PARALLEL MULTI-PURPOSE OPTIMIZATION WITH MODIFIED FAST MESSY GENETIC ALGORITHM

Pavel Kostka, Zbyněk Škvor
Dept. of Electromagnetic Field
Czech Technical University
Technická 2, Prague 6
Czech Republic

xkostka@fel.cvut.cz, skvor@fel.cvut.cz

Abstract

Fast messy genetic optimisation is found suitable for complex microwave circuit design. Increase in computation speed is achieved using several ordinary computers connected to a network. Calculations are running on background so that computers can be used for other purposes at the same time. When applied to microwave circuits, a modification to previous genetic optimisation methods proved suitable. Dynamic change of bounds has significantly improved convergence rate. The new method has found global minimum in each run, while classic methods failed for some starting points.

Keywords

fast messy genetic algorithm, parallel, optimisation, multi-purpose

1. Introduction

Circuit optimisation became one of the applications most used by microwave engineers. Due to the nature of microwave circuits, the optimisation task is far from easy. Error functions mostly suffer from local minimum problems, so that in many cases the optimisation gets unbelievably inefficient.
During larger microwave structure optimisation local iteration methods tend to fall into local minimums. Hence, the new - global - methods like simultaneous annealing, taboo method or genetic algorithms are being used lately. The more variables we try to tune, the bigger state area gets and the optimisation on a single machine becomes rather slow. For overall speedup, several computers connected to each other are used.

The genetic algorithm (GA) is a stochastic global search method that mimics the metaphor of natural biological evolution. GAs operate on a population of potential solutions applying the principle of survival of the fittest to produce (hopefully) better and better approximations to a solution.

2. Fast Messy Genetic Algorithm

Fast messy genetic algorithm (fmGA) is a special clone of common simple genetic algorithm (GA). This type represents new, more powerful kind in the GA branch. It resists premature local-minimum fall and solves problems in shorter time.

A gene is represented by a pair *(allele locus, allele value)* in messy algorithms, so that – for instance – chromosome (2,0),(0,1),(1,1) represents 3 bit-long chromosome 110. Operation cut and splice are used to bread new offspring. Incomplete specification of the chromosome (underspecification) or redundant specification (overspecification) could occur during evolution. Overspecification is solved by right-to-left scan, i.e. the only first occurence of an appropriate pair is taken into account. For example: ((0,0), (2,1), (1,0), (2,0)) represents chromosome 001. On the other hand, underspecification is harder to deal with. We must complete the chromosome in order to be able to evaluate its fitness function (FF). Hence a template is used – see Fig. 1.

The algorithm works in two main iteration cycles – in the inner and outer loops. Each new start of outer loop is called as an era. The change of building-block order together with inner cycle launching is the main aim of this cycle.

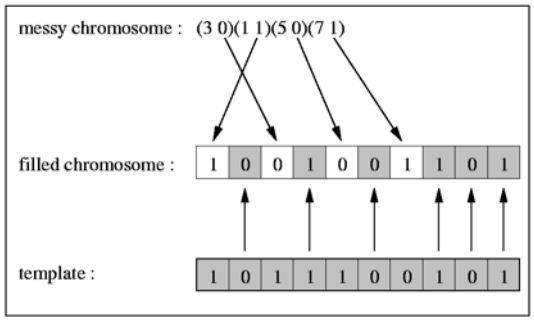

Fig. 1 Representation of a solution – chromosome.

Inner cycle consists of three phases:

1) initialisation
2) building block filtering phase
3) juxtapositional phase

Detailed description of the algorithm could be found in [1], [3] and [4].

3. Improvements

We have decided to use this algorithm for microwave circuit optimisation. In order to save time, circuit analysis and Fitness Function (FF) evaluation has been carried out using the source code of a microwave CAD program called MIDE (see [2]). Implemented algorithm is based on [1] with the following important modifications:

3.1 Parallel multitasking

The bottleneck of any common algorithm is in the time consumption of FF evaluation – even a simplest circuit gets evaluated hundreds to thousand times per second. If one uses almost any global search method (and GAs specially) hundreds of thousands evaluations are required. The local area network has been used and the task has been divided to many common workstations. The application has been running under multitasking environment so that computers could be used for ordinary tasks like writing reports or playing games. At no extra cost, performance exceeding any single PC has been made available.

