An. Şt. Univ. Ovidius Constanţa Vol. 9(2), 2001, 17–30

ON HEURISTIC OPTIMIZATION

Camelia Ciobanu and Gheorghe Marin

Dedicated to Professor Mirela Ştefănescu on the occasion of her 60th birthday

1. Introduction

In the late 1960's is was postulated that there exists a class of combinatorial problems with inherent complexity that any technique of solving such problems to optimality requires computational effort that increases polynomially with the size of the problem. These problems are called NP-hard problems.

Most of the real life problems fall into the category of NP-hard or NP-complete. The exact solution of these problems may be obtained through optimization algorithms (c.g., linear programming, dynamic programming etc.). However, if the problem is not amenable to optimization algorithms due to large computation time, another option is to go for solutions which can be obtained quickly at the risk of sub-optimality. These are called approximation or heuristic algorithms.

Heuristic algorithms are very efficient in handling combinatorial explosion encourented in situations where choices are sequentially compounded, leading to a large number of alternatives. The ewapon target allocation problem, discussed is an example from the area of military OR whereas large number of examples can be quoted from other areas, e.g., manufacturing operations, financial investment, capitol budgeting, resource management, scheduling, VLSI design, etc. Some of the important techniques (see Glover and Greenberg 1989) are Simulated Annealing (based on an analogy to statistical mechanics), Genetic Algorithsm (based on an analogy biological evolution), Neural Networks (based on an analogy to human nervous system), Tabu Search and Target Analysis (based on the concept of artificial inteligence). In the following sections the problem of weapon target allocation in multiple layer defense has been considered as an example through which some of these heuristic techniques are explained.

2. Weapon Target Allocation Problem in Multiple Layer Defense

The problem of weapon target allocation has been studied by several researches under different headings, viz., force mix studies, force-on-force modeling, M-on-N engagement models, weapon target allocation, battle management/command, control and communication (BM/C^3) etc. (see Bracken et al. 19871, 1987b, Bonachevsky et al. 1988, Beare 1987). This problem is complex due to a large number of factors, e.g., type and number of weapons of friendly and enemy forces, their capabilities, strategic values of the assets and enemy targeting plan. Along with this, limits on availability of weapons, manpower, operating cost and area for weapon deployment impose constraints adding to the complexity of the problem. In this chapter, we present a method of determining an optimal defense plan to protect a set of non-identical assets or sites that may come under attack by multiple types of weapons (see Jaiswal et al. 1993). The assets are to be defended by multiple types of weapons deployed in multiple layers, each layer containing identical weapons, i.e., the dth type of defending weapons provide defense in the dth ($1 \leq d \leq D$) layer; for example, defense may be provided by air defense guns, short range missiles, medium range missiles and long range missiles in four different layers.

The analytical model described here may help military commanders in providing them with rational attack or defense plans. Apart from obtaining optimal plans, it can be used to: (a) Predict the capability of existing weapon system against a perceived threat, and (b) Evaluate the potential improvement due to technological development in weapon system.

In the case of multiple layer defense, single attacking weapons pass through different defending weapons deployed in different layers. Attacking weapons that survive the interception by all layers have a chance to cause damage to the asset. If the attack is known and is simultaneous (as opposed to the sequential attack (see Nickel and Mangel 1985) in which attack takes place at different time points), the problem is to determine an optimal defense plan that maximizes a given objective function.

Let

D =Types of defending weapons available

S =Number of assets

A =Types of attacking weapons

k_{dsa} =Probability of successful interception by one defending weapon of type

d deployed to defend an asset s against an attacking weapon of type a

x_{dsa} =Number of defending weapons of type d that are develoyed to intercept

attacking weapon of type a to defend asset s (defense plan)

n_{sa} = Number of attacking weaponS of type a aimed at asset s (attack plan)

g_{sa} = The probability that a single attacking weapon of type a destroys the asset

s when it is able to penetrate the defending weapons

v_s = Value of asset s

c_d = Cost of operating one defending weapon of type d

m_d = Manpower required per defending weapon of type d

B_d = Number of defending weapons of type d

R_a = Number of attacking weapons of type a

G_s = Ground area available at asset s

t_d = Ground area required by a defending weapon of type d

$C_{\max}$ = Maximum operating cost of weapons deployed

$M_{\max d}$ = Maximum available manpower to operate defending weapons of type d.

