International Journal of Computer Information Systems and Industrial Management Applications.
ISSN 2150-7988 Volume 4 (2012) pp. 598-608
© MIR Labs, www.mirlabs.net/ijcisim/index.html

Optimizing ATM Cash Management by Genetic Algorithms

Roberto Armenise[1], Cosimo Birtolo[1], Eugenio Sangianantoni[1], and Luigi Troiano[2]

[1]Poste Italiane S.p.A. – TI - RS - Centro Ricerca e Sviluppo,
Piazza Matteotti 3, 80133 Naples, Italy
{*armenis5, birtoloc, e.sangianantoni*}*@posteitaliane.it*

[2]University of Sannio – Department of Engineering,
Viale Traiano, 82100 Benevento, Italy
troiano@unisannio.it

Abstract: **The paper investigates the optimization of ATM cash by means of genetic algorithm in order to produce optimal upload strategies able to minimize the daily amount of stocked money, still assuring cash dispensing service. We provide experimental evidence of benefits obtained by the proposed approach in terms of lower financial costs and more rational cash management. In particular we considered as test bench a pool of 30 ATMs with different characteristics in terms of location, position, cash capacity and usage, representative of the whole set of ATMs operated by Poste Italiane S.p.A.**

Keywords: ATM Cash Management, Genetic Algorithms, Optimization.

I. Introduction

An efficient management of cash can reduce financial costs due to unused stocked cash. This is particularly true in the case of automatic teller machines (ATMs) which operate as automatic cash dispensers. However predicting cash demand is challenging due unpredictability of withdrawals, but profitable because of the large number of tellers. Indeed, ATMs are ubiquitous especially in urban areas. It has been estimated that their number exceed 1.6 million worldwide [1]. Stocking cash in ATM entails costs that can be broadly divided in two contributions: financial costs and operational costs [2]. The first are mainly due to unused stock rated by annual passive interests, while time to perform and supervise the task, maintenance, out-of-service and risk of robbery are associated to the second.
Therefore performing an efficient refill of ATMs able at the same time to minimize the daily amount of stocked money and to assure cash dispensing service, is highly desirable, especially for large financial institutions, as outlined by recent studies [3, 4]. A survey aimed at understanding which ATM features are most desired by buyers [4], collected responses obtained by financial institutions worldwide, based in North America (36%), Europe (23%), Asia (19%), Middle East/Africa (12%), Latin America (8%) and Australia (2%). Respondents (206 in 2009, and 243 in 2010) were asked to choose three top desired features of next generation ATMs. According to this study, Tab.1, cash management and forecasting placed 3rd after remote monitoring and cross-platform multivendor ATM software in 2010, moving from the 6th place in the 2009 ranking.

Features	**2010**	**2009**
Remote monitoring of the ATM network	49%	45%
Multivendor ATM software	31%	22%
Cash management and forecasting	28%	22%
Envelope-free check deposit	25%	24%
One-to-one marketing/purchase gift cards	24%	26%
Customer preferences	23%	26%
Bulk-note cash deposit	22%	23%
Software distribution	22%	28%
Support for cash recycling	21%	21%
Automated test tools	15%	18%
Support for biometrics	13%	13%
Other	6%	7%

Table 1: The three most desired new features of ATM software (Percentage of financial institutions including the features among their top three priorities) [4].

Generally ATM cash management and optimization is performed manually, according to corporate policies and personnel experience. A non-optimal cash upload can lead to poor service when cash demand is underestimated and to un-

necessary costs when demand is overestimated. Therefore, finding the best match between cash stock and demand becomes crucial to improve.
According to Simutis et al. [5], effective cash management should rely on advanced algorithms able to accurately predict cash supply and demand, allowing banks to pro-actively manage the usage of money throughout their network. Recently, some authors attempted to optimize the cash by modeling and forecasting the demand. However, the high variance and non-stationarity of the underlying stochastic process can affect reliability of such an approach.
The more genral problem of modeling cash flow has been first investigated by Miller and Orr in 1966 [6], concluding that the cash flow is mostly unpredictable. According to their model, the cash balance fluctuates irregularly over the time. Later research [7] confirmed Miller and Orr conclusions, so that the lack of demand visibility is usually considered a major challenge in cash management and optimization. Therefore the approach based on forecasting withdrawals in the following days on the basis of stochastic modeling of demand, so that cash is uploaded in order to fit the future demand, is hard to implement due to high unpredictability and non-stationarity of demand.
In this paper we suggest the application of genetic algorithms as means for searching and generating optimal upload strategies, aimed at identifying a set of uploading rules able to minimize the residual stock and to guarantee service availability at the same time. This paper is organized as follows: Section 2 briefly overviews ATM cash management; Section 3 defines the problem and describes the proposed solution; Section 4 reports experimental results; Section 5 outlines conclusions and future directions.

II. ATM Cash Management

Cash management is generally related to problems in which the inventory level of cash can either increase or decrease, due to both external flows and financial decisions taken at the beginning of each period. By contrast ATM cash management generally allows only decisions regarding the increase of inventory level, so that only cash uploads are considered.
Cash drawing is a stochastic process whose characterization tends to change over the time due to several factors. For instance demand is subject to change according to period of time, position of ATM, socio-economic features of users. In addition, the quantity of cash drawn is generally larger before holidays. Cash drawing follows weekly, monthly and annual cycles. People tend to cash-out money during paydays or at the beginning of each month or at the end of each week.
Whatever, the demand of cash is not only influenced by time, but it follows different trends that make modeling even more difficult. For example, how holidays affect the use of ATMs depends on where the teller is located. So that, vacation destinations register an increase of demand during holidays, while ATMs in shopping centers are mostly used on Friday and Saturday.
In general, ATM cash management deals with finding the minimal amount of cash able to meet the demand over an ATM network. Management generally relies on human experience, often able to outperform automated software solutions. Some drawbacks can be identified in commercially available solutions. Model parameters are generally static and do not adapt during the operation, so that on-line cash optimization is not made possible. Moreover, cash forecasting is largely based on linear regression models with seasonality coefficients customized for each ATM. This makes parametrization time consuming.
Similarly to other large financial institutions, Poste Italiane holds a widespread networks of ATMs, whose cash is manually managed by skilled personnel. Therefore ATMs are uploaded following human experience and corporate policy, instead of an optimized strategy. In particular, Poste Italiane recommend to:

1. Guarantee availability of cash in each ATM, avoiding out-of-service status
2. Upload ATM at least two times a week respecting the fixed upper bound of money available in the ATM
3. Avoid two or more system alerts a week due to running out cash