3.2 Implementation

Parallel computation machine was implemented using common client-server model. The first idea led to launching single fmGA algorithm on the server and then distribute appropriate variables together with optimising circuit to the clients. Clients then perform evaluation (compute the solution) and return result to the server. In order to minimize the total network overhead we suggested batch processing (a bundle of waiting states are together send to the client and then – after evaluation – the same bundle of solutions are returned).

3.3 Quantization-step choice

During the process of FF discovering we encounter the problem with quantization. This process is comparable with converting analog signal to its digital form. Let's look at trivial example: let change FF progress according to figure 2 (i.e. FF is depends on only one variable). As one can see there exist three local minima in points 39, 68 and 101. Last mentioned point represents global minimum. Figures 3, 4 and 5 show FF curve under 4-, 5- and 6-bit sampling. Global minimum can be successfully discovered only in the last case (6 bits per variable). The others sampling rates are incapable of doing the same.

We can result from such example easy theorem: the more bits-per-variable we choose the better results we can expect.

In these days, when computers' performance reaches hundreds of MIPS, using of sampling with tens of bits per variable is quite common. After all, analyses of the time profile of the one iteration show that genetic operations consume at most 10% of total time.

However resolution we choose we can't prove that such adjustment is capable of true global minimum discovering (i.e. we can't prove that the minimum we have just fallen into is not only at the edge of a very steep "FF crater" where we can get into with a bit higher resolution).

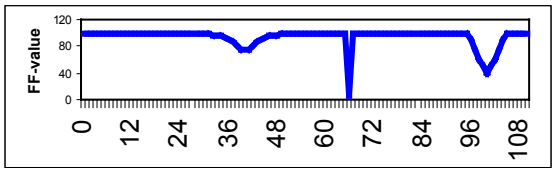

Fig.2. An example of 1D optimisation problem (1 variable); This curve is object of quantization process for purposes of computer processing.

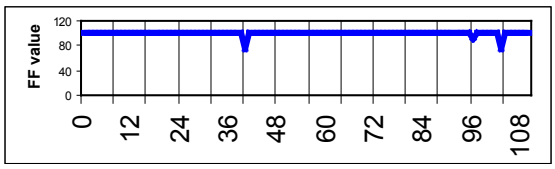

Fig. 3 Quantization process -16 steps

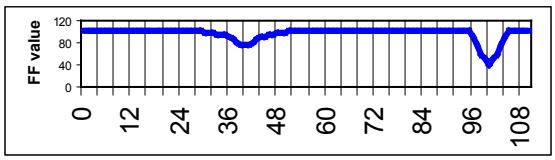

Fig. 4 Quantization process - 32 steps steps

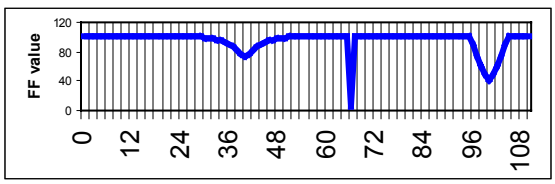

Fig.5 Quantization process: 64 steps

In spite of the above-mentioned consideration, the practical implementation brings very clear question: "How many bits are consequently sufficient?" We have incorporated new method to the structure of fmGA in order to solve satisfactorily this problem.

3.4 Mapping and dynamic change of bounds

Previous paragraph described possible reasons of failure during searching of global FF minimum among many local ones. This problem is one only one we hint at. Let us demonstrate one more imperfection (see figure 2). If we perform linear sampling at whole interval (what is necessary because of the lack of the knowledge of FF curve) we will obtain subinterval 0-30 where no minimum exists. So-spent bites uselessly occupy space at the expense of the rest of the interval (the remaining interval can be re-sampled with higher resolution with the same number of sampling bits).

That's the next argument that (with the other ones) led us to map variable via some additional layer during sampling process.

