# A new reconfiguration mechanism of bio-inspired self-repairing hardware based on variable neighborhood search algorithm

Xiubin Liu, Xiangli Feng, Yanling Qian, Long Wang, Yue Li* *Laboratory of Science and Technology on Integrated Logistics Support*
*National University of Defense Technology*
Changsha, China
Liuxiubin_nudt@163.com, liyue@nudt.edu.cn

***Abstract*—The bio-inspired self-repairing hardware greatly enhances the reliability of the circuit by endowing the circuit with the self-repairing property similar to that of the biological tissue, so it has broad application prospects. In order to overcome the shortcomings of current bionic self-repair hardware, such as not flexible enough, low utilization of spare cells, and insufficient consideration of wiring resource overhead and self-repairing time. This paper proposes a reconfiguration mechanism based on variable neighborhood search algorithm, which can restore the array function to normal only by changing the faulty electronic arrays in a small scale. At the same time, it can make full use of the spare cells in the array, and maintain a better wiring resource overhead and self-repairing time within a certain number of faulty cells.**

***Keywords*—bio-inspired electronic array, search algorithm, FPGA**

## I. Introduction

More and more electronic devices are being sent to space, deep sea and other places to explore the unknown world. These devices may face harsh environments such as electronic radiation, high temperature, corrosion, etc., resulting in equipment failure, and it is difficult to get repairs in time after failure. In order to improve the reliability of the task, it is necessary to find a way to enable the device to recover its functions autonomously after a failure. The traditional solution mostly adopts the technology of Triple Modular Redundancy(TMR) [1]. Based on the analysis of the failure mode in advance, the modules prone to be faulty will be set up two backups, and most of the same output will be used as the correct output of the system through the voting system. As long as two or more modules fail at the same time, the system can ensure the correct output of the system. TMR is simple to implement, but there are obvious deficiencies. On the one hand, the judgment of components that are prone to failure depends on the designer's awareness of the environment where the equipment is located, but in many cases, the working environment is complicated or even unknown. This creates uncertainty about the selection of components that require backups. On the other hand, backups of a single component will increase the size and quality of the device, increasing the design costs.

The proposal of bio-inspired self-repairing hardware provides a new solution to the above problems. Mange et al. proposed a new field programmable gate array based on the development of multicellular organisms [2]. Its basic idea is to apply bionic mechanism to the design of electronic circuits, so that electronic circuits can change their structure and parameters independently and dynamically according to the working environment to achieve desired performance. It has the characteristics of self-adaptation and self-repairing similar to organisms.

The bio-inspired self-repairing hardware is an electronic array composed of general-purpose reconfigurable electronic cells. The function of each cell is determined by the configuration information stored inside the cell, and the overall function of the electronic array is accomplished by the cooperation of each cell. The cell contains a cell-level self-test unit. When it fails, the faulty cell sends an "error" signal, triggering the electronic array to perform online reconfiguration, each cell reselects configuration information and performs new functions. At the same time, the communication channel between cells is rerouted to restore the array function.

The bio-inspired self-repairing hardware has changed the traditional redundant fault-tolerant scheme, giving the functional circuit self-repairing and self-adaptive capabilities, and has broad application prospects. However, in the fault condition, it is a major difficulty to reconfigure the function and communication connection of electronic cells.

The structure of the rest parts of this paper is as follows. Firstly, the related works of online reconfiguration mechanism are introduced. Then the proposed reconfiguration mechanism based on variable neighborhood search algorithm is described in detail. Following that, the simulation and discussion are done. Finally, the conclusion is drawn.

## II. Related works

As an important part of bio-inspired self-repairing hardware research, online reconfiguration mechanism has achieved some results. At present, the common reconfiguration mechanisms mainly include the column (row) removal mechanism, the single cell removal mechanism and the neighbor replacement mechanism.

978-1-7281-0860-5/19/$31.00 ©2019 IEEE

## A. The column removal mechanism

In the column removal mechanism, the whole column where the the faulty cells are in is replaced by the right columns [3], as shown in Fig. 1. This mechanism consumes a column of cells to repair a faulty cell, which wastes a lot of resources.

