Optimization Module for Abaqus/CAE based on Genetic Algorithm

K. Szajek[1], W. Kakol[2], T. Lodygowski[1] and M. Wierszycki[1,2]

[1]Institute of Structural Engineering, Poznan University of Technology, Piotrowo 5, Poland
[2]BudSoft, 61-807 Poznan, Sw.Marcin 58/64, Poland

Abstract: *Genetic algorithms have become one of successful tools in design and topology optimization. The optimization module based on genetic algorithms was developed and employed in Abaqus/CAE by GUI and kernel scripting. The new module extends advanced functionality of Abaqus/CAE allowing to perform optimization directly in Abaqus Unified FEA product suite from SIMULIA. The genetic algorithms implemented in optimization approach are based on available GPL libraries. Significant improvement in evolving into optimal solution can be achieved when genetic algorithms are combined with neural networks which one can train by running Abaqus jobs, and substantially improve the efficiency of computations. In the paper the shape optimization problem of a tooth implant will be presented and discussed in detail. The particular Abaqus features useful in this application will be highlighted, as well. The presented approach seems to be extremely efficient in parallel computations.*

Keywords: *Design Optimization, Optimization.*

1. Introduction

Genetic algorithms (GAs) have received wide popularity as optimization techniques during last decades and can compete successfully with the gradient-based approaches in many areas (Goldberg, 1989, Burczynski, 2004). GAs are stochastic search approaches which rely on the principle of the survival of fittest in natural selection. Unlike conventional optimization techniques GAs explore simultaneously the entire design space and therefore is likely to reach the global minimum. Improvement of global search process can be performed by incorporating in optimization neural networks (NN) which can learn and adapt changes over the time. In general, GAs require a lot computations (structural analysis in our case) and therefore high performance computing addresses ideally its needs, especially when combined with NN.
In the paper the process of optimization with the use of FEA, genetic algorithms and neural network is discussed. For a given criteria, based on finite number of solutions, better FE model is proposed. The existing open source libraries have been used: Galileo (for GA) and ffnet (for NN). The whole optimization procedure was implemented with the use of Python scripting language. The FE model, numerical analysis and post-processing of results were performed with the use of the Abaqus Unified FEA product suite from SIMULIA. The integration of GA and NN libraries with FE tools was done by using the Abaqus Scripting Interface (ASI). The optimization was first performed solely with GA. The GA was used for fitness evaluation of a set of results obtained

from numerical analysis. A fitness evaluation of each FEA structural analysis solution can be done with any tool which can return fitness value of a solution based on a genotype. At the beginning of each iteration of optimization loop a new population is defined. Genotypes created during genetic process are the starting point of the evaluation mechanism. For each genotype a numerical model is created, the analysis is defined and performed. The obtained results are graded according to a defined objective function. The presented approach is very general and can be used to optimization of any Abaqus model. However optimization of more complex problems requires modification of this algorithm. The assumption that GA does not demand a precise solution for each genotype is fundamental and the basis of modification. The crucial task of evaluation mechanism is extracting features of a genotype which improves the quality of the individual. Thus, it is recommended to use an estimation tool which can evaluate the fitness less accurately but in a faster way. In the second presented approach of optimization, the estimation based on NN is applied. In order to train NN, randomly generated genotypes and numerical analysis results were used as a training set. The training set can be obtained with the use of parallel computations. The modified algorithm reduces the key disadvantages of GA. Moreover, in this approach, each expensive FE analysis is used for the purpose of improving a mechanism of estimation – NN. An minimal number of FE analyses is determined by changing optimal solution in subsequent iterations.

2. Optimization Module

The algorithm for performing optimization in Abaqus was implemented using python environment, which is built in Abaqus/CAE. The Abaqus Scripting Interface (ASI) was used to communicate the new module with Abaqus/CAE. The data exchange between an user and Optimization Module (OM) is performed with Graphical User Interface (GUI). The GUI is implemented as an independent module (Figure 1).