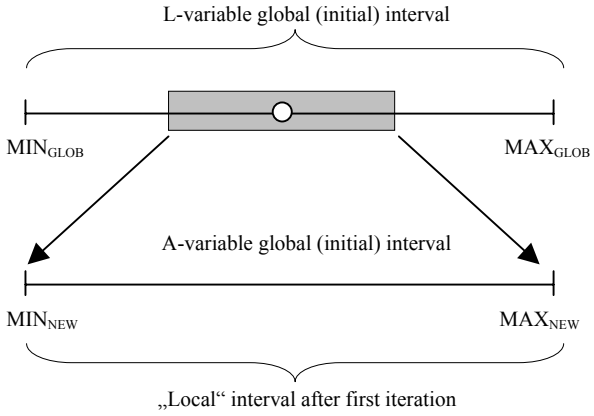

Fig. 6 "Local" interval after first iteration

Consider a variable on a constrained interval and let optimise this variable (see fig. 6). We can regard it (for explanation purposes) as the variable called L that stands for inductance of a real inductor. We estimate that desired solution must lie within 100pH - 100nH (the inductor simply can't be fabricated with values outside this interval).

Let's call such boundaries as global bounds. These bounds must be known for quantization process. Let's perform a quantization and launch the optimisation process. First, the algorithm will try to discover areas where some local minima lie (first run of the algorithm). Second, after finishing such first stage, program will mark the most promising (the best so-far found) minimum (i.e. a minimum with the lowest FF value) and recalculate bounds of area we perform searching according to equations 1 and 2.

$$MIN_{NEW} = \max(MIN_{GLOB}, FF_{BEST-F} - \frac{MAX_{OLD} - MIN_{OLD}}{4}) \quad (1)$$

$$MAX_{NEW} = \min(MAX_{GLOB}, FF_{BEST-F} + \frac{MAX_{OLD} - MIN_{OLD}}{4}) \quad (2)$$

MIN_{GLOB} a MAX_{GLOB} denote global bounds, (i.e. bounds, that user had entered before launching the application), FF_{BEST-F} stands for the best local minimum discovered so far, MAX_{OLD} a MIN_{OLD} describe bounds from previous run of optimisation (see below) and symbols min a max are functions (min (0,1)=1; max (10,20)=20).

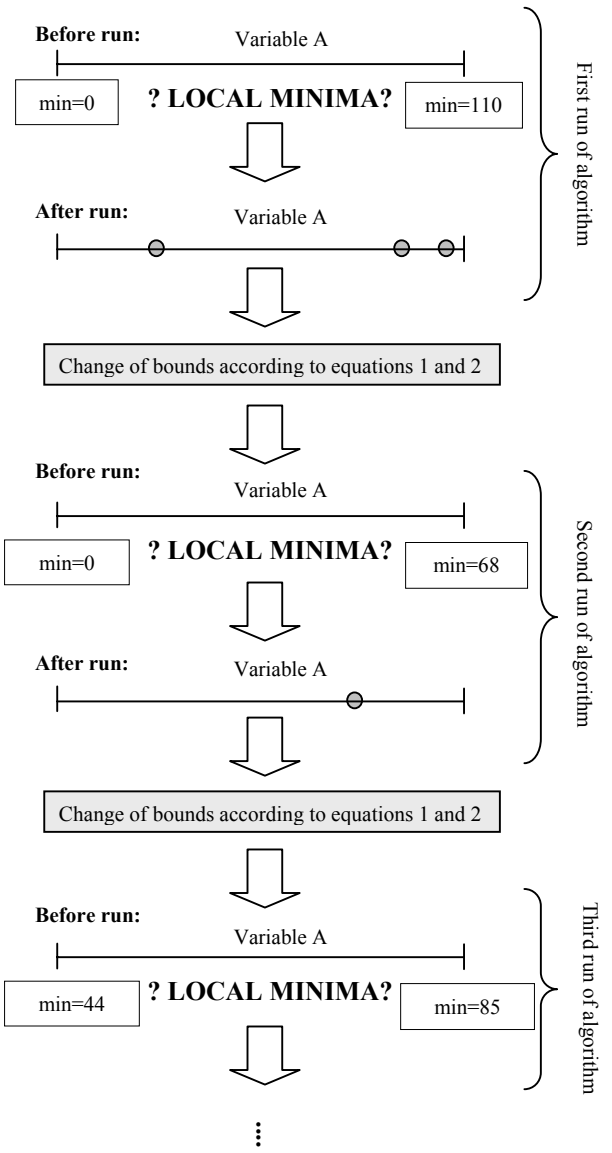

Fig. 7 Example of fmGA mapping process

Let's illustrate whole procedure with the following example (see fig. 7):

Variable A is quantizied with 4-bit resolution, i.e. 16 possible states (2^4). Global bounds are set within 0-110. First run of algorithm minima 40, 85 and 105 are discovered. The best-found minimum is chosen (imperfect 4-bit resolution causes that the best one is minimum 40). Now, bounds are changed so that the position of this minimum should be (if possible) centred and the range of bounds twice reduced (see equations 1 and 2) and new round of optimisation process is launched.

