Optimization of integrated circuits by means of simulated annealing

Jernej Olenšek, Janez Puhan, Árpád Bűrmen, Sašo Tomažič, Tadej Tuma

University of Ljubljana, Faculty of Electrical Engineering, Tržaška 25, Ljubljana, Slovenia

Key words: parametric optimization, simulated annealing, design of integrated circuits.

Abstract

The purpose of this paper is to test the efficiency of the modified orthogonal simulated annealing algorithm. The method is compared with the COMPLEX method on a set of mathematical functions. The method is then used on three real-world cases of integrated circuits and compared with a modified COMPLEX method that uses intelligent initial points selection.

Optimizacija integriranih vezij z algoritmom simuliranega ohlajanja

Ključne besede: parametrična optimizacija, simulirano ohlajanje, načrtovanje integriranih vezij.

Povzetek

Namen prispevka je preizkusiti učinkovitost modificiranega ortogonalnega simuliranega ohlajanja. Primerjamo ga z metodo COMPLEX na skupini matematičnih funkcij. Metoda je nato uporabljena na treh realnih primerih integriranih vezij in primerjana z modificirano metodo COMPLEX, ki uporablja pametno izbiranje začetnih točk.

1 Introduction

Optimization problems arise in virtually every field of engineering, science, and business. The parametric optimization problems are usually presented in the following form:

$$\begin{aligned}&\min_{\underline{x}\in R^n} f(\underline{x})\\ &f : R^n \to R \qquad (1)\\ &\underline{x} \in [\underline{L},\underline{U}]\end{aligned}$$

where f is the so-called cost function (CF) and $\underline{x}$ is a vector of parameter values. $\underline{L}$ and $\underline{U}$ are vectors of lower and upper parameter bounds, respectively. Unfortunately analytical solutions to (1) can only be obtained for some very simple and small problems. Most practical problems are complex and often include simulations and measurements, which are very expensive and time consuming. The complexity of the optimization problem depends on the dimensionality (i.e. the number of optimization parameters) and on the shape of the CF. The size of the solution space increases exponentially with the problem dimensionality, so locating good solutions becomes increasingly more difficult. But the real challenge arises from the CF itself. In most real-world applications the CF is nonlinear and has many local minima. Often the value of the CF is a result of numerical simulations or measurements that introduce noise to the CF. Noise makes the fast deterministic gradient based methods useless and derivative free direct methods become more attractive. Direct methods are usually divided in two major groups. Deterministic methods always produce the same final solution when they start with the same initial guess. One method from this group is the simplex method which is well known and popular due to its simplicity and speed. But the simplex method is a local downhill search method and its solution greatly depends on the initial guess. Stochastic methods, on the other hand, introduce randomness to the search process and are capable of escaping from the local minima in order to find better solutions. Simulated annealing is a stochastic method. In this paper we describe a recent version of simulated annealing referred to as Orthogonal Simulated Annealing (OSA) [1] and compare it with a modified simplex method also known as COnstrained siMPLEX (COMPLEX) [2]. The comparison is done on a set of mathematical test functions. OSA and modified COMPLEX methods are then used on three real-world integrated circuit (IC) design problems. The purpose of comparison is to establish the feasibility of circuit optimization with OSA.

The paper is organized as follows. In section 2 a brief description of the basic simulated annealing algorithm is given and in section 3 the OSA algorithm is described in detail. Section 4 compares the algorithm with the COMPLEX method on a set of mathematical test functions. In section 5 OSA is compared with a modifed COMPLEX methods on three cases of IC design. Section 6 gives the conclusions.

2 Simulated annealing algorithm

Downhill methods can easally get trapped in local minima. To escape from a local minimum uphill moves must be allowed from time to time to give the algorithm a chance to move to unexplored parts of the solution space. Simulated annealing [3] was developed for this purpose. It always accepts downhill moves but occasionally uphill moves are also accepted. The basic features of the algorithm come from the analogy with the movement of atoms in metal. When metal is heated up to a very high temperature, atoms can move freely even to a state with higher energy. When the material is cooled down slowly, atoms are more likely to move to low energy states. If the annealing is slow enough, the resulting metal has an uniform structure with very few defects and minimal free energy. The simulated annealing method mimics this process by introducing an artifical parameter to the search process often referred to as the temperature (T) which controlls the acceptance probability for the uphill moves. At the beginning of the search it is set to a high value and most transitions to higher CF values are accepted. As the search progresses, the temperature is slowly decreased so that the uphill moves become less frequent. If the annealing is done in a sufficiently slow manner, the final solutions reached by the algorithm are near the global minimum of the CF. The CF is an analogy of the free energy of the atoms in a metal. The basic steps of the simulated annealing algorithm are:

1. initialize - set algorithm parameters, inital point
2. generate new point - generation mechanism
3. acceptance criterion – transition
4. continue with 2 until end of temperature stage
5. annealing - cooling schedule (decrease temperature)
6. continue with 2 until stopping condition is met

These steps must be chosen carefully in order to ensure the probabilistic convergence to the global optimum. The obtained algorithms, however, are not very efficient in practice because the required cooling schedules are too slow or the generation mechanisms are too inefficient to get any useful result in a reasonable amount of time. That is why most practical versions of the algorithm use modified generation mechanisms and cooling schedules. This way the convergence proofs (i.e. [5]) no longer apply but good solutions can still be obtained in a reasonable amount of time.

3 Orthogonal simulated annealing algorithm (OSA)

Recently a new version of the simulated annealing algorithm was developed [1], taking advantage of a carefully designed set of experiments at every iteration that helps to choose a good point for the next iteration. Since the results reported in [1] were encouraging, this method was chosen for implementation and testing. All steps of the algorithm are described in this section.

3.1. Initialization

In the initialization step basic algorithm parameters are set. For this purpose several points (in our case 100) are randomly chosen and evaluated. The best of these points is set as the starting point for the algorithm. The initial value of the temperature parameter T is set to the standard deviation of CF values at these points. Our method differs slightly from the original one [1]. Instead of using the same parameter (temperature) for the acceptance criterion and the generation meachanism we use a separate parameter for generation of random moves. The allowed intervals $\left[\underline{L},\underline{U}\right]$ for different optimization variables can vary considerably so the generation mechanism must use a separate parameter for each variable. For this reason we introduce another vector parameter referred to as the range ($\underline{R}$). The initial values of the components of $\underline{R}$ are set to allowed interval widths of optimization variables. Another parameter that needs to be set is the number of moves at each temperature stage N_t. In theory it must be large enough for the algorithm to reach thermal equlibrium in every temperature stage. In our case N_t is set to 10.

3.2. Generation mechanism

The algorithm uses orthogonal experimental design (OED) to choose good candidates for the next iteration. A carefully designed set of experiments allows for an efficient factor analysis. The main idea is to evaluate a small number of points in order to estimate factor effects on the given CF. The selection of these points is done with the help of orthogonal arrays. In this context orthogonal means statistically independent so that the estimation of the effect of one factor does not affect the estimation of the effects of others. An example of such an experimental design for three factors and three levels per factor is given in table 1, which contains the generated orthogonal array and figure 1, which shows the distribution of the corresponding experimental points.

-- Table 1 --

-- Fig 1 --

There is a total of $Q^{N_f} = 3^3 = 27$ combinations of factor levels for this example, where Q is the number of levels per factor and N_f is the number of factors. In our case only nine experimental points have to be evaluated and the use of the orthogonal array assures that these points are evenly spread around the search space. The algorithm for generation of orthogonal arrays can be found in [4].

The use of orthogonal arrays also has drawbacks. The method works very well when there are no interactions between different factors. Unfortunately this is usually not the case. Furthermore optimization problems often include many optimization variables so the number of required experiments for an efficient factor analysis is large. Therefore the N_v variables are randomly grouped into N_f factors. The two most extreme cases are when $N_f = N_v$ and $N_f = 1$. In the former case there are many factors but because of the interaction effects between factors, the estimation of factor effects is less accurate. In the latter case there is only one factor and the estimated effect is accurate, but the optimization requires more iterations. A compromise is needed. The formula for determining the number of factors for a given problem is:

$$N_f = \left(3^{\lfloor \log_3(2\cdot N_v+1)\rfloor} - 1\right)/2 \qquad (2)$$

At the beginning of every iteration the variables are randomly divided into N_f groups and every group is considered as one factor. Then a random perturbation vector is generated according to a specified probability distribution (in our case the Cauchy distribution). For every optimization parameter x_i the probability distribution of perturbation dx_i is:

$$p(dx_i) = \frac{R_i}{\pi\left(R_i^2 + dx_i^2\right)}, \quad i = 1,2,...N_v \qquad (3)$$

where R_i is the range parameter of the i-th variable in the current temperature stage. To generate a random variable from this distribution the inversion method is used:

$$dx_i = R_i \cdot \tan\left(\pi \cdot (U - 1/2)\right), \quad i = 1,2,...N_v \qquad (4)$$

where U is an uniformly distributed random number from the interval [0,1]. After generating the perturbation vector $\underline{dx}$, the three levels for every optimization variable are determined as:

$$\begin{aligned} x_i^0 &= x_i^c \\ x_i^1 &= x_i^c + dx_i, \quad i = 1,2,...N_v \\ x_i^2 &= x_i^c - dx_i \end{aligned} \qquad (5)$$

where x_i^c is the current value of the i-th variable. If x_i^1 or x_i^2 violates box-constraints $[L_i, U_i]$, its value is chosen randomly from this interval. Since variables are grouped into factors, setting one factor to some level means setting all the variables x_i from that factor to the corresponding level (x_i^0, x_i^1 or x_i^2). This way the orthogonal array is converted into experimental points, which are then evaluated. The main effects of all factor levels are obtained by the following formula:

$$S_{j,k} = \sum_t y_t \cdot F_t \tag{6}$$

where $S_{j,k}$ denotes the effect of the k-th level of the j-th factor and y_t is the value of the CF from the t-th experiment. F_t has only two possible values. It is 1 if in the t-th experiment the j-th factor has k-th level. Otherwise F_t is 0. The new candidate solution can now be generated. For every factor (j) the level (k) with the minimum $S_{j,k}$ is chosen. The CF of the new candidate solution is then evaluated. The best of all the experimental points and the candidate solution is then submitted to the acceptance criterion as a potential solution for the next iteration. This process is repeated N_t times in every temperature stage.

3.3. Acceptance criterion

Most versions of the simulated annealing algorithm use the same transition acceptance criterion which is known as the Metropolis criterion. Downhill transitions are allways accepted. Uphill transitions are accepted with the probability:

$$P = e^{\left(-\frac{y'-y}{T}\right)} \tag{7}$$

where y' and y are the CF values at the new and the current point, respectively, and T is the value of the temperature parameter. At high temperatures almost all transitions are accepted but when the temperature is close to zero most of the uphill moves are rejected and the algorithm acts as a downhill method.

3.4. Annealing

The next step of the algorithm is the cooling schedule. Several schedules have been developed but the best known and also very popular is the original schedule of Kirkpatrick [3]. The temperature decreases exponentially:

$$T(k) = T(k-1)\cdot\alpha, \quad \alpha \in [0,1] \tag{8}$$

where k is the temperature stage index. Large values of α mean slow convergence but more reliable search for the global optimum whereas smaller values mean fast convergence with the increased risk of getting trapped in a local minimum. The empirically chosen value for α was 0.99. At the end of every temperature stage the number of moves N_t in a temperature stage is also decreased by α:

$$N_t(k) = N_t(k-1)\cdot\alpha \tag{9}$$

The probaility distribution for random moves must also be adapted. The range parameter $\underline{R}$ is reduced at the end of every temperature stage:

$$R_t(k) = R_t(k-1)\cdot\alpha, \quad i = 1,2,\ldots N_v \tag{10}$$

3.5. Stopping criterion

Several stopping criteria can be used. In our case the algorithm stops when the temperature reaches user specified minimal value $T_{\min}$ (in our case 10^{-6}) or when the number of CF evaluations exceeds the maximum allowed number of evaluations.

4 Optimization of mathematical test functions

Orthogonal simulated annealing was compared with the COMPLEX method [2] on a set of mathematical functions. The set includes unimodal functions, functions with a small number of local minima (considered as moderatelly difficult problems) and difficult problems with many local minima, noise, nonlinearity and strong interactions between variables. All of the tested functions can be found in [4]. The optimization was repeated 50 times for every function with randomly chosen initial points. Both methods had the same limited number of CF evaluations (50000 and 70000 for problems with $N_v = 30$ and $N_v = 100$, respectively). The optimization results are given in table 2.