Innovative approaches propose the adoption of neural network as means for predicting future cash demand. Among the different proposals, Dijonas et al. [2] elaborated a model based on combination of neural networks and multi-agent technology. In particular, data are gathered by agents from ATM network, and delivered to neural network for prognosis and optimization. The idea is to use the neural network to map the relationship between various factors influencing the cash demand. So, the input variables for neural network are weekday, day of the month, month of the year, holiday effect value and average cash demand for ATM in last week. The output is the demand of cash predicted for the next time interval. In a previous work, Simutis et al. [5] simulated the behavior of an ATM network made of 1,225 units in two different scenarios, at different annual interest rates and cash uploading costs. They showed that neural approach allowed to keep the average forecasting error of daily cash demand under 10%. In addition, optimization procedure allowed to decrease daily cost for ATM network approximately by 18% in both scenarios. In another work, authors compared Artificial Neural Network (ANN) approach to Support Vector Regression (SVR) [1]. ANN and SVR models were trained using data recorded by 15 real ATMs along 2 years. In their experimentation, forecasting error was between 15-28% (ANN), and 17-40% (SVR).
In order to improve prediction, it could be useful to group ATMs so that common behavior is emphasized and spurious behavior discarded. Seedig and Runkler [8] adopted a Fuzzy C-Neural Network Model (FCNNM) which combines

fuzzy clustering with recurrent neural networks. Data regarded daily withdrawals of about 30 ATMs during a period of roughly two years, provided by a European bank involved in a project at Siemens AG. Their approach is organized in two steps: (i) cash withdrawals are predicted and (ii) if accurate, find the optimal date for refilling the ATM. As experimentation, the authors focused on four time series, each specific of one ATM. The number of clusters was 2 and the neural network architecture was a time delay-recurrent neural network. The time series entailed a period of 620 days and the model used the past 28 days (4 weeks) to predict the following 7 days (1 week).
Instead of driving the cash refilling strategy by forecasting the future demand of cash, decisions could be made upon conditions regarding time, location, cash residual and other factors that could lead to consider a cash upload. In other terms we could be interested in finding a set of rules that if applied to current operation of ATMs can lead to the decision of when and how upload the machine. Therefore premises are conditions on some factors (e.g. time, inventory level, location, etc.) and conclusions are a set of possible recharges to apply when premises are met. The goal is to find the set of rules able to minimize the daily money left stocked in each ATM and to guarantee availability of service at the same time, so that machines do not go in out-of-service status. We call this approach Condition Based Cash Management (CBCM).
In this paper we adopt Genetic Algorithms (GA) as means to search the best solution. Although a rule can be well represented by tree-based chromosomes, we prefer GA to Genetic Programming (GP) based solutions (e.g. see [9, 10, 11]), as rules in our case have a regular structure, so that GA can better exploits such a regularity, in contrast to GP that is able to deal with more general structures but at cost of searching a larger space.
Similarly to other contributions in this are, we apply this approach to both individual ATMs [2, 5, 1] and to grouped ATMs [8].

III. Problem Definition

Each rule is structured as a conjunction of given conditions, entailing a level of recharge. For example a possible rule could be,

$$\texttt{IF}\ p_1(a_q,t)=c_{1,h}\ \texttt{AND}\ \ldots\ \texttt{AND}\ p_n(a_q,t)=c_{n,k}\ \ \texttt{THEN RECHARGE}\ l_m \tag{1}$$

meaning that if the ATM a_q at time t meets conditions $p_1(a_q,t)=c_{1,h}$, $p_2(a_q,t)=c_{2,k}$ and $p_n(a_q,t)=c_{n,k}$, then a_q is recharged of cash l_m.
Each condition represents one possible value predicate p can assume when applied to a specific ATM and date. Examples are “Summer holiday”,“Workday”, “Day before a festivity”, “Low amount of money in ATM”, etc.), while examples of recharge levels are “Upload at maximum”, “Upload your ATM of x euros”, etc. Formally,

$$p_i : ATM \times D \to C_i \tag{2}$$

where ATM is the set of possible ATMs, D the set of dates, and C_i is the set of possible outcomes of p_i.
Therefore, given a set of predicates, rule specifications belong to the space

$$R \equiv C_1 \times C_2 \times \ldots \times C_n \times L \tag{3}$$

where L is the set of possible recharge levels. A rule set is specified by $K \subseteq R$. We aim to find the subset of K that when applied to a single ATM or to group of them, minimizes the daily average exceeding stock (S) within the time interval $t_0 \ldots T$, that is w.r.t. ATM a_q defined as

$$S(a_q,t) = s_0(a_q) + \frac{1}{t-t_0}\sum_{\tau=t_0..t} l(a_q,\tau) - d(a_q,\tau) \tag{4}$$

where date $t \in t_0..T$, $s_0(a_q)$ is the initial stock, $l(a_q,\tau) \in L$ is the recharge applied at date τ if any, $d(a_q,\tau)$ is cash withdrawn at date τ. In particular, the level of recharge l is determined by applying the rule set specified by K. If no rule can be applied to a_q at time τ, then $l(a_q,\tau)=0$ and no recharge is applied.
Stock S should never exceed, the maximum load capacity of ATM a_q. In order to prevent this case, the quantity $l(a_q,\tau)$ is trimmed so that maximum capacity is never exceeded. In addition S should never go below 0, as this case entail out-of-service of a_q. We model this latter case by considering a contingency rule. This rule, states that if cash run out, a given extra refill (e.g. 50% of maximum load) is provided so that ATM can continue operations. Therefore the daily balance in Eq.(4) becomes

$$\lceil l(a_q,\tau)\rceil - d(a_q,\tau) + f(\tau)l_c \tag{5}$$

assuming f extra recharges l_c at date τ.
Given a subset $A \subseteq ATM$ the overall average daily stock, w.r.t. a rule set K is

$$S_K(t) = \sum_{a_q \in A} S(a_q,t) \tag{6}$$

Since we are interested to minimize the stock along the whole period of interest, the goal is to minimize $S(T)$, and problem can be stated as

$$K = \arg\min_{K \subseteq R} S_K(T) \tag{7}$$

A. Algorithm

The problem is to search the specification space R in order to find an optimal specification subset K that when applied to ATMs in A within the interval $t_0..T$ is able to minimize the

Algorithm 1 GA pseudo-code

```
1: Generate a population of random specifications
2: repeat
3:     Evaluate fitness of each specification
4:     Select specifications for reproduction
5:     Cross specifications selected
6:     Mutate some specifications
7: until $maxgen$ generations
```

overall daily average $S_K(T)$. In this paper we experimented genetic algorithms as means to explore R. Algorithm employed in this paper (see Algorithm 1) is inspired to the Simple GA as given by Goldberg [12].
When the algorithm is instanced, an initial population of rule specification sets is randomly generated. These specifications provide the set of condition and recharge target values required to build rules. After, rules so far specified are applied to the set of ATMs in order to find out what is the overall daily average stock $S_K(T)$. Thus, solutions are ranked and selected to reproduce according to their cash stock, so that rules sets able to minimize stock are privileged. Specifications obtained so far are pair crossed and some of them mutated. In particular, we adopted tournament selection, single point crossover and simple mutation as genetic operators. After $maxgen$ generations the best individual is obtained.

