Elsevier Editorial System(tm) for Expert Systems With Applications
Manuscript Draft

Manuscript Number:

Title: OPTIMIZING ANTI-SPAM FILTERS WITH EVOLUTIONARY ALGORITHMS

Article Type: Full Length Article

Keywords: anti-spam filters; multiobjective optimisation; evolutionary computation; genetic algorithms

Corresponding Author: Dr. José Ramón Méndez Reboredo, Ph.D.

Corresponding Author's Institution: University of Vigo

First Author: Iryna Yevseyeva, Ph. D.

Order of Authors: Iryna Yevseyeva, Ph. D.; Vitor Basto-Fernandes, Ph. D.; David Ruano-Ordás, Ph. D. Student; José Ramón Méndez Reboredo, Ph.D.

**Cover Letter**

University of
Vigo

Dr. José R. Méndez Reboredo
New Generation Computer Systems Group
University of Vigo

Escuela Superior de Ingeniería Informática
Edificio Politécnico
Campus Universitario As Lagoas, s/n.
32004 Ourense, Spain

Tfn.: +34 988 387015
Fax: +34 988 387001
email: moncho.mendez@uvigo.es
web: http://sing.ei.uvigo.es/

Expert Systems With Applications

Dear Dr. Liebowitz,

We wish to submit the attached paper titled ‘*OPTIMIZING ANTI-SPAM FILTERS WITH EVOLUTIONARY COMPUTATION ALGORITHMS*’ for possible publication in *Expert Systems With Applications* journal. A previous version of this work was submitted before and it was rejected because the manuscript was too long. Therefore, we have condensed the manuscript and size is now appropriate.

In the present work we evaluate the suitability of applying evolutionary algorithms to optimize spam-filtering software (such as SpamAssassin) and provide a further analysis of using simple and multiobjective evolutionary algorithms. Starting from our own experience in the field, we summarize previous work together with some novel alternatives for optimizing software.

The accuracy of two well-known multiobjective alternatives is compared against Grindstone4SPAM, a high accuracy filter optimizing technique, by using a raw and large e-mail corpus and following a fold-cross validation scheme. The comparison has been accomplished through a performance analysis (Pareto fronts) and benchmarking optimized filters.

We hope you find the paper sufficiently interesting within the scope of the journal and worthy of publication.

We await your decision with interest.

*Conflict of Interest*: none declared.

José R. Méndez Reboredo
**corresponding author*

# OPTIMIZING ANTI-SPAM FILTERS WITH EVOLUTIONARY ALGORITHMS

**Iryna Yevseyeva[1], Vitor Basto-Fernandes[1,2], David Ruano-Ordás[3], José R. Méndez[3*]**

[1] Computer Science and Communication Research Center, Polytechnic Institute of Leiria, 2411-901 Leiria, Portugal.

[2] Informatics Engineering Department, Technology and Management School, Polytechnic Institute of Leiria, Leiria, Portugal, 2411-901 Leiria, Portugal.

[3] University of Vigo, Campus As Lagoas S/N, 32004 – Ourense, Spain

Email addresses:

IY: iryna.yevseyeva@gmail.com

VBF: vitor.fernandes@ipleiria.pt

DRO: drordas@uvigo.es

JRM: moncho.mendez@uvigo.es

[*]Corresponding author:

José Ramon Mendez [Tlf.: +34 988 387015 – Fax: +34 988 387001]

ESEI: Escuela Superior de Ingeniería Informática. Edificio Politécnico. Campus Universitario As Lagoas s/n, 32004 – Ourense – Spain.

## Abstract

This work is devoted to the problem of optimizing scores for anti-spam filters, which is essential for the accuracy of any filter based anti-spam system, and is also one of the biggest challenges in this research area. In particular, this optimisation problem is considered from two different points of view: single and multiobjective problem formulations. Some of existing approaches within both formulations are surveyed, and their advantages and disadvantages are discussed. Two most popular evolutionary multiobjective algorithms and one single objective algorithm are adapted to optimisation of the anti-spam filters' scores and compared on publicly available datasets widely used for benchmarking purposes. This comparison is discussed, and the recommendations for the developers and users of optimizing anti-spam filters are provided.

**Keywords**: anti-spam filters, multiobjective optimisation, evolutionary computation, genetic algorithms.

# 1. Introduction

With the increasing proliferation of information and communication technologies and the growing information worldwide exchanges through Internet, making Internet services and resources controllable against malicious usage became vital. The growing of connections to exchange large amounts of data (such as videos, music, etc.) supported by Internet network introduced the need of improving both effectiveness (objectives oriented), and efficiency (optimal usage of resources for achieving specific goals). Recent developments on high-speed computer networks (by using *Fiber-to-the-x* (FTTx) technologies, such as Fiber-to-the-curb, Fibre-to-the-building, Fiber-to-the-house (Idate, 2012)) allowed fast exchanging of large volumes of information. However, the huge amount of spam contents distributed through networks has limited their benefits and currently, a lot of Internet physical resources, technical managers and end users time are wasted for deleting spam messages, closing spam web banners, and downloading unwanted spam information.

Spammers found and developed a wide variety of forms to distribute illegal and fraudulent advertisements. Due to the continuous changing of techniques used to distribute spam, anti-spam filters become obsolete in a short time period and need to be updated on a regular base. This situation created preconditions for the development and wide spreading of professional anti-spam filtering services. Aiming at customer satisfaction and accuracy of emails classification, the modern anti-spam filtering systems are desired to have the following properties: (*i*) ability of continuous updating of a default anti-spam filter and adding new rules to it with respect to customer preferences and (*ii*) ability to stay up to date with the latest spam spreading techniques. Behind these services there are teams of experts examining emails and updating anti-spam filters behaviour to detect the newest spam contents. Current filtering frameworks (including *SpamAssassin*

(The Apache SpamAssassin Project, 2011) or Wirebrush4SPAM (Pérez-Díaz, Ruano-Ordas, Fdez-Riverola & Méndez, 2012b) support filter customisation by using a filtering description syntax based on message fields and contents criteria.

Hundreds of enterprises develop and commercialise anti-spam filtering services. Most of these services are based on signup (gathering information about username, pay methods, mail transfer agent server and target domain) and *change mail exchange* (MX) register (Mockapetris,1987) of the target domain (AgentSPAM, 2012). Providing continuous updating of filtering services at affordable cost makes these services very attractive to *small and medium-sized enterprises* (SMEs).

From a technical point of view, the generation of rules to address new trends of spam emails is an easy process. However, discovering the relative importance of (thousands of) rules to assign individual scores for weighting each rule included in a filter, is a complex setup process, performed usually without any guidance or systematic support. This task should be done automatically, taking into account the need of possible reassignment of existing rules scores, when a new rule is added to the system. Currently, this task has been addressed by the techniques surveyed in (Basto-Fernandes, Yevseyeva & Méndez, 2012), such as *evolutionary algorithms* (The Apache SpamAssassin Group, 2010; The Apache SpamAssassin Group, 2009a), *logistic regression* (Dreiseitl & Ohno-Machado, 2002), *neural network* trained with error back propagation by gradient descent (*Perceptron*) (The Apache SpamAssassin Group, 2004) and Grindstone4SPAM (SING Group, 2007; Méndez, Reboiro-Jato, Díaz, Díaz & Fdez-Riverola, 2012). However there is still the need of solving existing drawbacks such as: (*i*) the absence of automatic customization processes to avoid rules execution when are useless in certain domains (Pérez-Díaz, Ruano-Ordas, Fdez-Riverola & Méndez, 2012b), (*ii*) the selection of the appropriate rule weights to handle user requirements and business area

(SING Group, 2007; Méndez, Reboiro-Jato, Díaz, Díaz & Fdez-Riverola, 2012) and finally, (*iii*) the elimination of the irrelevant filtering rules in order to avoid their execution and hence the reduction of the time needed for accomplish the filtering process (Pérez-Díaz, Ruano-Ordas, Fdez-Riverola & Méndez, 2012b).

In this work, we test the suitability of using different evolutionary computation approaches for automatic scores setting of rules in an anti-spam filter.

The rest of the paper is structured as follows: Section 2 introduces the target problem and surveys the techniques used for optimizing scores of anti-spam filters. Section 3 describes the experimental protocol and the experimental results are provided in section 4. Finally, the conclusions and future work are drawn in section 5.

# 2. Optimization of anti-spam filters

Recently, the open source SpamAssassin filtering system gained popularity among SMEs users and became a reference in the anti-spam filtering domain. Its popularity is not only due to its public availability to research and development (becoming a disadvantage being available to spammers), but also because of its performance. SpamAssassin introduced to the anti-spam filtering domain two major features (Pérez-Díaz, Ruano-Ordas, Fdez-Riverola & Méndez, 2012b): (*i*) the possibility of modelling the filter operation as a combination of rules of different types working together and (*ii*) the ability of updating the filter behaviour by introducing new rules into the system. These features have also been widely exploited to develop other advanced anti-spam filtering solutions such as Symantec Brightmail (Symantec Corporation , 2012) or McAffee SpamKiller (McAfee, 2012), addressed mainly to leading big companies and also some SMEs.

As we can see from (Pérez-Díaz, Ruano-Ordas, Fdez-Riverola & Méndez, 2012b) and (Pérez-Díaz, N., Ruano-Ordás, D., Méndez, J.R., Gálvez., Juan F., & Fdez-Riverola F, 2012) SpamAssassin is a plugin middleware and framework for the execution and development of new user defined anti-spam filters and techniques. Each SpamAssassin technique can be combined in a filter depending on user needs. These techniques are implemented in separate plug-ins. Each plug-in is treated as a different entity avoiding dependencies between plug-ins and guaranteeing high modularity to the whole anti-spam system. Moreover, this feature provides a great flexibility to the platform allowing easy creation, manipulation and deployment of new customized anti-spam filtering techniques.

Table 1 introduces a brief description of different types of filtering techniques provided by default in SpamAssassin, extracted from /usr/share/perl5/Mail/SpamAssassin/Plugin directory.

**** Table 1 here ****

As we can see from Table 1, the SpamAssassin techniques are divided into four different groups: (*i*) responsible for executing an intelligent analysis of message contents, (*ii*) reliable for querying collaborative networks and servers sharing information about spam senders and deliveries, (*iii*) in charge of validating senders legitimacy and finally, (*iv*) regular expressions and parsers for checking email structure and syntax. Using each type of technique on its own is not efficient and therefore, some combinations of techniques of different types are applied. Keeping in mind this idea, a SpamAssassin filter is combination of techniques through rules.

A SpamAssassin filter is mainly composed by a collection of rules and a threshold called *required_score*. Each rule contains a logical test (that works as a trigger condition and uses one of the available techniques) and a score. During the operation of the spam filter, a message is classified as spam when the amount of scores belonging to triggered rules is greater or equal than *required_score*. Due to this particular form of design filters, the adjustment of rule scores and *required_score* parameters emerged as a difficult optimization challenge.

Traditionally, scores setting and tuning is performed manually by system administrators based on their experience gained after years of applying a try-a nd-error approach. Constant race against spammers that invent new ways to distribute spam, leads to the need of automatic optimisation of scores setting process that would assist or even substitute system administrators in this task. For automatic scores setting, the advanced optimisation techniques could be used. Recent survey of literature on this subject (Basto-Fernandes, Yevseyeva & Méndez, 2012) has revealed some approaches proposed by researchers to optimisation of the scores setting for filtering rules.

For the sake of our research proposal contextualization, we present in section 2.1 some perspectives on the anti-spam filtering problem formulation. In section 2.2 and 2.3 we present the state of the art on single and multiobjective anti-spam filtering techniques.

### 2.1. Latest advances on filter optimisation

Naturally, the formulation of the scores setting optimisation problem is bi-objective: a typical user would wish to minimize both, the number of spam messages not identified by anti-spam filtering techniques, called false negative (FNs), and the number of legitimate messages classified as spam by mistake, called false positives (FPs) (as opposite to correctly classified spam messages, true positives (TPs), and correctly classified legitimates (TNs)). A business email is one of extreme cases of anti-spam systems setup with such objectives, where the number of FPs and FNs should be tuned to have lowest possible rate of lost legitimate messages (basically equal to zero), usually at the expenses of higher FN classifications. On the other extreme is *Content Management Systems* (CMSs) devoted to entertainment, e.g. similar to news ticker on TV that can dismiss some legitimate messages keeping or even improving the relevance and interest on their usage, while the acceptance of any spam message is not allowed. Probably, the majority of the cases between these two extremes would still be of high user interest for a variety of the problem areas.

However, as it is still often done with multiobjective problems, the formulation was initially simplified to a single objective problem by weighting objectives according to their importance. Such objective function used for evaluating efficiency of the anti-spam filters is called a performance index (Androutsopoulos, Koustias, Chandrinos, Paliouras & Spyropoulos, 2000). Different performance indexes were developed and the list of them can be found in (The Apache SpamAssassin Group, 2009b).

Assuming that keeping legitimate messages is much more important than having some spam messages to arrive at the email-box, Androutsopoulos, Koustias, Chandrinos, Paliouras & Spyropoulos (2000) suggested the *Total Cost Ratio* (TCR) performance index. TCR is the most often used metric that shows the relation between the total number of spam messages (*nspam*) in the testing corpus and the sum of FPs and FNs taking into account the relative importance of legitimate messages loss ( $\lambda_{TCR}$ ) when compared to the non detection of spam messages. TCR is calculated as follows:

$$TCR = \frac{nspam}{\lambda_{TCR} * fp + fn} \qquad (1)$$

Selecting the value of the $\lambda_{TCR}$ ratio is really ad hoc approach and depends on the problem to be solved and subjective preferences of the decision maker. In email spam filtering, typical values for $\lambda_{TCR}$ are 1, 9 and 999. For instance, for the filtering of business related email messages, losing any legitimate message is critical, and receiving some spam messages, even though uncomfortable, is allowed; that leads to high value for $\lambda_{TCR}$ . The maximal value of TCR provides the set of scores for the anti-spam filtering rules that is optimal for the current problem.

Other performance measures were developed taking into account several usually conflicting objectives. For instance, two other important performance measures, *precision* and *recall*, are usually optimized simultaneously, since they complement each other (van Rijsbergen, 1979). Recall is able to compute the ability of classifying spam e-mails (higher values of recall imply more spam detected) while precision calculates the competence of a given filter in generating low FP errors (higher values of precision imply a lower FPs rate):

$$precision = \frac{nspam - fn}{nspam - fn + fp} \qquad recall = \frac{nspam - fn}{nspam} \tag{2}$$

*f-score* (van Rijsbergen, 1979) combines the values of recall and precision in the interval [0-1], and takes value 1 only if the number of FP and FN errors generated by the filter is 0. Equation (3) shows how this measure is calculated.

$$f_\beta = (1+\beta^2) * \frac{2 * precision * recall}{(\beta^2 * precision) + recall} \tag{3}$$

The f-score with *β=1*can be interpreted as a weighted average of the precision and recall, reaching its best score at 1 and worst score at 0. The *balanced f-score* (*β=1*) is the harmonic mean of precision and recall. The f-score with *β*=2 set higher weight to recall than to precision, but the f-score with *β* =0.5 has opposite settings.

Other popular performance measures are proportions of the FPs (*FP%*) and FNs (*FN%*), when compared to the number of the known-to-be ham (*nham*) and spam (*nspam*) messages, respectively:

$$FP\% = \frac{fp}{nham} * 100 \qquad FN\% = \frac{fp}{nspam} * 100 \tag{4}$$

*Batting average* (Graham-Cumming, 2004) is popular method to show the connection between %FP and %FN measurements. It is built taking into account the hit rate and strike rate, where the former represents the proportion of detected spam messages and the latter the FP errors average.

