# Network Device Allocation Optimization Using Genetic Algorithms

1st Jason Michael Anthony Falbo
*R&.D Department*
*Mircom Group of Companies*
Vaughan, Canada
jfalbo@mircomgroup.com

2nd Stéphane Dedieu
*Mircom Group of Companies*
*bloo audio Inc*
Ottawa, Canada
stephane.dedieu@bloo-audio.com

***Abstract*—We present an optimization methodology for a pseudo-adaptive algorithm that assigns IOT devices to a set of networks. The age of The Industrial Internet of Things (IIOT) provides a fantastic opportunity to rethink traditional methods of device assignment amongst heterogeneous network options within an environment. To date, the only way to guarantee performance of a life-safety critical network was to invest in costly fixed and dedicated infrastructure that could not be a shared with non-critical building systems equipment. This paper describes a methodology to tune shared infrastructure communication pathways for reliability approaching fixed infrastructure solutions. The genetic algorithm (GA) developed maximizes devices' and networks' performance while minimizing infrastructure costs. We assume an assessment of each network available to each device in terms of QoS, and our GA running on a central controller at a remote server.**

***Index Terms*—Genetic Algorithms; Industrial Internet of Things (IIOT); Wireless Sensor Networks (WSN); Fire; Life-Safety;**

## I. Introduction

As the number of networks and connected devices grows, we face new challenges in deploying our devices onto unlicensed networks with an ad-hoc approach to network management. Devices are deployed today without coordination of airspace amongst users with competing interests. Many tools available to tune networks involve primitive methods to find a configuration that 'performs' well. The metrics of performance are ill-defined. In this paper we publish initial results of a methodology meant to autonomously balance and re-configure networks supported by quantitative grading of the satisfaction level of the devices assigned to the networks. Through simulations and results we demonstrate improvement of device satisfaction scores across a number of possible network configurations, and compare the results of our GA algorithm vs. exhaustive search and random search algorithms. Many applications are possible. The simplest and most common application could be for autonomous Wi-Fi-network selection in consumer devices. Industrial applications include guaranteed reliability deployments for life safety and health monitoring applications. There exists huge potential in the building controls industry to reduce labor costs of deployment in addition to minimizing service call-backs when deploying ad-hoc wireless networking equipment. The idea for our research arose from real world deployments of mission critical networks. In bringing these solutions to market, we identified a need to quickly and accurately resolve congestion issues in algorithmic fashion. The main goals of optimizing wireless device allocation to available networks in the building is to improve affordability, performance and reliability of intelligent building equipment, thus, driving adoption of life safety systems in emerging and economically depressed markets. Successful results will show the algorithm's capability to:

- support dynamic network selection and re-allocation of devices on networks by the centralized decision making engine
- improve the communication pathways on shared network such that devices are better balanced to support device communication requirements
- optimize affordability in infrastructure solutions with improved network performance in life-safety critical applications.

Fig. 1 describes Typical Devices Deployed in a Modern Intelligent Building. We make the following assumption: Optimization infrastructure exists consisting of devices deployed in the network that are meant to relay data, specifically QoS metrics [1], to the central controller in supporting the optimization algorithm.

- A passive infrastructure object will simply collect network performance data and report it to the central server used by the Genetic Algorithm
- An active infrastructure object can function as part of the main network application at the same time as it collects data on the network for the optimization server.

Passive infrastructure objects are envisioned as low power, low cost, IOT sensors that can be deployed in temporary or permanent manner during the project commissioning phase where network tuning is most critical. In order to minimize

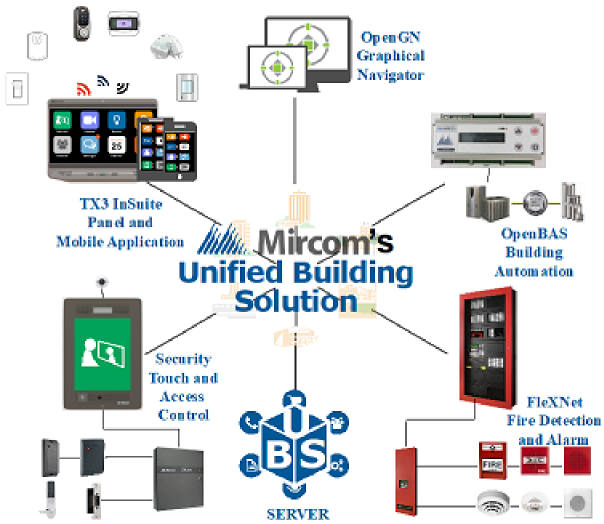


Fig. 1: Intelligent Building Systems.

