of convergence for several runs of the same task with unchanged parameters. The results show that the algorithm converges to the optimum after approximately 100 generations. The number of generations is plotted in a logarithmic scale for better resolution.

Tests were performed with the following setup of parameters:

- number of generations: 800,
- population size: 30,
- number of best individuals: 3,
- number of original individuals: 3,
- working group size: 24 (12 of best and 12 of original individuals),
- normal mutation probability: 0.2,
- additive mutation probability: 0.2,
- amplitude of the additive mutation: 5,
- number of normal crossover points: 3.

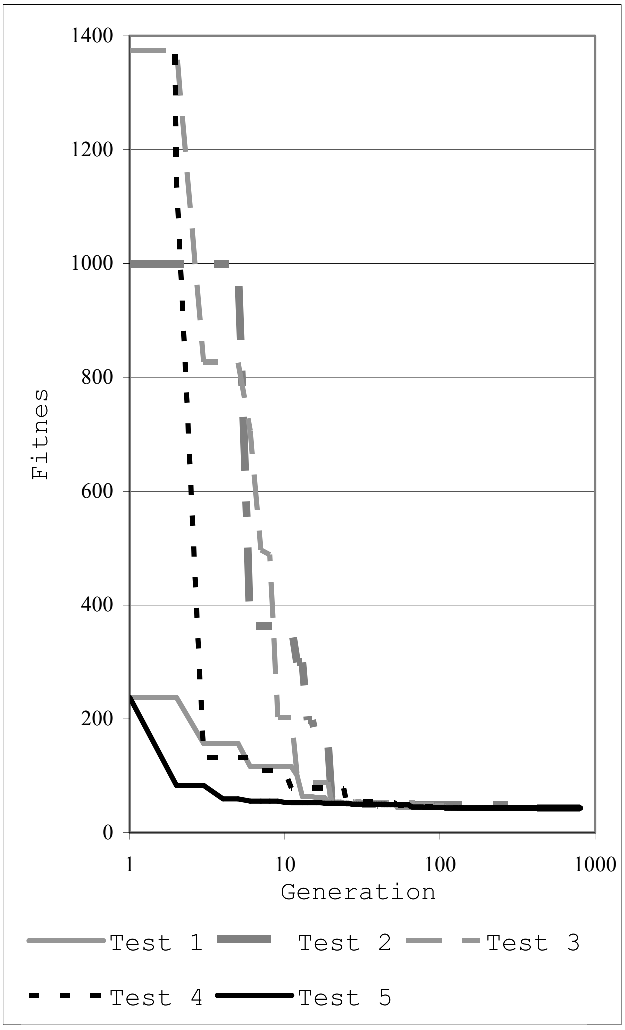

Fig. 3. The convergence of computation for n=6.

The searched state space was reduced using the TSC method with the threshold value σ = 0, which means that the size of basic state space was 598 candidates. This corresponds with the value of 61 936 603 045 317 possible combinations. During the computation 8 GA Slave modules was involved.

Results of experiments show the potential of genetic algorithms. Despite the slowness of calculation resulting from the nature of genetic algorithms, experiments proved that the designed architecture is effective. It is clear that by further tuning of parameters of the genetic algorithm, or by a suitable change of its structure, better results can be achieved.

V. Performance of Parallel Genetic Algorithms

This section describes the experimental verification of the functionality of a parallel genetic algorithm implemented in the context of the designed architecture. The structure of the migration of best subpopulations individuals is shown in Fig. 4. The migration of chromosomes takes place at random generation with some probability, which is freely adjustable for each GA Master module. The module managing A subpopulation, will request the best chromosome from the last generation of B subpopulation, when the condition of the probability is met. The module taking care of B subpopulation, works the same way. Two separate GA Master modules were used for the purpose of the experiment, as apparent from the description.

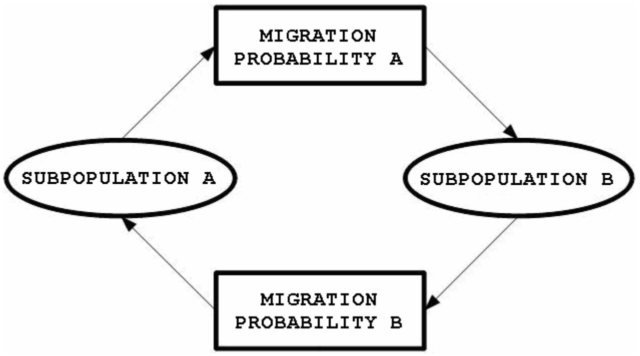

Fig. 4. The structure of tested parallel genetic algorithm. [2].