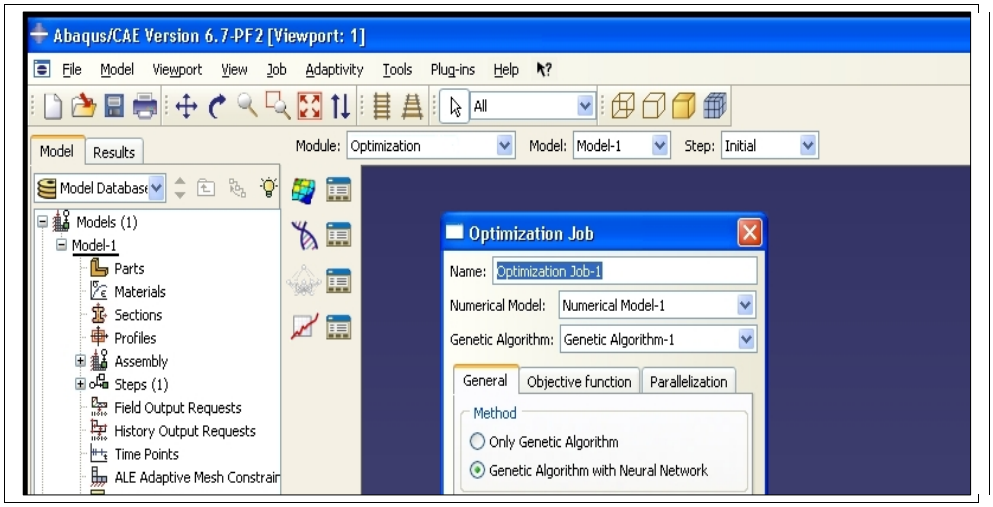

Figure 1. Main Optimization Module dialog box

ASI provides a very comfortable interface to control all data associated with a defined Abaqus/CAE model. It allows to use python scripts in order to redefine data in a desire way. This functionality allows an user to check a number of solutions for given configuration of design variables. The simplest idea of optimization bases on numerical models for randomly generated design variables. It is the simplest way but requires a large number of analyses and hence it is effective only when the dimension of a solution space is small. It is better to use a tool which can infer basing on a finite number of analysis results and propose a candidate solution deliberately instead of randomly. Such a tool should find relation between design parameters and objective function values. In Optimization Module for Abaqus a Genetic Algorithm is used. For large design problems which require heavy computations the usage of neural network as a estimation tool for genetic algorithm individuals is proposed.

2.1 Module structure

The hierarchy of OM objects is presented in Figure 2. The OM objects are organized in Tools group and Optimization Job. The Optimization Job is the key object which links tools elements and controls a whole optimization process. The tools (NN, GA, NM) are stored and managed with the use of CAE manager dialogs.

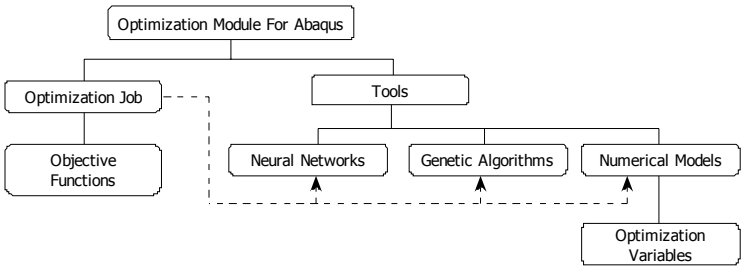

Figure 2. Object hierarchy structure of Optimization Module for Abaqus

2.2 Numerical Model object

The Numerical Model (NM) object is an interface to Abaqus/CAE model. Each NM object refers to a set of design variables. The most important attributes of NM object are design variables. Each variable represents only one value from Abaqus/CAE model. A special attention should be given when defining a range of variables in order to avoid unfeasibly configuration of design parameters. Variables can be of various types: part sketch parameters, material or amplitude data value, profile dimension etc. For GA a real variable value is encoded in binary form. The more bit number the better accuracy can be obtained. Unfortunately increasing bit number moderates optimization process. The set of design parameters is a basis of model modification. After running a rebuilding process, each design variable object generates and runs a macro in order to change a model value. An user can not change automatically generated macros but it is possible to define an additional one. An user defined macro is run at the end of modification process and gives a chance

to control meshing, detects unfeasible solution etc. In the case of rebuilding (regeneration) error raised by Abaqus/CAE, information is stored and such a model is excluded from further calculations. The rebuilding error is the information about unfeasible set of design parameters. After rebuilding process a new job is defined. A job object is created based on a current Abaqus/CAE model. For each defined job input file is generated. By reason of servicing parallel computations a given number of jobs and input files are created in turn. The program waits until all input files are created. The Abaqus solver is run then in independent thread in order to carry out all analyses simultaneously.