B. Chromosome

A chromosome represents a possible set of rule specifications. Each specification it is a point in $R \equiv C_1 \times C_2 \times \ldots \times C_n \times L$. Therefore it can be identified by an index $I \in 0..H_n - 1$, assuming

$$H_i = \prod_{i=1}^{i} B_i, \; i = 1..n \qquad H_0 = 1 \tag{8}$$

where $B_i = card(C_i)$. Instead, we assume $K = card(L)$. Index can be computed as

$$I = \left(\sum_{i=1}^{n} c_i H_{i-1} \right) + l H_n \tag{9}$$

where $c_i \in 0..B_i - 1$ and $l \in 0..K - 1$. Eq.(9) can be inverted so that given index I, it is possible to obtain the tuple $(c_1, \ldots, c_n, l)$, where l provides the recharge level and computed as

$$l = I / H_n \tag{10}$$

noting an integer division by the symbol by / symbol. Similarly, if we assume $r_{n+1} = I \bmod H_n$, we can recursively compute

$$c_i = r_{i+1} / H_{i-1} \qquad r_i = r_{i+1} \bmod H_{i-1} \tag{11}$$

Therefore, a solution is made of a list of indexes, each pointing to an element in R. In this paper we consider two possible codings of solutions: Extended and Compact Chromosome. In the first coding we adopt a chromosome made by a sequence of H_n integers (see Fig.1, in which a represent a generic value ranging from -1 and $L - 1$). Each value provides a recharge level, whilst the index if decoded provides the conjunction of predicates to be used as premise. When the value is -1 the rule is not considered by the strategy.

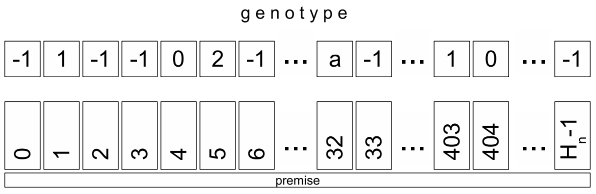

genotype
-1 1 -1 -1 0 2 -1 ... a -1 ... 1 0 ... -1
0 1 2 3 4 5 6 ... 32 33 ... 403 404 ... H_n-1
premise

Figure. 1: Extended Chromosome (EC).

The main drawback of Extended Chromosome is length, that makes slower the convergence towards an optimal strategy. This problem becomes more evident by increasing the number of predicates, cardinality of possible conditions and the number of recharge levels. In many cases of practical interest, we can limit the number of rules to be considered in strategy. For this purpose, we considered the Compact Chromosome made of a finite list of indexes, as depicted by Fig.2. Each value, ranging from 0 to $H_n K - 1$ represents a specification in R.

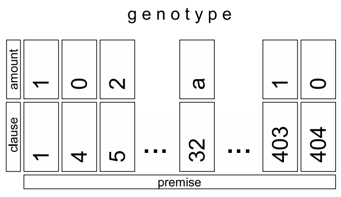


Figure. 2: Compact Chromosome (CC).

C. Fitness Function

The fitness function of an individual x is aimed at minimizing the stocked money in ATM and guaranteeing at the same time the cash-dispensing service for customers.

$$fitness(x) = \frac{1}{T} \cdot \sum_{i=1}^{T} (s_{i-1} - d_i + u_i(x)) \tag{12}$$

where T is the number of days in training phase, s_i is the amount of stocked money in ATM, s_0 is the initial stocked money in the target ATM, d_i is cash demand during day i, while u_i is the amount of cash-in flow.

IV. Experimentation

A. Equipment and Materials

The algorithm implementation made use of JENES [13], an open source Java library for genetic algorithms. Tests run on Intel Xeon 2.66 GHz machine with 4 GB of RAM equipped with operating system Windows Server 2003 Enterprise Edition Service Pack 2.

B. Data Collection

Experimentation made use of data collected by a pool of 30 Poste Italiane ATMs, chosen to provide a wide coverage of different kind of Poste Italiane ATMs. Poste Italiane S.p.A. is a leading operator of postal services and an innovative and competitive player in the arena of financial and payment services. It provides the public mail service and it boasts a network of nearly 14,000 post offices and 5,900 ATMs. Selected ATMs are placed in different Italian cities and in different areas and characterized by different transaction volumes. ATMs differ for location (i.e. urban area, shopping center, tourist location or other strategic areas), position (i.e. Throught-The-Wall ATM or Lobby) and cash capacity. Indeed, Poste Italiane assigns ATMs to 6 different classes, each entailing a maximum limit to cash load. Our sample contained at least 3 ATMs per class. We divided the whole set of 30 ATMs in two groups: first group contains 20 ATMs, while second group 10, with at least one representative for each class. ATMs recorder at least 70% uptime, that is the time period of service availability for cash withdrawals. Among them, 26 ATMs (80%) recorded an uptime higher than 92%. We referred to data from July 1st, 2009 to April 30th, 2010. In particular we split data in two intervals: from July 1st, 2009 to December 31st, 2009 (Period 1), and from January 1st, 2010 to April 30th, 2010 (Period 2). This led to split data in four quadrants as shown in Tab. 2.

	From July 1st, 2009 to December 31st, 2009 (Period 1)	**From January 1st, 2010 to April 30th, 2010 (Period 2)**
20 ATMs	Training Set	Testing Set 1
10 ATMs	Testing Set 2	Testing Set 3

Table 2: Training and Testing Sets.