An optimal triplet of pilot buses was searched in this experiment and following common parameters were used for separate tests:

- number of generations: 600,
- subpopulation size: 20,
- number of GA Master modules: 2,
- number of GA Slave modules: 8.

Based on the results of previous experiments, the searched state space was reduced using the reverse selection method. Thus the size of basic state space was 50 candidates, which corresponds with the value of 19 600 possible combinations. The parameters varied for individual tests during the experiment, are shown in the Table 3.

Convergences of performed tests in the presented experiment are captured in a few charts for the best possible comparison. Graphs in Fig. 5 and Fig. 6 compare the convergences of subpopulations A and B for all performed tests. The number of generations is plotted in a logarithmic scale for better resolution. Common trend of convergence of subpopulations A and B separately for each test is shown on graphs in Figs. 7 - 11.

The graphs show how subpopulations influenced each other. The influence of genetic algorithms parameters settings on subpopulations evolution or possibly the rate of result convergence to the global optimum is noticeable as well. This is best seen in the graph in Fig. 11, which belongs to Test no. 5. The mutation probability of A subpopulation in this test was set at quite a low value, resulting in an algorithm travels in quite a small space and long stuck in local extrema. On the other hand, the mutation probability of B subpopulation was set at quite a high value. The algorithm in this case traveled in a much larger space, but also stuck for a long time in local extrema. Nevertheless, the result eventually converges to the global optimum even if the number of generations is much higher than in other cases, as seen in the charts.

TABLE III
Parameters of Parallel GA for Separate Tests

Test nr.	Subpopulation	The number of the best individuals	The number of the original individuals	The work group size (best + original)	The normal mutation probability	The additive mutation probability	The additive mutation amplitude	The number of normal crossover points	The migration probability
1	A	2	2	16 (8+8)	0.2	0.2	1	1	0.3
	B	2	2	16 (8+8)	0.6	0.6	5	1	0.3
2	A	2	2	16 (8+8)	0.2	0.2	1	1	0.2
	B	2	2	16 (8+8)	0.4	0.5	8	1	0.15
3	A	2	2	16 (8+8)	0.2	0.2	1	1	0.2
	B	2	2	16 (8+8)	0.4	0.5	8	1	0.15
4	A	1	1	18 (9+9)	0.2	0.2	1	1	0.2
	B	2	2	16 (8+8)	0.4	0.5	8	1	0.1
5	A	1	1	18 (9+9)	0.1	0.1	1	1	0.2
	B	1	1	18 (9+9)	0.9	0.9	8	1	0.05

The presented results clearly show that the parallel genetic algorithm is valid and can improve the behavior of ordinary genetic algorithm mainly due to its ability to search through a much larger state space at the same time. The indisputable advantage is also its increased ability to get out of the local extrema.

Finally, the average total time of the defined task calculation is evaluated. The time is evaluated for the processing of 600 generations. However, it should be noted, that in this configuration, the system worked in parallel with two subpopulations, which corresponds with the number of 1200 generations processed within a single computing task. The time consumption of individual experiments is captured in Table 4.

TABLE IV
EXPERIMENT TIME CONSUMPTION EVALUATION

Test	The average time of one population evaluation [s]	The average time consumption of the task [h:m:s]
1	7.99	**1:19:55**
2	4.72	**0:47:12**
3	4.86	**0:48:36**
4	8.68	**1:26:49**
5	6.18	**1:01:48**

