Voting Systems and Automated Reasoning: the QBFEVAL Case Study

Massimo Narizzano and Luca Pulina and Armando Tacchella [1]

Abstract

Systems competitions play a fundamental role in the advancement of the state of the art in several automated reasoning fields. The goal of such events is to answer the question: "Which system should I buy?". Usually the answer comes as the byproduct of a ranking obtained by considering a pool of problem instances and then aggregating the performances of the systems on each member of the pool. In this paper, we consider voting systems as an alternative to other procedures which are well established in automated reasoning contests. Our research is aimed to compare methods that are customary in the context of social choice, with methods that are targeted to artificial settings, including a new hybrid method that we introduce. Our analysis is empirical, in that we compare the aggregation procedures by computing measures which should account for their effectiveness using the data from the 2005 evaluation of quantified Boolean formulas solvers that we organized. The results of our experiments give useful indications about the relative strengths and weaknesses of the procedures under test, and allow us to infer also some conclusions that are independent of the specific procedure adopted.

1 Introduction

Systems competitions play a fundamental role in the advancement of the state of the art in several automated reasoning fields. A non-exhaustive list of such events includes the CADE ATP System Competition (CASC) [1] for theorem provers in first order logic, the SAT Competition [2] for propositional satisfiability solvers, the International Planning Competition (see, e.g., [3]) for symbolic planners, the CP Competition (see, e.g., [4]) for constraint programming systems, the Satisfiability Modulo Theories (SMT) Competition (see, e.g., [5]) for SMT solvers, and the evaluation of quantified Boolean formulas solvers (QBFEVAL, see [6, 7, 8] for previous reports). The main purpose of the above events is to designate a winner, i.e., to answer the question: "Which system should I buy?". Even if such perspective can be limiting, and the results of automated reasoning systems competitions may provide less insight than controlled experiments in the spirit of [9], there is a general agreement that competitions raise interest in the community and they help to set research challenges for developers and assess the current technological frontier for users. The usual way to designate a winner in competitions is to compute a ranking obtained by considering a pool of problem instances and then aggregating the performances of the systems on each member of the pool. While the definition of performances can encompass many

[1] The authors wish to thank the Italian Ministry of University and Research (MIUR) for its financial support, the anonymous reviewers who helped us to improve the quality of the paper, and Elena Seghezza for making some relevant references available to us.

aspects of a system, usually it is the capability of giving a sound solution to a high number of problems in a relatively short time that matters most. Therefore, one of the issues that occurred to us as organizers of QBFEVAL, relates to the procedures used to compute the final ranking of the solvers, i.e., we had to answer the question "Which aggregation procedure is best?". Indeed, even if the final rankings cannot be interpreted as absolute measures of merit, they should at least represent the relative strength of a system with respect to the other competitors based on the difficulty of the problem instances used in the contest.

Our analysis of aggregation procedures considers three voting systems, namely Borda's method [10], range voting [11] and Schulze's method [12], as an alternative to methods which are well estabished in automated reasoning contests, namely CASC [1], the SAT competitions [2], and QBFEVAL [13] (before 2006). We adapted voting systems to the artificial setting of systems competition by considering the systems as candidates and the problem instances as voters. Each instance casts its vote on the systems in such a way that systems with the best performances on the instance will be preferred over other candidates. The individual preferences are aggregated to obtain a collective choice that determines the winner of the contest. Our motivation to investigate methods which are customary in the context of social choice by applying them to the artificial setting of systems competitions is twofold. First, although voting systems do not enjoy a great popularity in automated reasoning systems contests (one exception is Robocup [14] using Borda's method), there is a substantial amount of literature in social choice (see, e.g., [15]) that deals with the problem of identifying and formalizing appropriate methods of aggregation in specific domains. Second, voting can be seen as a way to "infer the candidates' absolute goodness based on the voters' noisy signals, i.e., their votes." [16]. Therefore, the use of voting systems as aggregation procedures could pave the way to extracting hints about the absolute value of a system from the results of a contest.

In the paper, we also propose a new procedure called YASM ("Yet Another Scoring Method")[2] that we selected as an aggregation procedure for QBFEVAL'06. YASM is an hybrid between a voting system and traditional aggregation procedures used in automated reasoning contests. Our results show that YASM provides a good compromise when considering some measures that should quantify desirable properties of the aggregation procedures. In particular, the measures we propose account for:

- the degree of fidelity of the procedures, i.e., given a synthesized set of raw data, evaluate whether a procedure distorts the results;
- the degree of stability of each procedure with respect to perturbations (i) in the size of the test set, (ii) in the amount of resources available (CPU time), and (iii) in the quality of the test-set;
- the representativeness of each procedure with respect to the state of the art expressed by the competitors.