Another possible way of reducing multiobjective problem to a single objective one is by moving some objectives into a set of constraints for the problem. For instance, when minimizing the number of spam messages arriving at an email-box (objective function) without losing a single legitimate message (constraint).

Due to the large and constantly growing number of filtering techniques to be applied in ensemble for anti-spam classification, both single and multiobjective formulations of this optimisation problem have combinatorial nature. Solving such problems to optimality by exact methods is time-consuming and hard, if possible, due to the large number of possible combinations of scores values for different filtering rules. That is why typically approximation methods, also called *metaheuristics*, are used to find near optimal and often optimal solutions in a feasible, suitable for the user time.

## 2.2. Single objective evolutionary techniques for optimizing anti-spam filters

The single objective problem can be presented as an optimisation (minimization is assumed) of some real-valued objective function *f(y)*, evaluated in decision space with a vector of decision variables, $y = (y_1, y_2, ..., y_n)$ such that $y_i \; i \in \{1, ..., n\}$. Some constraints may be imposed on the decision variables by the domain definition of objective function or by subjective preferences of the decision maker. The constraints of both equality and inequality type can be defined as inequality ones: $f(y) \geq c$, where $c$ is a constant value.

For anti-spam systems the scores vector is a vector of decision variables $y$ of length $n$ (the total number of filtering rules) with each variable $y_i$ corresponding to a score of one rule. Considering optimisation of some performance measure, e.g. TCR, such scores values for filtering rules should be adjusted to optimize the value of the selected performance measure, e.g. maximize the TCR value. For making reliable conclusions about the scores obtained, tests are usually done on the large enough sets of messages, called *corpora*. Moreover, cross-validation schemes are used to address training issues relative to some techniques used by rules (e.g., Naïve Bayes).

E*volutionary algorithms* (EAs) appear to be very powerful metaheuristics and gain popularity in industrial applications including those of combinatorial nature. The effectiveness of EAs is due to working with not a single solution but a population of potential solutions (also called individuals or chromosomes). EAs try to balance convergence and diversity dilemma in optimisation, by guiding the search towards possible multiple optimal solutions (natural for multimodal optimisation, and its particular case, multiobjective optimisation). In this way, EAs are able to study complex search spaces and functions by preserving the population from premature convergence to a local optima or undesired solutions.

The first attempt to automatic optimisation of filtering rules scores was made in the Apache SpamAssassin Project. A single objective evolutionary algorithm, also called *SpamAssassin Genetic Algorithm* (SAGA) was used to optimize scores of filtering rules in SpamAssassin versions 2.5 and 2.6. Even though there is commented source code of SAGA available at (The Apache SpamAssassin Group, 2010), it is confusing due to many changes done by several developers. SAGA adapts an open-source code on genetic algorithm; PGAPack (Levine, 1995), to anti-spam filtering rules scores setting. PGAPack is a parallel genetic algorithm library written in ANSI C that uses the *Message Passing Interface* (MPI).

The main disadvantage of SAGA is the extremely high running time required for setting scores, between 6 and 24 hours, on high-end machines reported in (The Apache SpamAssassin Group, 2004). This fact does not allow updating scores more often that at each release of SpamAssassin. Ideally, the possibility of performing such updates should be provided for any user of SpamAssassin. The need for the fast scores setting encouraged SpamAssassin developers to search for alternatives to SAGA.

As an attempt to improve the optimisation of scores in SpamAssassin and speeding it up, another version of EA was implemented in the framework of the open-source *Grindstone4SPAM* (SING Group, 2007; Méndez, Reboiro-Jato, Díaz, Díaz & Fdez-Riverola, 2012) developed at the University of Vigo. In addition, Grindstone4SPAM aims at saving administrators time while adding rules, optimizing the speed of Bayes database and offline filter evaluation.

The Grindstone4SPAM EA has been preconfigured to use a population of 200 individuals with the stopping criterion set to 100 generations. The individuals of initial population are generated randomly in the range of scores [-5; 5], although the possibility to generate scores from some given (e.g. by expert) configuration of filtering rules scores is provided. In the later case, the single individual is used as a seed for creating as many individuals as needed by some modifications with the help of random generator (having uniform distribution by default). In particular, this individual is modified for creating 199 new members of initial population according to the following sequence of operations. First, the number of genes to be altered is selected randomly between 1 and the maximal number of genes. Then, the position of the gene to be altered is selected randomly among those not yet modified. For the gene selected for alteration, the sign of alteration (addition “+” or negation “-”) is selected randomly. Then, the value of the change to be applied with the selected earlier sign is selected randomly among those in the range [-5; 5] with the step 0.5. From the initial population 10 best individuals are selected as parents for the next generation and from them 190 new offspring are reproduced as follows. First, from 10 best individuals, 2 parents are selected randomly. Then, the selected parents are used for creating a new individual by one of the following operations selected randomly: (*i*) to mutate a first parent; (*ii*) to mutate a second parent; (*iii*) for each filtering rule score (gene) to set a value equal to the average value between two parents; (*iv*) for each filtering rule score (gene) to set a value equal to minimal value between two parents; (*v*) for each filtering rule score (gene) to set value equal to the maximal value between two parents; (*vi*) just copy a first parent; (*vii*) just copy a second parent. The probabilities of three first operations are twice higher when compared to probabilities of the last four ones.

SpamAssassin version developers at headquarters in USA were also looking for alternatives to SAGA, and in the version 3.0.0 they have tried to substitute SAGA by simplest version of *Artificial Neural Network* (ANN), called *Perceptron* (The Apache SpamAssassin Group, 2004) trained with error back propagation by gradient descent. There was some role back to SAGA, but the most recent versions of SpamAssassin are using Perceptron that takes around 8 minutes to perform optimisation of the anti-spam filtering rules scores, maintaining the quality of solution similar as SAGA achieves (Dinter, 2004). Fast scores optimisation makes possible customisation of the scores setting process by users of SpamAssassin whenever they need it (e.g. when adding new rules to the anti-spam filter).

In Perceptron for SpamAssassin (The Apache SpamAssassin Group, 2004), scores of filtering rules are represented by *Perceptron input weights*. Initially, the weights values are generated randomly within predefined ranges, and are updated during training on a set of already classified messages of corpus. The Perceptron learning process terminates after a predefined number of iterations or when the predefined minimal value of the classification error is reached. The weights or score values of filtering rules obtained at the final iteration are fixed for classification of new (corpora of) messages. For training Perceptron in SpamAssassin, the stochastic gradient descent optimisation method is used. The logarithmic sigmoid (*logsig*) is used as an *activation function,* which determines when the Perceptron fires overcoming predefined threshold. By default the least square error is used as a *transfer function* (objective function in optimisation terms), but possibility of evaluating entropic error is also encoded.

In a struggle for the time efficiency another approach from statistics, *logistic regression* (LR), was suggested to perform scores setting for anti-spam filtering rules in (Findlay & Birk, 2007). Actually, LR can be considered as a generalization of ANN, see e.g. (Dreiseitl & Ohno-Machado, 2002), where LR and ANN performances are also compared to other popular classification algorithms from the machine learning field, and where main differences between them are highlighted. When compared to parametric LR with coefficients and intercept interpreted parameters, ANNs are considered to be semi-parametric or non-parametric, since it is not always possible to interpret their parameters (weights) (Dreiseitl & Ohno-Machado, 2002).

In (Findlay & Birk, 2007) two different algorithms, *Iteratively Re-weighted Least Squares Least Angle Regression* (IRLS-LARS) and *Truncated Regularized Iteratively Reweighted Least Squares* (TR-IRLS) are presented. Comparative analysis of these algorithms with SAGA demonstrated superiority of TR-IRLS and its fast running time similar to that of Perceptron. TR-IRLS is superior to SAGA when Bayes and network tests are disabled and when Bayes tests are disabled and network tests are enabled; however, it performs worse than SAGA when Bayes tests are enabled and network tests are disabled and when both Bayes and network tests are enabled.

### 2.3. Multiobjective evolutionary techniques for optimizing anti-spam filters

Simplification of the multiobjective form of the scores setting optimisation problem with one of the approaches discussed in the previous subsection may look attractive and intuitive, but usually is not reliable. Since relative importance values of objectives (also called weights) given by different users (or even for the same user) may vary significantly. This fact leads to significantly different final trade-off solutions obtained by a single objective algorithm.

To obtain the set of all trade-offs between several objectives, multiobjective approaches should be used. In multiobjective problem formulation, several independent and usually conflicting objectives are optimized simultaneously. The result of multiobjective optimisation is rarely a single solution, it is rather a set of compromise solutions that present trade-offs between objectives, called *Pareto optimal set* or simply *Pareto set*. The mapping of Pareto set solutions to their corresponding evaluation on all objectives is called *Pareto front*. The solutions belonging to the Pareto set are optimal in a sense that moving from one solution to its neighbour on the Pareto front improves one or more objectives, but only at the cost of deteriorating other(s).

In case of the multiobjective optimisation, *m* objective functions $f = (f_1, f_2, ..., f_m)$ are optimized simultaneously (minimization of all functions is assumed), such that $f_k$, $k \in \{1,...,m\}$ is a real-valued function of a vector of decision variables *y*. Some constraints may be imposed on the decision variables $f_k(y) \geq c_k$, where $c_k$ is a constant value. The Pareto set of optimal solutions is constructed using the Pareto dominance relation. This relation assumes that one solution *y* is better (having smaller values assuming minimization of objectives) than the other one $y'(y, y' \in R^m)$, if it is strictly better on at least one objective and not worse on the rest of objectives: $y \succ y'$ *(y dominates y')*, $\forall k \in \{1,...,m\}: \mathrm{f}(y_k) \leq f(y_k^{'})$ *and* $\exists l \in \{1,...m\}: \mathrm{f}(y_l) < f(y_l^{'})$.

Typically, only a single solution among those belonging to the Pareto set should be selected as the final. It may be addressed directly by the decision maker according to his or her preferences or with the help of a decision aiding tools (Belton & Stewart, 2002).

Figure 1 shows an example of multiobjective presentation of the anti-spam classification problem with 3 different filtering rules F1, F2, F3. The 6 points represent 6 different configurations (with different scores for each filtering rule) of this 3-filtering rules anti-spam system. All 6 configurations are evaluated in 2-objectives space with number of FP and FN errors objectives to be minimized. The 3 black points with minimal values on these objectives define Pareto front and corresponding Pareto set of best configurations.

**** Figure 1 here ****

The choice of techniques for multiobjective formulation of the scores setting optimisation for anti-spam filtering problem was in the favour of EAs, called *multiobjective evolutionary algorithms* (MOEAs), due to their collective learning nature that allows performing simultaneous or parallel search for good solutions. This property is particularly important for the multiobjective optimisation problems with several Pareto optimal solutions. The main difference and main difficulty of MOEAs, when compared to single objective EAs, is the calculation of fitness for each individual. In principle, such fitness should allow aggregating evaluations of a solution on multiple objectives into a single value. Even if fitness is not calculated directly by MOEA, the non-dominated solutions should be considered “fitter” and should be preferred to the dominated ones for both selection of parents and selection of the next population. At the same time, the diversity of solutions chosen at the selection stages should be preserved.

Currently, only one multiobjective approach was found in the literature on optimisation of scores for anti-spam systems (Dudley, 2007; Dudley, Barone & While, 2008) called *Multi-Objective Spam Filtering* (MOSF). It is a multiobjective genetic algorithm based on a well-known *Non-dominated Sorting Genetic Algorithm II* (NSGA-II) (Deb, Pratap, Agarwal & Meyarivan, 2002) and adapted to setting scores for SpamAssassin filtering rules. When compared to the original NSGA-II, different selection scheme is used (Fonseca & Fleming, 1993) and variation (crossover and/or mutation) operators are adapted to the problem area. There was no justification given in favour of the chosen selection scheme, and there is no evidence on the priority of this scheme when compared to the original NSGA-II selection scheme in the literature.

In our research, two most popular MOEAs were selected for optimizing scores of SpamAssassin filtering rules, just mentioned NSGA-II (Deb, Pratap, Agarwal & Meyarivan, 2002) and *Strength Pareto Evolutionary Algorithm 2* (SPEA2) (Zitzler, Laumanns & Thiele, 2002). Both methods belong to the group of elitist Pareto-based MOEAs. It is due to the fact that they incorporate Pareto dominance relation directly into their selection schemes and preserve the non-dominated solutions found so far (this effect is called elitism). Keeping track of the best solutions found and passing them to the subsequent generations guaranties that at least a part of offspring population will be of the quality not worse than that of the parents.

In NSGA-II at each iteration the population is sorted according to the *non-dominated sorting procedure* suggested by Goldberg (Goldberg, 1989). The procedure consists of selecting all non-dominated solutions and assigning the first rank to them. After removing the first-rank solutions from the population, the non-dominance relation is applied to the rest of population; the non-dominated solutions obtained at this stage are assigned with the second rank and removed from further consideration. The process is repeated until the whole population is sorted. For preventing premature convergence (e.g., to the local optima) the diversity of the population is preserved with *crowding distance technique*. It is calculated for each individual of the same rank with respect to its two closest neighbours in the objective space. Individuals located in the most crowded regions (with the smallest values of the crowding distance) are discarded.

The *binary tournament* is used in the selection of parent solutions for mating or reproduction of offspring from parents by crossover and mutation operators. For setting up binary tournament, two members are selected from entire population at random for the competition. Then two winners of such tournaments compete against each other and the winner goes to the mating pool, from which parents are selected for crossover at random. By default NSGA-II uses polynomial mutation and simulated binary crossover for real-value vector representation. The next generation is selected from merged populations of parents and offspring based on non-dominated sorting and crowding distance evaluation for individuals belonging to the same front as described above.

SPEA2 inherits elitism and archiving introduced in its predecessor SPEA (Zitzler & Thiele, 1998). After each iteration of both, SPEA and SPEA2, the archive is updated by adding new non-dominated solution and removing dominated ones. In SPEA2, the fixed size archive is maintained by fill it with dominated solutions with best fitness when the number of non-dominated solutions is not enough, or truncating non-dominated solutions with *k*-nearest neighbours clustering method (Dreiseitl & Ohno-Machado, 2002) when there are more non-dominated solution than need in archive.

For assigning fitness, SPEA2 calculates rank called *strength* for each individual in both the archive and the population as a number of individuals in the union of the archive and the population that the individual dominates. Then, the fitness is assigned to each individual of the population as a sum of strengths of all individuals in both the archive and the population that dominate it. The fitness of an individual is increased by its “density” value that is estimated based on *k*-nearest neighbour method. When compared to NSGA-II, the *binary tournament with replacement* selection (that allows selecting the same individual for the tournament) is used for selecting mating parents in SPEA2.

NSGA-II and SPEA2 outperformed SPEA on a number of benchmark problems, and on higher dimensions SPEA2 outperformed NSGA-II (Zitzler, Laumanns & Thiele, 2002).

# 3. Experimental protocol

For setting up a protocol for testing efficiency of the selected optimisation algorithms, two major choices were made *(i)* of dataset for testing the algorithms, and *(ii)* of performance measures used for efficiency evaluation of each of the algorithms. For anti-spam filter optimisation using EAs it means selecting a corpus of messages to be used for classification (in two classes: spam or legitimate), and choosing among large amount of EAs performance measures.