The probability that weapons deployed in dth $(1 \leq d \leq D)$ layer will not be able to intercept a single attacking weapon of type a $(1 \leq a \leq A)$ on asset s $(1 \leq s \leq S)$ is given by

$$(1 - k_{dsa})^{x_{dsa}/n_{sa}} \,. \tag{1}$$

The probability that a single attacking weapon of type a is not intercepted by any layer on asset s is given by

$$\prod_{d=1}^{D} (1 - k_{dsa})^{x_{dsa}/n_{sa}} . \tag{2}$$

The probability that a single attacking weapon of type a destroys the asset s is given by

$$\left\{\prod_{d=1}^{D}(1-k_{dsa})^{x_{dsa}/n_{sa}}\right\}g_{sa}. \tag{3}$$

The survival probability of asset s by multiple layer defense when attacked by all types of attacking weapons is given by

$$H(s)=\prod_{a=1}^{A}\left[1-\left\{\prod_{d=1}^{D}(1-k_{dsa})^{x_{dsa}/n_{sa}}\right\}g_{sa}\right]^{n_{sa}}. \tag{4}$$

The objective from the defending side is to maximize the total expected value of the surviving assets which is given by

$$\sum_{s=1}^{S}v_sH(s) \tag{5}$$

subject to the following constraints:

(a) Weapon availability

$$\sum_{s=1}^{S}\sum_{a=1}^{A}x_{dsa}\le B_d \quad \text{for} \quad d=1,2,...,D$$

(b) Area availability

$$\sum_{d=1}^{D}\sum_{a=1}^{A}t_dx_{dsa}\le G_s \quad \text{for} \quad s=1,2,...,S$$

(c) Cost

$$\sum_{d=1}^{D}\sum_{s=1}^{S}\sum_{a=1}^{A}c_dx_{dsa}\le C_{\max}$$

(d) Manpower

$$\sum_{s=1}^{S}\sum_{a=1}^{\Lambda}m_dx_{dsa}\le M_{\max d} \quad \text{for} \quad d=1,2,...,D.$$

From the attacking side, the problem is to minimize the same objective function subject to the constraints imposed on the resources of the attacker against a given defense plan.

From this problem definition it is clear that the classical methods of determining the optimal value of a real analytical function of several variables may not be feasible. Therefore, in the following section we use Simulated Annealing an heuristic technique for solving the above mentioned problem.

3. Simulated Annealing

Metropolis et al. (1953) introduced a simple Monte Carlo algorithm for simulating the thermal motion of atoms of solid in contact with a heat bath. At each iteration, an atom is given a small random displacement and the resulting change δ in the energy of the solid is calculated. If $\delta \leq 0$, the change is accepted, but if $\delta > 0$, the change is accepted with the probability $exp(-\delta/K_B T)$ where T is the temperature of the heat bath and K_B is a physical constant called the Boltzmann constant. If large number of iterations are carried out at each temperature, the solid attains thermal equilibrium.

Annealing is a thermal process for obtaining the low energy state of the solid by initially melting the solid to a high temperature and then slowly decreasing the temperature, spending a long time at temperature close to the freezing point. There exists an analogy between the combinatorial optimization problem and the above defined thermal process in the sense that different states of the solid correspond to feasible solutions and the energy which is to be minimized corresponds to the objective function $f(x)$ to be optimized. Therefore, Metropolis algorithm can be used to generate a population of configurations of a given optimization problem at some effective temperature which is called the control parameter. This technique is called simulated annealing and can be treated an iteration of Metropolis algorithm evaluated at decreasing values of the control parameter (see Kirkpatrick et al. 1983, Bohachevsky et al. 1987, 1988).