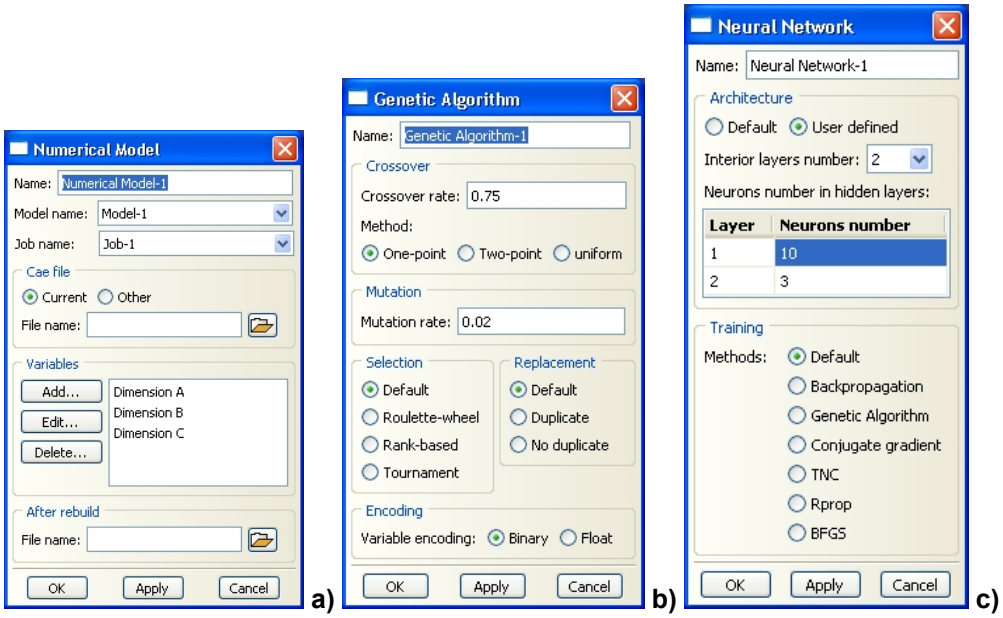

Figure 3. Tools object definition window

2.3 Genetic Algorithm object

The Genetic Algorithm (GA) object defines a genetic optimization process. The most important advantages of GA over classical methods of optimization is that it works basing on problem solution instead of analytical relation, and can optimize linguistic variables and uses parallel computations by nature. GA operates on individuals which are abstract representation of a real solution. Each individual consists of a set of encoded design variables, which is called a genotype. The values usually are binary encoded but other types are also possible. The main idea of GA approach is creating a set of initial solutions and using evolutionary operators to improve them in successive iterations, called generations. An initial set of individuals, called a population, is usually randomly created. A key parameter defined by an user is a size of first population. A number of initial individuals has to be large enough to proper covering a space solution and to guarantee a high level of diversity. Every population is subjected to evaluation. Evaluation process consists in assigning a fitness value to each individual. The fitness value describes a solution

quality and is calculated according to an objective function. Evaluation process uses Numerical Model (NM) object in order to modify Abaqus/CAE model according to a genotype, to create a job, to generate an input file, and to run analysis. After completion NM reads results and transfers them to the objective function as arguments. Fitness values are the basis of selection process. Individuals are chosen from population by selection which creates a new one set. Statistically, only the most fitted solutions are chosen in order to allow them taking a part in reproduction. Selection mechanism watches over improving a next population quality. After termination of selection, reproduction process starts. The reproduction based on selected individuals uses their genotypes in order to create a new design parameters configuration. The Optimization Module for Abaqus uses two reproduction operators: crossover and mutation. The primary function of crossover is mixing parent individuals genotypes and creating a new couple of children one. The second reproduction operator, mutation, is responsible for distorting randomly genotypes. Random distortion prevents from loosing proper genotypes configuration in the consequence of crossover and helps to escape from a local minimum. In the last stage of GA individuals are created as a result of crossover and mutation process replacing previous population. Depending on a chosen type of replacing either previous generation is changed on a new one or both are joined, sorted by fitness value and only the best individuals create a next population. The GA ends when either maximum fitness value for the best fitted individual is obtained or maximum number of generations is achieved. Both values are specified by an user.

2.4 Neural Network object

The Neural Network (NN) object simulates an artificial neural network processing. Generally, a NN is a mathematical structure to signal processing. In engineering field it can be used as a non-linear statistical data modeling tool, as well. The module takes advantage of NN to model complex relation between design variables (input data) and a objective function value (output data). In GA+NN type of optimization the NN is used as an individual estimation tool.

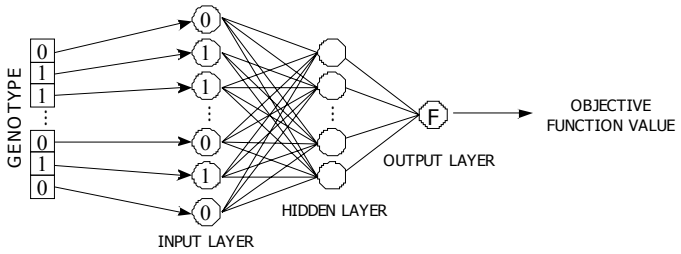

Figure 4. Neural Network structure

A Neural Network consists of interconnected artificial neurons, which are programming elements that mimics the properties of biological neuron. All neurons are organized in layers. The first one is called an input layer and the last one an output layer. Between output and input layers there can be hidden layers. Processing signals are modulated according to weights, which correct signals before they enter neurons. As a result an estimation output value is obtained. The weights are matched during a training process. Basing on input and output collection data, the weights are corrected in order to minimize square error between calculated and obtained output data for a

whole training set. Minimized square error is called learning error. There is a few training methods available in the program (Figure 3). Due to processing genotype, the number of neurons in input layer is equal to genotype length. The output layer consists of only one neuron which refers to an objective function value. Presence of hidden layers improves a NN possibility of modeling more complex problems. On the other hand, too extended hidden part of NN in relation to a size of training data set makes NN processing worst. The number of hidden layers and neurons can be specified by an user, however, it is not recommended. By default, the program matches NN architecture based on learning error. Moreover, an user can choose an Optimize Neural Network option to rebuild NN architecture during optimization process accordingly to training data set. In order to find the most accurately layers and neurons configuration a genetic algorithm is used. The best NN configuration is characterized by the lowest learning error.

2.5 Optimization Job object

The Optimization Job (OJ) object is the key element in the module. The OJ links all objects and organizes an optimization process. There are two methods of optimization available in the module. The first one is based solely on Genetic Algorithm object whereas the second incorporates Neural Network object, too. Objective function definition is a crucial point for an optimization processing.