C. Execution and Results

Each test run the algorithm along 1000 generations on populations made of 500 individuals and was setup with standard parameters[1] as follows: Tournaments 2, Crossover 0.8, Mutation 0.1, Elitism 1. Given the cash limit M, the stocked cash S, the cash upload levels employed are $L = 6$:

[1]Parameter have been chosen by the simple qualitative analysis, according to common values adopted for them, without any in-depth quantitative analysis for their optimization.

- Upload until the limit M is reached
- Upload 67% of the difference between maximum allowed stocked money M and current stocked money S
- Upload 50% of the difference between maximum allowed stocked money M and current stocked money S
- Upload 33% of the difference between maximum allowed stocked money M and current stocked money S
- Upload 10% of M
- Upload 20% of M

Furthermore, conditions referred to date (5) and to current stocked cash (6), so that $H_n = 5 \cdot 6 = 30$. In particular, conditions on dates regarded patterns of workdays (W) and holidays (O) with respect to the current date. They are reported in Tab.3.

	Pattern					
ID	D_{-1}	$\mathbf{D_0}$	D_1	D_2	D_3	Description
1	W	**W**	W	X	X	A generic W
2	O	**W**	W	X	X	W after a holyday
3	X	**W**	O	W	W	W before a short holyday
4	X	**W**	O	O	X	W before a long holyday
5	X	**W**	O	W	O	W before an extended holyday

Table 3: Time Statements.

Conditions on current availability of cash are 6: (i) Stocked cash is less than 20% of M, (ii) Stocked cash ranges between 20% and 30% of M, (iii) Stocked cash ranges between 30% and 40% of M, (iv) Stocked cash ranges between 40% and 80% of M, (v) Stocked cash ranges between 80% and 90% of M, and (vi) Stocked cash is greater than 90% of M.

We performed two quantitative analysis: analysis of convergence aimed at validating the algorithm and proving consistency of results; and analysis of performance aimed at assessing the advantages in relying decision making on genetic approach instead of human experience. Quantitative analysis was organized in test groups as reported in Tab.4.

Experiment 1 is aimed at comparing convergence of algorithm when Extended Chromosome (EC) and Compact Chromosome (CC) are employed. We collected the outputs of the 6 simulations and we compared fitness values of the best solutions at each generation. In Tab.5 are shown average value and standard deviation of the fitness values obtained by the 10 different runs in the case of EC and CC.

We performed Wilcoxon paired test. Looking at p-values, and assuming 0.05 as upper limit to reject the null hypothesis and 0.50 as lower limit to accept it, we can state that in some cases (ATMs 1,2,5) EC and CC behave in the same way (p-value bigger than 0.8). That means that obtained solutions are statistically equivalent, thus CC is able to provide good solutions similarly to EC. In other cases (ATMs 3,4) EC is

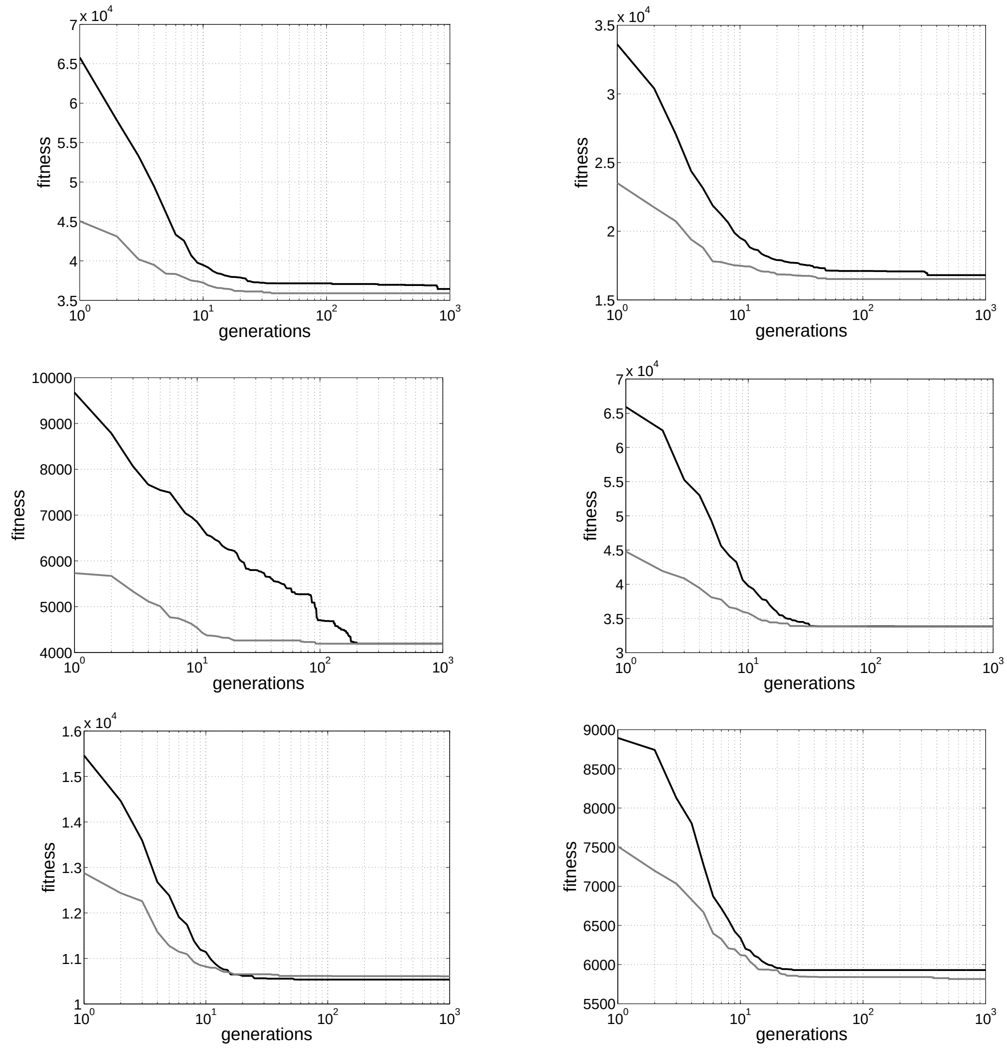


Figure. 3: EC (black) vs. CC (grey): Mean Fitness of best individual.

definitely better than CC (p-value is below 0.001) as it tends to keep less cash stocked. In remaining cases, we are not able to reach any conclusion from tests.

In order to understand how the coding affects the algorithm convergence along the different generations, Fig.3 outlines the solution fitness, i.e. the effect of the best upload strategy on stocked cash, of other 6 ATMs (each chosen randomly from each class). We can deduce that CC provides better solutions starting from lower fitness value than EC.