the deployment costs, the least number of infrastructure objects should be used. The allocation of multiple networks to multiple devices under constraints is a discrete, multi-convex problem. The idea is to find near-optimal solutions in an acceptable computational time. One of the most powerful heuristic approaches is based on Genetic Algorithms (GAs).

## II. Genetic Algorithms

### A. Generalities and Related work

GAs principle is based on natural mechanisms of the genetics. GA mathematics theory is presented in Goldberg (1989) [3]. We will insist on the concept of schemata and on the statistics of the Fitness function.

Mehboob et al. [4] propose a comprehensive literature survey of GA strategies applied to networks resource allocation to devices, while satisfying QoS constraints. Many references relevant to our problem are available in section 3 and 4.

Sherif et al. [5] proposed a GA-based approach to analyze the network bandwidth allocation using a predetermined range of QoS levels (high, medium, low) declared by each multimedia sub-stream. The proposed algorithm assigns the best available *QoS* level to each sub-stream depending upon the availability of bandwidth resources in network.

Ferentinos et al. [6] propose a GA for adaptive wireless sensor network design including network clustering with proper choice of the cluster-head and energy management. The fitness function incorporates many aspects of the network performance subject to energy conservation: density of the network, connectivity, operational energy, battery life. The problem is encoded with a vector and two bits per sensor, describing the type of sensor and its status.

Lin et al. [7] developed a GA optimization strategy for allocating WLAN's resources to patients devices in a hospital, for applications, data and videos. The approach accounts for EMI interference The GA is encoded with a bandwidth vector *B* in a time frame *(k)* and the optimal bandwidth allocation is subject to a set of constraints *(QoS)* including: *SNR*, data stored on each device and transmitted to the server, delays.

### B. Proposed Approach

The originality of our approach is to rely on a *QoS* assessment and ranking of all networks by each device, stored in a score matrix $W$. The $W$ matrix was inspired by the work of Brin and Page in 1998 [2] where they made effort to develop a method for rating Web pages objectively and mechanically. The problem consists of optimizing the device allocations based on "best fit" among networks, subject to a set of constraints. Our algorithm is completed with two local modules, briefly described in section IV. In section V, we introduce a quick statistical analysis of the Fitness function and constraints for determining the probability of finding an optimal solution and adjusting GA parameters. The algorithm is validated with a small size example. Optimization results are compared with an exhaustive search and a random search.

## III. Description of the Methodology

### A. Problem Outline

A Genetic Algorithm assigned to the problem of multiple devices allocation among multiple heterogeneous networks is setup as follows:

- "NetRank" - A quantitative scoring of a network's overall quality based on weighted assessment of QoS metrics
- "DeviceRank" - A quantitative multi-variate scoring of a device's overall impact on NetRank based on weighted assessment of resource consumption
- All accessible networks maximum "netranks" are assigned to a vector *N*. $card(N) = nn$. Note: $N_i = N(i)$.
- All devices' "rank" impact under management are assigned to a vector *D*. $card(D) = nd$. Note: $D_j = N(j)$.
- QoS parameters of the physical network parameters can be measured using tools like *wireshark*, *ping*, *Iperf*, or custom application code
- $QoS(m)$ $m = 1, ...nq$ parameters vector includes for example: burst rate *(BR)*, throughput *(TP)*, Bit Success Rate *(BSR)*, User Network Preference *(UNP)*, Jitter *(JR)*, Round trip Time *(RTT)*, Connection Wait Time *(CWT)*, Network Access Cost *(NAC)*.

### B. Computation of Score Matrix W

Based on their current activity and resource availability, at a time $t_k$ all networks $N_i$ are scored and assigned a Netrank by each device $D_j$. The score is computed by weighting each *QoS* parameter of network $N_i$. Note that weights $\beta_m$ depend on $t_k$, *i, j*. For each $j$, each column $W(;,j)$ is further normalized. The rating process will be detailed in an advanced paper.

$$W(i,j) = \sum_{m=1}^{nq} \beta_m QoS(m) \tag{1}$$

At this point we consider two types of *QoS* parameters.