Simulated annealing can be termed as a biased random walk that samples the objective function in the space of independent variables. It has the ability to migrate through a sequence of local extrema in search of a global solution and to recognize when the global extremum has been reached. Let X_I, X_T, X_0 and X_N belong to the feasible solution space and

$f(X_I)$ = Initial value of the objective function

$f(X_T)$ = Assumed or True value of the global extremum

$f(X_0)$ = Previously accepted value of the objective function

$f(X_N)$ = Presently derived value of the objective function

$$
\begin{aligned}
C_k &= \text{Control parameter value at kth step} \\
L_k &= \text{Number of iterations at control parameter value } \mathrm{C}_k \\
\Delta X &= \text{Step size for neighborhood selection} \\
p_1 &= exp[-\{\mid f(X_0) - f(X_N) \mid\}/C_k] \\
p_2 &= exp[-\{\mid f(x_T) - f(X_N) \mid\}/C_k] \\
u &= \text{Uniform random number.}
\end{aligned}
$$

The decision criterion states that accept X_N if either $f(X_N)$ is less than $f(X_0)$ (for minimization problem) or if a random number u is less than p_1 and also p_2. Here, the probability p_1 helps to migrate through a sequence of local extrema and the probability p_2 helps to recognize when the global extremum is reached. The combined effect of p_1 and p_2 decides whether the detrimental step is to be accepted or not. Therefore, the simulated annealing algorithm, besides accepting improvement in the value of the objective function, also accepts deterioration in the value of the objective function which is not so in local search algorithm. Initially, at large value of C_k, large deterioration will be accepted; as C_k decreases, only smaller deterioration will be accepted and finally, as the value of C_k approaches zero, no deterioration will be accepted. Also, a large step size is taken initially and it is decreased slowly after fixed number of iterations along with the control parameter values. This procedure is continued until the control parameter reaches a specified lower limit which can be used as a stop criterion. The pseudo-code for simulated annealing algorithm is given in Fig.1.

```
Procedure SIMULATED _ANNEALING
Begin
    Initialize X_1, C_0, ΔX, L_0;
    k = 0;
    X_0 = X_l
    Repeat
        for l = 1 to L_k
        begin
            GENERATE_NEIGHBOR (X_N, X_0, ΔX) ,
            if (f(X_N) < f(X_0)) then X_0 = X_N
            else begin
                p_1 = exp[−{|f(X_0) − f(X_N)|}/C_k];
                p_2 = exp[−{|f(X_T) − f(X_N)|}/C_k];
                u = random [0, 1];
                if (u < wf(p_1) and (u < wf(p_2)) then X_0 = X_N;
            end;
        end;
        k := k + 1;
        CALCULATE_LENGTH (L_k);
        CALCULATE_CONTROL (C_k);
        CALCULATE_STEP (ΔX) :
    until stop_criterion
end;
```

Fig.1: Pseudo-code for Simulated Annealing Algorithm (Function Minimization).

In this pseudo-code , the procedure GENERATE_NEIGHBOR determines the next feasible random direction by changing the values of the variables by ΔX. There are many ways of generating the neighborhood (see Aarts and Korst 1989).

One simple method which we have used here is given by

$$x'_i = x_i + \frac{d_i \Delta x_i}{\sqrt{\sum_{j=1}^{n} d_j^2}} \quad \text{for } i = 1, 2, ..., n$$

such that

$$\begin{aligned} X_n &= (x'_1, x'_2, ..., x'_n) \\ X_0 &= (x_1, x_2, ..., x_n) \end{aligned}$$

where

$d_i =$ Normal random number with mean 0 and standard deviation 1
$n =$ Number of optimizing variables
$\Delta x_i =$ Step size for ith optimizing variable.

Procedure CALCULATE_LENGTH (generally increases L_k as k increases) determines the number of iterations for which the control parameter is C_k, CALCULATE_CONTROL(decreases C_k as k increases) calculates the value of kth change in the value of the control parameter, and the procedure CALCULATE_STEP (decreases ΔX as k increases) determines the value of the step size for which the control parameter value is C_k.

Simulated annealing is a generalized heuristic algorithm which can handle large class of problems irrespective of the nature of the objective function as in the case of classical techniques. Some modifications for the basic simulated annealing algorithm have been made. Interested reader may refer to Aarts and Korst (1989) and Eglese (1990). For an extensive discussion and bibliography, reference may be made to Johnson (1988).