In subsection 3.1 and 3.2 we describe the options made with respect to the test dataset selection and two protocol comparison schemes for the purpose of NSGAII, SPEA2 and GrindStone4SPAM algorithms performance assessment.

## 3.1. Corpus selection

When designing experiments for a problem domain, it is a good practice to consider performance of the algorithms on typical tests as well as rare but still possible data and/or real-world cases. For anti-spam classification the tests could be performed on the sets of messages, also called corpora, already classified manually by users. Therefore, several researches and organizations have manually compiled and shared their own collection of emails in order to test the suitability and efficiency of new filtering techniques.

A thorough analysis of corpora available in Internet is outlined in (Pérez-Díaz, Ruano-Ordas, Fdez-Riverola & Méndez, 2012a). Table 2 presents a summary of the most suitable corpora for our experiments. As we can observe, each corpus contains distinct characteristics, such as proportions of ham and spam messages, single and multi-domain membership that are essential for validating the new anti-spam filtering techniques proposed.

**** Table 2 here ****

All corpora listed in Table 2 follow the RFC 2822 (Resnick, 2001) format specification. Such specification facilitates the analysis of the message content and grants the access to stored information. It is required that corpora contain both spam and legitimate emails. Moreover, when selecting corpora for tests we should keep in mind that medium-sized corpora are the most suitable for saving computational resources without compromising statistical significance of results. These features are the reason for the wide usage of SpamAssassin corpora in previous research works. Keeping in mind these conditions and in order to assure backward reproducibility of results, we have selected SpamAssassin corpora for learning and testing stages of our experiments protocol.

### 3.2. Design of experiments

This subsection outlines the measures used to compare optimisation methods considered in this work. To this end, the following tune-up algorithms will be compared: (*i*) Grindstone4SPAM, (*ii*) NSGA-II, and (*iii*) SPEA2. The protocol includes two different comparison schemes: (*i*) performance analysis and (*ii*) optimized filter benchmarking.

For the first comparison scheme, we plot Pareto fronts in order to visualize the efficiency of each analysed optimisation method. The obtained results provide a visual comparison of optimisation abilities of each algorithm.

Due to the stochastic nature of EAs, their comparison is not trivial, since results of not a single but multiple runs should be compared. The difficulty is related to the fact that comparing results of even two single runs of MOEAs leads to comparison of two Pareto fronts that is trivial only in case of domination of all solutions of one Pareto front over another one. Even more difficult becomes comparison of multiple runs of one MOEA to those of another MOEA, and results in comparison of areas covered by all solutions of resulting Pareto fronts. For two and three objectives, plotting resulting Pareto fronts allows obtaining rough pictures of the location of such areas, and we will use such images for initial visual comparison of algorithms performance in addition to more advanced tools for MOEAs comparison discussed in subsection 4.1.

Opposite to the first scheme, the second comparison scheme involves a separation of the training (learning) and testing message sub-sets, known as *cross-validation* in the classification domain research area, for assessing the performance achieved by the proposed algorithms. As shown in Figure 2, this scheme comprises two different fragments: (*i*) the creation of the training and learning datasets and (*ii*) the application of the obtained datasets in order to generate the improved rules scores and verify their suitability.

**** Figure 2 here ****

To facilitate the understanding of the second comparison scheme, we have included in Table 3 the fold-division process carried out for each experiment fragment of cross-validation process. As shown in Table 3, we chose a medium-sized corpus in order to minimize the overtraining drawbacks existing in big-sized datasets.

**** Table 3 here ****

In order to measure the performance of the proposed filter benchmarking scheme, we used the following optimization measures introduced in subsection 2.1: (*i*) percentage of FPs and FNs, (*ii*) total cost ratio, (*iii*) recall, and (*iv*) precision.

The next section shows in detail the results achieved by the execution of the two comparison schemes, defined by the experimental protocol.

# 4. Experimental results

For testing the selected algorithms, Grindstone4SPAM, NSGA-II and SPEA2, according to both comparison schemes, we adopted the SpamAssassin default configuration.

In particular, the threshold is set to 5 and scores range are selected in the interval [-5; 5]. SpamAssassin public mail corpus 2005 (The Apache SpamAssassin Group, 2005) is the dataset used in all experiments. NSGA-II (Deb, Pratap, Agarwal & Meyarivan, 2002) and SPEA2 (Zitzler, Laumanns & Thiele, 2002) default configurations are taken as reference configurations for comparisons and results analysis. The same configuration of parameters is set for both NSGA-II and SPEA2 on their common parameters. Population size is set to 100, number of function evaluations to 25000, number of independent runs of the algorithms with random seeds on the same problem instance to 30, binary tournament selection operator, polynomial mutation operator, mutation probability 1.0/*number_of_decision_variables*, mutation distribution index 20.0, SBX crossover operator, crossover probability 0.9, crossover distribution index 20.0, are set the same for both algorithms. Archive size is set to 100, which is specifically defined for SPEA2 (parameter non-existent in NSGA-II).

Grindstone4SPAM default configuration described in subsection 2.2 is used, 19000 function evaluations and 30 independent runs of the algorithms with random seeds on the same problem instance were performed. Grindstone4SPAM single objective function was set to TCR with $\lambda_{TCR}$ assigned to value 1 (for all runs), meaning equally importance of avoiding both FN and FP classifications.

NSGA-II and SPEA2 were performed with jMetal (Durillo & Nebro, 2011), an optimisation framework for the development of different multiobjective metaheuristics in Java, and Grindstone4SPAM original implementation developed in C by its authors was used for comparative analysis. Due to the difference in the implementation languages and types of optimization problem formulation (single or multiobjective) used, but assuming similar number of function evaluations in all tests, we consider reasonable to perform qualitative analysis of the solutions obtained as a result of algorithms simulation according to the two comparison schemes presented, rather than comparing computational time spent by algorithms execution.

### 4.1. Performance analysis

In this subsection we compare two most used general purpose algorithms from the multiobjective optimisation population-based metaheuristics group (NSGA-II (Deb, Pratap, Agarwal & Meyarivan, 2002), SPEA2 (Zitzler, Laumanns & Thiele, 2002), and a very recent, single objective genetic algorithm for anti-spam classification optimisation, developed at the University of Vigo (SING research group), named Grindstone4SPAM (SING Group, 2007; Méndez, Reboiro-Jato, Díaz, Díaz & Fdez-Riverola, 2012). We based our comparison  strategy on the approach proposed in (López-Ibáñez, Paquete & Stützle, 2012). First, we describe briefly why we followed this approach, and, second, we analyse the outcomes of the simulations performed.

Most of the known approaches for summarizing and comparing multiobjective optimisation algorithms, with respect to solutions quality, are based either on direct examination of non-dominated sets resulting from the optimisation process, or scalar quality indicators such as the *hypervolume* (Zitzler & Thiele, 1998). Since direct examination of results proved to be cumbersome and difficult to achieve for algorithms performance comparison, and scalar quality indicators can only measure specific, limited quality aspects, other approaches providing good trade-offs between these two extremes were introduced. In (López-Ibáñez, Paquete & Stützle, 2012) the *empirical attainment function* (EAF) is proposed as the means for summarizing both outcomes of multiple runs of an algorithm and also to illustrate the differences of two algorithms outcome. The *attainment function* computes the probability that an arbitrary objective vector is attained (dominated or equal) in a single run of a particular algorithm (Grunert da Fonseca, Fonseca & Hall, 2001). This is done by estimation, using results from several runs of an algorithm.

In (Fonseca & Fleming, 1996) the notion of *attainment surface* was proposed. It defines the boundary that splits the objective space in the region of vectors attained by the outcomes of the algorithm, and the region of vectors that are not (e.g. the median attainment surface delimits the region attained by 50 percent of the runs). The plotting of attainment surfaces (e.g. first quartile, median, etc.) summarizes the behaviour of an algorithm and can be used for algorithms comparison purposes. In (López-Ibáñez, Paquete & Stützle, 2012) the examination of differences between EAFs for algorithms behaviour and performance comparison is proposed. The difference of the estimated probability values of two algorithms (first algorithm minus second algorithm) at a certain point indicates a better performance of one algorithm over another at that point. Positive and negative differences are plotted separately, and the magnitudes of the differences between the EAFs are encoded using different shades of grey, the darker is a point, the larger is the difference.

Figure 3 to Figure 6 show the simulation outcomes with the NSGA-II, SPEA2 and Grindstone4SPAM algorithms configurations described at the beginning of this section.

**** Figure 3 here ****

Figure 3 represents the plots of NSGA-II (*i*) and SPEA2 (*ii*) attainment surfaces for 30 independent runs, showing the best, median and worst percentiles of attainment surfaces for both algorithms.

**** Figure 4 here ****

Figure 4 provides visual information to compare NSGA-II (*i*) and SPEA2 (*ii*) results (attainment surfaces) with Grindstone4SPAM single objective best results. In general Grindstone4SPAM presents better results than NSGA-II and SPEA2 for the minimization of FPs, while NSGA-II and SPEA2 reveal better results towards the minimization of FNs, however, at the cost of bigger numbers of FPs.

**** Figure 5 here ****

Figure 5 shows the EAFs associated to NSGA-II (*i*) and SPEA2 (*ii*) algorithms. Points in the graphics are assigned a gray level according to their probability (gray level encodes the value of the EAF), and attainment surfaces are also shown in both plots. Lower lines represent the best set of points attained over all runs of both algorithms and upper lines the set of points attained by any of the runs (differences between the algorithms are shown within these two lines). Dashed lines correspond to the median attainment surface of each algorithm.

**** Figure 6 here ****

Figure 6 shows the location of the differences between the EAFs of the two algorithms. The difference is encoded in a grey scale and the attainment surfaces are plotted similarly to Figure 5. The gray level encodes the magnitude of the observed difference. On the left it is shown the objective space regions where NSGA-II performs better than SPEA2. NSGA-II performs significantly better (between 20% and 40% better) towards the minimization of FPs. On the right it is shown that SPEA2 does not perform better than NSGA-II in any region of the objective space.

### 4.2. Optimized filter benchmarking

In this subsection, we compare the accuracy achieved during the execution of each optimisation algorithm, NSGA-II, SPEA2 and Grndstone4SPAM, based on the cross-validation analysis described in subsection 3.2. Table 4 presents a global summary showing in detail the amount of hits (TNs and TPs) and errors (FNs and FPs) achieved by using each algorithm.

**** Table 4 here ****

As we can observe from Table 4, NSGA-II shows better performance with respect to smaller number of FP errors. From other point of view, Figure 7 summarizes information included in Table 4 using percentage evaluations of FP and FN errors.

**** Figure 7 here ****

As we can see from Figure 7, MOEAs (especially NSGA-II) achieved a great level of accuracy (99.45 percent). Moreover, Figure 7 also shows that NSGA-II optimisation process provided the best results obtained by the optimised filter: it achieved the smallest level of FP errors and the highest rate of true hits $(TPs + TNs(OK))$. Grindstone4SPAM presents the lowest level of FNs at the cost of having the worst true hits value $(TPs + TNs(OK))$.

We also used recall and precision measures (see section 3.1) to compare the analysed algorithms. Figure 8 provides a graphical comparison of the achieved results.

**** Figure 8 here ****

As we can see from Figure 8, Grindstone4SPAM achieved the greatest recall scores. However, this algorithm presented the worst precision scores against all considered alternatives. We also found that NSGA-II achieved the best recall-precision ratio.

In order to get a unified view of the filtering performance achieved by all analysed optimisation algorithms, we combined recall and precision scores using f-score and balanced f-score measures for accuracy evaluation. Results are shown in Table 5.

**** Table 5 here ****

As we can observe from Table 5, f-score with $\beta = 1$ (balanced f-score) results by NSGA-II are better than those obtained when using the other alternatives. Moreover, Grindstone4SPAM achieved the best evaluation when using 1.5 and 2 as $\beta$ values, corresponding to the user preference of having lower values of FN classifications.

Finally, we have executed a performance comparison using a cost sensitive point of view. To this end, we used TCR measure to assess filter effectiveness assuming different FPs-FNs cost scenarios. Figure 9 shows a TCR benchmark using $\lambda_{TCR}$ equal to 1, 9 and 999.

**** Figure 9 here ****

As we can see from Figure 9, TCR scores achieved by filters optimized using NSGA-II are clearly higher for any cost configuration. Finally, we used batting average metrics to compare the performance of all analysed algorithms. Table 6 summarizes the scores for each optimisation algorithm.

**** Table 6 here ****

As we can see from Table 6, NSGA-II holds the best strike rate (capability of avoiding FP errors), while Grindstone4SPAM has the ability to achieve the highest sensibility rate (capability of detecting spam messages).

### 4.3 Results analysis

Performance analysis scheme described in subsection 4.1 shown that NSGA-II outperforms SPEA2 in minimizing both the number of FNs and FPs objectives, and Grindstone4SPAM in minimizing FNs objective, while Grindstone4SPAM outperforms NSGA-II and SPEA2 in minimizing FPs objective. On the other hand, optimized filter benchmarking scheme discussed in subsection 4.2, shown Grindstone4SPAM better performance in avoiding FN classifications by highest results on recall, balanced f-score with $\beta = 1.5$ or $\beta = 2$ and batting average hit rate. However, NSGA-II achieves better performance in minimizing FP classifications, corresponding to better metrics of accuracy, precision, f-score with $\beta =1$ (balanced f-score), TCR1, TCR9, TCR999 and batting average strike rate.

In other words, while subsection 4.1 reveals that Grindstone4SPAM performs significantly better towards the minimization of FPs when compared to two multiobjective algorithms, the results from subsection 4.2 indicate the best potential of NSGA-II to avoid FP errors, against Grindstone4SPAM highest sensitivity to detect spam messages. The Grindstone4SPAM accuracy changed from experiments in subsection 4.1 (without cross-validation) to the experiments in subsection 4.2 (with cross-validation), contrasting to the more stable classification behaviour shown by NSGA-II, lead to the conclusion that Grindstone4SPAM suffers from over-fitting effects (Sarle, 1995) when compared to the more stable NSGA-II classification outcome (higher generalization ability), within the anti-spam classification domain.

Next section presents the conclusions drawn from our work as well as future research directions for improving the optimisation of current anti-spam filters.

## 5. Conclusions and future work

In this work, optimisation of the anti-spam filtering system was analysed from single and multiobjective points of view. A detailed literature survey has shown potential of the evolutionary approaches when applied to this domain and lead to the selection of three evolutionary algorithms for performance evaluation comparison. Two most widely used multiobjective evolutionary algorithms, NSGA-II and SPEA2, were compared with a single objective evolutionary algorithm, Grindstone4SPAM, which was developed by one of the authors. NSGA-II revealed the most promising results among the three algorithms, taking into account the overall set of performance metrics most used in evolutionary algorithms comparison, namely empirical attainment function, and anti-spam research domain, namely percentage of correctly classified against wrongly classified messages, recall, precision, f-score, TCR and batting average. Comparison of experiments following two schemes, with and without fold-cross validation, demonstrated higher generalization ability of NSGA-II when compared to Grindstone4SPAM.

Although MOEAs (NSGA-II and SPEA2) provide a variety of optimal solutions (Pareto optima), in contrast to single objective algorithms (Grindstone4SPAM) that obtain only one optimum, this feature of MOEAs was not fully exploited in this work. The focus on comparing MOEAs with single objective Grindstone4SPAM required a more constrained comparison framework, not benefiting the full exploration of MOEAs variety of near-optimal solutions generated along the various objective space dimensions.