- "'Independent"' *QoS* parameters, depend on an individual device only, for example distance from an access point *(SNR, BSR, ..)* or network access cost *(NAC)*.

- ”Dependent”’ *QoS* parameters, are degraded by several devices competing for resources i.e. bandwidth. The Impact of allocating several devices to a Network is not obvious to each device but a server with better perspective can employ an algorithm to resolve conflicts by device re-allocation

Independent *QoS* parameter conflicts can sometimes be addressed by the device itself. For example networks are scored “0” when the *SNR* (link quality) is too low. Each device can score dependent *QoS* parameters, but not necessarily optimize for them; in case of conflict, optimization can only be solved from the server, having a picture of all network and device activity. The score information is synthesized in a “devices’ network selection score” matrix $W_k$. For simplifying notations, we drop the time index *k* and assume we work in a time frame $[t_k, t_{k+1}]$. For simplification purposes in this paper, we consider only one dependent *QoS* parameter ’available bandwidth’. An example of matrix $W$ is shown in Fig. 2. Vectors $D$ and $N$ are respectively seeded with $D$’s desired throughput rate (Mb/s) and $N$’s available bandwidth (Mb/s). Selection scores in matrix $W$ are calculated after available bandwidth.

| 240 | 220 | 140 | 110 | 90 | 65 | 50 | 35 | 5 | 4 | *D/N* |
|---|---|---|---|---|---|---|---|---|---|---|
| 0.6 | 0.5 | 0.3 | 0.3 | 0.2 | 0.3 | 0.3 | 0.3 | 0.2 | 0.2 | 500 |
| 0.4 | 0.5 | 0.3 | 0.4 | 0.3 | 0.2 | 0.3 | 0.2 | 0.2 | 0.2 | 250 |
| 0. | 0 | 0.4 | 0.3 | 0.2 | 0.3 | 0.2 | 0.2 | 0.2 | 0.2 | 150 |
| 0. | 0 | 0 | 0 | 0.3 | 0.2 | 0.2 | 0.3 | 0.2 | 0.2 | 100 |
| 0. | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.2 | 0.2 | 10 |

Fig. 2: Score selection matrix *W*.

On a regular basis, (every 30s-60s) the server monitors matrix $W$ and satisfaction level of current devices’ configuration. The optimization algorithm is triggered when conflicts occur and/or when a device is allocated to a low score network.

### *C. Genetic Algorithm (GA) - Encoding*

An “individual” is a configuration of devices $D_j$ allocated to networks $N_i$ available to them. Its chromosome is represented by a vector $X$ as shown in Fig. 3. Elements of $X$ are the indices of Networks $N_i \;\; i = 1, ..nn$ allocated to devices $D_j \;\; j = 1, ..nd$. Note: $max(X(j)) = nn$ and $X_j = X(j)$.

| *D=* | $D_1$ | $D_2$ | $D_3$ | $D_4$ | $D_5$ | $D_6$ | $D_7$ | $D_8$ | $D_9$ | $D_{10}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| *X=* | 3 | 4 | 4 | 4 | 3 | 2 | 5 | 1 | 1 | 1 |

Fig. 3: Individual’s chromosome or configuration vector *X*

### *D. Genetic Algorithm (GA) - Fitness Function*

In the following $\epsilon, \epsilon_1, \epsilon_2$ are small strictly positive real numbers introduced for bounding ratios. For each configuration vector $X$ we extract a score vector $XW$ from matrix $W$:

$$XW(j) = W(X_j, j) \;\; for \;\; j = 1, ....nd \tag{2}$$

Each term $XW(j)$ describes the selection score attributed by device $D_j$ to its current networks $N_i$. Based on vector $X$ in Fig. 3 and matrix $W$ in Fig 2:

| *XW=* | 0 | 0 | 0 | 0 | 0.2 | 0.2 | 0 | 0.3 | 0.2 | 0.2 |
|---|---|---|---|---|---|---|---|---|---|---|

Fig. 4: Configuration *X* selection score vector *XW*

*1) Objective function:* We want to maximize the score of $X$ by re-allocating each $D_j$ to it’s ’best fit’ network. A best fit network must satisfy all of $D_j$’s weighted QoS requirements. For each $D_j$, we define the satisfaction index $S_j$ as the score of $D_j$’s current network vs. the top score of $D_j$’s highest ranked ’best fit’ network:

$$S_j = max(W(:, j)) - XW(j) \tag{3}$$

If all devices are connected to their ’best fit’ network, all terms $S_j = 0$. Based on vector $X$ in Fig. 3:

| *S=* | 0.6 | 0.5 | 0.4 | 0.4 | 0.1 | 0.1 | 0.3 | 0 | 0 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|

Fig. 5: Configuration *X*: satisfaction vector *S*

For all devices $D_j$ in $X$, the objective function $CF$ includes the sum of all satisfaction indexes $S_j$ and is defined as follows:

$$CF = \frac{1}{\sum_{j=1}^{nd} S_j + \epsilon} \tag{4}$$

*2) Constraints:* Assuming configuration $X$ in Fig. 3 and the score vector $XW$ in Fig. 4, two constraints are defined below: (a) the conservation of devices: at all times, all devices must be allocated to a network. In vector $XW$, some devices are not allocated. We introduce the ’sgn’ function in Fig 6.

| *sgn(XW)=* | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|

Fig. 6: Devices accounted for in current configuration X.

For each $X$, the number of devices is stored in a variable $ICD$. The conservation of devices constraint $CD$ is:

$$CD = \frac{1}{nd - \sum_{j=1}^{nd} sgn(XW(j)) + \epsilon_1} \tag{5}$$

(b) bandwidth congestion: Given a configuration $X$, let $u(n) \subset \{1, ...nn\}$ be the subset of indexes of networks subject to congestion and $s_n(m) \subset \{1, ...nd\}$ be the subset of devices connected to each of them. Congestion violations of configuration $X$ are stored in a variable $ICV < 0$:

$$ICV = \sum_{n \in u(n)} \left( N_n - \sum_{j \in s_n(m)} D_j \right) \tag{6}$$

$X$ is admissible only if $ICV = 0$ i.e. the subset $u(n)$ is empty. The congestion constraint $CV$ is defined as:

$$CV = \frac{1}{|ICV| + \epsilon_2} \tag{7}$$

We define the Fitness function $F$. We can add and weight functions $CF$, $CD$, $CV$. $CV$ is in general way "smaller" than $CD$ and $CF$ and would require some scaling. In this paper, we link both constraints with a proper choice of the bounding parameters: $\epsilon_1 = \epsilon_2 = 0.1$ and $\epsilon = \epsilon_1 \epsilon_2$.

$$F = \alpha CF + (1-\alpha)\, CD \times CV \quad 0 \leq \alpha \leq 1 \tag{8}$$

$F$ is maximized when devices are allocated to their 'best fit' networks and $ICD = 0$ & $ICV = 0$. $F$ theoretical maximum is 100. For formatting the graphs when plotting $F$ we will consider: $20 log_{10}(F)$ in *dB*. Maximum= 40 *dB*.

## IV. Description Of The Algorithm

The GA population is stored in a matrix $XPOP(np, nd)$. GA parameters are:

- *np*: number of individuals $X$ in the population
- *ncf*: number of couples crossover
- *nmut*: number of mutations
- α: weight of objective function $CF$
- *Ts*: threshold norm of satisfaction vector $S$
- *Tcg*: threshold "mild" congestion violation

The algorithm, described in Fig. 7, works with the usual GA operators. Special features are described in section V.

**Input:** At time $t_k$ download vectors $X, N, D, S_k$ score matrix $W$

**Output:** Optimal configuration: $X_{opt}$ Satisfaction vector $S_{k+1}$

If $\|S_k\|_2 > Ts$

1. Order $N, D$, in descending order. Pre-process score matrix $W$.

2. Quick random search and statistics analysis on constraints and Fitness random variables.
   - compute probability of finding a solution: "solution probability"
   - Set GA Parameters: *np, ncf, nmut, Tcg.*

3. Randomly select $XPOP1$, set of configurations $X$

while iteration $kga < Kmax$ or $(ICV < 0 \; and \; ICD > 0)$

4. Compute Fitness $F$ for all individuals in $XPOP$ as per eq. (2)-(8)

5. Select new population $XPOP$ based on normalized Fitness $\phi_k$

6. Crossover operator

7. Mutation operator

end

8. The individual with the strongest Fitness, $Xopt$ is a solution.

9. Local operator *LO1*: If congestion violations $ICV < Tcg$, *LO1* reallocates devices until no device can find network with headroom.