-- Table 2 --

The results show that the OSA method is promising when compared to COMPLEX. The COMPLEX method exhibits very fast convergence but gets stuck in a local minimum in almost every tested case. It outperforms the OSA method in some cases of unimodal functions and functions with strong noise. The latter is not unexpected since the method maintains a population of points between iterations. Simulated annealing, on the other hand, starts every iteration from a single point. On multimodal functions OSA outperformed the COMPLEX method in terms of global search capabilities. Due to the modified generation mechanism and cooling schedule the algorithm was not able to locate the global minimum in all optimization runs, but the solutions that were found were still fairly good when compared to the global minimum.

5 Optimization of integrated circuits

Since the OSA performance on mathematical functions was very promising, the next step was to test it on real-world electronic circuit design problems and compare its performance with the COMPLEX method. For this purpose SPICE OPUS circuit simulator was used [6]. In SPICE OPUS a modified COMPLEX method is already integrated as one of the available optimization methods. Since the original method has very fast convergence, restart with intelligent initial points selection is conducted every time the basic method reaches its stopping criterion [7]. This process is repeated untill the given limit of CF evaluations is reached. The final result is the best solution of all the runs. The OSA method had to be implemented in C langunage and added as one of the available optimization methods in SPICE OPUS.

Three cases of electronic circuit design were considered. The first circuit was a simple delay element, the second an operational amplifier, and the third a rather complex amplifier circuit. Circuit topologies for all three cases are given in figures 2, 3, and 4. The key properties of all three optimization problems are given in table 3.

-- Fig2 --

-- Fig3 --

-- Fig4 --

-- Table 3 --

Optimization parameters were resistances, capacitances and transistor channel lengths, widths, and multiplier factors. For every circuit several design goals were set. A single CF is constructed as a combination of all the design goals [8]. Optimization is conducted across several corner points to account for different envirenmental conditions (supply voltage, temperature, process parameter variations, ...). Since every CF evaluation requires a separate circuit simulation for each corner point, a large number of simulations is expected resulting in very long run times. Therefore every circuit was optimized only once. The results of the optimization are given in table 4.

-- Table 4 --

Since the modified COMPLEX method uses restarts and can explore several local solutions within the given number of CF evaluations, the number of the run in which a solution was found is given in brackets. The number of CF evaluations after which the COMPLEX method was manually stopped, is also given. The OSA method stopped automatically when the temperature reached its final value. Both tested methods were compared in terms of the solution quality and the number of CF evaluations (FE).

The first case is the most simple of the three cases considered. It only has a few design goals and does not include corner points. It also has the least optimization variables. All this makes the solution space smaller and the CF less complex. For this case the modified COMPLEX method performed considerably better than OSA. It did however require more CF evaluations and several restarts to reach a good solution. In the second case multiple corner points and more design goals were considered. OSA outperformed the modified COMPLEX method in terms of solution quality and number of required CF evaluations. The third case has the largest number of optimization variables, design goals, and corner points. In this case OSA was also more successfull than the modified COMPLEX method. These results show that for simpler cases the modified COMPLEX method clearly is a better choice. But when it comes to complex circuits, many design goals, and, above all, a large number of corner points, it does not perform as good as OSA. Not even restarts helped the COMPLEX method to find a better solutions than the one OSA found in a single run.

6 Conclusions

A recently developed optimization method called Orthogonal Simulated Annealing (OSA) is described and compared against a version of the simplex algorithm

(COMPLEX method). Both methods are first tested on a set of mathematical test functions. The results showed that OSA performs better when the CF has many local minima. On the other hand, the COMPLEX method is a good choice when finding a local minimum quickly is more important than finding a global minimum. OSA and modified COMPLEX method were then tested on three IC design cases. The results showed that on the simpler case the modified COMPLEX method using restarts outperformed the OSA method. As the problem complexity increased, the ability of the OSA to explore the search space more thoroughly resulted in better performance (compared to the modified COMPLEX method). But in order to obtain a good solution in a reasonable amount of time, probabilistic global convergence of the algorithm had to be sacrificed (modified generation mechanism and cooling schedule). Therefore there is no guarantee as to when and if the global minimum will actually be found. Nevertheless OSA is well suited to IC optimization and design, particularly for problems with many variables and corner points.