Although most of the state of the art MOEAs use a generational scheme, recent proposals using a steady-state scheme have been developed and studied. While in the generational scheme the algorithm creates a new population of individuals from an old population, using the typical genetic operators, in the steady-state scheme typically only one new individual is created and tested for becoming (or not) a new member of the population at each step of the algorithm.

Steady-state versions of NSGA-II (Nebro & Durillo, 2009) and especially *S Metric Selection Evolutionary Multiobjective Algorithm* (SMS-EMOA) (Beume, Naujoks & Emmerich, 2007) have been studied and shown higher performance when compared to their generational scheme counterparts, in several benchmarking scenarios. The improved quality of the resulting approximations of the Pareto front and better convergence properties of these algorithms are achieved at the cost of higher computation time and computational resources. In future work, particular attention will be given to the recently developed MOEAs such as steady-state version of the NSGA-II and SMS-EMOA.

# References

AgentSPAM. (2012). Spam & Virus Filtering. <www.agentspam.com>

Allman, E., Callas, J., Delany, M., Libbey, M., Fenton, J., & Thomas, M. (2007). RFC-4871: DomainKeys Identified Mail Signatures. <www.ietf.org/rfc/rfc4871.txt>

Androutsopoulos, I., Koutsias, J., Chandrinos, K. V., Ch, K. V., Paliouras, G., & Spyropoulos, C. D. (2000). An evaluation of Naive Bayesian anti-spam filtering. In Proceedings of the workshop on Machine Learning in the New Information Age, 11th European Conference on Machine Learning (pp. 9–17).

Basto-Fernandes, V., Yevseyeva, I., & Méndez, J. R. (2012). Anti-spam multiobjective genetic algorithms optimization analysis. *International Resource Management Journal (In press)*

Belton V., & Stewart, T. (2002). *Multiple Criteria Decision Analysis: An Integrated Approach.* (1st ed). Dordrecht: Kluwer Academic Publishers.

Beume, N., Naujoks, B., & Emmerich, M. (2007). SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research* (pp. 1653–1669). Vol 181.

Cisco Systems. (2010). SpamCop. <www.spamcop.net>

CSMINING Group. (2010). CSDMC2010 Spam corpus: Spam email datasets. <http://csmining.org/index.php/spam-email-datasets-.html>

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* (pp. 182–197). Vol 6.

Dinter, T. V. (2004). New and upcoming features in SpamAssassin v3. ApacheCon - Conference of the Apache Software Foundation (ASF).

Dreiseitl, D., & Ohno-Machado, L. (2002). Logistic regression and artificial neural network classification models: A methodology review. *Journal of Biomedical Informatics* (pp. 352–359). Boston: Elsevier.

Dudley, J. (2007). Improving the performance of heuristics spam detection using a multiobjective genetic algorithm. Master's thesis. University of Western Australia, <http://undergraduate.csse.uwa.edu.au/year4/Current/Students/Files/2007/-JamesDudley/CorrectedDissertation.pdf >

Dudley, J., Barone, L., & While, R. L. (2008). Multi-objective spam filtering using an evolutionary algorithm. *IEEE Congress on Evolutionary Computation* (pp. 123–130).

Durillo, J. J., & Nebro, A. J. (2011). jMetal: A Java framework for multi-objective optimization. *Advances in Engineering Software* (pp. 760–771). Vol 42.

ECUE. (2011). ECUE Spam Datasets. <www.comp.dit.ie/sjdelany/Dataset.htm>

Fdez-Riverola, F., Iglesias, E. L., Díaz, F., Méndez, J. R. & Corchado, J. M. (2007). SpamHunting: An instance-based reasoning system for spam labelling and filtering. *Decision Support Systems* (pp. 722–736). Elsevier.

Findlay, D., & Birk, S. (2007). Logistic regression and spam filtering. Master's thesis, Queen's University, Kingston, Ontari. <ww.duncf.ca/FindlayBirkThesis.pdf>

Fonseca, C. M, & Fleming, P. J. (1993). Genetic algorithms for multi-objective optimization: Formulation, discussion and generalization. In *Proceedings of the Fifth International Conference on Genetic Algorithms* (pp. 141–153). San Mateo, California, USA: Morgan Kaufmann.

Fonseca, C. M., & Fleming, P. J. (1996). On the performance assessment and comparison of stochastic multiobjective optimizers. In *Proceedings of the 4th International Conference on Parallel Problem Solving from Nature* (pp 584-593). London, UK: Springer-Verlag.

Freed, N., & Borestein, N. (1996a). RFC-2045: Multipurpose Internet Mail Extensions part 1. <www.ietf.org/rfc/rfc2045.txt>

Freed, N., & Borestein, N. (1996b). RFC-2046: Multipurpose Internet Mail Extensions part 2. <www.ietf.org/rfc/rfc2046.txt>

Freed, N., & Borenstein, N. (1996c). RFC-2049: Multipurpose Internet Mail Extensions part 5. <www.ietf.org/rfc/rfc2049.txt>

Freed, N., & Klensin, J. (2005a). RFC-4288: Media Type Specifications and Registration Procedures. <www.ietf.org/rfc/rfc4288.txt>

Freed, N., & Klensin, J. (2005b). RFC-4289: Multipurpose Internet Mail Extensions part 4. <www.ietf.org/rfc/rfc4289.txt>

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Massachusetts, USA: Addison-Wesley.

Graham-Cumming, J. (2004). Understanding spam filter accuracy. *JGC Spam and Antispam Newsletter*. <www.jgc.org/antispam/11162004-baafcd719ec31936296c1fb3d74d2cbd.pdf >

Grunert da Fonseca V., Fonseca, C. M., & Hall, A. O. (2001). Inferential performance assessment of stochastic optimisers and the attainment function. In *Proceedings of the 1st International Conference on Evolutionary Multi-Criterion Optimization (EMO 2001)* (pp. 213–225). London, UK: Springer-Verlag.

Guenter, B. (1998). SPAM archive. http://untroubled.org/spam/

Idate. (2012). Fiber-to-the-x (FTTx2012). <www.idate.fr/private/idate/UserFiles/File/-telechargements_associes/pages/FreeDownload/FTTx2012_WhitePaper_web.pdf>

Levine, D. (1995). PGAPack Parallel Genetic Algorithm Library.

<ftp://ftp.mcs.anl.gov/pub/pgapack/pgapack.tar.Z>

Levine, J. (2010). RFC-5782: DNS blacklists and whitelists.<ietf.org/rfc/rfc5782.txt>

López-Ibáñez, M., Paquete, L., & Stützle, T. (2012). Exploratory analysis of stochastic local search algorithms in biobjective optimization. *Experimental Methods for the Analysis of Optimization Algorithms* (pp. 209–222). Berlin, Germany: Springer-Verlag.

McAfee. (2012). McAfee SpamKiller product family. <http://exsult.com/-ds_spamkiller_family.pdf>

Méndez, J. R., Reboiro-Jato, M., Díaz, F., Díaz, E., & Fdez-Riverola, F. (2012). Grindstone4Spam: An optimization toolkit for boosting e-mail classification. *Journal of Systems and Software*. Elsevier.

Metsis, V., Androutsopoulos, I., & Paliouras, G. (2006). Spam filtering with Naive Bayes - Which Naive Bayes?. In *CEAS 2006 - The Third Conference on Email and Anti-Spam*.Mountain View, California, USA. www.ceas.cc/2006/15.pdf

Mockapetris, P. (1987). RFC-1035: Domain names - Implementation and specification.<www.ietf.org/rfc/rfc1035.txt>

Moore, K. (1996). RFC-2047: Multipurpose Internet Mail Extensions part 3. www.ietf.org/rfc/rfc2047.txt

Nebro, A. J., & Durillo J. J. (2009). On the effect of applying a steady-state selection scheme in the multi-objective genetic algorithm NSGA-II. *Nature-Inspired Algorithms for Optimisation* (pp. 435–456). London, UK: Springer-Verlag.

Orăsan, C., & Krishnamurthy, R. (2002). A corpus-based investigation of junk emails. In *Proceedings of The Third International Conference on Language Resources and Evaluation (LREC-2002)*. Las Palmas de Gran Canaria. Spain. <clg.wlv.ac.uk/-papers/orasan-02b.pdf >

Pérez-Díaz, N., Ruano-Ordas D., Fdez-Riverola, F. & Méndez, J. R. (2012a). SDAI: An integral evaluation methodology for content-based spam filtering models. *Expert Systems With Applications* (pp. 12487–12500). Vol 39.

Pérez-Díaz, N., Ruano-Ordas, D., Fdez-Riverola, F., & Méndez, J.R. (2012b). Wirebrush4SPAM: a novel framework for improving efficiency on spam filtering services. *Software: Practice and Experience*.

Pérez-Díaz, N., Ruano-Ordás, D., Méndez, J.R., Gálvez., Juan F., & Fdez-Riverola F. (2012). Rough sets for spam filtering: Selecting appropriate decision rules for boundary e-mail classification. *Applied Soft Comput*ing (pp. 3671-3682). Vol 12.

Prakash V., & Ritter, J. (2007). Vipul's razor. <http://razor.sourceforge.net>

Resnick, P. (2001). RFC-2822: Internet message format. <www.ietf.org/rfc/rfc2822.txt>

Rhyolite Software. (2000). Distributed Checksum Clearinghouses. <rhyolite.com/dcc/>

Sarle, W. S. (1995). Stopped training and other remedies for overfitting. In *Proceedings of the 27th Symposium on the Interface of Computing Science and Statistics* (pp. 352–360).

SING Group. (2005). SING Public Corpus. <http://sing.ei.uvigo.es>

SING Group. (2007). Grindstone for SPAM. <http://sing.ei.uvigo.es/grindstone4spam>.

Symantec Corporation. (2012).Symantec Brightmail Gateway.<symantec.com/resellers-/support/enterprise/gateway/salestools/-14529676_Sales_Cheat_Sheet_SBG%5B1%5D.pdf>

Text REtrieval conference. (2009). TREC Spam Corpus. <trec.nist.gov/data/spam.html>

The Apache SpamAssassin Group. (2004). Perceptron for SpamAssassin. <www.spamassassin.org/full/3.0.x/dist/masses/README.perceptron>

The Apache SpamAssassin Group. (2005). SpamAssassin public corpus. <http://-spamassassin.apache.org/publiccorpus/>

The Apache SpamAssassin Group. (2009a). Genetic Algorithm for SpamAssassin.http://wiki.apache.org/spamassassin/GeneticAlgorithm

The Apache SpamAssassin Group. (2009b). Measuring filter spam accuracy. http://-wiki.apache.org/spamassassin/MeasuringAccuracy

The Apache SpamAssassin Group. (2010). SAGA Implementation algorithm. <spamassassin.apache.org/full/2.6x/dist/masses/craig-evolve.c>

The Apache SpamAssassin Project. (2011). The Powerful #1 Open-Source Spam Filter. SpamAssassin <http://spamassassin.apache.org>

The Apache SpamAssassin Group. (2012). Using network tests. <http://-wiki.apache.org/spamassassin/UsingNetworkTests>

Tobin, F. (2009). Pyzor distributed network system. <sourceforge.net/apps/trac/pyzor/>

van Rijsbergen, C. J. (1979). *Information retrieval*. (2nd ed). London: Butterworths. < www.dcs.gla.ac.uk/Keith/Preface.html >

Wong M., & Schlitt, W. (2006). RFC-4408: Sender Policy Framework. <www.ietf.org/-rfc/rfc4408.txt>

Zitzler, E., & Thiele, L. (1998). Multiobjective optimization using evolutionary algorithms - A comparative case study. In *Proceedings of the 5th International Conference on Parallel Problem Solving from Nature* (pp 292-304). London, UK: Springer-Verlag.

Zitzler, E., Laumanns, M., & Thiele, L. (2002). SPEA2: Improving the strength pareto evolutionary algorithm for multiobjective optimization. In Giannakoglou, K.C., Tsahalis, D.T., Periaux, J., Fogarty,T (Eds.), *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems* (pp. 95–100). CIMNE.

# FIGURE LEGENDS

**Figure 1.** Multiobjective presentation of the scores setting for anti-spam filtering problem

**Figure 2.** Second comparison stage

**Figure 3.** EAF NSGA-II and SPEA2 attainment surfaces

**Figure 4.** EAF NSGA-II and EAF SPEA2 attainment surfaces vs Grindstone4SPAM

**Figure 5.** EAFs associated to the outcomes of NSGA-II and SPEA2 algorithms

**Figure 6.** Differences between the NSGA-II and SPEA2 EAFs

**Figure 7.** Percentage of FPs, FNs and TPs+TNs (OK)

**Figure 8.** Recall and precision measures comparison

**Figure 9.** TCR measures using distinct values of λ

# TABLE LEGENDS

**Table 1.** SpamAssassin filtering techniques description

**Table 2.** Available corpora

**Table 3.** Fold-division for cross-validation process

**Table 4.** FNs, FPs, TNs and TPs obtained from the experimental results

**Table 5.** F-score measures comparison

**Table 6.** Batting average measure comparison

# Vitae

**Iryna Yevseyeva** received her PhD degree in computer science and optimisation from the University of Jyvaskyla, Finland, in 2007, for the research on multicriteria classification. She was a post-doctoral researcher on multiobjective optimization at the University of Algarve and INESC Porto (Portugal), and Leiden University (The Netherlands). Her main research interests are in multicriteria decision analysis and optimisation (algorithms development and applications). She has been involved in several international research projects and published a number of scientific papers.

**Vitor Basto-Fernandes** got his PhD on multimedia transport protocols from the University of Minho, Portugal, in 2006. He has been lecturing on distributed systems, computer networks, information system integration, and cognitive networks at the Universities of Minho and University ofTrás-os-Montes e Alto Douro, Portugal. In 2008 he joined Polytechnic Institute of Leiria, Portugal, as assistant professor. He has been involved in international research projects, published a number of scientific papers, and performed industry based activities in the area of e-commerce and web-based systems.

**David Ruano-Ordás** is Ph.D. student from the University of Vigo, Spain. He has large experience on Linux administration and software development in C. He collaborates as a researcher with the SING group at the University of Vigo. His main research interests are in data mining techniques applied to different AI problems, such as spam filtering.

**José R. Méndez** is Ph.D. assistant professor from the University of Vigo, Spain. He worked as system administrator, software developer and IT consultant in the civil service and IT Industry during 10 years. He received his PhD in 2006 from the University of Vigo, Spain. He is co-author of several scientific papers in the domain of anti-spam filtering and developer of patented SpamHunting technique. His main research interest is in development and improvement of anti-spam filters.