10. Local operator *LO2*: Rearrangements of devices for improving vector $S$ without congestion violation.

If $\|S_{k+1}\|_2 < Ts$ end.

Fig. 7: Description of the algorithm.

All parameters depend on the size of the problem to be optimized, $D, N, W$. *np* should be kept as small as possible.

## V. Simulations and Results

At time $t_k$, we assume vectors $D, N$ and matrix $W$ in Fig. 2 and vector $X$ in Fig. 3. Note $\|S\|_2 \approx 1$. The "utilization rate" is: $\sum D_j / \sum N_i = 95\%$. The size of the problem is small, the GA can be validated with an exhaustive search. All simulations are run with Matlab. Processor: IntelCore i7-7700HQ, CPU@ 2.8Ghz, RAM 8.0GB.

### A. Exhaustive Search Algorithm EXST

The *EXST* algorithm is built with a generator of all permutations of devices $D_j$ on networks $N_i$ with repetitions. Total number of configurations: $KC = 5^{10}$.
In Fig. 10 & 11 we plot the Fitness function $F$, the objective function $CF$, and physical constraints violations $ICD$ and $|ICV|$. Fitness $F$ maximums are $\approx 77 (38 \; dB)$ meaning that maximums of $CF$ and constraints $CD, CV$ were not reached at the same time. $F$ is discrete and multi-convex which justified the use of a heuristic algorithm in the first place. The number of maximums and their "size" depend on vector $D, N$ and matrix $W$. In the present example, there are 8 very localized maximums. Schemata $Sc_i \in Sc$ involved in the best solutions are linked to the optimal allocation of the largest devices $D_1$, $D_2$, $D_3$, $D_4$ to largest networks $N_1$,$N_2$, $N_3$. Two are shown in Fig. 8.

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $Sc_1$ | 1 | 1 | 2 | 2 | * | * | * | * | * | * |
| $Sc_8$ | 2 | 1 | 3 | 1 | * | * | * | * | * | * |

Fig. 8: Schemata linked to absolute maximums.

The probability of finding a configuration $X$ belonging to one of these 8 schemata is $P(X \in Sc) = 8 \times 5^6 / 5^{10} = 0.012$. The GA is able to extract schemata rapidly. They carry thousands of configurations with lower Fitness, surrounding the absolute maximums. In Fig. 11, we can see a relatively large area under iteration $10^6$ with congestion violations up to 200 Mb/s. After extraction of related schemata and a threshold *Tcg*=-200, the local module is likely to converge towards one of the maximums. Based on the distributions of $F$ and the constraints, we compute the "solution probability", (*EXST* results in Table I).

### B. Random Search RS

Based on convergence considerations, a sample of $2.10^3$ configurations is generated at random. Fig. 12 & 13 show the Probability Distributions (PDs) of random variables $Y_F$, $Y_{ICD}$, $Y_{ICV}$. In Fig. 12, $Y_F$ varies between 0 and 1.4, very far from $F$ maximums. The graph always looks like a Gamma distribution strongly shifted to the left. Depending on its shape and scale factor, valuable information about the Fitness function can be extracted: density/"size" of maximums. In large problems, along with the joint distribution $(Y_{ICD}, Y_{ICV})$ in Fig. 13, it is the only information available for determining the existence of solutions and for adjusting GA parameters. In Fig 13, the PD of $Y_{ICV}$ always resembles a binomial or normal distribution. Intuitively: $ICV$ collects sum of combinations of $kd = 1, 2, ..nd$ devices, $\binom{nd}{kd}$. These are binomial coefficients. We will try to demonstrate that result and extract a PD. The interest is that for large problems, even for a low sample Random Search, the PD of $Y_{ICV}$ can be locally extended by continuity close to zero and give access to the "solution probability" (necessary condition). For a low sample Random Search, $Y_F$

maximum value is always very far from $F$ maximums. Its PD cannot be used for computing the "solution probability". With *EXST* and *RS* we find the following probabilities (local extension for *RS*):