[2]The terminology "scoring method" is somewhat inappropriate in the context of social choice, as it recalls a positional scoring procedure such as Borda's method and range voting; we decided to keep the original terminology for consistency across the previous works [17, 18, 21].

We compute the above measures using part of the results from QBFEVAL'05 [8]. In particular, the results of our experiments give useful indications about the relative strengths and weaknesses of the aggregation procedures, and allow us to infer also some conclusions that are independent of the specific method adopted.

This paper builds on and extends previous work by one of the authors [17]. First, the version of YASM that we present here improves on the one presented in [17]. In particular, the new YASM is simpler and more effective when compared to the old one. Moreover, the comparison of aggregation procedures is broadened by the addition of new effectiveness measures (fidelity, see Section 4), and an improved definition of State-Of-The-Art (SOTA) relevance (see Section 4).

The paper is structured as follows. In Section 2 we introduce the case study of QBFEVAL'05 [8], and we introduce the state of the art aggregation procedures. In Section 3 we introduce our new aggregation procedure, and then we compare it with other methods in Section 4 using several effectiveness measures. We conclude the paper in Section 5 with a discussion about the presented results.

2 Preliminaries

2.1 QBFEVAL'05

QBFEVAL'05 [8] is the third in a series of non-competitive events that preceded QBFEVAL'06. QBFEVAL'05 accounted for 13 competitors, 553 quantified Boolean formulas (QBFs) and three QBF generators submitted. The test set was assembled using a selection of 3191 QBFs obtained considering the submissions and the instances archived in QBFLIB [19]. The results of QBFEVAL'05 can be listed in a table RUNS comprised of four attributes (column names): SOLVER, INSTANCE, RESULT, and CPUTIME. The attributes SOLVER and INSTANCE report which solver is run on which instance. RESULT is a four-valued attribute: SAT, i.e., the instance was found satisfiable by the solver, UNSAT, i.e., the instance was found unsatisfiable by the solver, TIME, i.e., the solver exceeded a given time limit without solving the instance (900 seconds in QBFEVAL'05), and FAIL, i.e., the solver aborted for some reason (e.g., a run-time error, an inherent limitation of the solver, or any other reason beyond our control). Finally, CPUTIME reports the CPU time spent by the solver on the given instance, in seconds. In the analysis herewith presented we used a subset of QBFEVAL'05 RUNS table, including only the solvers that, as far as we know, work correctly (the solvers of the second stage of the evaluation) and the QBFs coming from classes of instances having fixed structure (see [8] for more details). Under these assumptions, RUNS table reduces to 4408 entries, one order of magnitude less than the original one. This choice allows us to disregard correctness issues, to reduce considerably the overhead of the computations required for our analysis, and, at the same time, maintain a significant number of runs. The aggregation procedures that we evaluate, the measures that we compute and the results that we obtain, are based on the assumption that a table identical to RUNS as described above is the only input required by a procedure. As a consequence, the aggregation procedures (and thus our analysis) do not take into

account (i) memory consumption, (ii) correctness of the solution, and (iii) "quality" of the solution.

2.2 State of the art aggregation procedures

In the following we describe in some details the state of the art aggregation procedures used in our analysis. For each method we describe only those features that are relevant for our purposes. Further details can be found in the references provided.

CASC [1] Using CASC methodology, the solvers are ranked according to the number of problems solved, i.e., the number of times RESULT is either SAT or UNSAT. Under this procedure, solver A is better than solver B, if and only if A is able to solve at least one problem more than B within the time limit. In case of a tie, the solver faring the lowest average on CPUTIME fields over the problems solved is the one which ranks first.

QBF evaluation [13] QBFEVAL methodology is the same as CASC, except for the tie-breaking rule, which is based on the sum of CPUTIME fields over the problems solved.

SAT competition [2] The last SAT competition uses a *purse-based method*, i.e., the measure of effectiveness of a solver on a given instance is obtained by adding up three purses:

- the solution purse, which is divided equally among all solvers that solve the problem;
- the speed purse, which is divided unequally among all the competitors that solve the problem, first by computing the speed factor $F_{s,i}$ of a solver s on a problem instance i:
$$F_{s,i} = \frac{k}{1 + T_{s,i}} \tag{1}$$
where k is an arbitrary scaling factor (we set $k = 10^4$ according to [20]), and $T_{s,i}$ is the time spent by s to solve i; then by computing the speed award $A_{s,i}$, i.e., the portion of speed purse awarded to the solver s on the instance i:
$$A_{s,i} = \frac{P_i \cdot F_{s,i}}{\sum_r F_{r,i}} \tag{2}$$
where r ranges over the solvers, and P_i is the total amount of the speed purse for the instance i.
- the series purse, which is divided equally among all solvers that solve at least one problem in a given series (a series is a family of instances that are somehow related, e.g., different QBF encodings for some problem in a given domain).