***Highlights (for review)**

Analysis of current spam-filter software operation
Review of spam filter optimization schemes
Multiobjective evolutionary techniques for optimizing anti-spam filters

**Table 1**

| Method | Filter Type Technique | Plug-in name | Description |
|---|---|---|---|
| *Content-based* | *Naïve Bayes* (NB)<br>(Metsis, Androutsopoulos & Paliouras, 2006; Androutsopoulos, Koustias, Chandrinos, Paliouras & Spyropoulos, 2000) | Bayes.pm | Calculate the probability of an email being spam by computing NB probability. |
| | Language Guessing | TextCat.pm | Guesses the language of the received message. |
| *Collaborative* | Vipul´s Razor<br>(Prakash & Ritter, 2007) | Razor2.pm | Distributed, collaborative, spam detection and filtering network. |
| | Pyzor<br>(Tobin, 2009) | Pyzor.pm | Collaborative, networked system to detect and block spam using digests of messages. |
| | Distributed Checksum Clearinghouses<br>(Rhyolite Software, 2000) | DCC.pm | Collaborative, networked system to detect and block spam using checksums of messages. |
| | DNS-based *Blackhole List* (RBL)<br>(Levine,2010) | DNSEval.pm | Lists of server *Internet Protocol* (IP) addresses from *Internet Service Providers* (ISPs) whose customers are responsible for the spam and from ISPs whose servers are hijacked for spam relay. |
| | SpamCop<br>(Cisco Systems, 2010) | SpamCop.pm | Free spam reporting service, allowing recipients of *Unsolicited Bulk Email* (UBE) and *Unsolicited Commercial Email* (UCE) to report offenders to the ISPs senders. |
| *Domain-authentication* | *Sender Policy Framework* (SPF)<br>(Wong & Schlitt, 2006) | SPF.pm | Is able to detect message spoofing by verifying sender IP addresses. |
| | *DomainKeys Identified Mail* (DKIM)<br>(Allman, Callas, Delany, Libbey, Fenton & Thomas, 2007) | DKIM.pm | DKIM implements sender verification scheme using *Public Key Infrastructure* (PKI) mechanisms. |
| *RFC2822 structure and syntax* | *Regular Expressions* (REGEX) | MIMEEval.pm | Allows regular expression rules to be written against *Multipurpose Internet Mail Extensions* (MIME) (Freed & Borestein, 1996a; Freed & Borestein, 1996b; Moore, 1996; Freed & Klensin, 2005a; Freed & Klensin, 2005b, Freed & Borenstein, 1996c) headers in the message. |
| | | MIMEHeader.pm | Performs regular expressions tests against MIME headers. |
| | | URIEval.pm | Checks and evaluates message URI (Uniform Resource Identifier) type. |
| | Content parsers | BodyEval.pm | Checks the correctness of the message body structure. |
| | | HTMLEval.pm | Checks the structure of *HyperText Markup Language* (HTML) code embedded inside the message. |

**Table 2**

| Collection name | | message source | percentage of legitimate emails | percentage of spam emails | total number of messages |
|---|---|---|---|---|---|
| SpamAssassin<br>(The Apache SpamAssassin Group, 2005) | | Public forums and user donations | 69% | 31% | 6047 |
| Junk-Email<br>(Orăsan & Krishnamurthy, 2002) | | Multiple domains | 0% | 100% | 1.563 |
| Bruce Guenter (Guenter, 1998) | | Own contributions | 0% | 100% | 171000 |
| SING (SING Group, 2005) | | University environment | 69.7% | 39.3% | 20130 |
| CSDMC2010 (CSMINING Group, 2010) | | ICONIP 2010 dataset | 68.1% | 31.9% | 4327 |
| 2005 | TRECSpam<br>(Text REtrieval conference, 2009) | Multiple domains | 43.0% | 57.0% | 92189 |
| 2006 | | | 35.0% | 65.0% | 37822 |
| 2007 | | | 33.5% | 66.5% | 75419 |
| Enron-Spam corpus<br>(Metsis, Androutsopoulos & Paliouras, 2006) | | Multiple domains | 37.0% | 63.0% | 52076 |
| Static ECUE Spam (ECUE, 2011) | | Individual user | 50.0% | 50.0% | 5000 |

**Table 3**

| | | | **SpamAssassin corpus** | |
|---|---|---|---|---|
| | **Step name** | **corpus ratio** | **ham messages** | **spam messages** |
| **First fragment** | Test instances | $\frac{1}{10} < corpus\_size >$ | 415 | 189 |
| | Filter optimisation | $\frac{9}{10} < corpus\_size >$ | 3735 | 1701 |
| **Second fragment** | Train Bayes | $\frac{8}{10} < corpus\_size >$ | 3320 | 1512 |
| | Train Bayes test | $\frac{9}{10} < corpus\_size >$ | 3735 | 1701 |

**Table 4**

| System | FNs | FPs | TNs | TPs |
|---|---|---|---|---|
| Grindstone4SPAM | 17 | 42 | 6909 | 2381 |
| NSGA-II | 31 | 20 | 6931 | 2367 |
| SPEA 2 | 33 | 22 | 6929 | 2365 |

**Table 5**

| | f-score | | |
|---|---|---|---|
| **System** | **β =1** | **β =1.5** | **β =2** |
| Grindstone4SPAM | 0.987761875 | 0.989735883 | 0.990844777 |
| NSGA-II | 0.989341693 | 0.988467716 | 0.987978963 |
| SPEA 2 | 0.988505747 | 0.987632509 | 0.987144169 |

## Table 6

| | Batting average | |
|---|---|---|
| **System** | **Hit rate** | **Strike rate** |
| Grindstone4SPAM | 0.99291076 | 0.006042296 |
| NSGA-II | 0.98707256 | 0.002877284 |
| SPEA 2 | 0.98623853 | 0.003165012 |

Figure 1
Click here to download high resolution image

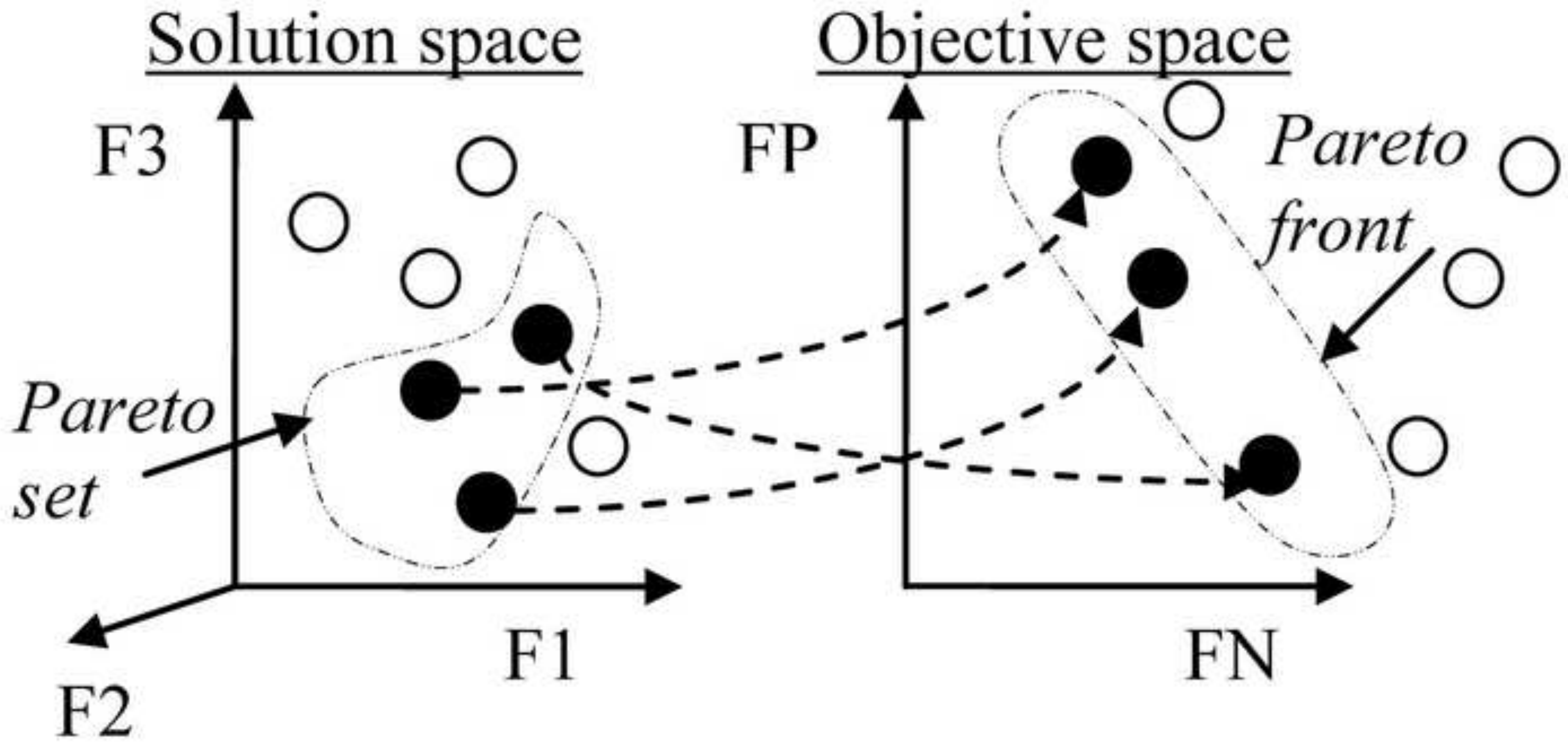


Figure 2

Click here to download high resolution image

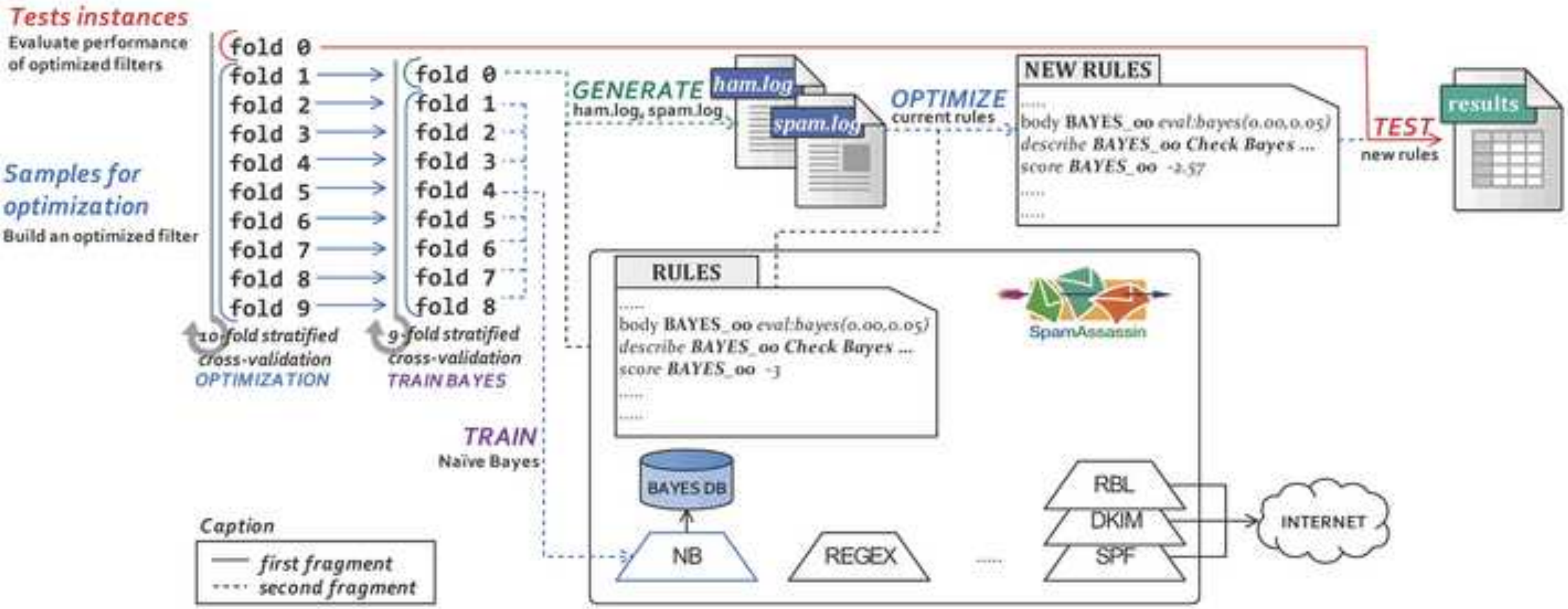
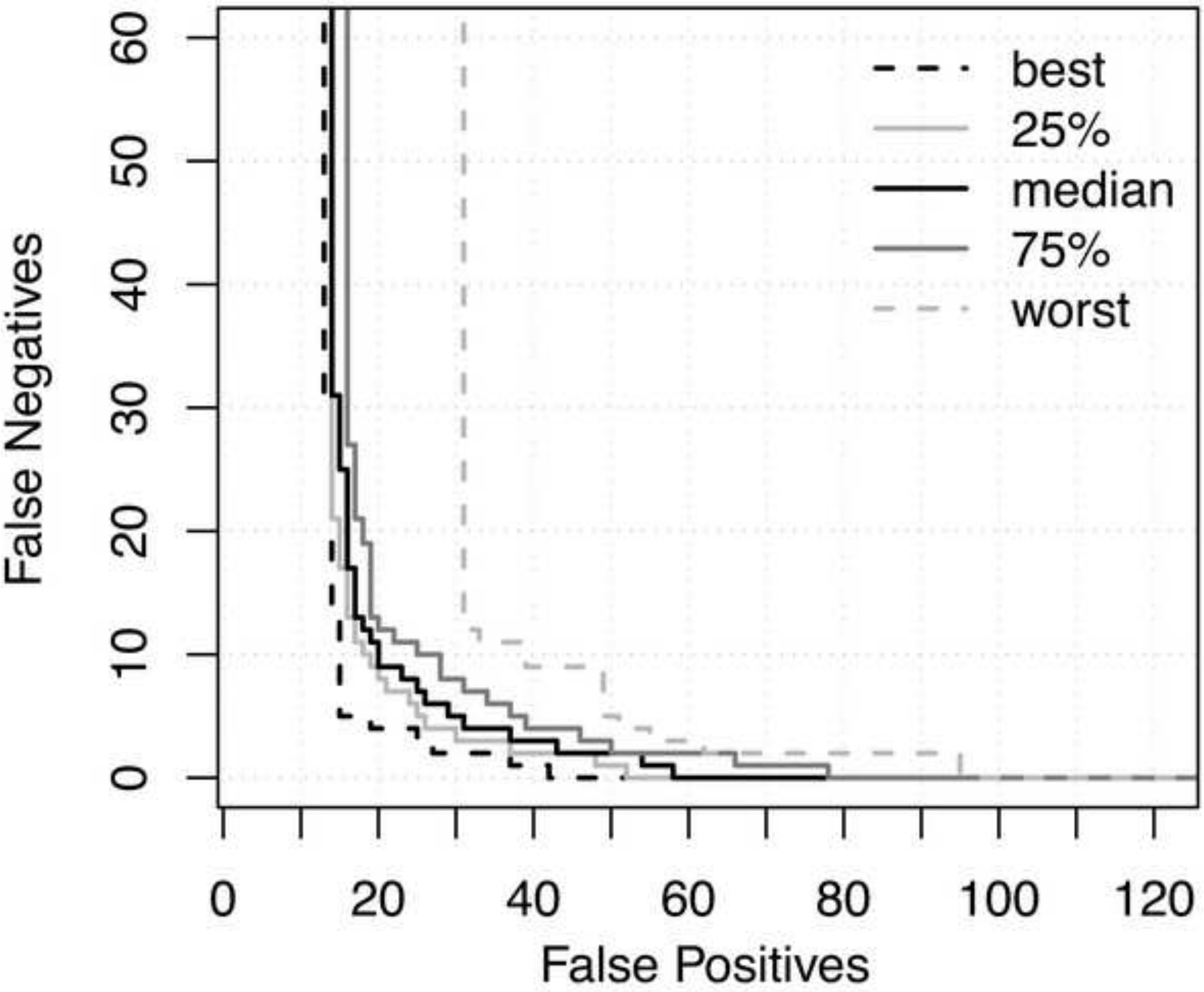