TABLE I: EXISTENCE OF A SOLUTION. PROBABILITIES.

| | *EXST* | *RS* |
|---|---|---|
| $P(Y_F \geq 76.5)$ | $1.5\ 10^{-5}$ | *N/A* |
| $P(Y_{ICD} = 0 \mid Y_{ICV} \leq -9.5)$ | $3.3.10^{-5}$ | $6.8\ 10^{-5}$ |

In both cases, *EXST, RS* we find the same order of magnitude. The "solution probability" is low. Probability to find a good schema is way higher and the best solution found with the quick *RS* belongs to schema $Sc_1$. This configuration can be introduced in the GA mutation operator for accelerating the convergence. Currently the "acceleration option" is available for small/medium size problems.

### *C. Genetic Algorithm GA*

Based on results in Table I, GA parameters are set such as *np*, *nmut*, *Tcg* are large for increasing the probability of extracting good schemata and converge to a maximum:
*np*=40, *ncf*=12, *nmut*=8, α=0.25, *Tcg*=-200 *Ts*=0.25.
We run the GA algorithm multiple times. The success rate is about 50%. Including the schema found with the quick *RS*, the success rate is close to 100%. A successful run is shown in Fig. 14 & 15, no "acceleration option", *LO1* was not triggered, *LO2* couldn't improve vector *S*. The results are in Fig. 9.

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| *D* | 240 | 220 | 140 | 110 | 90 | 65 | 50 | 35 | 5 | 4 |
| $X_{opt}$ | 1 | 2 | 3 | 1 | 4 | 1 | 1 | 1 | 3 | 3 |
| *S* | 0 | 0 | 0 | 0.1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Initial available bandwidth *N* | | | | | | 500 | 250 | 150 | 100 | 10 |
| Optimal Allocation with *Xopt* | | | | | | 500 | 220 | 149 | 90 | 0 |

Fig. 9: Genetic Algorithm - Results.

The Satisfaction index is improved $\|S\|_2 = 0.1$. In practice at time $t_{k+1}$, the devices will return a matrix $W_{k+1}$. New iterations can be performed if $\|S_{k+1}\|_2 > T_s$.

Table II shows a synthesis of algorithms performance. For the GA we consider 50% successful runs, 10 iterations on average i.e about 800 configurations. The performance improvement over a 3 $10^4$ samples Random Search is significant.

TABLE II: ALGORITHMS PERFORMANCE

| | *EXST* | *Random Search* | *Genetic Algorithm* |
|---|---|---|---|
| *Fitness* | 38 dB | 38 dB | 38 dB |
| *Configurations* | 9.8 $10^6$ | 3 $10^4$ | 800 |
| *CPU time* | 480 s | 1.7 s | 0.12 s |

We have tested ways to improve the GA performance when utilization rate is high $> 95\%$:

- through the mutation operator, include schemata of the best Random Search solutions, "acceleration option".
- run the GA several times, collect best solutions and introduce them in the mutation operator in next runs.
- pre-process score matrix $W$ when a device $D_j$ scores all networks about the same (low variance), increase its scores of lower rank networks.
- weight each terms $S_j$ with $Max(W(:,j))^p \ \ 0 \leq p \leq 1$.
- if the solution probability is too low, get rid of a set of low priority devices (introduce devices "weighting").

If the solution probability is too high, the Genetic Algorithm is not really required. A quick Random Search and the local operator *LO1* can quickly converge to an optimal solution. This has happened with large size examples and utilization rate under 75%.

## VI. CONCLUSIONS

We proposed an innovative optimization algorithm using real-time assessment of network QoS metrics by connected devices. The results show improved performance compared to initial static configurations largely used in practice today.

- The approach was validated with a small size example and an absolute criteria.
- *EXST* algorithm is an interesting tool for visualizing the Fitness function, understand schemata and improve the Genetic Algorithm.
- A quick random search and statistics analysis of $F$ and constraints allows an approximation of the "solution probability" and proper tuning of GA parameters, both dependent on $D$, $N$, $W$, utilization rate.