Analysis of performance is aimed at comparing the results offered by our approach compared to historical upload actions as performed by humans. Results offered by runs of Experiment 2 are reported in Tab.6. We can notice how genetic algorithm constantly offer solutions that lead to an average daily stocked cash that is lower than those entailed by human action.

Goodness of results are confirmed by Experiment 3, aimed at testing strategies found in the past when they are applied to a future period within an horizon of 4 months. Again, historical cash management performed worse than the upload strategy suggested by the algorithm, as outlined by Tab.7. Therefore, we can state that strategies found in the training phase are consistent and valid also when applied in the following period, well facing non stationarity of cash demand.

Experiment 4 and 7 are aimed at evaluating the possibility of training the algorithm with respect to groups of ATMs in order to find a more general strategy, thus less sensitive to specific characteristics.

In Experiment 4 we clustered ATMs in 7 classes corresponding to different allowed maximum limit of available cash. For each group of ATMs we found an uploading strategy that can be applied at every ATM belonging to the same class (i.e., Group strategy). We trained the algorithm on Period 1 and we tested the solutions for 10 different ATMs on the Period 1 (Experiment 5) and Period 2 (Experiment 6). In particular, in Fig.4 we compared the performance of human upload-

ID	ATM1	ATM2	ATM3	ATM4	ATM5	ATM6	ATM7	ATM8	ATM9	ATM10
Class	7	5	4	6	7	3	1	2	6	2
Run 1	35623.97	17683.37	12669.78	36899.08	38389.35	10756.80	4191.79	6608.64	33928.10	10349.02
Run 2	37040.43	16751.25	12998.80	37140.27	36664.02	10196.90	4191.79	5728.15	34122.99	10492.28
Run 3	36392.77	16614.89	12669.78	37273.53	37528.80	10197.66	4191.79	5900.33	33928.10	10391.79
Run 4	36475.92	16790.87	13492.61	37140.27	36355.65	9699.29	4191.79	5728.15	33279.08	10946.85
Run 5	36641.96	16152.28	12875.16	42355.71	36355.65	9644.40	4191.79	5685.87	33928.10	10492.28
Human	76252.01	68575.49	46332.82	77521.84	86214.67	34855.19	11256.46	18456.73	53606.63	19269.34

ID	ATM11	ATM12	ATM13	ATM14	ATM15	ATM16	ATM17	ATM18	ATM19	ATM20
Class	7	5	3	3	4	6	3	3	4	1
Run 1	29597.22	17630.43	10322.88	8662.55	12835.16	33140.92	9159.40	9633.75	13060.92	4683.21
Run 2	29597.23	17941.85	10456.09	8605.38	13713.10	33793.10	8770.71	9374.29	13201.79	4705.38
Run 3	29597.23	17460.54	10176.14	8605.38	12835.16	33140.92	8877.12	9361.52	12848.53	4820.87
Run 4	29597.23	17630.43	10428.91	8605.38	12835.16	33793.10	9559.46	9361.52	13083.42	4653.75
Run 5	29597.23	17630.43	10155.92	8826.74	12835.16	33861.39	8770.71	9626.79	12990.49	4744.08
Human	52756.91	58520.16	31673.42	36897.86	45019.94	64975.43	22005.05	36726.35	42120.27	14113.31

Table 6: Experiment 2, average daily stocked cash of strategies in the Training Set (Period 1).

ID	ATM1	ATM2	ATM3	ATM4	ATM5	ATM6	ATM7	ATM8	ATM9	ATM10
Class	7	5	4	6	7	3	1	2	6	2
Run 1	52116.33	23193.16	15564.83	120691.41	45762.66	14111.25	4976.66	7668.66	35019.83	10519.25
Run 2	50965.75	18050.00	17752.58	49713.08	50319.25	11858.41	4874.58	6373.00	40630.33	9812.08
Run 3	49064.75	15688.16	14921.25	44970.75	43338.41	16754.00	15924.83	6747.75	34507.50	10332.41
Run 4	43366.91	16032.00	16514.91	39272.25	50922.58	11784.91	4874.58	6697.00	33584.00	9611.91
Run 5	49348.00	18372.66	15375.50	44261.41	41005.08	12930.25	5180.83	6463.00	34808.33	10474.00
Human	77549.75	66643.41	48501.41	72117.41	89386.08	34147.25	12186.08	18897.66	56355.50	19474.66

ID	ATM11	ATM12	ATM13	ATM14	ATM15	ATM16	ATM17	ATM18	ATM19	ATM20
Class	7	5	3	3	4	6	3	3	4	1
Run 1	170188.25	20918.91	20918.91	11170.00	22635.83	38102.58	10348.66	19690.66	15822.00	6423.16
Run 2	184535.16	22043.08	22043.08	11302.91	15628.25	93247.50	10670.50	22569.16	15771.33	6447.41
Run 3	183037.91	21060.25	21060.25	11549.41	16627.25	37729.08	10486.58	23942.00	16855.83	6462.75
Run 4	180264.33	23010.58	23010.58	11389.41	19668.75	33062.41	10913.16	19736.08	19153.16	6401.16
Run 5	36027.58	22427.25	22427.25	11547.08	17611.25	34484.83	10670.50	18653.58	25240.08	5905.50
Human	54181.73	54677.20	32083.16	34751.41	45090.75	67033.61	25068.66	38056.98	38988.83	17094.75

Table 7: Experiment 3, average daily stocked cash of strategies in the Testing Set 1 (Period 2).

ing strategy (reported in Tab.6 where average daily stocked cash is depicted in the related row) and Group strategy (median values of the 5 different runs depicted in Tab.6 for every ATM) by means of a scatter plot.
Comparison of Group strategy and human one is provided in Fig.5 where we tested the 7 discovered uploading rules (a single rule per group of ATMs) for 10 different ATMs. Considering the same ATM, computer-based Group strategy outperformed human-based strategy in both Period 1 and Period 2 even if different ATMs (i.e., out-of-sample ATMs) are taken into account (see Fig.5).
Results confirmed that ATM cash management can improve by application of genetic algorithm, leading to a lower amount of stocked cash.