The overall ranking of the solvers under this method is obtained by considering the sum of the purses obtained on each instance, and the winner of the contest is the solver with the highest sum.

Borda's method [10] Suppose that n solvers (candidates) and m instances (voters) are involved in the contest. Consider the sorted list of solvers obtained for each instance by considering the value of the CPUTIME field in ascending order. Let $p_{s,i}$ be the position of a solver s ($1 \leq s \leq n$) in the list associated with instance i ($1 \leq i \leq m$). According to Borda's method, each voter's ballot consists of a vector of individual scores given to candidates, where the score $S_{s,i}$ of solver s on instance i is simply $S_{s,i} = n - p_{s,i}$. In cases of time limit attainment or failure, we default $S_{s,i}$ to 0. The score of a candidate, given the individual preferences, is just $S_s = \sum_i S_{s,i}$, and the winner is the solver with the highest score.

Range voting [11] Again, suppose that n solvers and m instance are involved in the contest and $p_{s,i}$ is obtained as described above for Borda's method. We let the score $S_{s,i}$ of solver s on instance i be the quantity $ar^{n-p_{s,i}}$, i.e., we use a positional scoring rule following a geometric progression with a common ratio $r = 2$ and a scale factor $a = 1$. We consider failures and time limit attainments in the same way (we call this the failure-as-time-limit model in [21]), and thus we assume that all the voters express an opinion about all the solvers. The overall score of a candidate is again $S_s = \sum_i S_{s,i}$ and the candidate with the highest score wins the election.

Schulze's method We denote as such an extension of the method described in Appendix 3 of [12]. Since Schulze's method is meant to compute a single overall winner, we extended the method according to Schulze's suggestions [22] in order to make it capable of generating an overall ranking.

3 YASM: Yet Another Scoring Method (Revisited)

While the aggregation procedures used in CASC and QBF evaluations are straightforward, they do not take into account some aspects that are indeed considered by the purse-based method used in the last SAT competition. On the other hand, the purse-based method used in SAT requires some oracle to assign purses to the problem instances, so the results can be influenced heavily by the oracle. In [17] a first version of YASM was introduced as an attempt to combine the two approaches: a rich method like the purse-based one, but using the data obtained from the runs only. As reported in [17], YASM featured a somewhat complex calculation, yielding unsatisfactory results, particularly in the comparison with the final ranking produced by voting systems. Here we revise the original version of YASM to make its computation simpler, and to improve its performance using ideas borrowed from voting systems. From here on, we call YASMv2 the revised version, and YASM the original one presented in [17]. YASMv2 requires a preliminary classification whereby a hardness degree H_i is assigned to each problem instance i using the same equation as in CASC [1] (and YASM):

$$H_i = 1 - \frac{S_i}{S_t} \tag{3}$$

	CASC	QBF	SAT	YASM	YASMv2	Borda	r.v.	Schulze
CASC	–	1	0.71	0.86	0.79	0.86	0.71	0.86
QBF		–	0.71	0.86	0.79	0.86	0.71	0.86
SAT			–	0.86	0.86	0.71	0.71	0.71
YASM				–	0.86	0.71	0.71	0.71
YASMv2					–	0.86	0.86	0.86
Borda						–	0.86	1
r. v.							–	0.86
Schulze								–

Table 1: Homogeneity of aggregation procedures.

where S_i is the number of solvers that solved i, and S_t is the total number of participants to the contest. Considering equation (3), we notice that $0 \leq H_i \leq 1$, where $H_i = 0$ means that i is relatively easy, while $H_i = 1$ means that i is relatively hard. We can then compute the measure of effectiveness $S_{s,i}$ of a solver s on a given instance i (this definition changes with respect to YASM):

$$S_{s,i} = k_{s,i} \cdot (1 + H_i) \cdot \frac{L - T_{s,i}}{L - M_i} \tag{4}$$

where L is the time limit, $T_{s,i}$ is the CPU time used up by s to solve i ($T_{s,i} \leq L$), and $M_i = min_s\{T_{s,i}\}$, i.e., M_i is the time spent on the instance i by the *SOTA solver* defined in [8] to be the ideal solver that always fares the best time among all the participants. The hybridization with voting systems comes into play with the coefficient $k_{s,i}$ which is computed as follows. Suppose that n solvers are participating to the contest. Each instance ranks the solvers in ascending order considering the value of the CPUTIME field. Let $p_{s,i}$ be the position of a solver s in the ranking associated with instance i ($1 \leq p_{s,i} \leq n$), then $k_{s,i} = n - p_{s,i}$. In case of time limit attainment and failure, we default $k_{s,i}$ to 0, and thus also $S_{s,i}$ is 0. The overall ranking of the solvers is computed by considering the values $S_s = \sum_i S_{s,i}$ for all $1 \leq s \leq n$, and the solver with the highest sum wins.