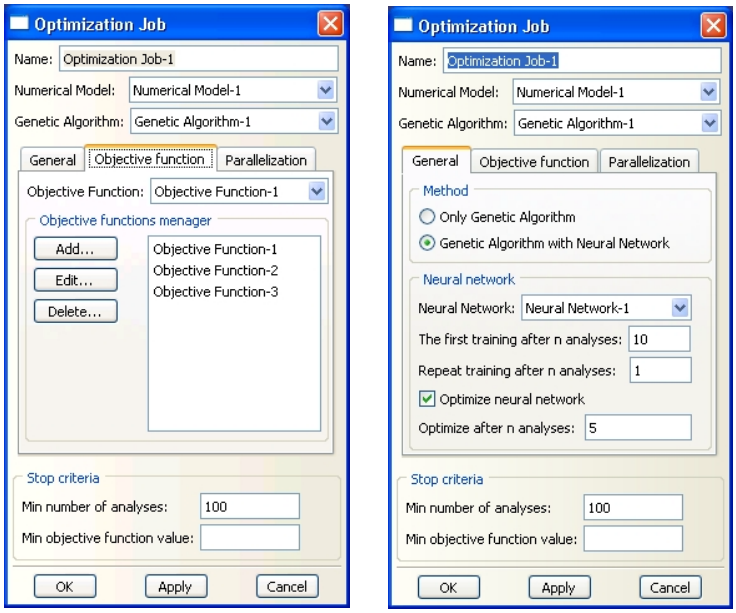

Figure 5. Optimization job object definition window

The OJ object allows an user to define more than one objective function, however only one is taken into consideration during an optimization. The objective function requires variables definition. Each variable has an unique name and refers to only one value from odb file. Every

variable has to have a specified step, frame and output key that refers to. Depending on chosen variable type, more detailed information is required, such as a name of instance or instance set for example. If an user indicates result subset, which consists of more than one value, an additional filters are available. The highest, the lowest value, average or sum, for example, can be considered depending on used filter. Based on defined variables, objective function is calculated according to an equation given by an user. The equation consists of variables names, which are replaced with real values read from odb file before evaluation. It allows to define expression consisting of any configuration of variables. The module also services parallel computations. The number of parallel analyses can be specified in parallelization tab. Optimization is processed until stop criteria are not achieved. An user can define minimal number of analyses and minimal objective function value. Specification of minimal objective function value is recommended, however it is not necessary.

2.6 Genetic algorithm optimization

A genetic optimization process is serviced with Genetic Algorithm (GA) object. The optimization algorithm is described in Genetic Algorithm section in detail. This section describes a problem of population evaluation with the use of Abaqus suite programs. However, many elements such as model rebuilding, job definition and submitting are explained in Numerical Model (NM) object section. The evaluation mechanism has to fetch individuals and assign fitness values to them based on objective function. In the module whole population is sent to evaluation in one time. The GA object waits until an evaluation terminates. Every population is partitioned into groups. The maximum number of individuals in each group is limited by a given parallel analyses number. Groups are sent to evaluation successively one by one.

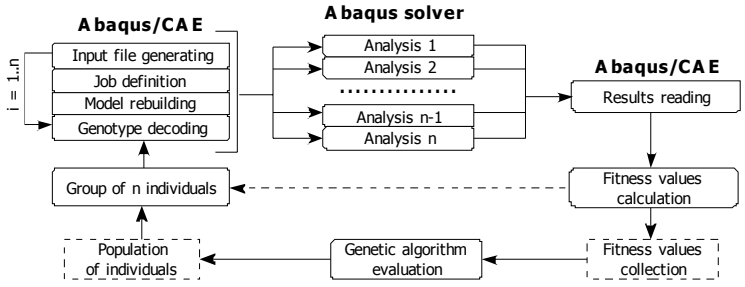

Figure 6. Scheme of Genetic algorithm evaluation processing

For each individual in a group, basing on genotype, input file is created in the loop. The genotype is decoded and optimized design variables are obtained in the first place. The Numerical Model object runs rebuilding mechanism, which reconfigures Abaqus/CAE model according to a current genotypes. Based on actual model a job is created and after that an input file is written. The loop is repeated until input files for all individuals are generated. Every prepared input file is submitted in an independent thread. In the result, analyses are carried out in parallel. The calculation time of group of individuals depends on the longest analysis. In order to prevent for delay caused by untypical time-consuming analysis, maximum analysis time can be defined. An analysis is stopped when the limit time is exceeded. This option should be used carefully because it is possible to exclude a valuable solution from checking. After the longest parallel analysis is finished, the result

reading procedure is started. Necessary data is read from odb file and an objective function value is calculated for each individuals in the current group. In the case when analysis is stopped because of exceeded maximum time or analysis exited with an error or rebuilding failure, a zero fitness value is returned. In this way, the unacceptable individuals are eliminated from further optimization. The fitness values for every group are collected and after the last one is evaluated the whole collection is returned to the GA object. The described loop is repeated for all population.

2.7 Genetic algorithm + neural network optimization

The primary goal of modification is reducing a number of analyses, which are necessary to find a better solution, event if it is not the best one. The basis of a proposed modification is an assumption that accurate evaluation of each individual is not necessary. A GA algorithm infers based on whole population. An evaluation tool should only detect features which make individual better fitted. Thus, it is advantageous to use an evaluation tool which evaluate less accurate but faster. The neural network (NN) is proposed. A properly prepared NN can return a fitness value estimation based on a genotype. The higher number of information for a NN learning, the better accuracy is obtained. However it is impossible to eliminate an error at all. The neural network is serviced by the Neural Network object and some elements such as a NN architecture, learning and an architecture optimization are described in the corresponding section in detail. This section raises a problem of a general optimization algorithm and learning data preparation.