Finally, Experiment 7 considered the whole set of ATMs to train the algorithm, testing resulting strategies in Period 1 (see Fig. 6). In this case, we added predicates to rules able to distinguish ATMs. We let the algorithm to find a generic strategy. In spite of that, the algorithm failed in most of cases showing a behavior worse than human approach, in Training (Experiment 7) and Testing Sets (Experiment 8). This result is according to extreme heterogeneity of ATMs cash demand profiles. Therefore, an a priori classification of ATMs is mandatory in order to model different ATM classes and to get reliable results. Moreover, observing Fig.7, we proved that a Specific strategy (i.e., an uploading strategy for each ATM) increase the goodness of prediction if it is compared to a general one.

Experiment	Description
1	The algorithm is trained on 5 ATMs belonging to Training S using both EC (10 runs per ATM) and CC (10 runs per ATM). Convergence is studied.
2	The algorithm is trained and tested w.r.t. each ATM (5 runs per ATM) belonging to Training Set.
3	Solutions found in Group 2 are tested on Testing Set 1.
4	Training Set is clustered in k classes of similar ATMs and the algorithm is trained and tested in Period 1 (5 runs per ATM group).
5	ATMs in Testing Set 2 are associated to the k classes of Group 4 and tested in Period 1.
6	Same classification as Group 5, but tested w.r.t. Testing Set 3 (Period 2).
7	The algorithm is trained and tested considering the whole set of ATMs in Training Set (5 runs).
8	Solutions found in Group 7 are tested on: (i) Testing Set 1, (ii) Testing Set 2, (iii) Testing Set 3.

Table 4: Experiments.

	EC		CC		
ATMs	Average	Std. Dev	Average	Std. Dev	p-value
1	45729.9	3075.1	47432.8	6299.2	1
2	42239.8	1715.3	43296.1	2963.4	9.68e-01
3	7073.4	151.6	8728.0	1474.9	1.83e-04
4	14240.4	1402.8	17674.3	2667.8	8.61e-04
5	16830.5	637.3	16500.2	2116.3	8.80e-01
6	9300.3	1078.5	4680.3	81.8	1.75e-04

Table 5: EC vs. CC: Mean best solution and its standard deviation of 10 different runs at generation 1000.

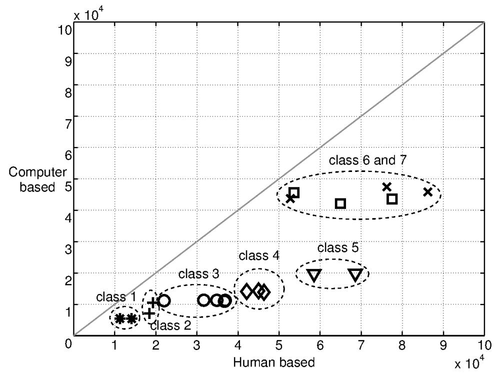
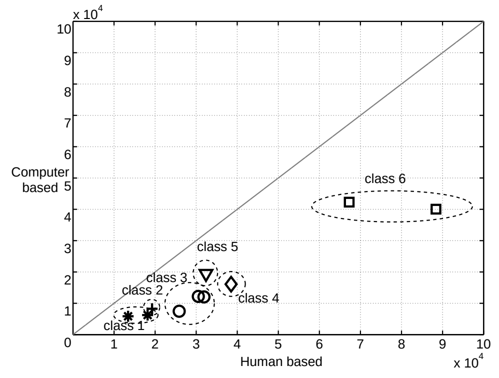


Figure. 4: Group strategy vs. Human strategy: Performance on Training set.

With the purpose of outlining the recharge strategy, we chose one ATM from first group. Fig.8 outlines the daily stocked cash in Period 1 and Period 2. Although different codings (i.e. EC and CC) are adopted, we obtain the a similar behavior of genetic approach (in the first figure EC and CC



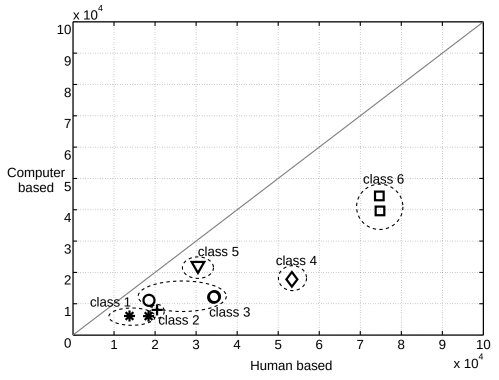


Figure. 5: Group strategy vs. Human strategy: Comparison on Testing set 2 (on the top) and Testing set 3 (on the bottom).

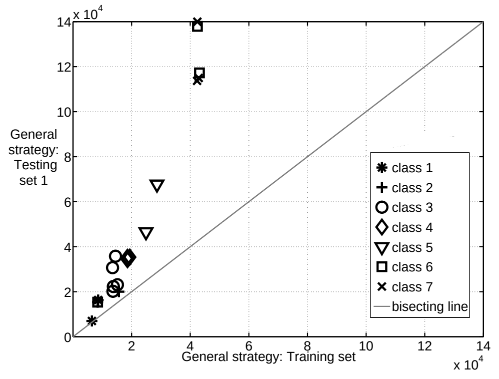


Figure. 6: General strategy: Comparison of the quality of prediction in Training set and in Testing set 1.

overlap), improving in both cases historical cash stocks. In particular, in this case the optimal solution is reached uploading ATM when the daily cash stock reaches the 20% of the allowed maximum limit of available cash in specific days, such as $ID2$ and $ID3$ (see Tab.3)

Observing how proposed approach differs from human behavior, we notice the frequency and the amount of cash uploading operations. In particular the number of cash uploading operation complies with Poste Italiane's guidelines and the proposed algorithm guarantees at the same time cash

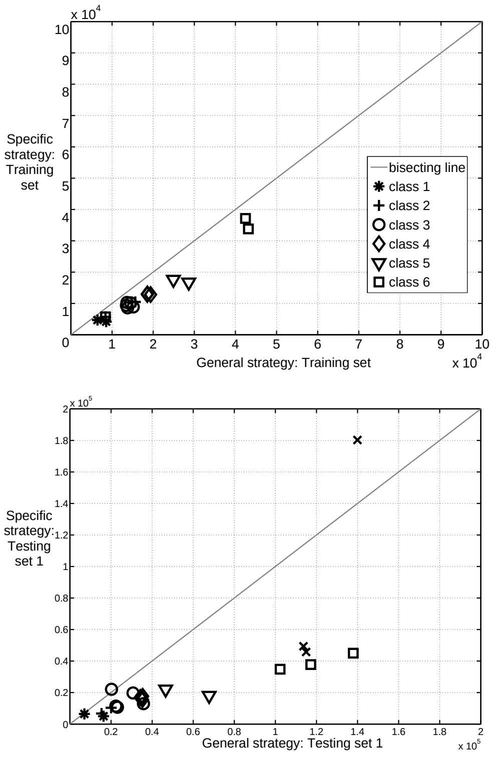


Figure. 7: Finding a general strategy: Comparison between general strategy and specific strategy in Training set (on the top) and Testing set 1 (on the bottom).