7 Acknowledgment

The research has been supported by the Ministry of Higher Education, Science and Technology of Republic of the Slovenia within programme P2-0246 - Algorithms and optimization methods in telecommunications.

8 References

[1] Li-Sun Shu, Shinn-Ying Ho, *A Novel Orthogonal Simulated Annealing Algorithm for Optimization of Electromagnetic Problems,* IEEE transactions on magnetics, Vol. 40, No. 4, pp. 1790-1795, July 2004.

[2] M.J. Box, *A new method of constrained optimization and a comparison with other* methods, Computer Journal, Vol. 8, pp. 42-52, 1965.

[3] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, *Optimization by simulated annealing,* Science, Vol. 220, pp. 1277-1292,1983.

[4] Yiu-Wing Leung, *An orthogonal Genetic Algorithm with Quantization for Global Numerical Optimizaion,* IEEE transactions on evolutionary computation, Vol. 5, No. 1, pp. 41-53, 2001.

[5] R.L. Yang, *Convergence of simulated annealing algorithm for continuous global optimization*, Journal of optimization theory and applications, Vol. 104, No. 3, pp. 691-716, 2000.

[6] SPICE OPUS circuit simulator homepage:
URL: http://www.fe.uni-lj.si/spice/ ,
Faculty of Electrical Engineering, Electronic Design Automation Laboratory:
URL: http://www.fe.uni-lj.si/edalab/\hspace{1mm}.

[7] Puhan J, Bűrmen A, Tuma T. *Analogue integrated circuit sizing with several optimization runs using heuristics for setting initial points*, Canadian journal of electrical and computer engineering, Vol. 28 (3-4): pp. 105-111 JUL-OCT 2003.

[8] Bűrmen A, Strle D, Bratkovič F, Puhan J, Fajfar I, Tuma T. *Automated robust design and optimization of integrated circuits by means of penalty functions*, AEU-International journal of electronics and communications, Vol. 57 (1), pp. 47-56, 2003.

univ. dipl. ing. el. Jernej Olenšek
Univerza v Ljubljani
Fakulteta za elektrotehniko
Tržaška 25, SI-1000 Ljubljana
E-mail: jernej.olensek@fe.uni-lj.si
Tel: (01) 4768 724

doc. dr. Janez Puhan
Univerza v Ljubljani
Fakulteta za elektrotehniko
Tržaška 25, SI-1000 Ljubljana
E-mail: janez.puhan@fe.uni-lj.si
Tel: (01) 4768 322

doc. dr. Árpád Bűrmen
Univerza v Ljubljani
Fakulteta za elektrotehniko
Tržaška 25, SI-1000 Ljubljana
E-mail: arpadb@fides.fe.uni-lj.si
Tel: (01) 4768 322

prof. dr. Sašo Tomažič
Univerza v Ljubljani
Fakulteta za elektrotehniko
Tržaška 25, SI-1000 Ljubljana
E-mail: saso.tomazic@fe.uni-lj.si
Tel: (01) 4768 432

izr. prof. dr. Tadej Tuma
Univerza v Ljubljani
Fakulteta za elektrotehniko
Tržaška 25, SI-1000 Ljubljana
E-mail: tadej.tuma@fe.uni-lj.si
Tel: (01) 4768 329

experiment number	factor 1 level	factor 2 level	factor 3 level
1	1	1	1
2	1	2	2
3	1	3	3
4	2	1	2
5	2	2	3
6	2	3	1
7	3	1	3
8	3	2	1
9	3	3	2

Table 1: Orthogonal array for 3 factors and 3 levels per factor.