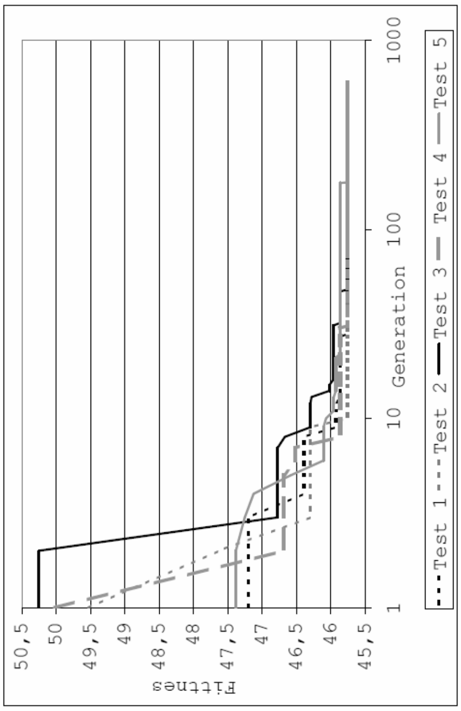

Fig. 5. The convergence of the subpopulation A.

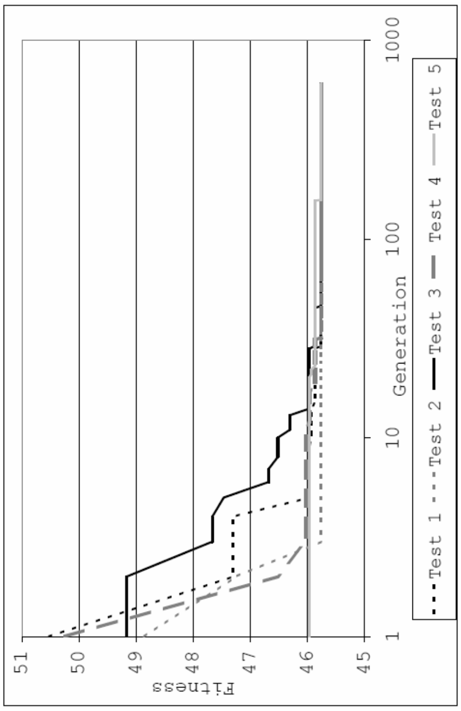

Fig. 6. The convergence of the subpopulation B.

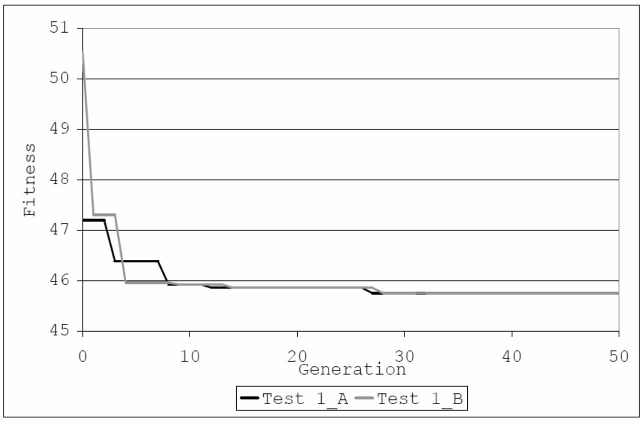

Fig. 7. The convergence of subpopulations A and B in Test 1.

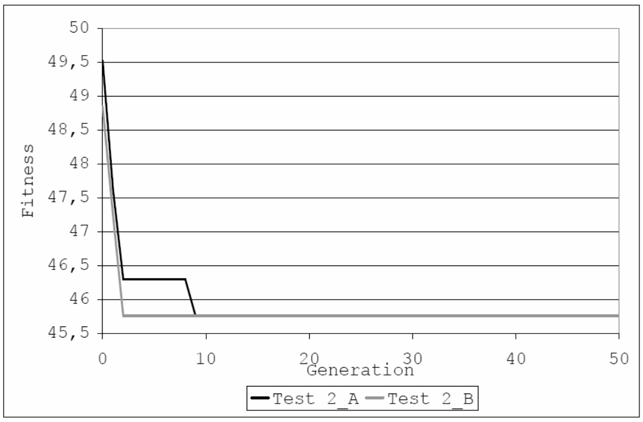

Fig. 8. The convergence of subpopulations A and B in Test 2.