We can see from equation (4) that in YASMv2 the effectiveness of a solver on a given instance is influenced by three factors, namely (i) a Borda-like positional weight ($k_{s,i}$), (ii) the relative hardness of the instance ($1 + H_i$), and (iii) the relative speed of the solver with respect to the fastest solver on the instance ($\frac{L-T_{s,i}}{L-M_i}$). Intuitively, coefficient (ii) rewards the solvers that are able to solve hard instances, while (iii) rewards the solvers that are faster than other competitors. The coefficient $k_{s,i}$ has been added to stabilize the final ranking and make it less sensitive to an initial bias in the test set. As we show in the next Section, this combination allows YASMv2 to reach the best compromise among different effectiveness measures.

4 Experimental Evaluation

4.1 Homogeneity

The rationale behind this measure (introduced in [17]) is to verify that, on a given test set, the aggregation procedures considered (i) do not produce exactly the same solver

Method	Mean	Std	Median	Min	Max	IQ Range	F
QBF	182.25	7.53	183	170	192	13	88.54
CASC	182.25	7.53	183	170	192	13	88.54
SAT	87250	12520.2	83262.33	78532.74	119780.48	4263.94	65.56
YASM	46.64	2.22	46.33	43.56	51.02	2.82	85.38
YASMv2	1257.29	45.39	1268.73	1198.43	1312.72	95.11	91.29
Borda	984.5	127.39	982.5	752	1176	194.5	63.95
r. v.	12010.25	5183.86	12104	5186	21504	8096	24.12
SCHULZE	–	–	–	–	–	–	–

Table 2: Fidelity of aggregation procedures. As far as **SAT** is concerned, the series purse is not assigned.

rankings, but, at the same time, (ii) do not yield antithetic solver rankings. Thus, homogeneity is not an effectiveness measure per se, but it is a preliminary assessment that we are performing an apple-to-apple comparison and that the apples are not exactly the same.

Homogeneity is computed as in [17] considering the Kendall rank correlation coefficient τ which is a nonparametric coefficient best suited to compare rankings. τ is computed between any two rankings and it is such that $-1 \leq \tau \leq 1$, where $\tau = -1$ means perfect disagreement, $\tau = 0$ means independence, and $\tau = 1$ means perfect agreement. Table 1 shows the values of τ computed for the aggregation procedures considered, arranged in a symmetric matrix where we omit the elements below the diagonal (r.v. is a shorthand for range voting). Values of τ close to, but not exactly equal to 1 are desirable. Table 1 shows that this is indeed the case for the aggregation procedures considered using QBFEVAL'05 data. Only two couples of methods (QBF-CASC and Schulze-Borda) show perfect agreement, while all the other couples agree to some extent, but still produce different rankings.

4.2 Fidelity

We introduce this measure to check whether the aggregation procedures under test introduce any distortion with respect to the true merits of the solvers. Our motivation is that we would like to extract some scientific insight from the final ranking of QBFEVAL'06 and not just winners and losers. Of course, we have no way to know the true merits of the QBF solvers: this would be like knowing the true statistic of some population. Therefore, we measure fidelity by feeding each aggregation procedure with "white noise", i.e., several samples of table RUNS having the same structure outlined in Subsection 2.1 and filled with random results. In particular, we assign to RESULT one of SAT/UNSAT, TIME and FAIL values with equal probability, and a value of CPUTIME chosen uniformly at random in the interval [0;1]. Given this artificial setting, we know in advance that the true merit of the competitors is approximately the same. A high-fidelity aggregation procedure is thus one that computes approximately the same scores for each solver, and thus produces a final ranking where scores have a small variance-to-mean ratio.

The results of the fidelity test are presented in Table 2 where each line contains the statistics of a aggregation procedure. The columns show, from left to right, the mean,

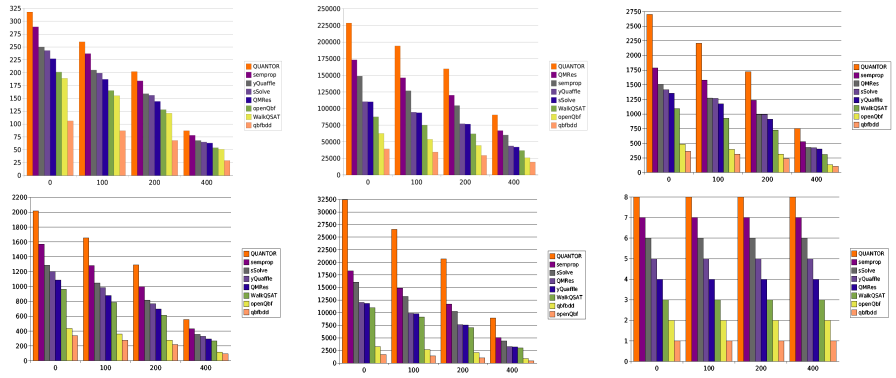

Figure 1: RDT-stability plots.