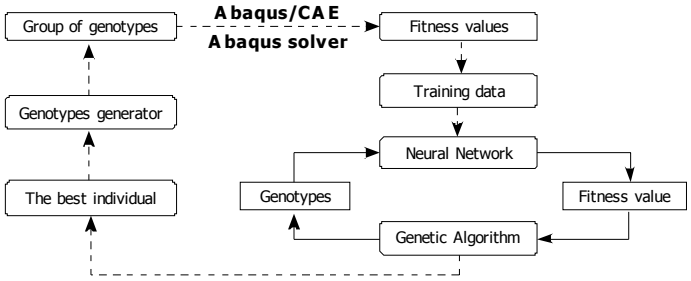

Figure 7. Modified optimization algorithm processing scheme

The optimization is carried out in loops. During each loop a batch of learning data is generated and the NN is a subject for learning. After NN learning, GA is carried out using a NN to estimate fitness value. The best solution from a GA is the first approximation of design parameters. Each loop improves NN quality and causes that successive approximations are closer to the optimal solution in next iterations. In order to create learning data, almost the same algorithm as in the Genetic Algorithm optimization is used. The only difference is genotypes creation. In the GA+NN optimization a genotypes creator is used instead of a population of individuals. The training data is a set of input-output vectors, which consist of genotypes and objective function values. In the first attempt only random genotypes are generated. With the effect from the second iterations, various strategies of learning genotypes can be used. The Only Random option is a continuation of the first attempt. It minimizes local extreme finding, however, usually do not provide the optimal

solution. For this reason, another two strategies are available. In both cases a training set consists of the best individual genotype from a previous iteration. This additional data is used, firstly, for verification and secondly, for NN correction. As a result the next optimal solution is calculated taking into consideration all previous attempts. In the case of parallel computations the rest of genotypes can be random generated or created as a result of the best genotypes mutation. The first option minimizes a risk of local extreme but moderates the best solution influence on the direction of a next approximation searching. The second option makes possible to find a better solution quickly but unfortunately strongly favors the first detected direction and is endangered for local extreme finding.

3 FE implant model

One of the first application for which presented Optimization Module was used, is optimization of implantological system OSTEOPLANT® (Kakol et al., 2002). It is the two-component system commonly used which consists of a root and an abutment. They are both connected with a non-rotational hexagonal slot, assembled by a screw (Figure 8). The majority of implant parts are axisymmetric. The main goal of this study is to find a the new shape of screw head and hexagonal slot. The next step will be to increase fatigue life of implant. The existing results of fatigue calculations of these implants give information about required reduction of stress to achieve assumed design life (Wierszycki et al., 2006c). The objective function was defined to reduce the equivalent Mises stress at notches.

Figure 8. Dental implant OSTEOPLANT

Genetic algorithm and neutral network learning would require a huge number of analysis to be carried out. For this reason the crucial characteristic of the numerical model of implant which is used in optimization process is a performance. In practice, the time of calculation can not exceed several dozen minutes. This limitation causes that a fully three-dimensional model of implant can not be used (Wierszycki et al., 2006c). For this study a special approach was proposed. The modeling approach described in detail below enabled us to carry out a nonlinear 3D simulation of dental implant in acceptable time with satisfied level of accuracy of results. A 3D model of implant has been created by revolving an axisymmetric model about its axis of symmetry (Figure 9). The symmetric model generation capability of Abaqus/Standard enables to create automatically 3D model (Abaqus, 2006a). The nodes, elements, section definitions, material and contact definitions of the three-dimensional model are created automatically base on axisymmetric model description. Only kinematic constrains and boundary conditions must be redefined. In order to

reduce time of calculation the asymmetric deformation of 3D model was assumed to be symmetric with respect to the radial – symmetry axis plane at an angle equal 0 or π. The symmetric results transfer capability of Abaqus/Standard was used to transfer the results from an axisymmetric simulation of assembly to the final 3D model (Abaqus, 2007a).

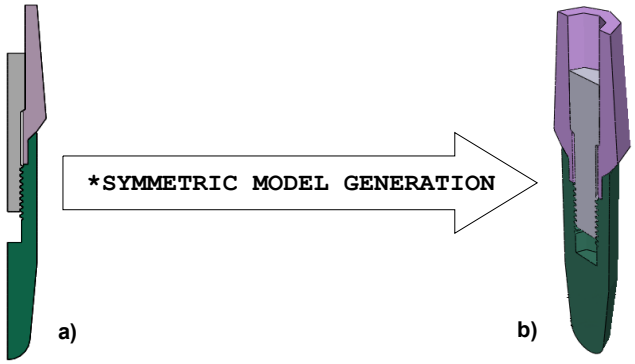

Figure 9. Numerical models of implant: (a) axisymmetric and (b) 3D

3.1 Geometry

The geometry of numerical model was simplified to axisymmetric description (Figure 9a). The internal threads of implant body and screw were simplified to axisymmetric, parallel rings. Because the main goal of this study is optimization of a screw connection the external threads of implant body were omitted. The parametric Abaqus/CAE model of implant consists of three axisymmetric parts. The shape of the each part corresponds to the cross-section of dental implant components: an abutment, a body and a screw. The 2D sketches of these parts have fully parametric geometry description and are fully constrained. These constraints with dimensions and parametric equations added to a sketch enabled us automatically modification of the shape of implant components (Abaqus, 2007c). The six global independent parameters were defined:

- screw head diameter
- screw diameter
- screw head conic surface opening angle
- screw head height
- hexagonal slot diameter
- hexagonal slot height
- hexagonal slot conic surface opening angle.

All parts share the same parameters, so the instances of this part are always consistence. The implant geometry of 3D model was not defined directly. The FE three-dimensional model is automatically created based on axisymmetric model.