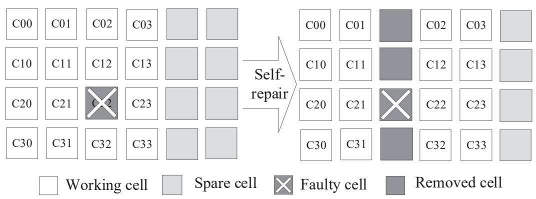


Fig. 1. The column removal mechanism

## B. The single cell removal mechanism

In the single cell removal mechanism, as shown in Fig 2, when a cell fails, the function of the faulty cell on the side of the spare cell is shifted backward, and as a whole, the faulty cell is replaced by an spare cell [4]. Compared with the column removal mechanism, the single cell removal mechanism consumes fewer spare cells, but there may be problems such as congestion of the routing channels and excessive wiring length.

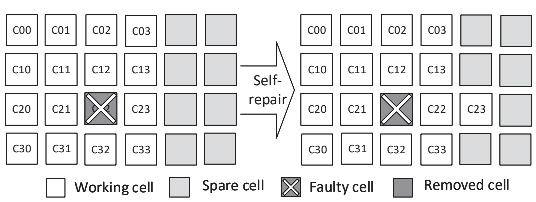


Fig. 2. The single removal mechanism

## C. The neighbor replacement mechanism

The neighbor replacement mechanism pre-arranges the spare cells around the working cells, and when the working cells fail, the spare cells around them replace the faulty cells according to their priorities.

*1) Szasz's neighbor replacement mechanism:* The neighbor replacement mechanism proposed by Szasz et al., as shown in Fig. 3, a group is formed of 9 cells, and cell arrays were formed by two-dimensional uniform distribution of these groups. In one group, 5 cells located at the four corners of the matrix and the middle of the matrix are in working state, and the remaining 4 cells are used for backup. Each cell contains five genes(denoted as A, B, C, D, E), and five working cells decompose one of the genes (denoted by underlined and bold) to achieve five functions. There are at least 2 spare cells around each working cell. If a cell fails, it can be quickly replaced by small-scale routing adjustment [5].

*2) Lala's neighbor replacement mechanism:* In the cell array structure proposed by Lala et al., as shown in Fig. 4,there are two spare cells S and two routing cells R located around each function cell F, and each spare cell S can replace any one

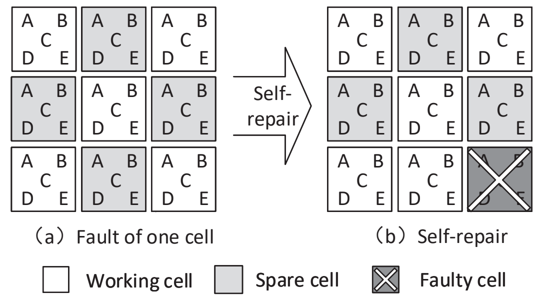


Fig. 3. Szasz's neighbor replacement mechanism

of the four function cells F around it. The cell R preserves the possible interconnection information during cell replacement, which facilitates rapid local routing [6].

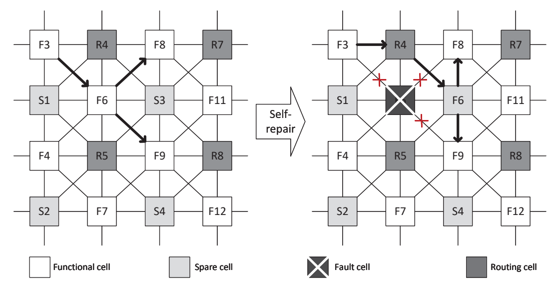


Fig. 4. Lala's neighbor replacement mechanism

However, when local cells fail continuously, the difficulty of implementing the reconfiguration mechanism proposed by Szasz and Lala et al. will be greatly improved.

To summarize, the traditional reconfiguration mechanism has the advantages of less computation and easy implementation, which meets the real-time requirement of bio-inspired self-repairing hardware. However, there are also some shortcomings.