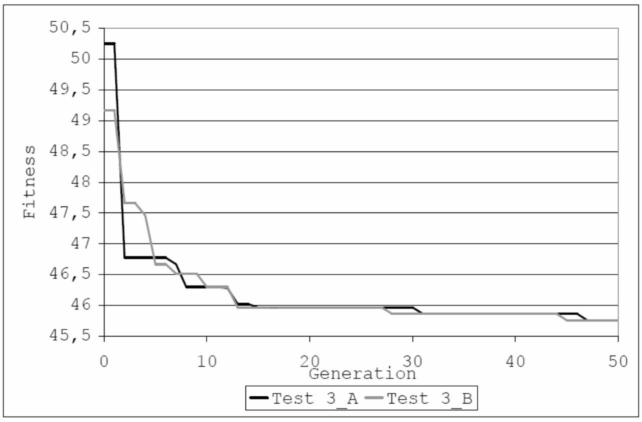

Fig. 9. The convergence of subpopulations A and B in Test 3.

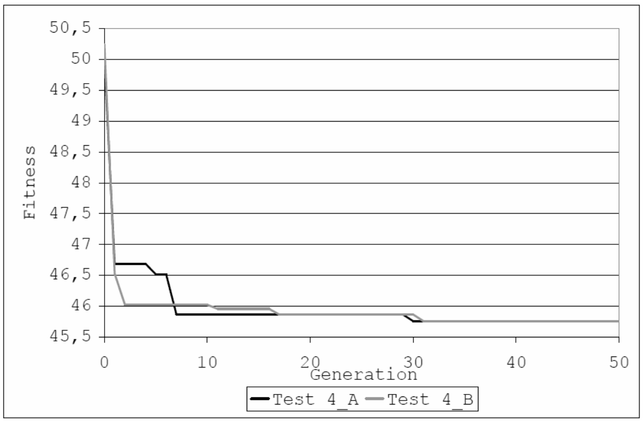

Fig. 10. The convergence of subpopulations A and B in Test 4.

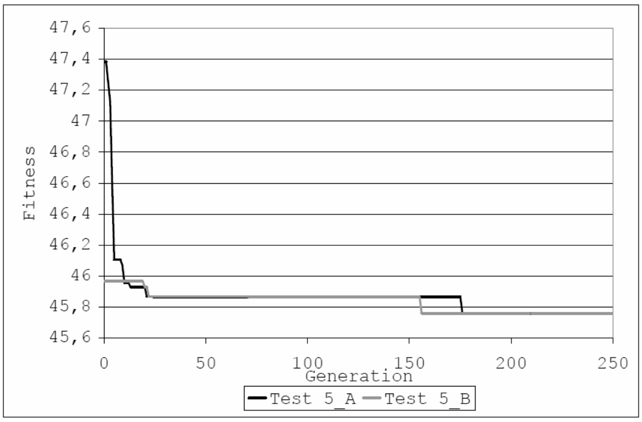

Fig. 11. The convergence of subpopulations A and B in Test 5.

VI. Conclusion

The experiments described in this paper clearly prove the assumptions that were taken into account during the design of the architecture of the optimal pilot bus selection system for the needs of the power system supervisory control in real time. It should be noted that the proof of concept implementation of the architecture cannot be considered definitive. Its purpose was purely experimental and should only serve to verify the proposed architecture.

Experiments have shown the potential of the GA and the parallelism for solving the problem of optimal pilot bus selection in real time. With regards to the achieved results the designed architecture could be considered as verified.

References

[1] M. Kopcek, D. Mudroncik, "Using GA and computation parallelism for the optimal pilot bus selection," in: Lecture Notes in Electrical Engineering. - ISSN 1876-1100. - Vol. 122. Advances in Automation and Robotics, Vol. 1: International Conference on Automation and Robotics, ICAR 2011, Dubai, December 2011. - ISBN 978-3-642-25552-6, p. 323-328

[2] M. Kopcek "Optimization of the power system processes for primary and secondary voltage control," in Slovak, Ph.D. dissertation, UIAM, MTF STU, Trnava, SR, 2011.

[3] S. Kunik, "Optimal pilot bus selection in the secondary voltage control of a large scale power system," in Slovak, Ph.D. dissertation, KAIA, MTF STU, Trnava, SR, 2008.

[4] M. Rakovsky "Reduction of the pilot bus selection in secondary voltage control," in Slovak, Ph.D. dissertation, KAIA, MTF STU, Trnava, SR, 2006.