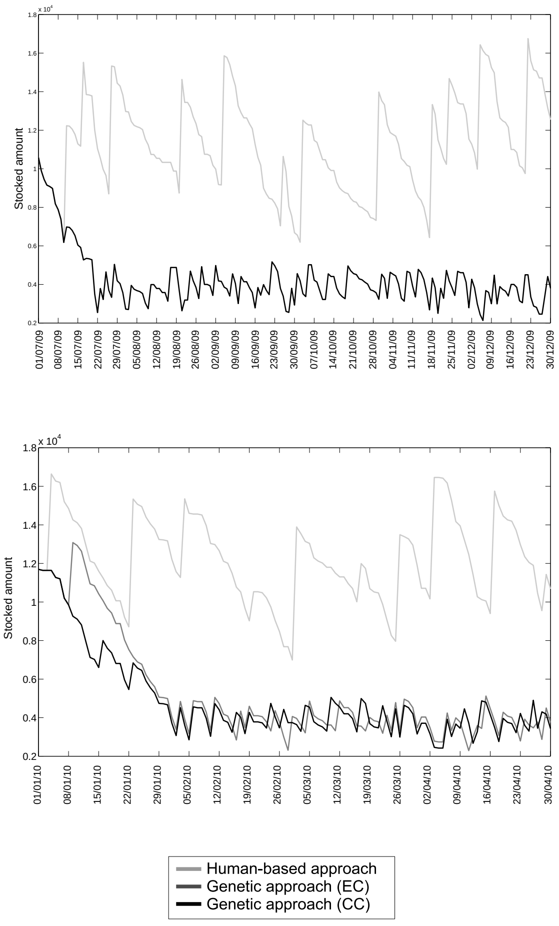


Figure. 8: Comparison between human-based (grey curve) and computer-based (black curve) uploading strategies by analyzing daily amount of stocked cash (in euros) for an ATM with low cash capability (Class 1).

availability and a minimized amount of stocked cash (see Fig.9).
In order to confirm our findings we reported in Fig.10 and in Fig.11 daily amount of stocked cash of other 2 ATMs. In particular, Fig.10 presented an ATM with a standard cash capability and located in a small town, while Fig.11 presented an ATM with a high cash capability and located in a big town.

V. Conclusions

Cash management and forecasting is becoming an important feature of ATM networks. Generally this problem has been studied in terms of stochastic modeling of future cash demand, on which to size the ATM cash upload. Main limitations come from high unpredictability and non-stationarity of demand. In this paper we presented a genetic algorithm aimed at searching optimal strategies to refill ATM cash stocks on the basis of conditions, that if met suggest to refill the ATM cash of a given amount. Experimental results proved this approach to be feasible and able to improve cash management if compared to human expertise, even for a limited extent of time. Since genetic algorithms are computationally expensive, scalability was taken into account by a further experimentation which assessed how performances are affected when the uploading strategy is determined for a single ATM, a group, or the whole network. In future, we aim at investigating the multi-objective evolutionary optimization as means to take into account the frequency of uploads and related costs.

Acknowledgments

The authors would like to thank Eugenio Manfredonia and Giovanni Capasso of R&D Center of Poste Italiane, whose contributions made this work possible.

References

[1] R. Simutis, D. Dilijonas, and L. Bastina, "Cash demand forecasting for ATM using neural networks and support vector regression algorithms," in *EurOPT 2008 - Proc. of the 20th Euro Mini Conf. on Continuous Optimization and Knowledge-Based Technologies*, May 20-23, 2008, pp. 416–421.

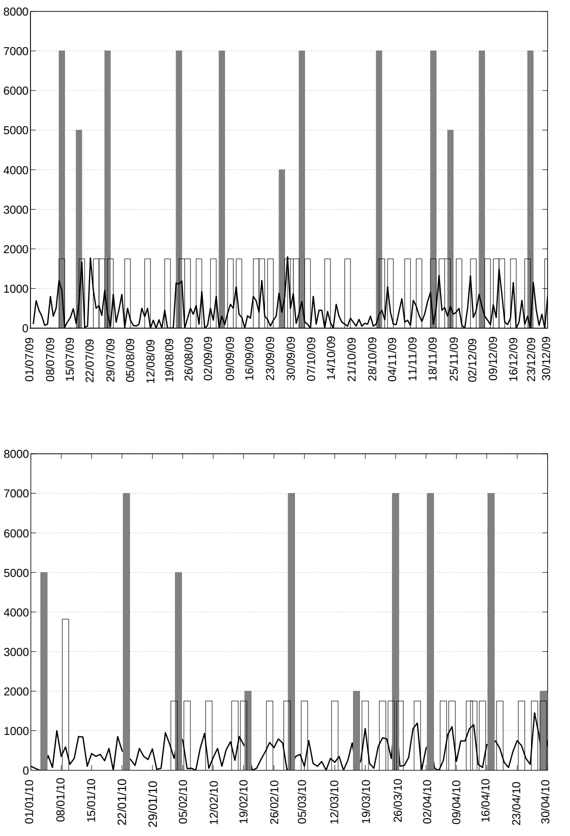


Figure. 9: Comparison between human-based (dark grey bars) and computer-based (light grey bars) uploading strategies adopted in order to satisfy cash demand (black curve).

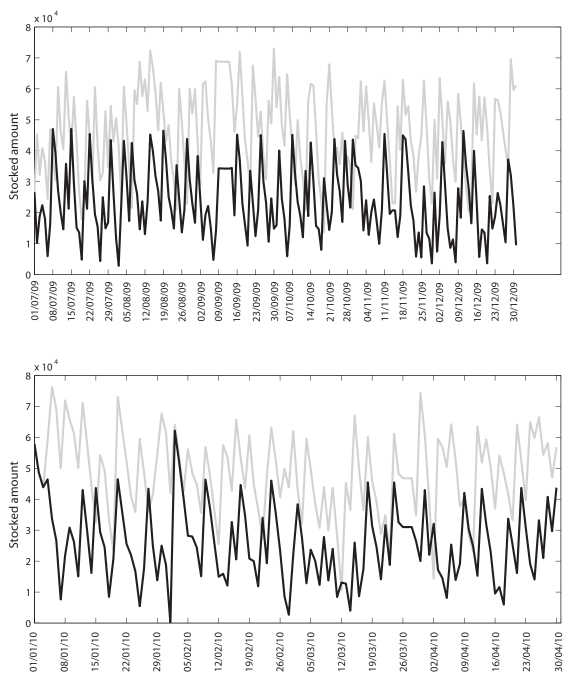


Figure. 10: Comparison between human-based (grey curve) and computer-based (black curve) uploading strategies by analyzing daily amount of stocked cash expressed in euros for an ATM with standard cash capability (Class 4).