1) The reconfiguration circuit can not be optimized, which makes the initial placement of the functional circuit more limited and the design flexibility is not high.
2) The spare cells can not replace and repair the faulty cells at any position, so the resource utilization rate is low.
3) The long reconfiguration time makes the repair speed slow, which is not conducive to the smooth operation of the equipment.
4) The reconfiguration process can not take into account the length of the circuit network, and the timing characteristics of the circuit decline after the repair.

Therefore, it is necessary to improve the existing bio-inspired self-repairing hardware reconfiguration mechanism, taking into account both optimization and real-time characteristics, so that it can better solve practical problems.

# III. Proposed reconfiguration mechanism based on variable neighborhood search algorithm

The bio-inspired self-repairing hardware studied in this paper is based on an array formed by the uniform distribution of identically constructed electronic cells, which contain function cells and spare cells. function cells can implement logical functions such as addition, subtraction, AND, OR, and so on. Multiple function cells can cooperate to accomplish complex tasks through interconnected communication. Spare cells are used to replace faulty cells. When the electronic array fails, the array needs reconfiguration, the placement of the original electronic array and the communication connection between the cells will change. So function circuits need to be mapped to the electronic arrays and reroute the communication between cells.

Local search algorithms such as hill climbing algorithm, simulated annealing algorithm(SA) and tabu algorithm are often used to determine the mapping relationship between the logic function of the circuit and the electronic array. Variable Neighborhood Search (VNS) is an improved local search algorithm, which is used to solve a series of combinatorial optimization problems. By defining different neighborhood structures, neighborhood changing is carried out according to evaluation criteria in the search process, so as to obtain better local optimal solutions.

In this paper, the variable neighborhood search algorithm is used to reconfigure the electronic array with the total resource overhead as the optimization objective.

$$\begin{array}{ll} \min & \sum w\left(i\right) \\ s.t. & \exists v_i \rightarrow n_i \\ & \exists e_{i,j} \rightarrow l_{i,j} \\ & vol(e_{i,j}) \leq bw(l_{i,j}) \end{array} \quad (1)$$

where,$w_i$ represents the resource overhead of the network $i$; the constraint condition of the function node is that there is a corresponding fault-free electronic cell $n_i$ for any function graph node $v_i$; the communication constraint is that for any data stream $e_{i,j}$, it can find the corresponding channel $l_{i,j}$, and the data flow $vol(e_{i,j})$ on any channel does not exceed its communication bandwidth $bw(e_{i,j})$.

The algorithms are as follows, where *Algorithm 1* is the overall framework of the reconfiguration mechanism based on VNS, *Algorithm 2* is used to calculate the resource cost change value, and *Algorithm 3* is used to calculate the wiring length.

**Algorithm 1:** Array relocation based on variable neighborhood search algorithm(VNS)

**Input:**
Optimal feasible solution: $x_{best} = x_0$;
Neighborhood parameter: $k = 1$;
Number of outside loop: $c_{out} = 1$
**Output:**
Optimal feasible solution: $x_{best}$

Initialize the local search parameters: local search count value $i = 1$ , minimum resource cost change value $\Delta f_{min} = 0$ ;
**repeat**
  Randomly generate a set of exchange sequence $S_k$ according to $x_{best}$ with $k$, generate the latest solution $x_{now}$ according to $S_k$, and calculate the resource cost change value $\Delta f_{now}$ ;
  **if** $\Delta f_{now} \leq \Delta f_{min}$ **then**
    $\Delta f_{min} = \Delta f_{now}$;
    $S_{min} = S_k$
  **end**
  $i = i + 1$ ;
**until** $i > i_{max}$;
**if** $\Delta f_{min} < 0$ **then**
  $x_{best} = x_{min}$;
  $k = 1$;
  $c_{out} = c_{out} + 1$;
  **if** $c_{out} \leq c_{max}$ **then**
    **goto** *Line 1*
  **end**
**else**
  $k = k + 1$;
  **if** $k \leq k_{max}$ **then**
    **goto** *Line 1*
  **end**
**end**
Output:$x_{best}$

## IV. Experimental verification and result analysis

### A. Determine the experimental object

The alu4 reference circuit of MCNC20 benchmark is mapped to the adaptive hardware. There are 106 CLB modules, 22 I/O modules, 685 line networks. A single line network connects up to 83 function nodes.