3.2 Material

All components of an implant are made of medical alloys of titanium. For general stress-strain analyses, isotropic, non-linear elastic-plastic characteristics of material models were taken into account. The material properties were based on the certificate of conformity and from the literature (Wang, 1996). The material models and characteristics definitions are automatically transferred from axisymmetric to three-dimensional model during symmetric model generation procedure (Abaqus, 2007a).

3.3 Assembly

The assembly is done in the first axisymmetric stage of implant model building. The three two-dimensional instances of implant parts were positioned relative to each other in a global coordinate system. Relative position constraints were applied that align:

- conic surface of hexagonal slot and outside of abutment,
- conic surface of screw head and inside of abutment.

Fully relative position constraints of implant part instances make possible automatic redefinition of implant model assembly (Abaqus, 2007a).

3.4 Contact

The tightening simulation involves solving a contact problem (Wierszycki et al., 2006b). For this purpose it is necessary to define three contact pairs between: root and abutment, root and screw, abutment and screw. These contact conditions produce typical assembly problem, so small-sliding contact formulation can be used (Abaqus, 2007a). In order to minimize dependence on a mesh density the surface-to-surface contact discretization was used. Moreover, surface-to-surface discretization provides more accurate stress and pressure results, especially in contact at corners like on the threads. For three dimensional model with two cylindrical elements in 180° segment node-to-surface discretization causes a doubled number of increments and sever discontinue iterations. The penalty method as the contact constraint enforcement method has been selected for both normal and tangential direction. The tangential surface behavior has been defined as a classical isotropic Coulomb friction model. The friction coefficient is assumed the same for all contact pairs and was equal to 0.19. The contact pairs definitions are automatically transferred from axisymmetric to three-dimensional model during symmetric model generation procedure (Abaqus, 2007a).

3.5 Loads

Loading of the implant model is a two-steps process (Wierszycki et al., 2006a). In the first step simulation of tightening is performed. In this step both model and its response are axisymmetric. This simulation can be carried out with the use of the axisymmetric model. The second step is bending, which is caused by the worst component of service load, perpendicular to axisymmetric axis. In this case the model which can describe asymmetric deformation is needed. For a two-component implant, the crucial aspect of numerical modeling is the simulation of mechanical assembly – tightening of the implant screw process (Lang et al., 2003, Wierszycki et al. 2006a).

For axisymmetric or simplified three-dimensional model, tightening simulations cannot be performed as a real physical process. A work-around approach is necessary. For simulation of tightening prescribed assembly load has been used. The pre-tension section was defined as a surface inside the middle part of the screw. The tightening load (500N) was applied to the pre-tension section by means of the pre-tension node as a concentrated load. During calculation the screw length is reduced in its middle part to achieve the assumed tightening force. The implant body and abutment were tightened as results of the change of screw length. The value of axial force in a tightened screw was calculated from the empirical equation (Bozkaya and Mufut 2005, Lang et al., 2003). It was verified during full simulation of screw tightening with the help of a fully three dimensional FE model of an implant (Wierszycki et al., 2006a). In the second step bending of the tightened implant is performed. The results obtained in the axisymmetric simulation are transferred onto the final three-dimensional model. The bending force is applied to the tip of abutment by means of surface-based coupling constrain. This constrain is defined in the three-dimensional model. The surface which defines coupling nodes has been created in axisymmetric model. The reference node was created in three-dimensional model. The concentrated load (100N) perpendicular to axisymmetric axis of implant was applied to this node.

3.6 Mesh

The mesh of the axisymmetric model was created with the use of the free mesh technique and advancing front method (Abaqus, 2007c). For all mesh regions quad-dominated shape of element was used. Each instances were partitioned and seeded in such a way to assume the enough fine meshes. On each edge of mesh region the seed was defined to control mesh density.

Table 1. Selected parameters of comparable simulation of 3D analysis with the use of cylindrical (CCL-x) and 3D solid element (C3D-x).

	Floating point operations per iteration [-]	Minimum memory required [MB]	Required diskspace [MB]	Number of equations [-]	Number of increments [-]	Number of SDI [-]	Number of EI [-]	Wallclock time [sec]
CCL-1	6.51E+009	47.31	150.73	56226	7	33	7	160
CCL-2	2.81E+010	87.25	366.96	93588	7	32	13	401
CCL-4	1.61E+011	176.93	1034.24	168312	9	47	15	1565
C3D-16	9.48E+011	447.38	2969.60	317760	13	84	20	8730
C3D-32	5.15E+012	1157.12	8427.52	616656	13	82	27	47685

The correct mesh density must be ensured for different geometry configuration. The seeds were defined by specifying average element size along an edge. To ensure proper mesh density for different geometry configurations the seed density was partially constrained. This approach ensures that even if the number of elements along an edge was changed its size remains such as was defined (Abaqus, 2007c). The mesh of the three-dimensional model was generated automatically during symmetric model generation procedure. In whole model of implant CCL9 and CCL12 elements were used (Abaqus, 2007a). The number of cylindrical elements along the

circumferential direction is the compromise between time of calculation and accuracy of results. The five test analyses were carried out to evaluate which number of element is optimum. The models with three-dimensional solid element C3D8 and C3D6 were used as a reference solution. In these models 32 and 16 elements per 180° segment were used. The maximum values of the equivalent Mises stress at characteristic notches and global bending stiffness of whole implant structure were used to compare results. The detail results of comparable studies are shown in Table 1. Because there are no significant differences in results for two and four cylindrical elements two elements were used for optimization process. This approach enables us to describe nonlinear asymmetric deformation for axisymmetric geometry and simultaneously significantly reduces the size of the problem (ca. 94 000 dof) in comparison with a full three dimensional model (ca. 600 000 dof).