Such change helps us to discover better local minimum (point 66), that is in the last step marked as global... On the other side, this process can lead to the paradoxical situation, where periodical repetition will cause locking in a local minimum. That's the reason why the algorithm "soften" bounds as long as the FF goes rapidly down. Otherwise one or more steps are rolled back.

3.5 Working with real numbers

Above-mentioned mapping procedure can be used while working with real numbers. First, let me shortly remind representation of decimal integers and real numbers in binary system (see tab. 1 and fig. 8).

While representation of integers in binary system is the same as in decimal one – i.e. it is performed one digit place by another, the real numbers are represented differently. First of all, each number is normalized (i.e. its format is modified as: [sign; mantissa; exponent]) and after it transferred into binary form.

Developer 's aim was quite clear – they tried to reach the best resolution as possible at given bit rate.

However, such representation makes troubles to common simple GAs while the process of sampling. Change of only one bit of exponent can represent huge change of variable value. The change can lead to leave hardly found local minimum. Yes, it's effective method that helps us recovering from local minima, but it also degrades the whole algorithm to almost random search method (each new crossing-over brings very different results; these results are either rapidly worse and then they are eliminated or the better ones – but they are discovered more or less by chance).

0	1	2	3	4	5	6	7
000	0001	010	011	100	101	110	111

Tab. 1. Representation of decimal integer in binary digit system (3 bits number).

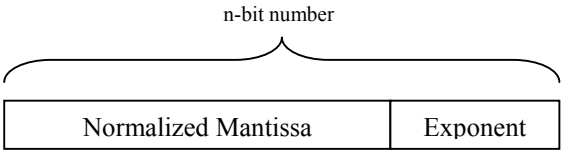

Fig. 8. Representation of real number in binary digit system.

Mapping method such disproportion effectively eliminates. Interval is linearly divided (quantizied) and the searching method doesn't differ from the one used in example that works with integers. Consider fig. 9 as an example: 3 bits are available for one (real) number; minimum is 3.05, maximum 14.56.

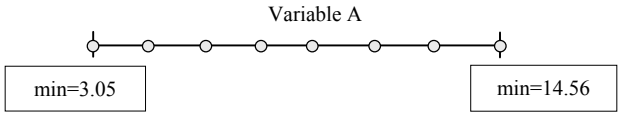

Fig. 9 Real-number mapping

Other points are linearly interpolated according to the equation 3 (results are in table 1).

$$STEP_N = MIN_{REAL} + N \cdot \frac{MAX_{REAL} - MIN_{REAL}}{N_{COUNT} - 1} \qquad (3)$$

$STEP_N$ denotes particular quantization step, MIN_{REAL}, MAX_{REAL} stands for minimum a maximum of actual interval, N_{COUNT} number of quantization steps.

K_{POS}	I_{VALUE}	K_{POS}	I_{VALUE}
000	3.05	100	9.63
001	4.69	101	11.27
010	6.34	110	12.92
011	7.98	111	14.56

Tab. 2. Result of interpolation

K_{POS} denotes quantization position, I_{VALUE} is interpolated value.

This example clearly shows that described method is versatile and it is usable for all kinds of real numbers (single, real, extended) – i.e. it can be used for arbitrary length of the pair [mantissa; exponent].

3.6 Results

We intend to construct a robust method, that would be able to found a solution in all cases (if a suitable one exists) apart from the starting point. Even such assumption looks quite easy, there are countless number of problems where classical methods fail. As an example that is easy to understand but not rather easy to optimise, we chose a textbook optimisation problem, see Fig. 10.