As shown in Fig. 5, the bio-inspired self-repairing hardware used in the experiment is similar to the island structure of the FPGA. There are 144 functional electronic cells arranged in a $12 \times 12$ uniform electronic cell array on a two-dimensional plane, and the array surrounded by 48 I/O dedicated electronic cells. The input and output pins of each electronic cell are connected to any interconnected line segment in the adjacent channel through the programmable switch in the connection box. The basic unit of switch box is programmable switch, which distributes at the intersection of horizontal and vertical channels, and plays a connecting role between routing resources. The width of horizontal and vertical channels is 50. The module with numbers represents the function nodes associated with the electronic cells, and the lower left is the coordinate origin position.

At the beginning of the experiment, the initial placement and route of the bio-inspired self-repairing hardware need to be performed by double-cycle simulated annealing algorithm. The placement results are shown in Fig. 5. The digital module represents the function nodes associated with the electronic cell, and the coordinate origin position is at the lower left.

### B. Inject fault into the specific cell and reconfigure

Inject fault to the cell in coordinates $(6, 6)$ to verify the reconfiguration mechanism.

1) The electronic cell has an associated functional nodes $(\#124)$, and start the reconstruction mechanism.
2) Find the nearest spare electronic cell to the faulty cell. The coordinate of the spare cell is $(6, 12)$. The function node $(\#124)$ of the sub-cell of coordinate $(6, 6)$ is

**Algorithm 2:** Calculate the resourse cost change value $\Delta f_{now}$

**Input:** current network length: $w_c(net_i)$;
historical network length: $w_h(net_i)$;
maximum network number: $net_{max}$
**Output:** the resourse cost change value $\Delta f_{now}$

```
Initialize parameters: resource cost change value
  Δf_now = 0, network number net_i = 0;
while net_i < net_max do
    if there is any node that adjusts the position in net_i then
        w_h(net_i) = w_c(net_i);
        Use the Dijkstra algorithm to calculate the w_c(net_i);
        Calculate the length change of the network
          Δw(net_i) = w_c(net_i) − w_h(net_i);
        Δf_now = Δf_now + Δw(net_i)
    end
    net_i = net_i + 1
end
```

**Algorithm 3:** Use the Dijkstra algorithm to calculate the current wiring length $w_c(net_i)$

**Input:** set $S$: record the nodes that has obtained the shortest path;
set $U$: record the node that has not yet obtained the shortest path.;
maximum network number: $net_{max}$
**Output:** network length $w_c(net_i)$

```
Initialize the sets S and U,there is only node of
  network net_i in S, and all target nodes u(j) in U, let
  the length increment of node ΔD(j) = 0;
Calculate the distance D(u(j), source) between each
  node u(j) and the node source in U, and let
  ΔD(j) = D(u(j), source);
while U is nonempty set do
    Find node with the shortest distance u_min in U,
      delete node u_min in U, and add it to S;
    Calculate the distance from each node u(j) in U to
      source by passing u_min:
      DP(u(j), u_min) = D(u_min, source) + D(u(j), u_min)
    if DP(u(j), u_min) < D(u(j), source) then
        D(u(j), source) = DP(u(j), u_min);
        ΔD(j) = D(u_min, u(j))
    end
end
Calculate the network length: w_c(neti) = ∑ ΔD(j) ;
Output: w_c(neti)
```

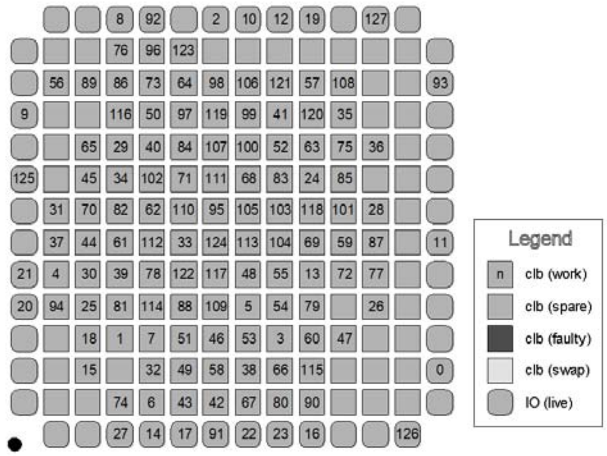


Fig. 5. The adaptive hardware used in the experiment.