We will investigate in-depth and report all GA improvements tested at the end of section V. The approach must now be "calibrated" for large size problems: "solution probability", fine tuning of GA parameters, thresholds *Tcg, Ts*, results sensitivity to $D$, $N$, $W$. Also, we will introduce more dependent *QoS* in the optimization process.

## REFERENCES


[1] Winargo Sugeng, Jazi Eko Istiyanto, Khabib Mustofa, Ahmad Ashari, "The Impact of QoS Changes towards Network Performance" International Journal of Computer Networks and Communications Security Vol. 3, n. 2, February 2015, 48–53

[2] Sergey Brin, Lawrence Page, "The Anatomy of a large-scale hypertextual Web search engine", Compyter Networks and ISDN Systems 30, 1998 - 107-117.

[3] David Goldberg, "Genetic Algorithms in Search, Optimization, and Machine Learning" Hardcover. Addison-Wesley Professional; 1 edition (Jan. 1 1989)

[4] Usama Mehboob · Junaid Qadir · Salman Ali · Athanasios Vasilakos "SURVEY: Genetic algorithms in wireless networking: techniques, applications, and issues" Published online: 18 February 2016 © Springer-Verlag Berlin Heidelberg 2016 Soft Comput (2016) 20:2467—2501 DOI 10.1007/s00500-016-2070-9

[5] Sherif MR,Habib IW,Nagshineh M,Kermani P (2000) "Adaptive allocation of resources and call admission control for wireless atm using genetic algorithms". Sel Areas Commun IEEE J 18(2):268-–282

[6] Konstantinos P. Ferentinos, Theodore A. Tsiligiridis "Adaptive design optimization of wireless sensor networks using genetic algorithms" . Journal of Computer Networks 51 (2007) 1031–1051

[7] Lin D, Labeau F (2012) Accelerated genetic algorithm for bandwidth allocation in view of EMI for wireless healthcare. In: Wireless communications and networking conference (WCNC), 2012 IEEE. IEEE, pp 3312–3317