4 Procedure of FE implant model building during optimization

The procedure of FE implant model building during optimization process is shown in Figure 10. At the beginning of the optimization process the CAE file with a fully parametric axisymmetric model of implant is opened. This model was described in detail in the previous section. When the optimization loop is started the axisymmetric model of implant is modified. The Abaqus/CAE creates the input file with modified axisymmetric model of implant and Optimization Module submits jobs. The results of these axisymmetric simulations are assembled implant structures. In the next step the Optimization Module generates PARAM files and simultaneously runs three-dimensional jobs. The input file of the three-dimensional model with definition of symmetric model generation procedure and second step of implant simulation are prepared by an user. The

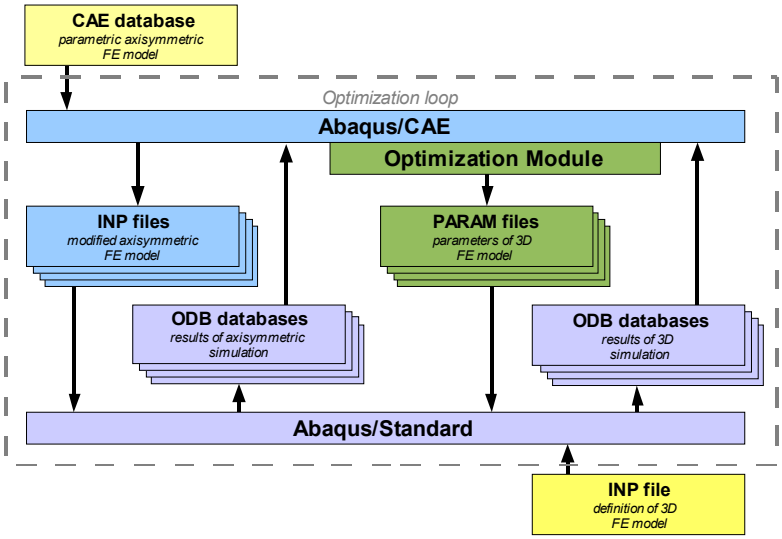

Figure 10. The procedure of FE implant model building during optimization.

PARAM files are created based on information from axisymmetric odb files and options which are defined by user. These files contain parameters of three-dimensional models of implant, such as:

- segment angle through which the cross-section to be revolved
- number of elements to be used in the segment
- offset for element numbering
- offset used for node numbering
- global number of node in axisymmetric model.

Finally, the results of the three-dimensional simulations are read from odb files and the Optimization Module starts Genetic Algorithm process. This procedure must be repeated for each population. The described modeling approach with the use of the axisymmetric geometry description and the semi-analytical discretization enables us to carry out a large number of implant simulations in the realistic time.

5 Results

The presented approach was used to optimization of the upper part of implant screw (thread was excluded form consideration) to reduce the Mises stress. The fitness function was defined as follows:

$$f(A,B,C,D,E,F) = \left(\frac{1}{\sigma_{mises}} \right)^2 \cdot 10^8$$

where σ_{mises} is the maximum Mises stress in the upper part of implant screw, A - F are the design parameters, as shown in Figure 11.

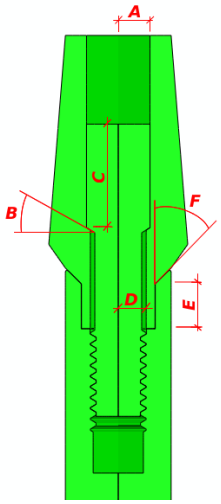

Figure 11. Design variables of implant model.

The results presented hereafter were obtained with the use of Genetic Algorithm only. The number of population was 60 and the number of generations was 100. Because of usage of long genotype (28 genes) a two-point crossover method was chosen. A default rate of crossover (0.75) and mutation (0.02) values were applied. In order to stronger promotion of a better individual in uniform population the rank-based selection was used. For each design variable a range and a number of bits for encoding were defined. The design variables are shown in Figure 11 and their values are placed in Table 2.

Table 2. Design variables of implant.

Symbol	Design variables				Number of bits
	initial	min	max	optimal	
A [mm]	1.05	0.95	1.6	1.42	4
B [deg]	31	0.775	80	29.20	6
C [mm]	3.45	1.5	6.4	3.6	3
D [mm]	1.26	1.0	1.5	1.5	4
E [mm]	1.5	0.05	2.65	0.223	4
F [deg]	45	11	90	88.7	6

The optimization was performed using 5 cpus for computations. A single analysis took 5 minutes in average. During a whole optimization process 6300 FEA jobs were run. The total time of optimization was 105 hours.