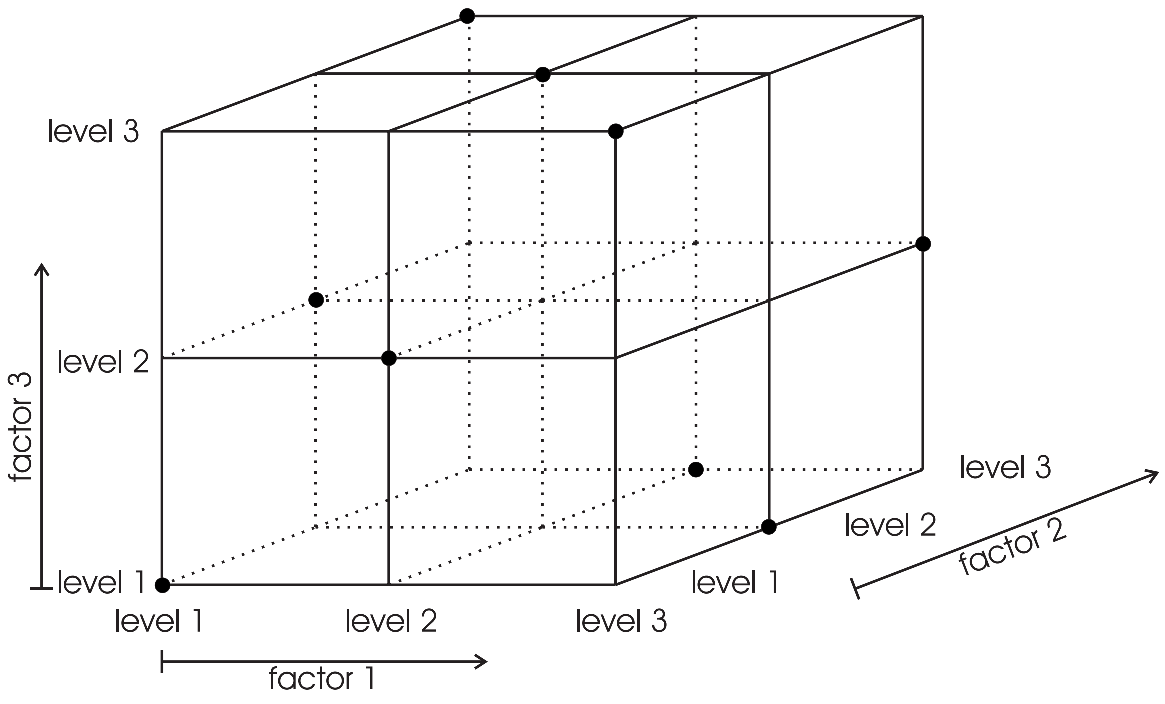

Fig. 1: Distribution of experimental points when the orthogonal array from Table 1 is used.

	N_v	COMPLEX	OSA	global minimum
f_1	30	-9596.2 -3012.3	-12569.5 -12569.5	-12569.5
f_2	30	6.9709 224.67	$3.5527 \cdot 10^{-14}$ 1.9899	0
f_3	30	$1.6714 \cdot 10^{-2}$ 3.93447	$1.5234 \cdot 10^{-7}$ $2.7337 \cdot 10^{-7}$	0
f_4	30	$2.2362 \cdot 10^{-5}$ 3.04840	$1.7375 \cdot 10^{-13}$ $1.6971 \cdot 10^{-1}$	0
f_5	30	$2.1584 \cdot 10^{-5}$ 1.59433	$1.6672 \cdot 10^{-16}$ $2.0732 \cdot 10^{-1}$	0
f_6	30	$1.9551 \cdot 10^{-5}$ 6.99148	$3.1601 \cdot 10^{-15}$ $1.0987 \cdot 10^{-2}$	0
f_7	100	-61.124 -29.261	-98.861 -97.860	-99.2784
f_8	100	-70.408 -63.311	-78.332 -78.331	-78.33236
f_{10}	100	113.13 196.95	283.59 795.04	0
f_{11}	30	$2.5981 \cdot 10^{-8}$ $5.3587 \cdot 10^{-5}$	$5.4955 \cdot 10^{-14}$ $8.2124 \cdot 10^{-13}$	0
f_{12}	30	$4.1618 \cdot 10^{-3}$ $2.0489 \cdot 10^{-2}$	$2.4360 \cdot 10^{-2}$ $1.5660 \cdot 10^{-1}$	0
f_{13}	30	$1.9705 \cdot 10^{-1}$ 3.6034	$1.8169 \cdot 10^{-7}$ $8.2303 \cdot 10^{-7}$	0
f_{14}	30	$8.9943 \cdot 10^{-2}$ 39.777	102.83 1928.0	0
f_{15}	30	1.8026 9.4910	$1.2809 \cdot 10^{-5}$ $8.4080 \cdot 10^{-1}$	0

Table 2. Table shows the best and the worst solution found in 50 optimization runs. The functions used are defined in [4].