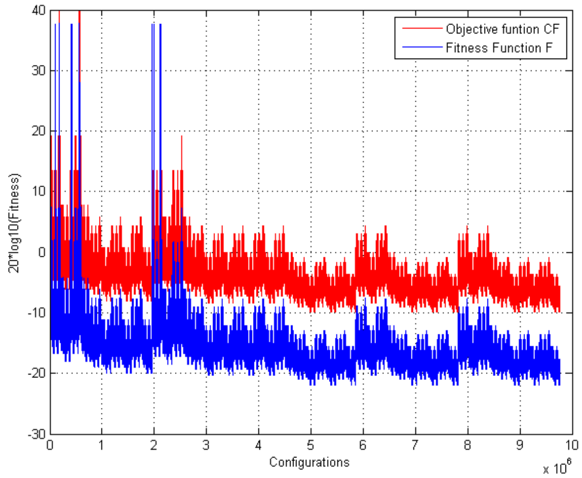


Fig. 10: *EXST*: Fitness function $F$

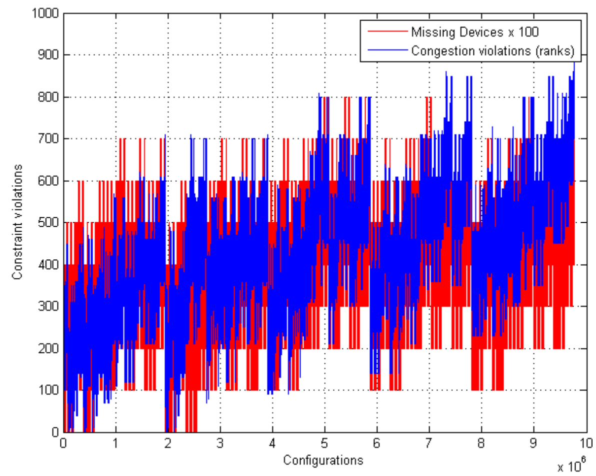


Fig. 11: *EXST*: Constraints: $100 \times ICD, |ICV|$

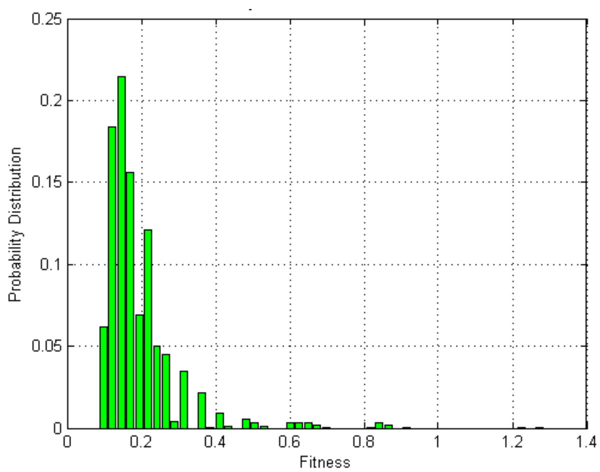


Fig. 12: *RS*: Probability Distribution $Y_F$

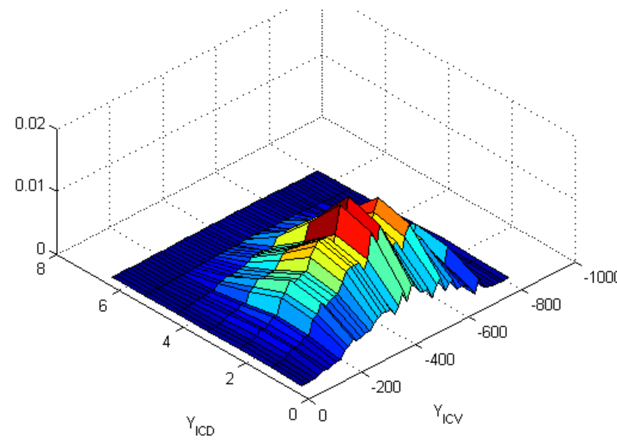


Fig. 13: *RS*: Joint probability distribution $(Y_{ICD}, Y_{ICV})$

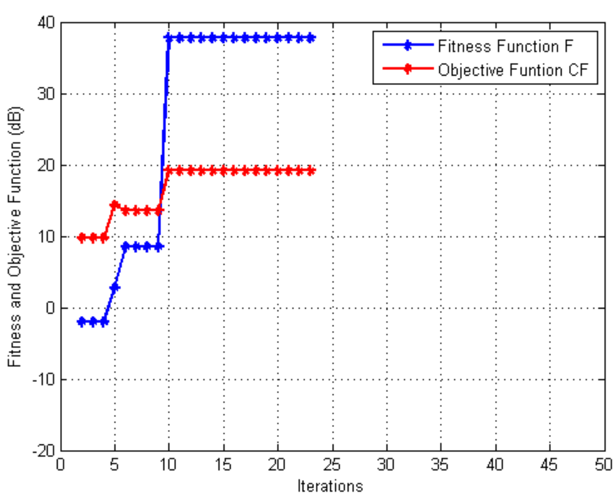


Fig. 14: *GA*: Fitness v. iterations

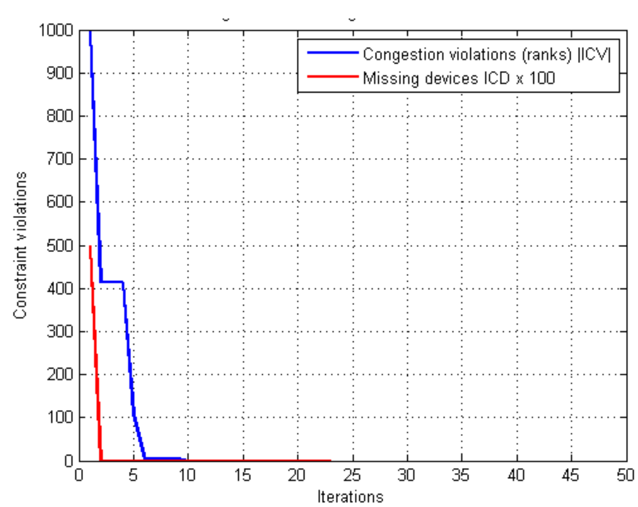


Fig. 15: *GA*: Constraints *100*ICD,* |*ICV*| v. iterations