Example 7.1: Consider that two types of weapons are available to defend three assets against two types of attacking weapons. Let us suppose that the maximum number of available defending weapons of the first type is 100 and that of the second type is 50. The number of attacking weapons of the first and second type are 50 and 29 respectively. The values of the first, second and third assets are 400, 300 and 200 respectively. Effectiveness of defending weapons and damage pobabilities of attacking weapons are given in Table 1. Determine an optimal defense plan against the known attack plan using simulated annealing technique assuming the attack plan to be $n_{11} = 5,\ n_{12} = 9,\ n_{21} = 25,\ n_{22} = 7,\ n_{31} = 20$ and $n_{32} = 13$.

Table 1: Effectiveness Values of Defending Weapons and Damage Probabilities of Attacking Weapons.

Defending Weapon Type (d)	*Asset* (s)	*Attacking Weapon Type* (a)	k_{dsa}	g_{sa}
1	1	1	0.20	0.015
2	1	1	0.60	0.015
1	1	2	0.35	0.055
2	1	2	0.50	0.055
1	2	1	0.25	0.075
2	2	1	0.50	0.075
1	2	2	0.20	0.040
2	2	2	0.45	0.040
1	3	1	0.35	0.060
2	3	1	0.45	0.060
1	3	2	0.25	0.075
2	3	2	0.65	0.075

This problem is basically an integer programming problem but to solve it through simulated annealing technique, it is considered as a continuous variable problem. The value of L_0 is assumed to be 200 and it is successively incremented by 100 for every increment in the value of k. The value of control parameter C_0 is assumed to be 0.5 and it successively decrements by 0.005 with every k until it reaches 0.005. The initial step size ΔX is assumed to be 1.0 and it decrements successively by 0.1 until it reaches the value 0.1.

The illustrate the procedure, let the initial feasible solution be generated by using uniform random numbers. The initial solution consisting of the number of weapons of dth type to be deployed on asset s against attacking weapon of type a (x_{dsa}) is calculated by using the relation

$$x_{dsa} = u_{dsa} \times (B_d/2.0) \tag{6}$$

where u_{dsa} is a uniform random number between 0 and 1 and B_d is the number of dth type of defending weapons available. On generating the values of x_{dsa}, it is checked for weapon availability constraint. If the constraint is satisfied, the solution is accepted as initial feasible solution for simulated annealing algorithm. Otherwise, another set of values for x_{dsa} is generated using the above equation and this procedure is continued until feasible initial solution is obtained. For example, let us suppose that the set of twelve random numbers generated are:

$$\begin{array}{llll} u_{111} = 0.5139, & u_{112} = 0.1757, & u_{121} = 0.3086, & u_{122} = 0.5345 \\ u_{131} = 0.9476, & u_{132} = 0.1717, & u_{211} = 0.7022, & u_{212} = 0.2264 \\ u_{221} = 0.4948, & u_{222} = 0.1247, & u_{231} = 0.0839, & u_{232} = 0.3896 \end{array}$$

then the value of x_{dsa} obtained from the equation 6 are

$$\begin{array}{llll} x_{111} = 25.69, & x_{112} = 8.79, & x_{121} = 15.43, & x_{122} = 26.73 \\ x_{131} = 47.38, & x_{132} = 8.59, & x_{211} = 17.56, & x_{212} = 5.66 \\ x_{221} = 12.37, & x_{222} = 3.12, & x_{231} = 2.10, & x_{232} = 9.74 \end{array}$$

Therefore the number of weapons of first type is equal to 132.61 and second type is equal to 50.55. These many number of weapons are not available and hence the weapon availability constraint is not satisfied. Another set of random numbers is then generated and the weapon availability constraint is checked. After few trials, the following set of random numbers is generated:

$$\begin{array}{l} u_{111} = 0.0136,\ u_{112} = 0.7396,\ u_{121} = 0.4183, u_{122} = 0.3620 \\ u_{131} = 0.2039, u_{132} = 0.1831, u_{211} = 0.0763,\ u_{212} = 0.1155 \\ u_{221} = 0.1591,\ u_{222} = 0.7883,\ u_{231} = 0.0403,\ u_{232} = 0.7906. \end{array}$$

The values of x_{dsa} can be read from the first row of Table 2. The value of the objective function $f(X_0)$ where $X_0 = (x_1, x_2, ..., x_n)$ is 61.44 per cent.