migrated to the spare cell of coordinate $(6, 12)$. The configuration scheme at this time is called the initial feasible solution.

3) According to the K-Neighborhood structure, generate the switching node and the maximum neighborhood value $k_{max} = 5$ . Let the maximum number of external loops be $c_{max} = 10$ , the number of loops of the local search be $i_{max}$ at each iteration , and calculate the relative overhead of routing resources. As shown in Table I, the relative resource overhead of each iteration is lower than that of the previous iteration.
4) After variable neighborhood search, the replacement of bio-inspired self-repairing array is obtained as shown in Fig. 6. The light grey module represents the electronic cells that need to update the information of function nodes before and after configuration. A total of 10 electronic cells adjust the information of function nodes.
5) Reroute for changing network and communication mapping.

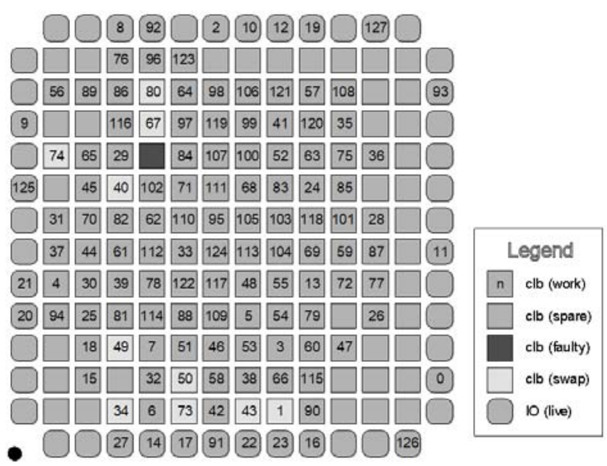


Fig. 6. Array placement in single cell failure mode.

TABLE I
RESULTS OF EACH ITERATION

| Iteration | Neighborhood parameter | The relative overhead of routing resources | Switching Nodes (Source cell coordinates ->target cell coordinates) |
|---|---|---|---|
| 1 | 1 | -42 | (6, 12)->(8, 3) |
| 2 | 1 | -38 | (8, 3)->(6, 10) |
| 3 | 1 | -16 | (6, 10)->(10, 7) |
| 4 | 1 | -28 | (10, 7)->(6, 5) |
| 5 | 1 | -6 | (6, 5)->(5, 7) |
| 6 | 1 | -8 | (5, 7)->(5, 5) |
| 7 | 3 | -14 | (5, 5)->(8, 11),(7, 4)->(10, 3),(8, 1)->(5, 1) |
| 8 | 1 | -30 | (8, 11)->(6, 8) |
| 9 | 1 | -16 | (6, 8)->(9, 4) |
| 10 | 1 | -26 | (9, 4)->(7, 5) |

### *C. Inject fault into the array randomly*

Fault nodes are generated randomly, and faults are injected into bio-inspired self-repairing hardware repeatedly until the spare CLB electronic cell modules are exhausted or multiple routing channels are congested, resulting in routing failure. In this experiment, 38 faulty cells can be allowed, as shown in Fig. 7.