Figure 3i

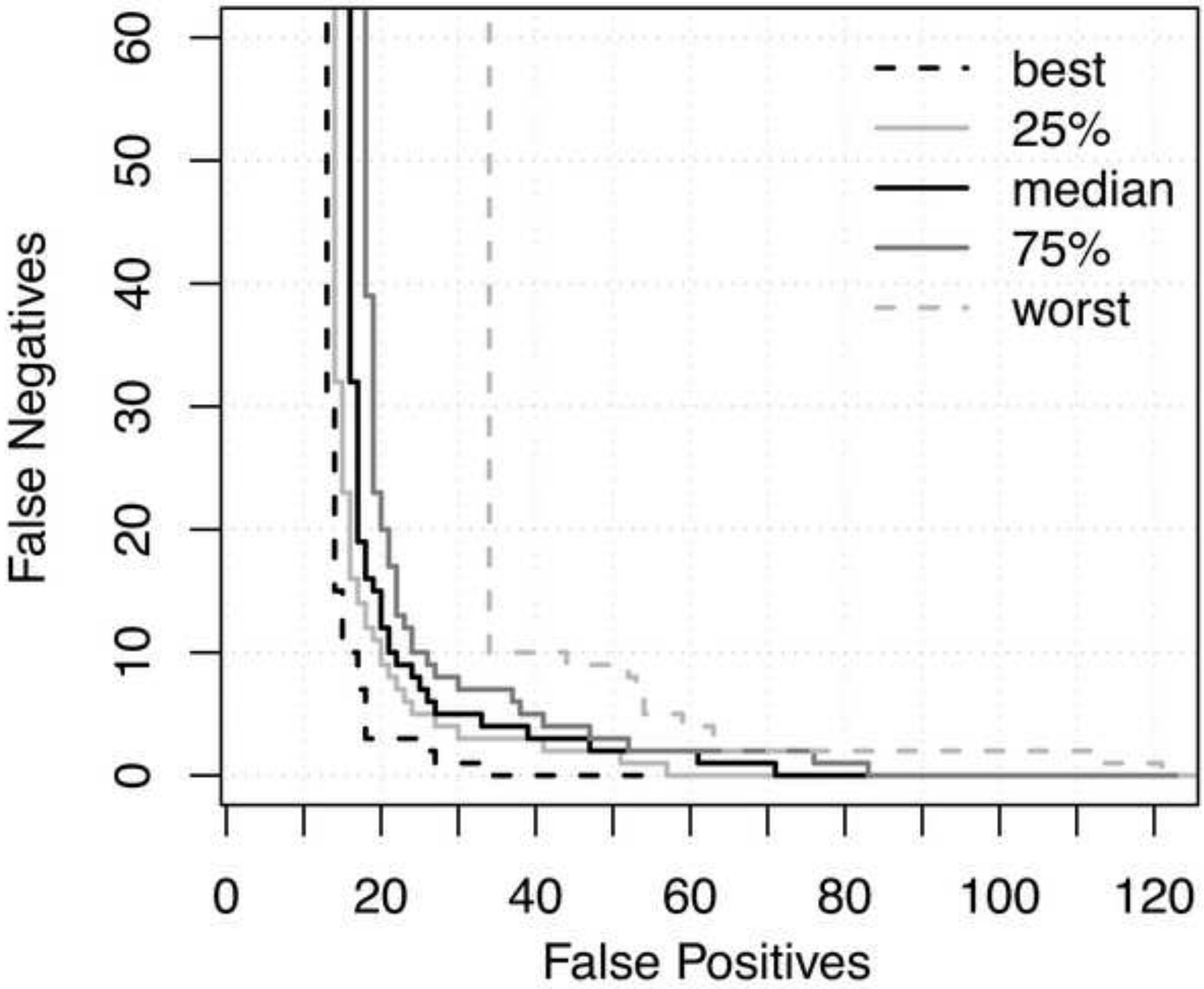
Click here to download high resolution image



*i)* NSGA-II

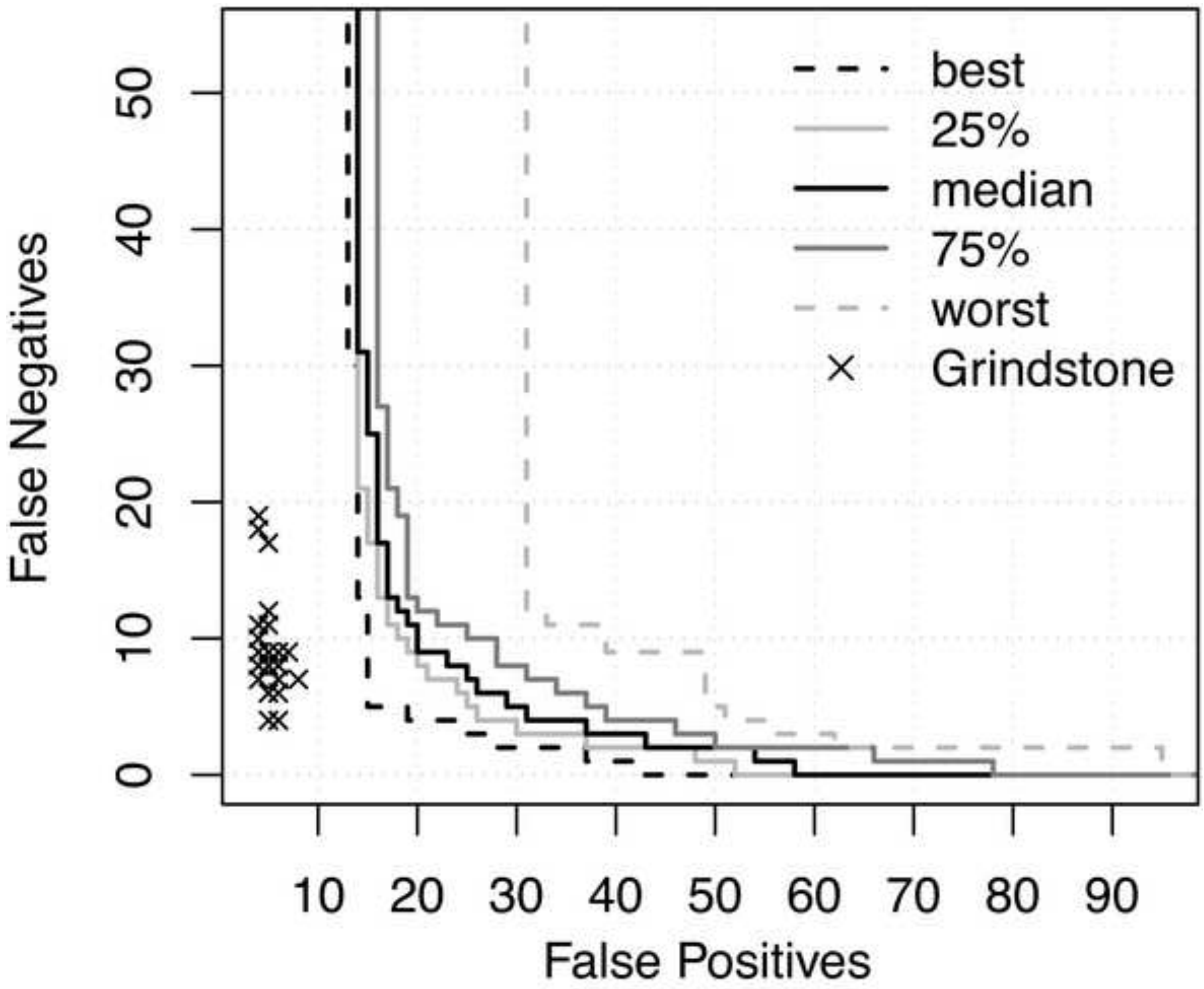
Figure 3ii

Click here to download high resolution image



*ii)* SPEA2

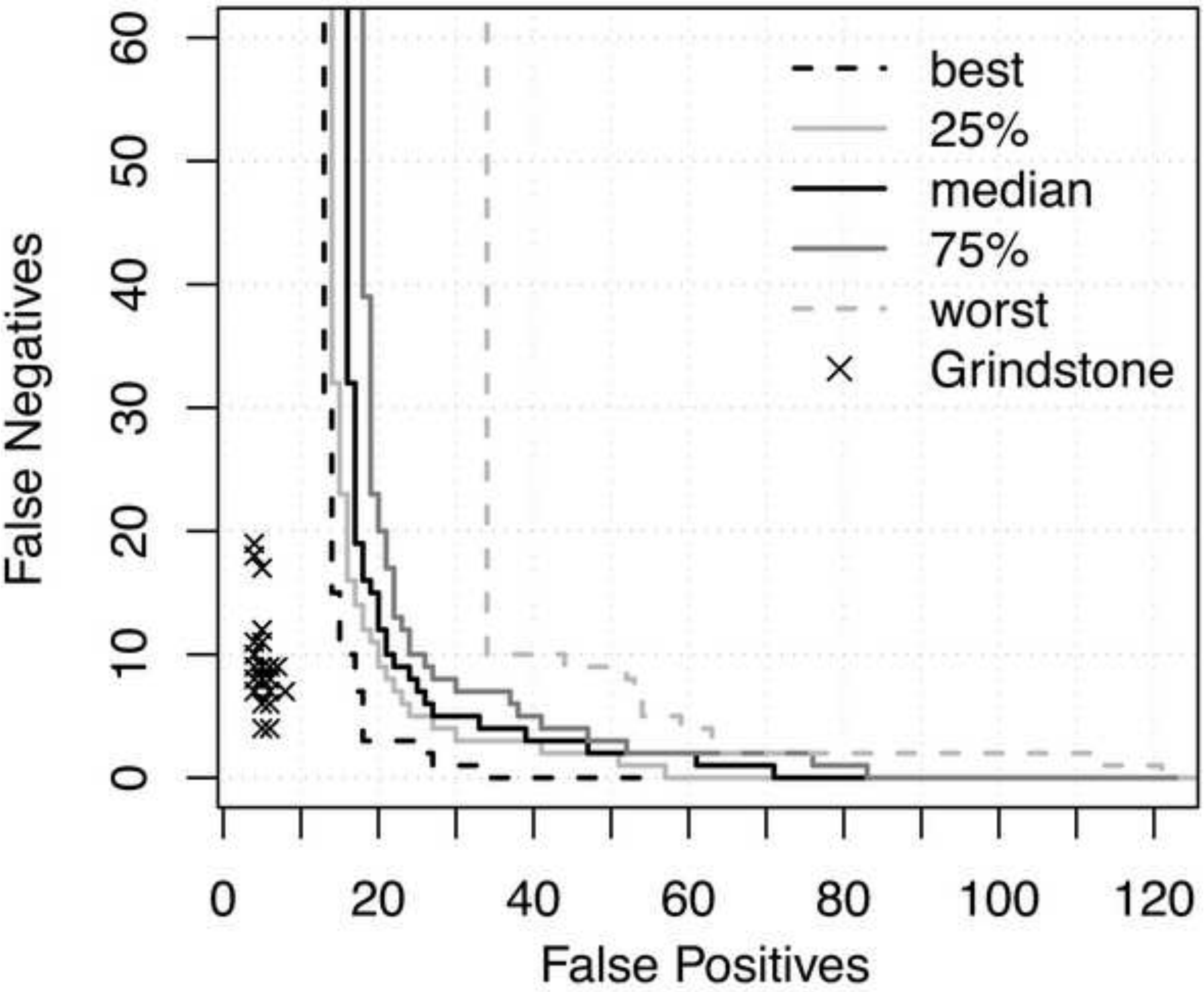
Figure 4i
Click here to download high resolution image