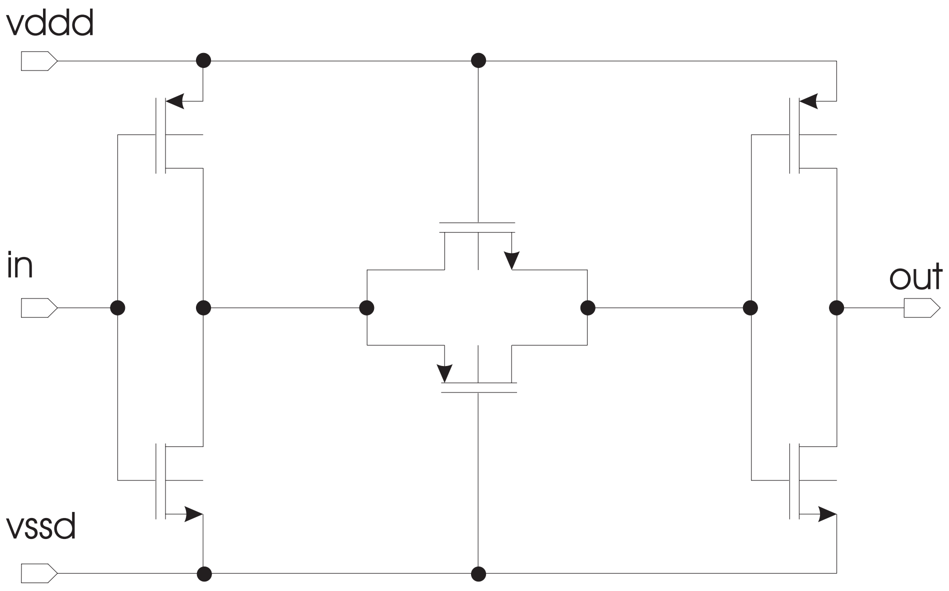

Fig. 2: Topology of the first circuit.

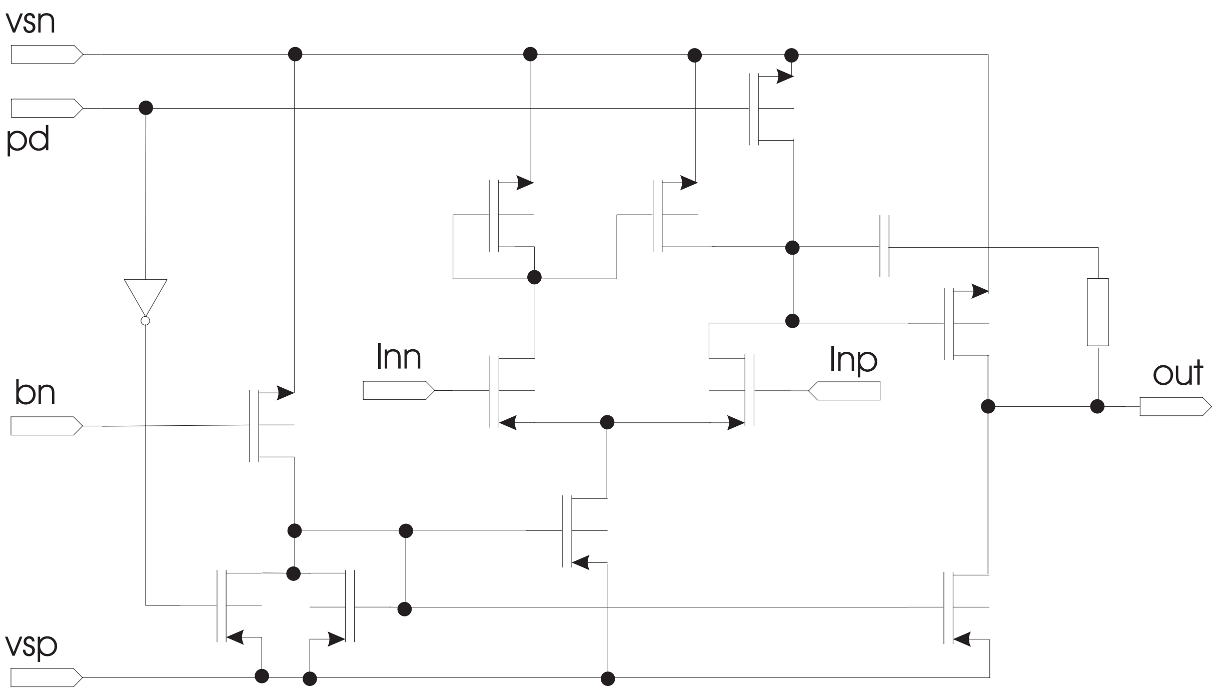

Fig 3: Topology of the second circuit.