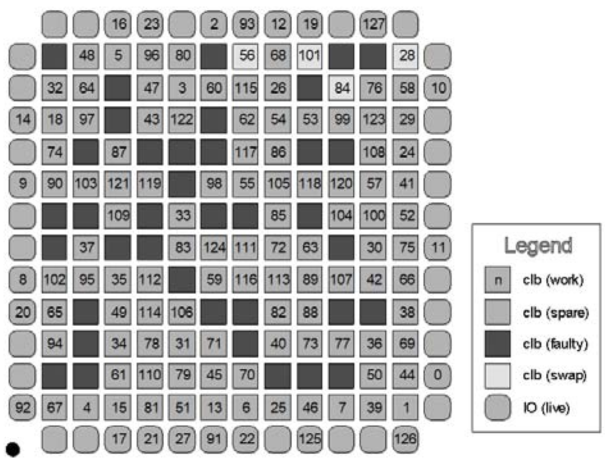


Fig. 7. Array placement in multi-cell failure mode.

### *D. Experimental results and analysis*

*1) Resource utilization:* Count the number of faults in each row, and the number of faults in a single row/column of electronic cells can be up to 5 times. If row-and-column removal mechanism or single cell removal mechanism is adopted, 5 columns (rows) of spare electron cells, i.e. 60 cells, need to be added. In the self-repairing mechanism proposed by Lala, the number of spare cells is not less than the number of working cells. The fault reconfiguration mechanism based on the self-routing algorithm proposed in this paper can make the best use of the spare electronic cells. Without considering the constraints, the number of spare cells is equal to the number of faults.

*2) Placement flexibility:* In the process of reconfiguration, the time cost $t_{vns}$ based on the variable neighborhood search algorithm:

The column removal mechanism or the single cell removal mechanism requires that each row or column must have enough spare cells. If the spare cells of the corresponding row or column are exhausted, the whole electronic array will not be able to achieve functional recovery. In the neighborhood replacement mechanism, there must be a certain number of spare cells around the function cells to replace the faulty cell. The above reconfiguration mechanisms depend too much on the placement of working cells and spare cells in the array and are not flexible enough. The proposed reconfiguration mechanism based on variable neighborhood search algorithm (VNS) in this paper does not depend on the placement of working cells and spare cells in the initial electronic array. When the working cells fail, the function of the array can be restored by adjusting the location and communication connection of some cells in the array.

*3) Resource overhead:* The resource overhead of the bio-inspired self-repairing hardware in the reconstruction process mainly include the wiring resource overhead and time overhead. In the initial layout of the electronic array, the reconfiguration mechanism based on the simulated annealing algorithm (SA) is often used to determine the position of each function cell. The idea of the algorithm is to update and optimize the solution by exchanging the position of the cells in the array. The algorithm can accept poor solutions at a certain probability and jump out of the local optimal solution, so it has better layout quality, but this method requires a long calculation time, it does not satisfy the real-time performance in the fault-repairing process. The proposed reconfiguration mechanism based on VNS only exchanges the positions of a small number of cells in the array during fault-repairing process, so it has a better real-time performance.

The working platform of this experiment is a computer equipped with Intel i7-3770 quad-core CPU, 16GB RAM and Windows 7 operating system. Taking the network resource consumption as the optimization goal, we obtain the wiring resource overhead and time overhead of the reconfiguration mechanism based on VNS and SA in each fault situation. The results are shown in Fig. 8 and Fig. 9.

As shown in Fig. 8, with the increase of faults number in electronic arrays, the wiring resource overhead of the reconfiguration mechanism based on VNS is gradually increasing, while the reconfiguration mechanism based on SA is less affected. And in Fig. 9, the reconfiguration mechanism based

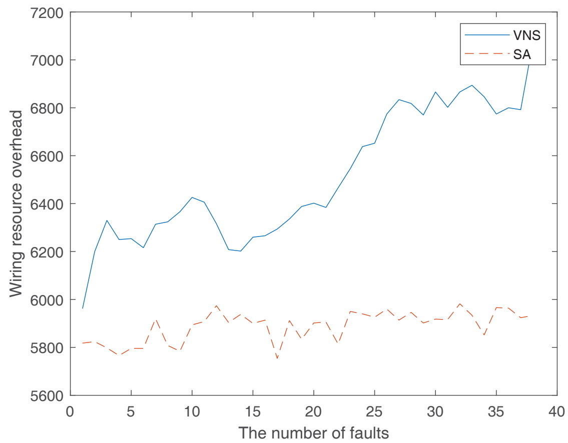


Fig. 8. Wiring resource overhead.