[2] D. Dilijonas, V. Sakalauskas, D. Kriksciuniene, and R. Simutis, "Intelligent systems for retail banking optimization - Optimization and management of ATM network system," in *ICEIS 2009 - Proc. of the 11th Int. Conf. on Enterprise Information Systems, Volume AIDSS*, J. Cordeiro and J. Filipe, Eds., May 6-10, 2009, pp. 321–324.

[3] H. Snellman and M. Virn, "ATM networks and cash usage," Bank of Finland, Research Discussion Papers 21/2006, November 2006.

[4] ATMmarketplace.com, "2010 ATM software trends and analysis," NetWorld Alliance, Tech. Rep., 2010.

[5] R. Simutis, D. Dilijonas, L. Bastina, J. Friman, and P. Drobinov, "Optimization of cash management for ATM network," *Information Technology And Control, Kaunas, Technologija*, vol. 36, no. 1A, pp. 117 – 121, 2007.

[6] M. H. Miller and R. Orr, "A model of the demand of money for firms," *Quart. J. Econ.*, vol. 80, pp. 413–435, August 1966.

[7] I. M. Premachandra, "A diffusion approximation model for managing cash in firms: An alternative approach to the miller-orr model," *European Journal of Operational Research*, vol. 157, no. 1, pp. 218–226, 2004, smooth and Nonsmooth Optimization.

[8] H. Seedig, R. Grothmann, and T. Runkler, "Forecasting of clustered time series with recurrent neural networks and a fuzzy clustering scheme," in *Neural Networks, 2009. IJCNN 2009. Int. Joint Conf. on*, June 14-19, 2009, pp. 2846 –2853.

[9] T. Yu, S.-H. Chen, and T.-W. Kuo, "Discovering financial technical trading rules using genetic programming with lambda abstraction," in *Genetic Programming Theory and Practice II*, Una-May O'Reilly, Tina Yu, Rick L. Riolo, and Bill Worzel, Eds. Ann Arbor: Springer, 13-15 May 2004, ch. 2, pp. 11–30.

[10] J.-Y. Potvin, P. Soriano, and M. Vallée, "Generating trading rules on the stock markets with genetic programming," *Computers and Operations Research*, vol. 31, no. 7, pp. 1033–1047, June 2004.

[11] Y. L. Becker and U.-M. O'Reilly, "Genetic programming for quantitative stock selection," in *GEC '09: Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation*. New York, NY, USA: ACM, 2009, pp. 9–16.

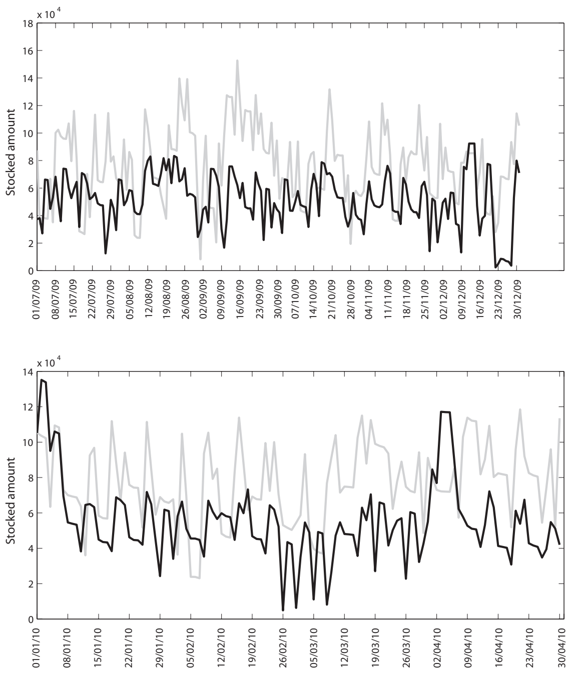


Figure. 11: Comparison between human-based (grey curve) and computer-based (black curve) uploading strategies by analyzing daily amount of stocked cash (in euros) for an ATM (Class 7).

[12] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley, 1989.

[13] L. Troiano and D. De Pasquale, “A JAVA library for Genetic Algorithms addressing memory and time issues,” in *Proc. of the World Congress on Nature Biologically Inspired Computing, 2009. NaBIC 2009*. IEEE, Dec. 2009, pp. 642 –647.

Author Biographies

Roberto Armenise He was born in Bari, Italy, in June 1978. He received the master’s degree in Electronic Engineering in 2006 at Politecnico di Bari, with specialization in microelectronics. He works at R&D Center of Poste Italiane and now is focusing on research activities concerning information technology field. His work is focused on evolutionary computation applied in industrial domain.

Cosimo Birtolo He was born in Martina Franca, Italy, in September 1979 and received the master’s degree in Electronic Engineering in 2006 at Politecnico di Bari, Italy. He is a PhD student in IT Engineering at University of Sannio and works as IT Specialist at R&D Center of Poste Italiane. He matured research interests on Computational and Intelligent systems. Since 2010, he has been participating as Member at UPU Standards Board activities for the development of technical and communications standards.

Eugenio Sangianantoni He was born in Avellino, Italy, in January 1969 and lives in Pagani. He graduated in Computer Science in 1996 at University of Salerno, Italy. He began his career as a consultant, initially, at metal industries and then at financial institutions. Now he works at R&D Center of PosteItaliane as Project Manager concerning the development of prototypes.

Luigi Troiano Ms.Eng.(2000), Ph.D.(2004). He is Assistant Professor at University of Sannio, and coordinator of Computational and Intelligent System Engineering Lab (CISELab). He obtained Laurea (master degree) in IT Engineering at University of Naples Federico II in 2000, and Ph.D. in IT Engineering at University of Sannio in 2004. His research interests are mainly related to Computational and Intelligent Systems, focusing on how to apply mathematical models and advanced algorithms to Industry.