The next step is to generate the neighborhood solution from the initial solution. There are many ways of generating the neighborhood. One simple method used here is given by

$$x'_{dsa} = x_{dsa} + (\Delta X \times d_{dsa}) / \sqrt{\Sigma d^2_{dsa}} \tag{7}$$

where d_{dsa} is a normal random number with mean equal to 0 and standard deviation equal to 1. The x'_{dsa} values are checked for constraint satisfaction and if it not satisfied, a new set of values are generated. This procedure is continued until we get feasible neighborhood. For example, let the set of standard random normal deviates be

$$\begin{array}{llll} d_{111} = -1.15, & d_{112} = -0.95, & d_{121} = -0.37, & d_{122} = -2.34 \\ d_{131} = 0.96, & d_{132} = 1.19, & d_{211} = 0.27, & d_{212} = -1.58 \\ d_{221} = 0.78, & d_{222} = -1.18, & d_{231} = 1.67, & d_{232} = -1.56 \end{array}$$

and by substituting them in the equation 7, we get the x'_{dsa} values which are feasible (see second row of Table 2). The objective function value $f(X_N)$ where $X_N = (x'_1, x'_2, ..., x'_n)$ is evaluated for the feasible neighborhood and is compared with $f(X_0)$. If $f(X_N)$ is greater than $f(X_0)$, the solution stored in X_N is copied on to X_0. Otherwise, p_1 and p_2 are evaluated as explained above. Uniform random number $u[0, 1]$ is generated and is compared with p_1 and p_2. If $u < p_1$ and $u < p_2$ then solution X_N is copied on to solution

X_0(*i.e.*, detrimental step is accepted); otherwise, solution X_N is rejected, and the procedure is continued with another feasible neighborhood solution. This procedure is continued for L_k number of iterations and then the value of k is incremented. The number of iterations L_k, the control parameter C_k and the step size ΔX are calculated as explained above and the procedure is repeated till the stop criterion (when there is no further improvement) is satisfied.

Table 2. Results for Twenty Iterations of Simulated Annealing

Itn	Defense Plan (x_{dsa})												Obj_Func.	Status
No	111	112	121	122	131	132	211	212	221	222	231	232	Value	
0	0.68	1.91	36.98	2.89	20.92	3.98	18.10	19.71	10.19	1.01	9.16	19.77	0.61440	
1	0.42	36.77	20.83	17.58	10.41	9.42	1.97	2.54	4.15	19.44	1.38	19.42	0.61353	Accepted(d)
2	0.35	37.14	20.71	18.18	10.35	9.24	2.12	2.79	3.85	19.74	1.08	19.69	0.61420	Accepted
3	0.55	36.69	20.89	18.68	10.78	9.12	2.48	3.02	3.80	19.75	1.27	19.46	0.61609	Accepted
4	0.39	36.65	21.34	18.28	10.36	8.57	2.57	3.02	3.95	19.45	1.27	19.40	0.61645	Accepted
5	0.91	36.58	21.67	18.38	10.41	8.46	3.05	2.48	3.78	19.64	1.22	19.35	0.61826	Accepted
6	0.69	36.32	21.26	18.05	10.95	8.24	2.81	2.30	3.79	19.30	0.99	19.46	0.61534	Accepted(d)
7	0.46	36.76	21.18	17.91	10.84	8.08	2.40	2.75	3.69	19.28	1.40	19.84	0.61482	Accepted(d)
8	0.28	36.42	21.15	18.08	10.73	7.54	2.57	3.16	3.67	19.52	1.00	19.52	0.61589	Accepted
9	0.62	35.97	21.16	18.26	11.24	7.76	2.82	3.16	3.36	19.81	0.70	19.66	0.61533	Accepted(d)
10	0.43	36.33	20.62	18.07	11.34	7.99	3.02	3.01	3.34	20.02	0.30	20.08	0.61515	Accepted(d)
11	0.72	35.97	20.39	18.40	11.42	7.61	2.84	2.58	3.57	20.23	0.02	19.79	0.61357	Accepted(d)
12	0.22	35.86	19.77	1834	11.68	7.65	2.90	2.72	3.76	20.42	0.29	19.46	0.61328	Accepted(d)
13	0.02	35.57	19.52	18.50	11.78	7.81	2.35	2.58	4.04	20.43	0.80	19.74	0.61050	Accepted(d)
14	0.17	35.79	19.21	19.01	11.36	7.77	2.25	2.44	4.09	20.97	0.56	19.63	0.61008	Accepted(d)
15	0.13	35.72	18.95	19.33	11.73	8.10	2.25	2.39	3.41	20.78	0.35	19.44	0.60649	Rejected
16	0.33	36.00	19.67	18.42	11.18	7.32	2.16	2.52	3.89	20.79	0.78	19.57	0.61026	Accepted
17	0.35	36.46	19.10	18.66	10.99	7.11	1.86	2.57	3.49	20.99	0.98	19.52	0.60726	Accepted(d)
18	0.48	36.49	18.50	19.06	11.04	6.97	1.92	2.43	3.49	21.54	0.93	19.20	0.60651	Rejected
19	0.57	36.12	18.88	18.92	11.33	6.82	2.46	2.80	3.55	20.92	0.73	19.34	0.61028	Accepted
20	0.75	36.20	18.66	18.73	11.77	6.30	2.16	2.78	3.58	20.97	0.69	18.78	0.60890	Rejected