*i)* EAF NSGA-II attainment surface vs Grindstone4SPAM

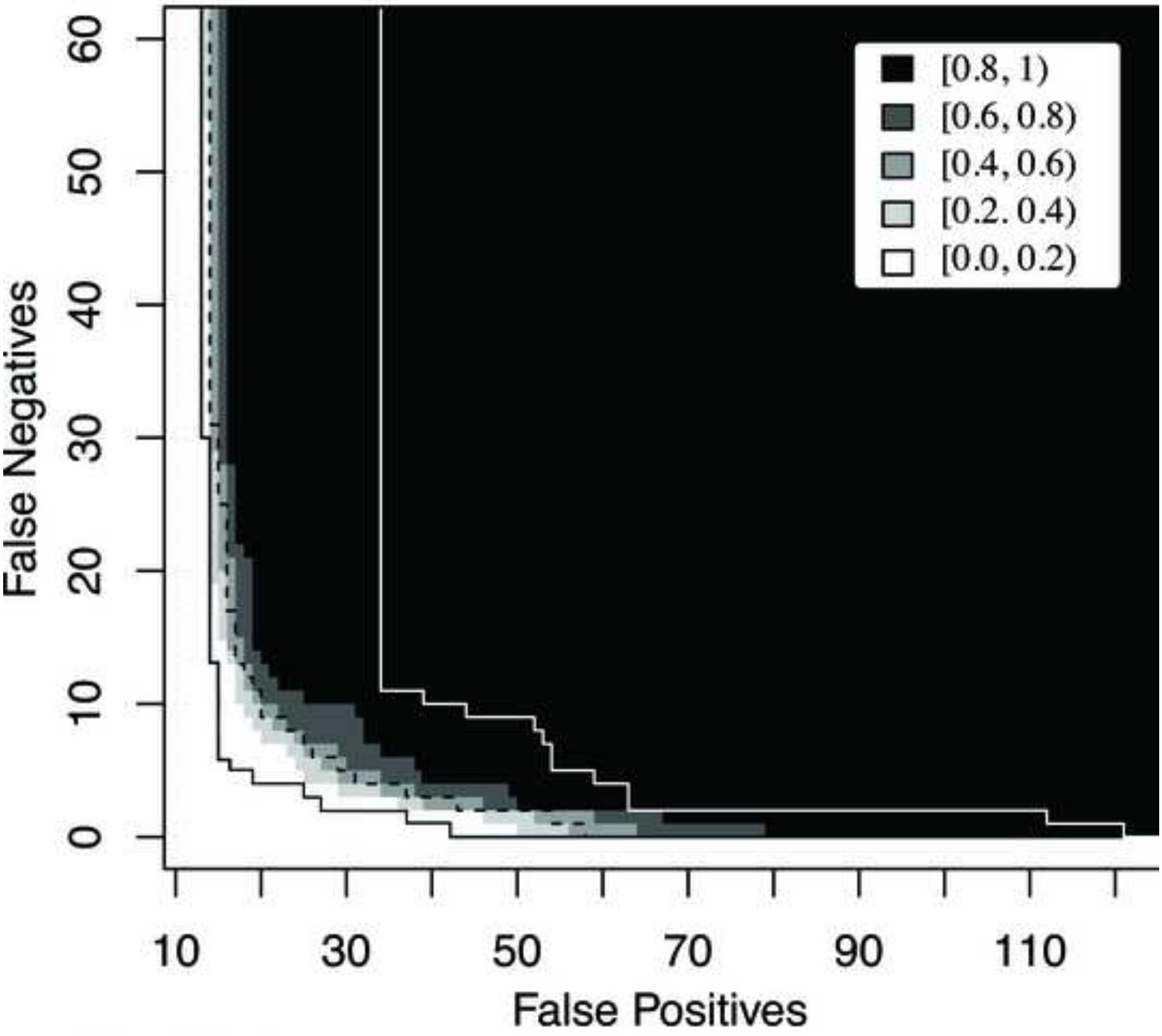
Figure 4ii

Click here to download high resolution image



*ii)* EAF SPEA2 attaintment surface vs Grindstone4SPAM

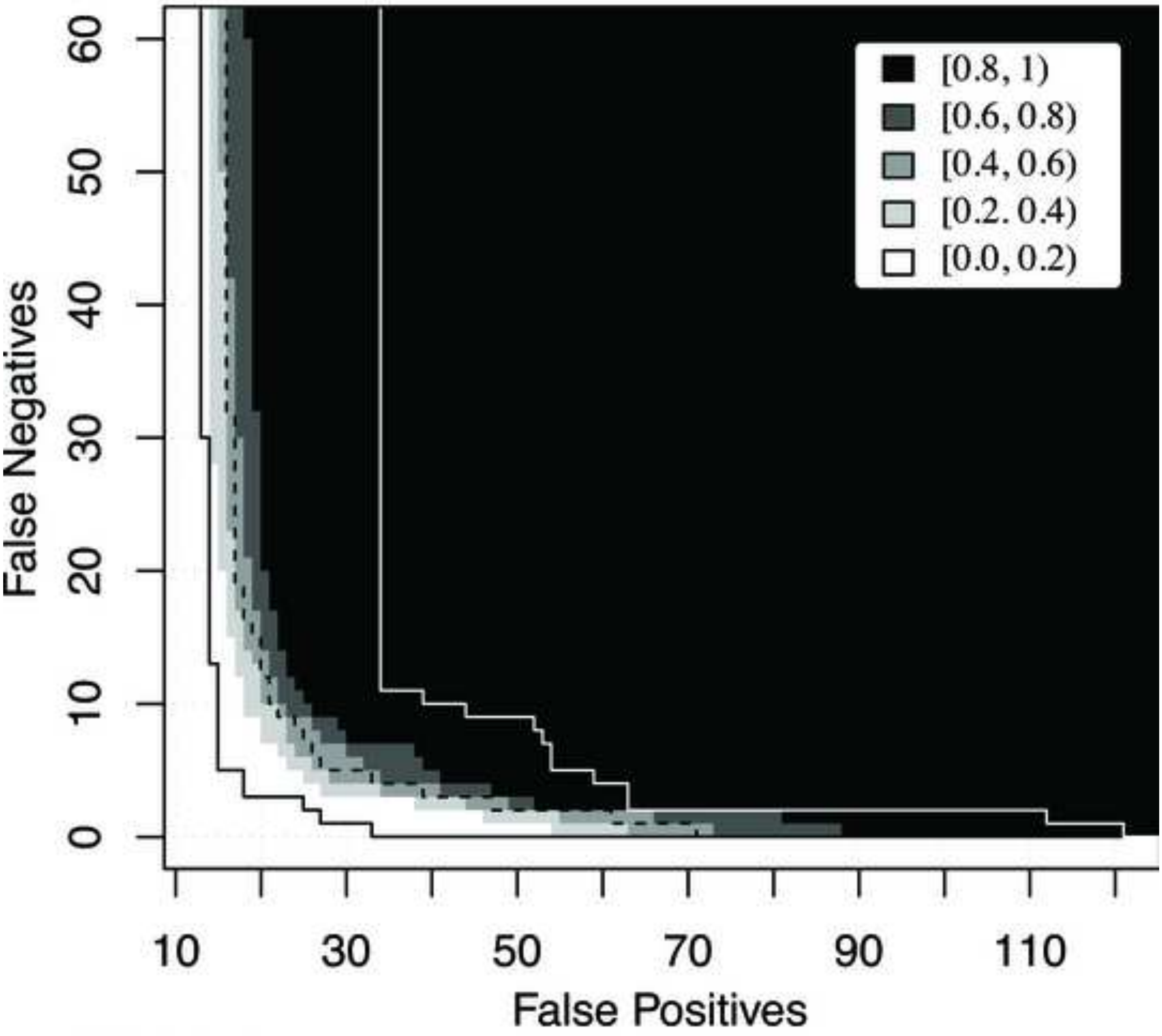
Figure 5i
Click here to download high resolution image



*i)* NSGA-II EAF

Figure 5ii

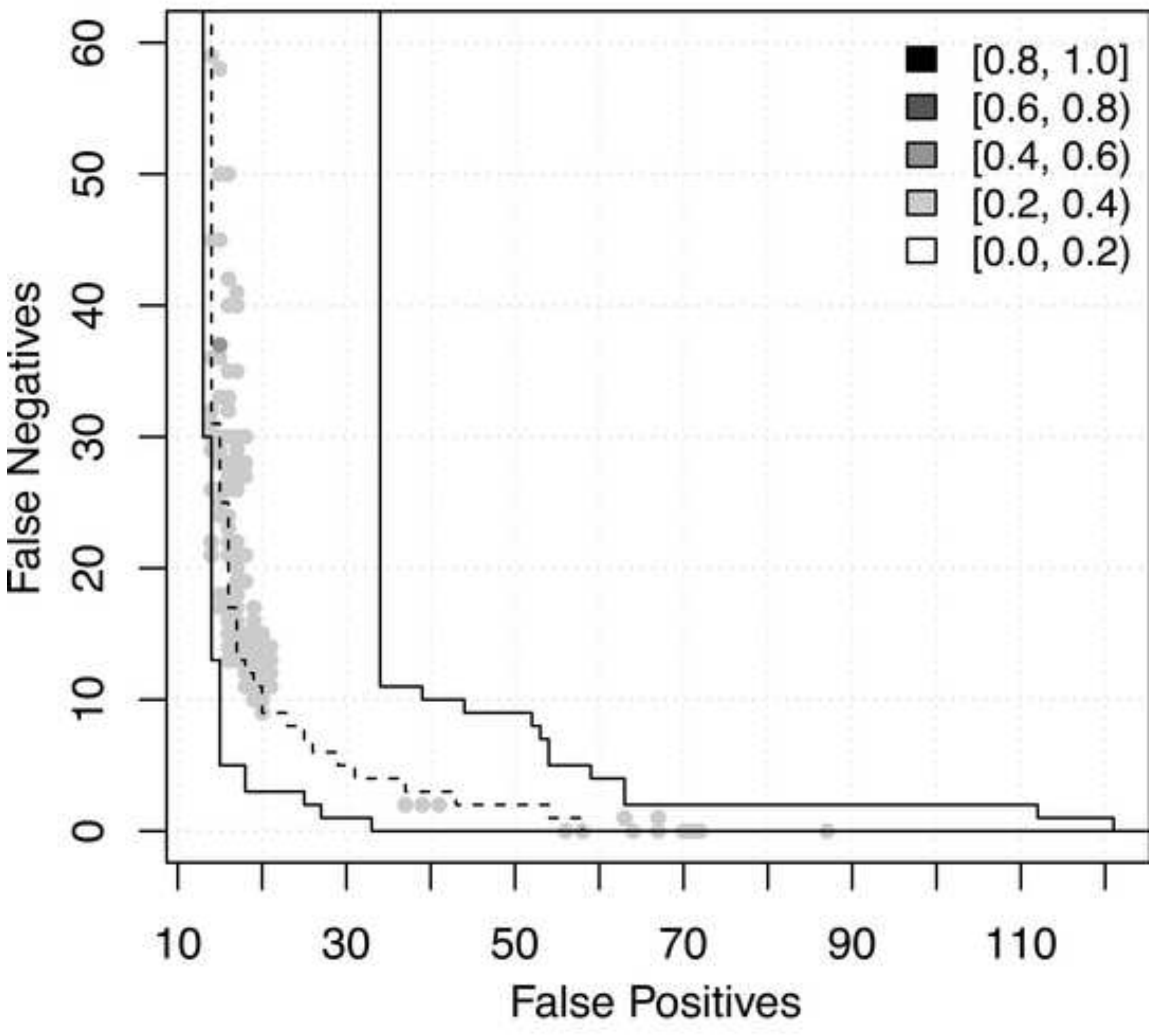
Click here to download high resolution image



*ii)* SPEA2 EAF

Figure 6i

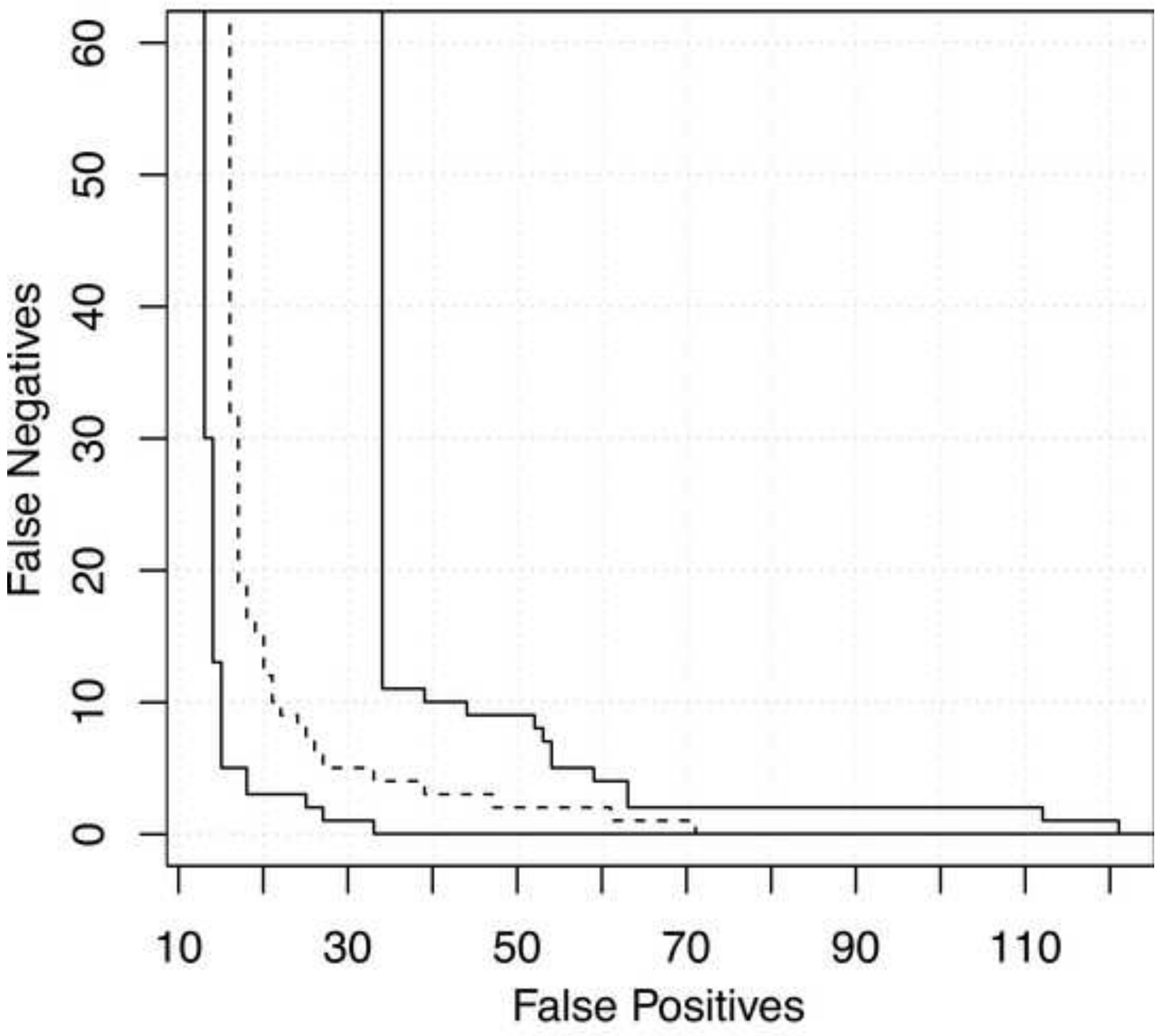
Click here to download high resolution image