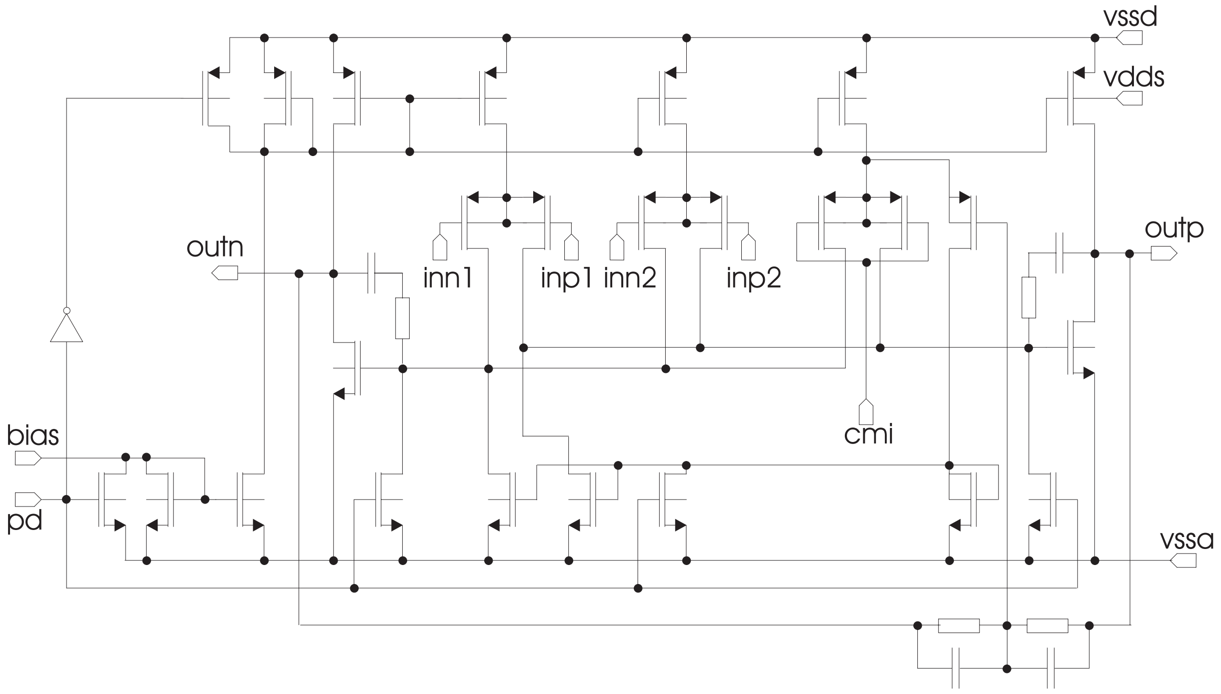

Fig. 4: Topology of the third circuit.

case	N_v	design goals	corner points
1	12	7	1
2	15	14	14
3	17	32	17

Table 3: Summary of the optimization cases: number of optimization parameters, number of design goals, and number of corner points.

case		mdified COMPLEX	OSA
1	FE until cost < $100\cdot10^3$	253 (1)	1011
	FE until cost < $20\cdot10^3$	660 (1)	12758
	best cost	$6.39\cdot10^3$	$11.3\cdot10^3$
	FE until best cost	21409 (28)	14703
	final FE	> 100 000	16122
2	FE until cost < 50	124 (1)	58
	FE until cost < 10	2483 (2)	33595
	best cost	8.07	7.37
	FE until best cost	96912 (69)	47605
	final FE	> 100 000	47768
3	FE until cost < 10	3672 (3)	32014
	FE until cost < 1	26688 (21)	34248
	best cost	0.282	0.088
	FE until best cost	41131 (32)	43877
	final FE	> 45 000	44164

Table 4: Optimization results: number of function evaluations (FE) to find a solution of the given quality, and final solutions. For modified COMPLEX method the number of the run in which a solution was found, is also given in brackets.