Table 2 list results of the first 20 iterations. The initial value of the objective function is 61.44. In the first iteration, feasible neighborhood of the initial solution is obtained and the objective function is evaluated as 61.35 which is less than the initial value. These values are substituted in decision criterion and a uniform random number is generated which decides to accept this detrimental step (denoted by Accepted (d)). Therefore, in the second iteration feasible neighborhood of solution obtained from first iteration is generated and the procedure is continued. After 20 iterations, the current objective function value is equal to 61.02 (solution generated at 19th iteration) because of the fact that the solution generated at 20th iteration is rejected. The best solution obtained so far is equal to 61.83 which is obtained at the 5th iteration.

By continuiting this procedure the optimal defense plan obtained is given in Table 3 which indicates that against the known attack plan, the maximum expected surviving value of the asset is 60.54% of the total asset value. Thus, the optimal defense plan is to deploy 47 weapons of type 1 on asset 1,39 and 19 weapons of types 1 and 2 on asset 2, and 14 and 31 weapons of type 1 and 2 on asset 3, respectively. In these calculations we have considered only weapon availability constraint but constraints on cost, manpower and available area can also be considered. If the attacker is interested in minimizing the value of the surviving assets the same objective function can be minimized to obtain the optimal attack plan (n_{sa}) against a known defense plan.

Table 3. Optimal Defense Plan Obtained through Simulated Annealing

Defending Weapon Type (d)	Asset (s)	Attacking Weapon Type (a)	Optimal Defense Plan
1	1	1	0
2	1	1	0
1	1	2	47
2	1	2	0
1	2	1	39
2	2	1	14
1	2	2	0
2	2	2	5
1	3	1	11
2	3	1	16
1	3	2	3
2	3	2	15

References

[1] Aarts, E.H.L. and Korst, J., *Simulated Annealing for Boltzmann Machines*, John Wiley, New York, 1989.

[2] Beare, G.C., *Linear Programming in Air Defence Modelling, Journal of Operational Research Society,* Vol. 38, No. 10, 899-905, 1987.

[3] Bohachevscky, I.O., Johnson, M.E. and Stein, M.L., Generalized Simulated Annealing for Function Optimization, *Technomatrics,* Vol. 28, 209-217, 1987.

[4] Bohachevscky, I.O., Johnson, M.E. and Stein, M.L., Optimal Deployment of Missile Interceptors, *American Journal of Mathematical and Management Sciences,* Vol.8, Nos. 3&4, 371-387, 1988.

[5] Bracken, J., Brooks, P.S. and Falk, J.E., Robust Preallocated Preferential Defense, *Naval Research Logistics,* Vol. 34, No. 1, 1-22, 1987a.

[6] Bracken, J., Falk, J.E. and Tai, A.J.A., Robustness of Preallocated Preferential Defense with Assumed Attack Size and Perfect Attacking and Defending Weapons, *Naval Research Logistics,* Vol. 34, 23-41, 1987b.

[7] Cohen, M.A. and Grossberg, S., Absolute Stability of Global Pattern and Parallel Memory Storage by Competitive Neural Network, *IEEE Trans. on Systems, Man and Cybernetics*, Vol. 13, No. 5, 815-825, Sep.-Oct., 1983.