the standard deviation, the median, the minimum, the maximum and the interquartile range of the scores produced by each aggregation procedure when fed by white noise. The last column is our fidelity coefficient F, i.e., the percent ratio between the lowest score (solver ranked last) and the highest one (solver ranked first): the higher the value of F, the more the fidelity of the aggregation procedure. As we can see from Table 2, the fidelity of YASMv2 is better than that of all the other methods under test, including QBF and CASC which are second best, and have higher fidelity than YASM. Notice that range voting, and to a lesser extent also SAT and Borda's methods, introduce a substantial distortion. In the case of range voting, this can be explained by the exponential spread that separates the scores, and thus amplifies even small differences. Measuring fidelity does not make sense in the case of Schulze's method. Indeed, given the characteristics of the "white noise" data set, Schulze's method yields a tie among all the solvers. Thus, checking for fidelity would essentially mean checking the tie-breaking heuristic, and not the main method.

4.3 RDT-stability and DTL-stability

Stability on a randomized decreasing test set (RDT-stability), and stability on a decreasing time limit (DTL-stability) have been introduced in [17] to measure how much an aggregation procedure is sensitive to perturbations that diminish the size of the original test set, and how much an aggregation procedure is sensitive to perturbations that diminish the maximum amount of CPU time granted to the solvers, respectively. The results of RDT- and DTL-stability tests are presented in the plots of Figures 1 and 2. We obtained such plots considering the CPU time noise model in [18], and considering YASMv2 instead of YASM and the Schulze's method instead of the sum of victories method.

On Figure 1, the first row shows, from left to right, the plots regarding QBF/CASC, SAT and YASMv2 procedures, while the second row shows, again from left to right, the plots regarding Borda's method, range voting and Schulze's method. Each histogram reports, on the x-axis the number of problems m discarded from the origi-

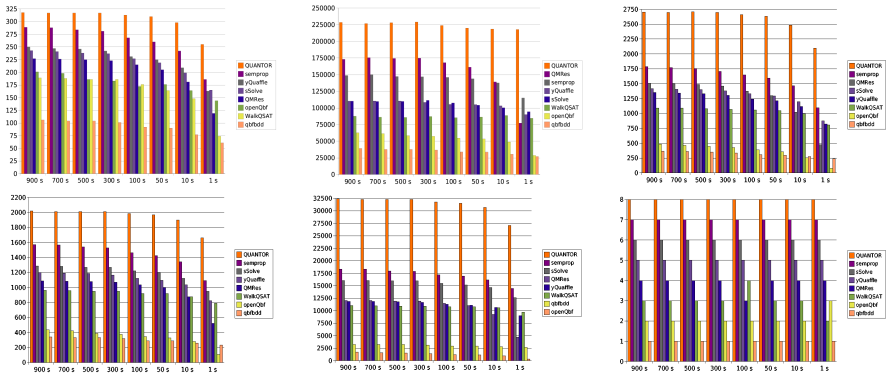

Figure 2: DTL-stability plots.

nal test set (0, 100, 200 and 400 out of 551) and on the y-axis the score. Schulze's scores are the straightforward translation of the ordinal ranking derived by applying the method which is not based on cardinal ranking. For each value of the x-axis, eight bars are displayed, corresponding to the scores of the solvers. The legend is sorted according to the ranking computed by the specific procedure, and the bars are also displayed accordingly. This makes easier to identify perturbations of the original ranking, i.e., the leftmost group of bars in each plot corresponding to $m = 0$. On Figure 2, the histograms are arranged in the same way as Figure 1, except that the x-axis now reports the amount of CPU time seconds used as a time limit when evaluating the scores of the solvers. The leftmost value is $L = 900$, i.e., the original time limit that produces the ranking according to which the legend and the bars are sorted, and then we consider the values $L' = \{700, 500, 300, 100, 50, 10, 1\}$.
The conclusion that we reach are the same of [17], and precisely:

- All the aggregation procedures considered are RDT-stable up to 400, i.e., a random sample of 151 instances is sufficient for all the procedures to reach the same conclusions that each one reaches on the heftier set of 551 instances used in QBFEVAL'05.