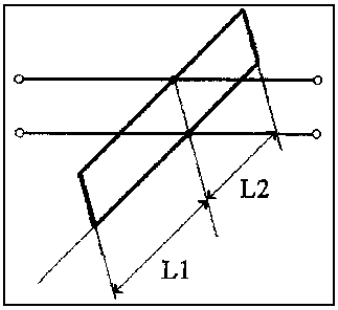

Fig.10. Test case – two shorted stubs in parallel to a line

The task follows: find electrical lengths L1 and L2 of two short-ended stubs, connected to a transmission line in parallel, so that the transmission coefficient would be lower than –25 dB between frequencies 6 GHz and 6.2 GHz. Simultaneously we want to achieve transmission better than -0.5 dB at 3 GHz. These demands are schematically shown in Fig. 11.

In the case the lengths are to be found in a range between 1 and 180 degrees for both stubs, the task is quite easy and any optimisation method works. To obtain a good test case, we try to find the optimum in a wider range, say 1 – 1000 degrees. FF shape then exhibits a number of local minima. The reason is obvious – electrical character of both stubs is repeated every 180 deg. Global minimum discovery in such selected area is almost impossible for common optimisation methods – see Fig. 12. Typical simplex methods will mostly find a local minimum, however, FF is always bigger than 0 (not all goals are met). A few examples of such solutions are shown in Fig. 13.

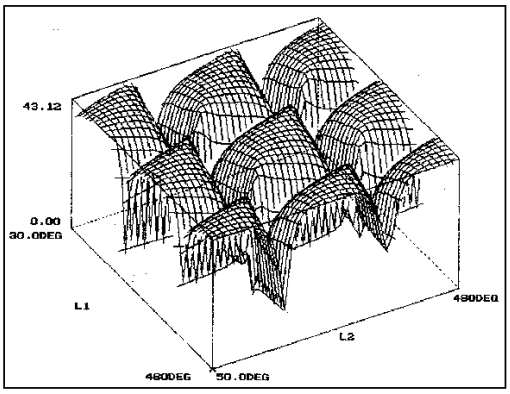

Fig. 11 Fitness function surface plot shows a number of local optima.

As can be observed from these examples – classical methods are capable of discovering minimum sometimes – dependable on starting (mostly randomly generated) point. Our experiments proved that in only 7 of 10 performed simulation runs classic methods discovered minimum.

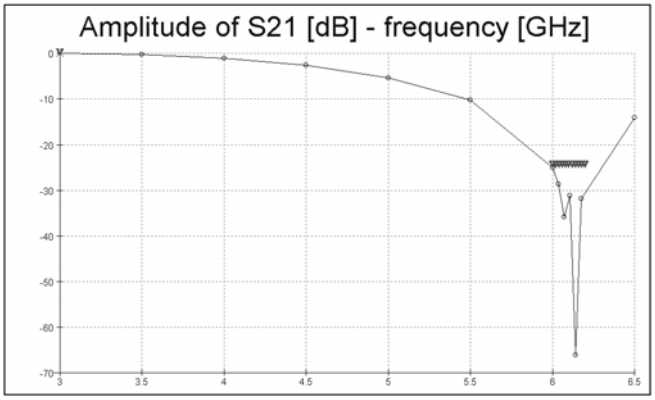

Fig.12. Forward transmission for correct solution (approx 175 and 178 degrees @ 6GHz

Using the new method, we started from the same point (we filled up variables with the same start values similarly to the above mentioned optimisation method) and carried out several optimisation runs. In all runs the global minimum has been found.

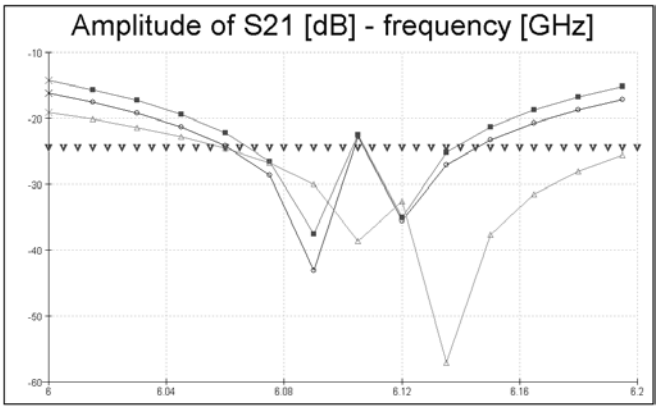

Fig.13. Some of the false solutions produced by ordinary optimisation methods.