[8] Davis, L., *Handbook of Genetic Algorithms,* Van Nostrand Reinhold, New York, 1991.

[9] Eglese, R.W., Simulated Annealing: A Tool for Operational Research, *European Journal of Operational Research,* Vol. 40, 271-281, 1990.

[10] Fogen, D.B., An introduction to Simulated Evolutionary Optimization, *IEEE Trans. on Neural Networks,* Vol.5, No. 1, 3-14, January 1994.

[11] Freeman, J.A. and Skapura, D.M., *Neural Networks, Algorithms, Applications and Programming Techniques,* Addison Wesley, New York, 1991.

[12] Glover, F., Tabu Search-Part I, *ORSA Journal on Computing,* Vol. 1, No. 3, 190-207, 1989.

[13] Glover, F., Future Paths for Integer Programming and Links to Artificial Intelligence, *Computer and Operations Research,* Vol. 13, No. 5, 533-549, 1986.

[14] Glover, F. and Greenberg, H.J., New Approaches for Heuristic Search: A Bilateral Linkage with Artificial Intelligence, *European Journal of Operational Research,* Vol. 39, No.2, 119-130, 1989.

[15] Goldberg, D.E., *Genetic Algorithms in Search, Optimization and Machine Learning,* Addison Wesley, New York, 1988.

[16] Holland, J.H., Genetic Algorithms, *Scientific American*, 66-72, July 1992.

[17] Jaiswal, N.K., Shrotri, P.K. and Nagabhushana, B.S., Optimal Weapon Mix, Deployment and Allocation Problems in Multiple Layer Defense, *American Journal of Mathematical and Management Science,* Vol. 13, Nos. 1&2, 53-82, 1993.

[18] Johnson, M.E. (ed), Simulated Annealing (SA) & Optimization: Modern Algorithms with VLSI, Optimal Design & Missile Defense Applications, *American Journal of Mathematical and Management Sciences,* Vol. 8, Nos 3 & 4, 205-450, 1988.

[19] Kirkpatrick, S., Gelatt C.D., Jr. and Vechhi M.P., Optimization by Simulated Annealing, *Science,* Vol. 220, 671-680, 1983.

[20] Lippmann, R.P., An Introduction to Computing with Neural Nets, *IEEE ASSP magazine,* 4-22, April 1987.

[21] Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H. and Teller, E., Equations of State Calculations by Fast Computing Machines, *Journal of Chemical Physics*, Vol. 21, 1087-1092, 1953.

[22] Miller, J.A., Potter, W.D., Gandham, R.V. and Lapena, C.N., An Evaluation of Local Improvement Operators for Genetic Algorithms, *IEEE Trans. on Systems, Man and Cybernetics,* Vol. 23, No. 5, 1340-1351, Sep.-Oct. 1993.

[23] Nickel, R.H. and Mangel, M., Weapon Acquisition with Target Uncertainty, *Naval Research Logistics Quarterly,* Vol. 32, 567-588, 1985.

[24] Potts, J.C., Giddens, T.D. and Yadav, S.B., The Development and Evaluation of an Improved Genetic Algorithm Based on Migration and Artificial Selection, *IEEE Trans. on Systems, Man and Cybernetics,* Vol. 24, No.1, 73-86, January 1994.

[25] Simpson, P.K., *Artificial Neural Systems, Foundations, Paradigms, Applications and Implementation,* Pergamon Press, London, 1990.

[26] Tagliarini, G.A., Fury, J.F. and Page, E.W., Optimization using Neural Networks, *IEEE Trans. on Computers*, Vol. 40, No. 12, 1347-1358, December 1991.

[27] Wacholder, E., A Neural Network- Based Optimization Algorithm for the Static Weapon-Target Assignment Problem, *ORSA Journal on Computing,* Vol. 4, 232-245, 1989.

[28] Wasserman, P.D., *Neural Computing, Theory and Practice,* Van Nostrand Reinhold, New York, 1989.

[29] Woodruff, D.L., Simulated Annealing and Tabu Search: Lessons from a Line Search, *Computers and Operations Research,* Vol. 21, No.8, 823-839, 1994.

Department of Mathematics and Informatics,
"Mircea cel Bătrân" Naval Academy,
Fulgerului 1
8700 - Constantza
Romania