- Decreasing the time limit substantially, even up to one order of magnitude, is not influencing the stability of the aggregation procedures considered, except for some minor perturbations for QBF/CASC, SAT and Schulze's methods. Moreover, independently from the procedure used and the amount of CPU time granted, the best solver is always the same.

Indeed, while the above measures can help us extract general guidelines about running a competition, in our setting they do not provide useful insights to discriminate the relative merits of the procedures.

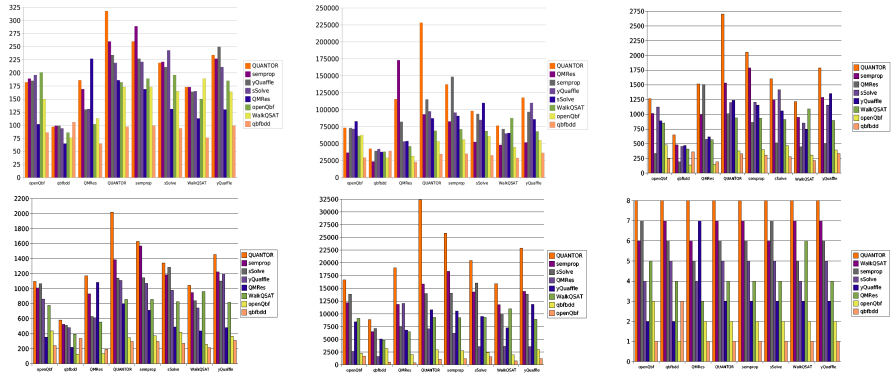

Figure 3: SBT-stability plots.

4.4 SBT-stability

Stability on a solver biased test set (SBT-stability) is introduced in [17] to measure how much an aggregation procedure is sensitive to a test set that is biased in favor of a given solver. Let Γ be the original test set, and Γ_s be the subset of Γ such that the solver s is able to solve exactly the instances in Γ_s. Let $R_{q,s}$ be the ranking obtained by applying the aggregation procedure q on Γ_s. If $R_{q,s}$ is the same as the original ranking R_q, then the aggregation procedure q is SBT-stable with respect to the solver s. Notice that, contrarily to what stated in [17], SBT-stability alone is not a sufficient indicator of the capacity of an aggregation procedure to detect the absolute merit of the participants. Indeed, it turns out that a very low-fidelity method such as range voting is remarkably SBT-stable. This because we can raise the SBT-stability of a ranking by decreasing its fidelity: in the limit, a aggregation procedure that assigns fixed scores to each solver, has the best SBT-stability and the worst fidelity. Therefore, an aggregation procedure showing a high SBT-stability is relatively immune to bias in the test set, but it must also feature a high fidelity if we are to conclude that the method provides a good hint at detecting the absolute merit of the solvers.

Figure 3 shows the plots with the results of the SBT-stability measure for each aggregation procedure considering the noise model and YASMv2 (the layout is the same as Figures 1 and 2). The x-axis reports the name of the solver s used to compute the solver-biased test set Γ_s and the y-axis reports the score value. For each of the Γ_s's, we report eight bars showing the scores obtained by the solvers using only the instances in Γ_s. The order of the bars (and of the legend) corresponds to the ranking obtained with the given aggregation procedure on the original test set Γ. As we can see from Figure 3 (top-left), CASC/QBF aggregation procedures are not SBT-stable: for each of the Γ_s, the original ranking is perturbed and the winner becomes s. Notice that on Γ_{QUANTOR}, CASC/QBF yield the same ranking that they output on the complete test set Γ. The SAT competition procedure (Figure 3, top-center) is not SBT-stable, not even on the test set biased on its alleged winner QUANTOR. YASMv2 is better than both CASC/QBF and SAT, since its alleged winner QUANTOR is the winner on

	CASC/QBF	SAT	YASM	YASMv2	Borda	r. v.	Schulze
OPENQBF	0.43	0.57	0.36	0.64	0.79	0.79	0.79
QBFBDD	0.43	0.43	0.36	0.64	0.79	0.86	0.79
QMRES	0.64	0.86	0.76	0.79	0.71	0.86	0.79
QUANTOR	1	0.86	0.86	0.86	0.93	0.86	0.93
SEMPROP	0.93	0.71	0.71	0.79	0.93	0.86	0.93
SSOLVE	0.71	0.57	0.57	0.79	0.86	0.79	0.86
WALKQSAT	0.57	0.57	0.43	0.71	0.64	0.79	0.79
YQUAFFLE	0.71	0.64	0.57	0.71	0.86	0.86	0.93
Mean	0.68	0.65	0.58	0.74	0.81	0.83	0.85

Table 3: Kendall coefficient between the ranking obtained on the original test set and each of the rankings obtained on the solver-biased test sets.

biased test sets as well. Borda's method (Figure 3, bottom-left) is not SBT-stable with respect to any solver, but the alleged winner (QUANTOR) is always the winner on the biased test sets. Moreover, the rankings obtained on the test sets biased on QUANTOR and SEMPROP are not far from the ranking obtained on the original test set. Also range voting (Figure 3, bottom-center), is not SBT-stable with respect to any solver, but the solvers ranking first and last do not change over the biased test sets and it is true for the Schulze's method (Figure 3, bottom-right) too.