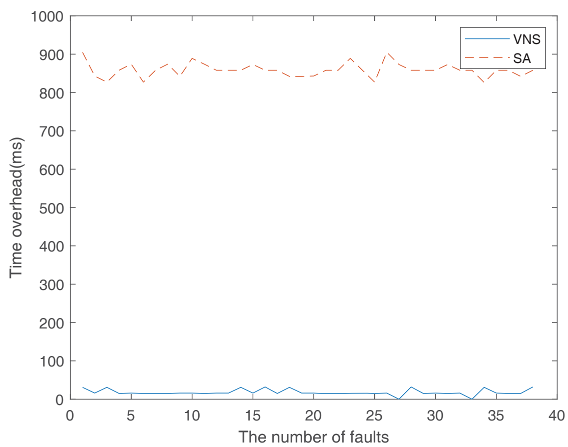


Fig. 9. Time overhead.

on VNS is much smaller than the reconfiguration mechanism based on SA. The reason is that the reconfiguration mechanism based on SA re-arranges the position of each cell by changing the original array placement on a large scale, while the reconfiguration mechanism based on VNS is to change the placement of the original cell array in a small range, so the resource cost of the reconstruction mechanism based on SA is relatively small and stable, but its time overhead is much larger than that based on VNS.

In summary, the reconfiguration mechanism based on VNS proposed in this paper is more flexible than column removal mechanism, single cell removal mechanism and the neighbor replacement mechanism. It can make full use of the spare cells in the array to repair, and has faster reconfiguration time than the reconfiguration mechanism based on SA in a certain number of faults. It ensures the real-time performance of array self-repairing.

## V. Conclusion

In order to overcome the shortcomings of current bio-inspired self-repairing hardware reconfiguration mechanism, such as inflexibility, low utilization of spare cells, inability to take both resource consumption and self-repairing time into account, this paper proposed a reconfiguration mechanism based on variable neighborhood search algorithm to reconfigure the fault array and restore the function of the array with little change in the original placement. Through experimental verification and analysis, the proposed reconfiguration mechanism has better flexibility, can effectively improve the utilization of spare cells, and can take into account the wiring resource overhead and reconfiguration time under a certain number of faulty cells. In the future, we need to do further research and optimization on the reconfiguration mechanism in the case of multiple faults.

## Acknowledgment

This research work is supported by National Natural Science Foundation of China under Grant No. 51675523.

## References

[1] R. E. Lyons and W. Vanderkulk, "The use of triple-modular redundancy to improve computer reliability," *IBM Journal of Research and Development*, vol. 6, no. 2, pp. 200–209, 1962.

[2] D. Mange, S. Durand, E. Sanckez, A. Stauffer, G. Tempesti, P. Marchal, and C. Piguet, "A new paradigm for developing digital systems based on a multi-cellular organization," in *Proc. ISCAS'95 - Int. Symp. Circuits and Systems*, vol. 3, Apr. 1995, pp. 2193–2196 vol.3.

[3] D. Mange, E. Sanchez, A. Stauffer, G. Tempesti, P. Marchal, and C. Piguet, "Embryonics: a new methodology for designing field-programmable gate arrays with self-repair and self-replicating properties," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 6, no. 3, pp. 387–399, Sep. 1998.

[4] C. Ortega-Sanchez, D. Mange, S. Smith, and A. Tyrrell, "Embryonics: A bio-inspired cellular architecture with fault-tolerant properties," *Genetic Programming & Evolvable Machines*, vol. 1, no. 3, pp. 187–215, 2000.

[5] C. Szasz and V. Chindris, "Development strategy and implementation of a generalized model for FPGA-based artificial cell in bio-inspired hardware systems," in *Proc. 5th IEEE Int. Conf. Industrial Informatics*, vol. 2, Jun. 2007, pp. 639–643.

[6] P. K. Lala and B. K. Kumar, "An architecture for self-healing digital systems," in *Proc. Eighth IEEE Int. On-Line Testing Workshop (IOLTW 2002)*, Jul. 2002, pp. 3–7.