*i)* EAF differences in favour of NSGA-II

Figure 6ii

Click here to download high resolution image



*ii)* EAF differences in favour of SPEA2

Figure 7

Click here to download high resolution image

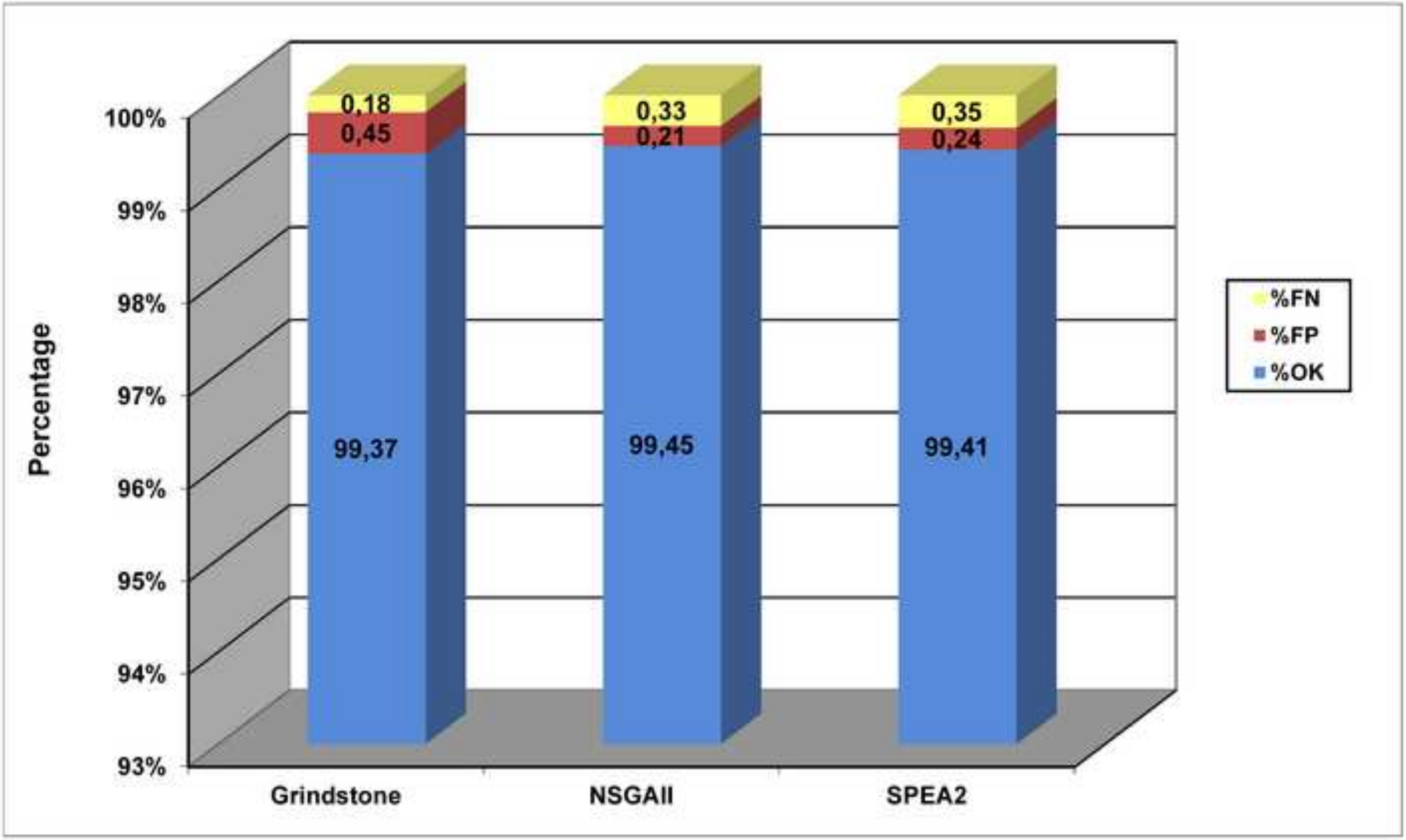


Figure 8

Click here to download high resolution image

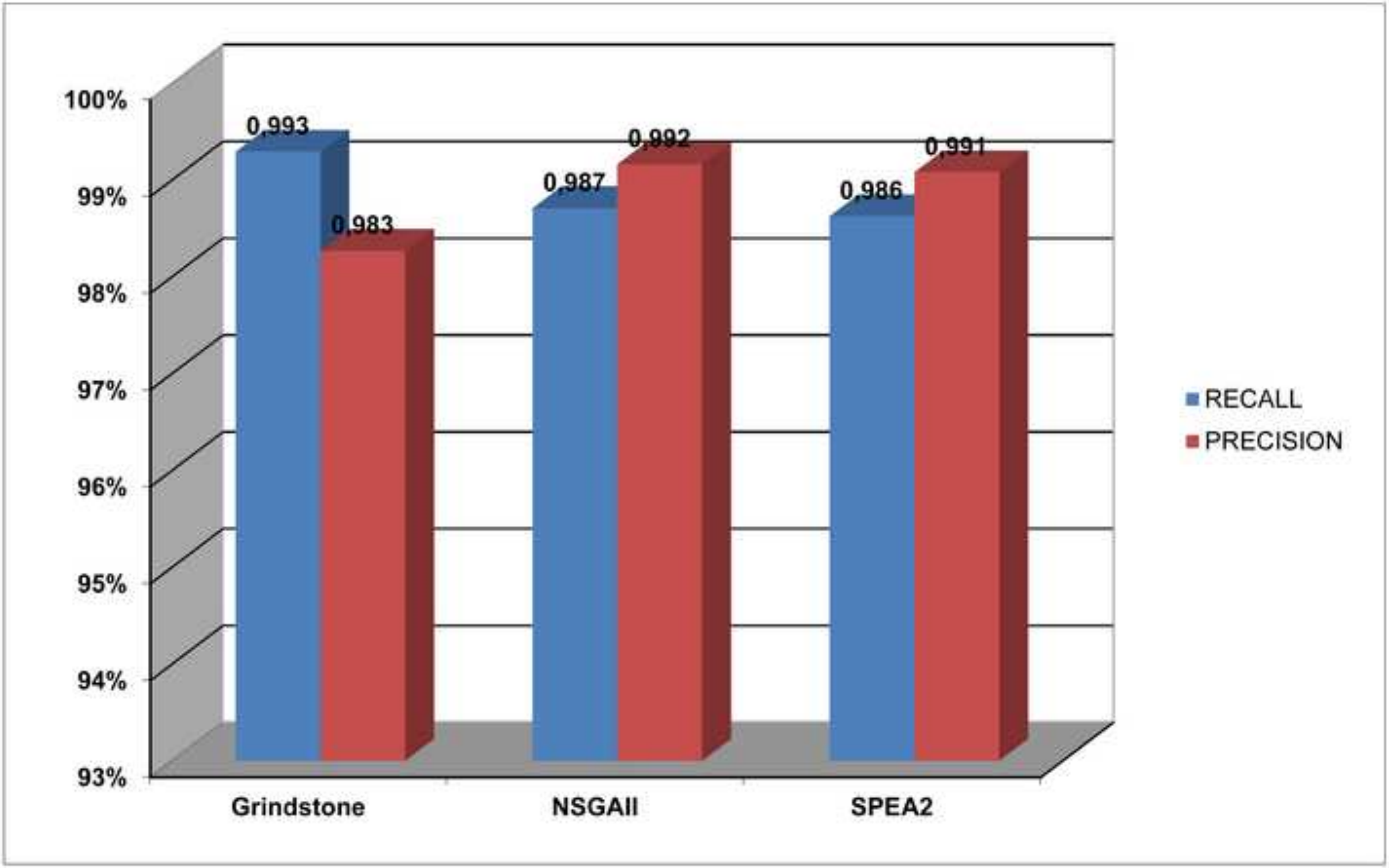


Figure 9i

Click here to download high resolution image

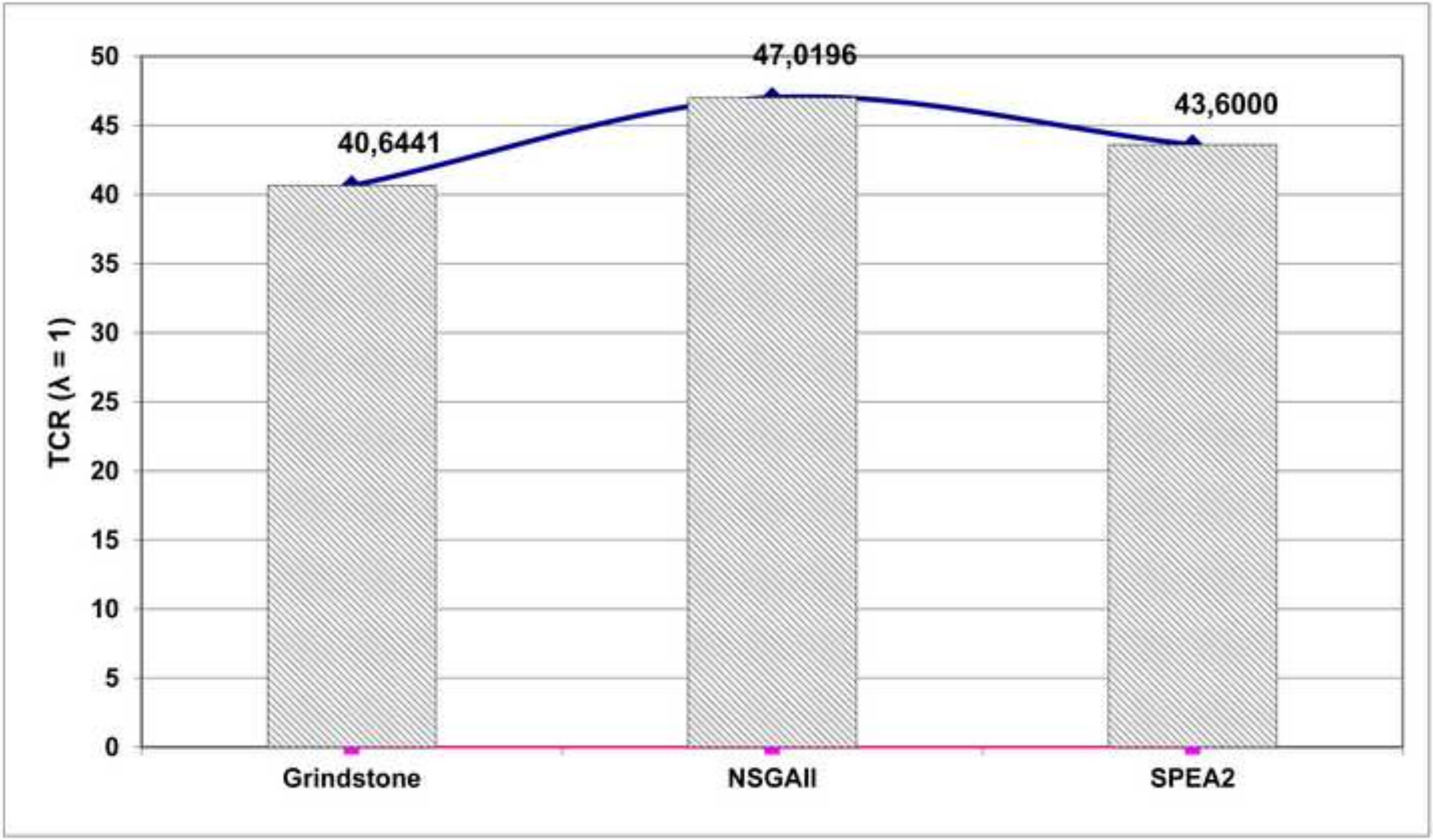


Figure 9ii

Click here to download high resolution image

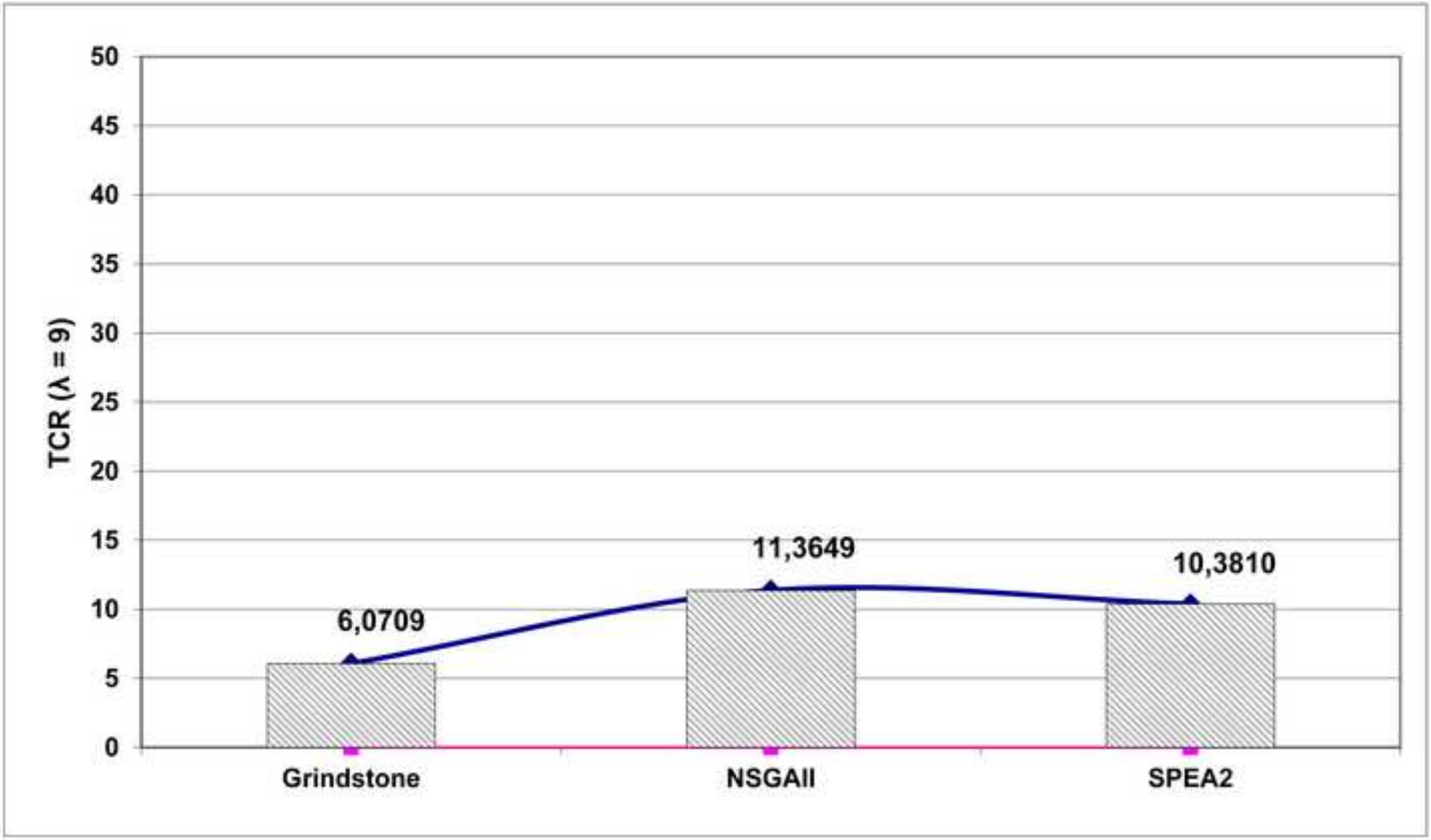
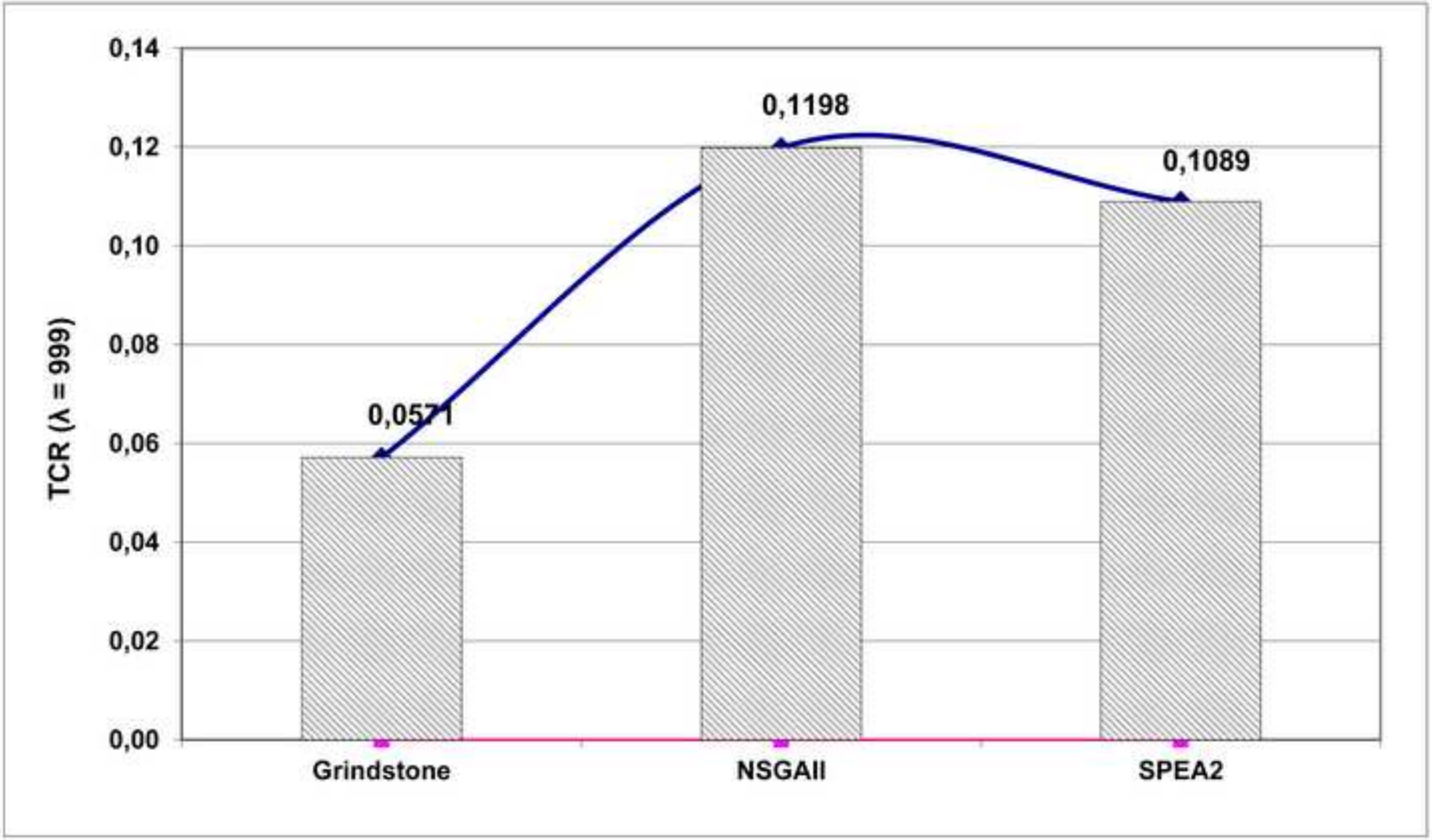


Figure 9iii

Click here to download high resolution image


TCR (λ = 999)
Grindstone
NSGAII
SPEA2