Looking at the results presented above, we can see that YASMv2 performance in terms of SBT stability lies in between classical automated reasoning contests methods and methods based on voting systems. This fact is highlighted in Table 3, where for each procedure we compute the Kendall coefficient between the ranking obtained on the original test set Γ and each of the rankings obtained on the Γ_s test sets, including the mean coefficient observed. Overall, YASMv2 turns out to be, on average, better than CASC/QBF, SAT, and YASM, while it is worse, on average, than the methods based on voting systems. However, if we consider also the results of Table 2 about fidelity, we can see that YASMv2 offers the best compromise between SBT-stability and fidelity. Indeed, while CASC/QBF methods have a relatively high fidelity, they perform poorly in terms of SBT-stability, and SAT method is worse than YASMv2 both in terms of fidelity and in terms of SBT-stability. Methods based on voting systems are all more SBT-stable that YASMv2, but they have poor fidelity coefficients. We consider this good performance of YASMv2 a result of our choice to hybridize classical methods used in automated reasoning contests and methods based on voting systems. This helped us to obtain an aggregation procedure which is less sensitive to bias, and, at the same time, a good indicator of the absolute merit of the competitors.

4.5 SOTA-relevance

This measure was introduced in [17] to understand the relationship between the ranking obtained with an aggregation procedure and the strength of a solver, as witnessed by its contribution to the SOTA solver. As mentioned in Section 3, the SOTA solver is the ideal solver that always fares the best time among all the participants. Indeed, a participant contributes to the SOTA solver whenever it is the fastest solver on some instance. In [17] SOTA-relevance was obtained by counting the number of such events for any given solver, and then computing the Kendall coefficient between the ranking

	SOTA-distance
CASC	1
QBF	1
SAT	0.71
YASM	0.86
YASM v2	0.79
Borda	0.86
range voting	0.71
Schulze	0.86

Table 4: SOTA-relevance.

thereby induced and the ranking obtained with any given procedure. However, it turns out that evaluating the SOTA-contribution of each solver by simply counting the number of times that it is faster than other solvers can be misleading. To understand this, consider the following example. Suppose that a solver A solves 50% of the test set using time *at most* t_A and times out on the rest, and that solver B, on the contrary, solves all the problems where A times out using time *at most* t_B but it does time out on the problems that A solves. Finally, suppose that a solver C is able to solve all the problems in the test set using time *at least* t_C where $t_C > t_A$ and $t_C > t_B$. Given our definition of SOTA solver, it turns out that C is never contributing to it. Evaluating the SOTA contribution using a simple count as described in [17] would induce a ranking where C is last. However, C is, on average, better than both A and B and this will probably be correctly spotted by high-fidelity methods, which would turn out to have a very low SOTA-relevance.

In order to overcome the above problem we redefine here SOTA-relevance in terms of SOTA-distance. SOTA-distance is the distance metric obtained by computing the Euclidean norm between the CPU times of any given solver and the SOTA solver. The resulting values of the metrics induce a ranking that can be used to compute the Kendall coefficient yielding the SOTA-relevance. Table 4 shows the values of the coefficients thereby obtained for each procedure. Notice that according to our new definition of SOTA-relevance, CASC/QBF methods turn out to have the highest such relevance possible, i.e., $\tau = 1$. Therefore the other coefficients correspond to the first row of Table 1 about homogeneity results. Notice that YASMv2 has a better SOTA relevance than SAT and range voting, but worse than all the other methods, including YASM. Given the positive results of YASMv2 insofar fidelity and SBT-stability are concerned, we consider this result as a matter for further investigation either in the quality of YASMv2, or in the explanatory power of the SOTA-distance metric.

5 Conclusions

Summing up, the analysis presented in this paper allowed us to make some progress in the research agenda associated to QBFEVAL. Indeed, in [17] improving YASM was cited as one of the future directions, and in this paper we have presented YASMv2, which features a simpler calculation, yet it is more powerful than YASM in terms of SBT-stability and fidelity. Our empirical evaluation tools of aggregation procedures have also improved with the addition of the fidelity measure and the improved def-

inition of SOTA-relevance. We confirmed some of the conclusions reached in [17], namely that independently of the specific procedure used, a larger test set is not necessarily a better test set, and that a higher time limit does not necessarily result in a more informative contest. On the other hand, while aggregation procedures based on voting systems emerged from [17] as "moral" winners over other procedures, the analysis presented in this paper shows that better results could be achieved using hybrid techniques such as YASMv2.