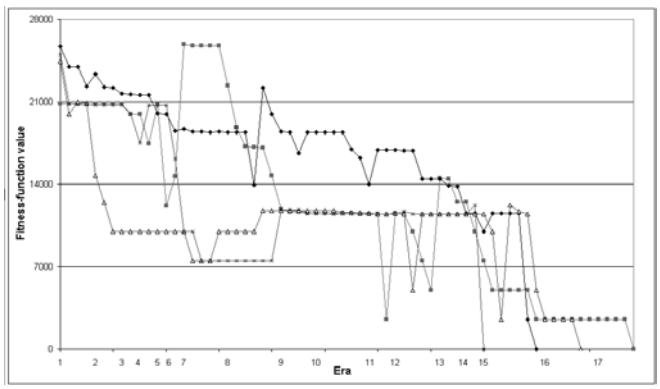

Fig.14 Fitness function plotted against eras.

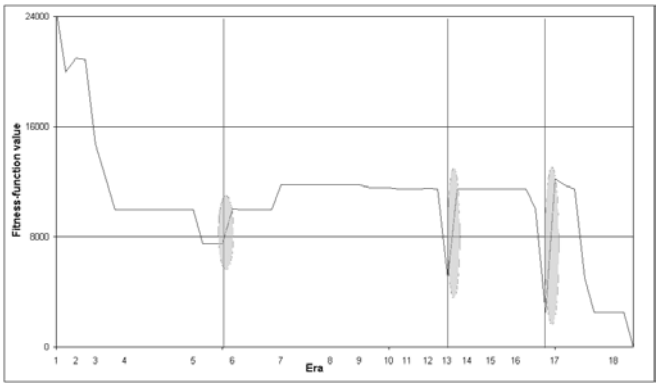

Fig.15. Detailed sketch of solution evolution describing method behaviour in local optima

The process is illustrated at Fig. 14. Notice one important feature: during the launching of a new era, the best so-far found chromosome is copied into template. It results in temporary degradation of FF (you can see bumps in the graph), however, in a few moments a new – often better - solution is found. Such iterative progress is typical for messy algorithms.

Figure 15 shows effect of dynamical change of bounds. It arises at the end of 6^{th}, 12^{th} and 17^{th} era (vertical red lines). Please note, that there are significant temporary FF degradations, but the local minima are left (ellipse areas).

3.7 Parallel fmGA server

The new method combines both mentioned improvements together: clients no more behave only as silly slaves (whose only task is fitness function evaluation) but each of them runs its own fmGA algorithm. Server side must solve the area division problem. Server can than send to each client only some sub-bounds of original problem. Each client receives different bounds and thus totally independently tries to discover solution(s) in its own sub-area.

3.8 Area division problem

Division of an 1-D area is obviously the simplest task – see Fig. 16. Appropriate areas are a little bit overlapped. It was found useful in cases when the right solution lay near the border of a sub-range.

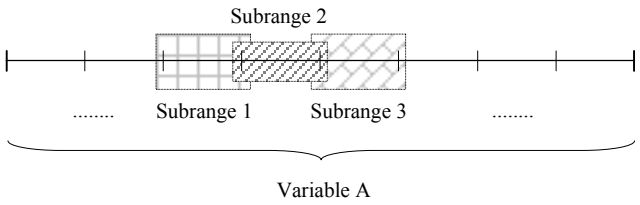

Fig. 16. Example of division and overlapping in 1-D area.

Situation appears rather harder for n-dimensional area (where N>1). Of course, we can perform the same division, but the less clients we have the coarser the division is (especially in case of many variables with large bound limits). A right solution can be discovered with much less probability. Keeping the uniform division step for all variables is a better technique, however we must cope with the situation that we create more intervals than cooperating clients. The above mentioned reasons made us to implement so-called *computational stack*, i.e. a structure that temporarily saves areas which wait for exploration – see Fig. 17 and 18.

As soon as the stack is empty (and no suitable solution has been found), results from all sub-areas are compared together and area bounds are changed so that they more tight surround so-far best found solution (i.e. dynamical change of bounds). If there are more than one area, where partial results are comparable, all of these areas are processed – one is explored and the others are saved into stack for later exploration.