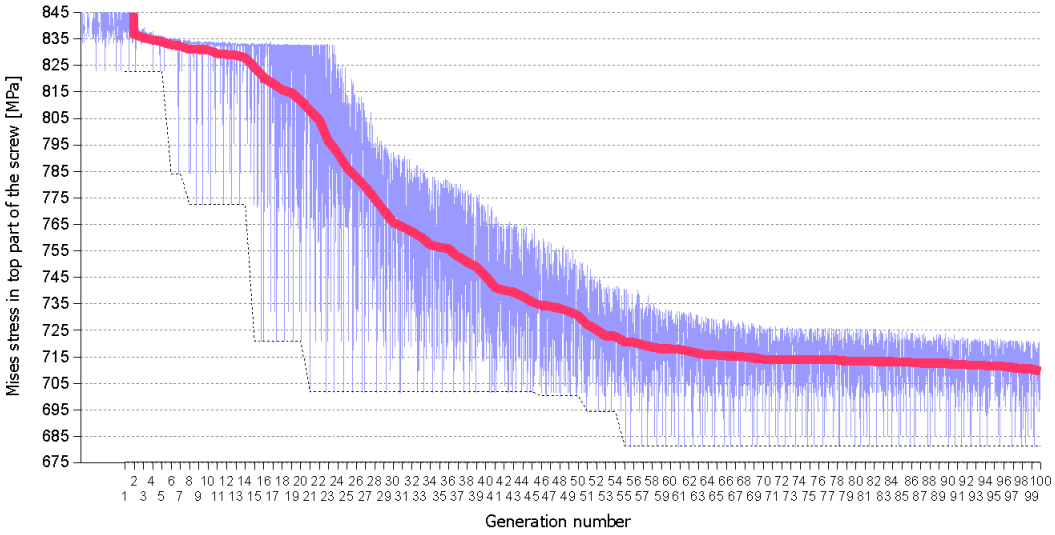

Figure 12. Fitness value for successive generated individuals.

Figure 12 presents the main results of optimization process. It shows the evolution of the maximum Mises stress in upper part of implant screw for successively generated solutions. The average value of the Mises stress is drawn by a solid (red) curve. It can be observed that the designs are improving – the maximum Mises stress is decreasing. Starting with an average value of 835 MPa, the Mises stress was reduced to 700 MPa. Note that at the beginning of optimization the initial population has to be established first. GA is run when the whole population of correct individuals is created. The design parameters from the best solution (peaks) are promoted stronger and thus they influence on population improvement. Because of chosen reproduction strategy, only the best individuals from current and newly created generation are basis for creation of a new population. It results in a low diversity of fitness values within each population and in constant reduction of maximum stress limit in whole optimization process. After the 55th generation the best solution is established and no further essential reduction of stress is observed.

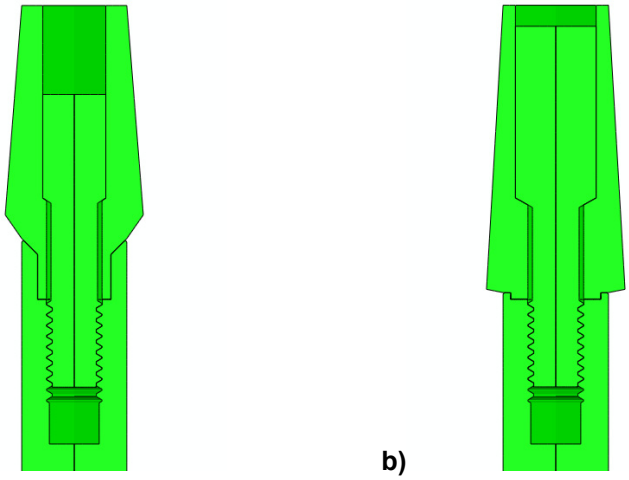

Figure 13. Initial (a) and final (b) shape of dental implant.

The shape of optimized and original implants are shown in Figure 13. The obtained optimal design parameters are placed in Table 2. In the new implant the reduction of the Mises stress at screw head and hexagonal slot is about 18%.

6 References

1. Abaqus Analysis User's Manuals, SIMULIA, Pawtucket, 2007.
2. Bozkaya D., Mufut S., Mechanics of the taper integrated screwed-in (TIS) abutments used in dental implants, Journal of Biomechanics, Vol. 38, pp. 87–97, 2005.
3. Burczynski T., Kus W., Dlugosz A. Orantek P., Optimization and defect identification using distributed evolutionary algorithms, Engineering Applications of Artificial Inteligence, Vol. 17, 4, pp. 337-344, 2004.
4. Chao H.K., Rowlands R.E., Reducing tensile stress concentration in performed hybrid laminate by genetic algorithm, Composites Science and Technology, Vol. 67, 13, pp. 2877-2883, 2007.
5. Goldberg D.E., Genetic Algorithm in Search, Optimization and Machine Learning, 1st Edition, Addison-Wesley Professional, 1989.
6. Kakol W., Lodygowski T., Wierszycki M., Hedzelek W., Zagalak R., Numerical Analysis of the Behavior of Dental Implant, ABAQUS Users' Conference Proceedings, CD-ROM, 2002
7. Lang L. A., Kang B., Wang R. F., Lang B. R., Finite element analysis to determine implant preload, The Journal of Prosthetic Dentistry, Vol. 90, 6, pp. 539-546, 2003.
8. Wang .K, The use of titanium for medical applications in the USA, Materials Science and Engineering, Vol. A213, pp. 134-137, 1996.

9. Wierszycki M., Kakol W., Lodygowski T., The screw loosening and fatigue analyses of three dimensional dental implant model, ABAQUS Users' Conference 2006, Boston MA, 2006.

10. Wierszycki M., Kakol W., Lodygowski T., Numerical complexity of selected biomechanical problems, Journal of Theoretical and Applied Mechanics, Vol. 44, 4, pp. 797-818 , 2006.

11. Wierszycki M., Kakol W., Lodygowski T., Fatigue algorithm for dental implant, Foundations of Civil and Environmental Engineering, Vol. 7, pp. 363-380, 2006.