References

[1] G. Sutcliffe and C. Suttner. The CADE ATP System Competition. `http://www.cs.miami.edu/~tptp/CASC` [2006-6-2].

[2] D. Le Berre and L. Simon. The SAT Competition. `http://www.satcompetition.org` [2006-6-2].

[3] D. Long and M. Fox. The 3rd International Planning Competition: Results and Analysis. *Artificial Intelligence Research*, 20:1–59, 2003.

[4] M.R.C. van Dongen. Introduction to the Solver Competition. In *CPAI 2005 proceedings*, 2005.

[5] C. W. Barrett, L. de Moura, and A. Stump. SMT-COMP: Satisfiability Modulo Theories Competition. In *CAV*, volume 3576 of *Lecture Notes in Computer Science*, pages 20–23, 2005.

[6] D. Le Berre, L. Simon, and A. Tacchella. Challenges in the QBF arena: the SAT'03 evaluation of QBF solvers. In *Sixth International Conference on Theory and Applications of Satisfiability Testing (SAT 2003)*, volume 2919 of *Lecture Notes in Computer Science*. Springer Verlag, 2003.

[7] D. Le Berre, M. Narizzano, L. Simon, and A. Tacchella. The second QBF solvers evaluation. In *Seventh International Conference on Theory and Applications of Satisfiability Testing (SAT 2004)*, Lecture Notes in Computer Science. Springer Verlag, 2004.

[8] M. Narizzano, L. Pulina, and A. Tacchella. The third QBF solvers comparative evaluation. *Journal on Satisfiability, Boolean Modeling and Computation*, 2:145–164, 2006. Available on-line at `http://jsat.ewi.tudelft.nl/`.

[9] J. N. Hooker. Testing Heuristics: We Have It All Wrong. *Journal of Heuristics*, 1:33–42, 1996.

[10] D. G. Saari. *Chaotic Elections! A Mathematician Looks at Voting*. American Mathematical Society, 2001.

[11] W. D. Smith. Range voting. Available on-line at`http://www.math.temple.edu/~wds/homepage/rangevote.pdf` [2006-9-29].

[12] M. Schulze. A New Monotonic and Clone-Independent Single-Winner Election Method. *Voting Matters*, pages 9–19, 2003.

[13] M. Narizzano, L. Pulina, and A. Taccchella. QBF solvers competitive evaluation (QBFEVAL). `http://www.qbflib.org/qbfeval`.

[14] RoboCup. `http://www.robocup.org`.

[15] K. J. Arrow, A. K. Sen, and K. Suzumura, editors. *Handbook of Social Choice and Welfare*, volume 1. Elsevier, 2002.

[16] V. Conitzer and T. Sandholm. Common Voting Rules as Maximum Likelihood Estimators. In *6th ACM Conference on Electronic Commerce (EC-05)*, Lecture Notes in Computer Science, pages 78–87, 2005.

[17] L. Pulina. Empirical Evaluation of Scoring Methods. In *Proc. STAIRS 2006*, volume 142 of *Frontiers in Artificial Intelligence and Applications*, pages 108–119, 2006.

[18] M. Narizzano, L. Pulina, and A. Tacchella. Competitive Evaluation of QBF Solvers: noisy data and scoring methods. Technical report, STAR-Lab - University of Genoa, May 2006.

[19] E. Giunchiglia, M. Narizzano, and A. Tacchella. Quantified Boolean Formulas satisfiability library (QBFLIB), 2001. `www.qbflib.org`.

[20] A. Van Gelder, D. Le Berre, A. Biere, O. Kullmann, and L. Simon. Purse-Based Scoring for Comparison of Exponential-Time Programs, 2006. Unpublished draft.

[21] M. Narizzano, L. Pulina, and A. Tacchella. Competitive Evaluation of Automated Reasoning Tools: Statistical Testing and Empirical Scoring. In *First Workshop on Empirical Methods for the Analysis of Algorithms (EMAA 2006)*, 2006.

[22] M. Schulze. Extending schulze's method to obtain an overall ranking. Personal communications.

Massimo Narizzano
DIST, Università di Genova
Viale Causa, 13 – 16145 Genova, Italy
Email: `mox@dist.unige.it`

Luca Pulina
DIST, Università di Genova
Viale Causa, 13 – 16145 Genova, Italy
Email: `pulina@dist.unige.it`

Armando Tacchella
DIST, Università di Genova
Viale Causa, 13 – 16145 Genova, Italy
Email: `tac@dist.unige.it`