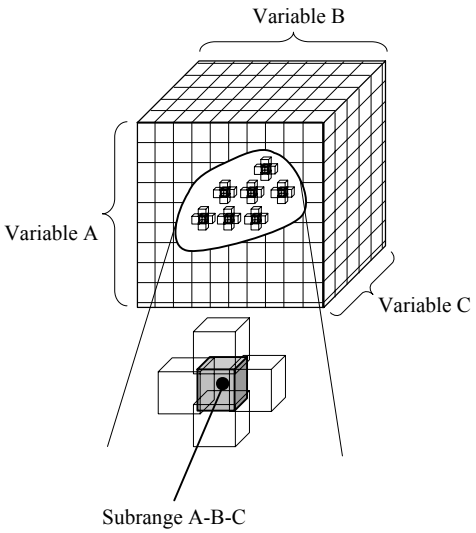

Fig.17 Example of division and overlapping in 3-D area.

3.9 Gradient method incorporation

First experiments of suggested method shown that time spent on discovering near neighbourhood of FF minimum and time spent on falling into the minimum are for algorithm fmGA almost comparable. I.e. the algorithm is capable of discovering the surrounding of solution in quite large search space, but at the present there exist better algorithms (typically gradient methods) for quick discovering of *FF - peek*.

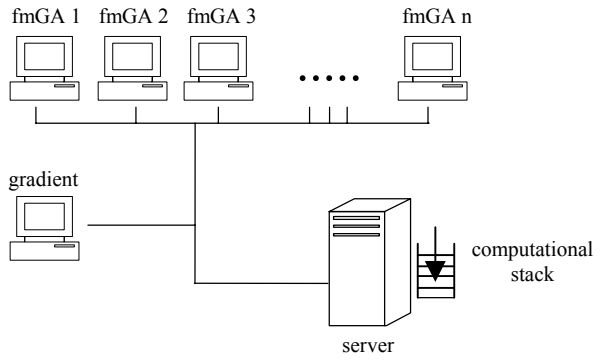

Fig.18 Sketch of parallel computation engine.

3.10 Conclusions

Fast messy genetic optimisation is a promissive method found suitable for complex microwave circuit design. High computational load can be overcome at nearly no cost using several ordinary computers connected to a network. Running calculations on background enables other users to do their jobs at the same time.

A modification to previous genetic optimisation methods, suitable for microwave circuits, proved to work and significantly improved convergence rate.

Above mentioned example of two radial stubs is only one of the tasks that the method is currently used for. We have used it because of its simplicity. The method seems to be very useful in cases, where an equivalent circuit of some real problem is known but the circuit has many unknown parameters.

Acknowledgements

This research and publication have been sponsored by Czech Grant Agency, contracts no. 102/01/0571 and 102/01/0573, and by Czech Ministry of Education in the frame of project MSM 210000015.

References

[1] KNJAZEW, D.: Application of the Fast Messy Genetic Algorithm to Permutation and Scheduling Problems, Illigal Report 2000022, University of Illinois, May 2000, available also at http://gal4.ge.uiuc.edu/pub/

[2] SKVOR, Z.: CAD pro vf. techniku, textbook, Czech Technical University, Prague 1998.

[3] Goldberg, E. – Deb, K. – Kargupta, H. – Harik, G.: Rapid, Accurate Optimisation of Difficult Problems Using Fast Messy Genetic Algorithms, Illigal Report 93004, University of Illinois, February 1993.

[4] PELIKAN M. – SASTRY, K. – GOLDBERG, D.: Evolutionary Algorithms + Graphical Models = Scalable Black-box Optimisation, Illigal Report 2001029, University of Illinois, November 2001.

About authors...

Pavel KOSTKA was born in Prague in 1977. In May, 2000 he graduated from Czech Technical University (CTU) in Prague and in these days he is approaching the finish of his PhD. studies at the department of Electromagnetic field of the same university. Optimization of microwave structures, parallel computation and algorithm-migration processes are his main points of interest.

Zbyněk Škvor graduated from CTU in Prague, 1985. He is with the Department of Electromagnetic Field, CTU Prague. His field of interest includes numerical electromagnetics, CAD for radio-frequency and microwave circuits and microwave measurements.