Which system should I buy?
A case study about the QBF solvers competition

Cristiano Ghersi, Luca Pulina, Armando Tacchella

Machine Intelligence for the Diagnosis of Complex Systems

Systems and Technologies for Automated Reasoning

DIST - University of Genoa

Why running a competition is a such a (big) deal?

Why running a competition is a such a (big) deal?

- Seemingly tiny problems which will indeed drive you crazy

Why running a competition is a such a (big) deal?

- Seemingly tiny problems which will indeed drive you crazy

- Input/Output formats
- Choosing the problem instances
- Running the systems
- Interacting with the developers
- Reporting the results
- ...

Why running a competition is a such a (big) deal?

- Seemingly tiny problems which will indeed drive you crazy
    - Input/Output formats
    - Choosing the problem instances
    - Running the systems
    - Interacting with the developers
    - Reporting the results
    - ...
- Not exactly your favorite experimental setup either

Why running a competition is a such a (big) deal?

- Seemingly tiny problems which will indeed drive you crazy
  - Input/Output formats
  - Choosing the problem instances
  - Running the systems
  - Interacting with the developers
  - Reporting the results
  - ...
- Not exactly your favorite experimental setup either
  - Proper experimental design is not that easy
  - It is systems you are comparing, not algorithms

Why running a competition is a such a (big) deal?

- Seemingly tiny problems which will indeed drive you crazy

  - Input/Output formats
  - Choosing the problem instances
  - Running the systems
  - Interacting with the developers
  - Reporting the results
  - ...

- Not exactly your favorite experimental setup either

  - Proper experimental design is not that easy
  - It is systems you are comparing, not algorithms
  - The runtime distributions of the underlying algorithms are unknown, or if they are known, they are probably ill-behaved

What this presentation is NOT about

Everything you need to know before running a competition...

What this presentation is NOT about

Everything you need to know before running a competition...
... otherwise you will not run any for scheduling systems!

What this presentation is about

Which system should I buy?*

Even if a systems competition is (mostly) an ill-posed experiment, we would like to

- rank the systems to reflect their true relative merit, and
- know how much confidence we can have in the results

(*) D. Long and M. Fox. The 3rd International Planning Competition: Results and Analysis. *Journal of Artificial Intelligence Research* – 20(2003).

Our contributions (still ongoing work)

- Research about ranking and reputation (RaRe) systems
  - investigating different aggregation procedures
  - using statistical testing to validate the results
- An in-depth account of QBFEVAL'05 results using both aggregation procedures and statistical testing

Outline

1. The case study
   - QBFEVAL'05 dataset
   - Working hypotheses
2. RaRe systems
   - State-of-the-art
   - Yet another scoring method (YASM)
   - Comparing aggregation procedures
3. Statistical testing
   - Modelling QBFEVAL'05
   - Experimental results

What is a quantified Boolean Formula?

Consider a Boolean formula, e.g.,

$$(x_1 \vee x_2) \wedge (\neg x_1 \vee x_2)$$

Adding existential "$\exists$" and universal "$\forall$" quantifiers, e.g.,

$$\forall x_1 \exists x_2 (x_1 \vee x_2) \wedge (\neg x_1 \vee x_2)$$

yields a quantified Boolean formula (QBF).

What is the meaning of a QBF?

A QBF, e.g.,

$$\forall x_1 \exists x_2 (x_1 \vee x_2) \wedge (\neg x_1 \vee x_2)$$

is true if and only if

for every value of x_1 there exist a value of x_2 such that $(x_1 \vee x_2) \wedge (\neg x_1 \vee x_2)$ is propositionally satisfiable

Given any QBF ψ:

- if $\psi = \forall x \varphi$ then ψ is true iff $\varphi_{|x=0} \wedge \varphi_{|x=1}$ is true
- if $\psi = \exists x \varphi$ then ψ is true iff $\varphi_{|x=0} \vee \varphi_{|x=1}$ is true

Some details about QBFEVAL'05

- 8 solvers on 551 instances

Some details about QBFEVAL'05

- 8 solvers on 551 instances
- Resource constraints
  - time limit: 900s (15 minutes)
  - memory limit: 900MB

Some details about QBFEVAL'05

- 8 solvers on 551 instances
- Resource constraints
  - time limit: 900s (15 minutes)
  - memory limit: 900MB
- The dataset has 4408 entries with four attributes
  - SOLVER, the name of the solver
  - INSTANCE, the name of the instance
  - RESULT, one of {SAT, UNSAT, TIME, FAIL}
  - CPUTIME, the amount of CPU time consumed

Some details about QBFEVAL'05

- 8 solvers on 551 instances
- Resource constraints
  - time limit: 900s (15 minutes)
  - memory limit: 900MB
- The dataset has 4408 entries with four attributes
  - SOLVER, the name of the solver
  - INSTANCE, the name of the instance
  - RESULT, one of {SAT, UNSAT, TIME, FAIL}
  - CPUTIME, the amount of CPU time consumed
- TIME means that the time limit was exceeded
- FAIL is a catchall for any ill behaviour

Factors that we disregarded

- Memory consumption
  - Difficult to define precisely
  - Difficult to measure precisely

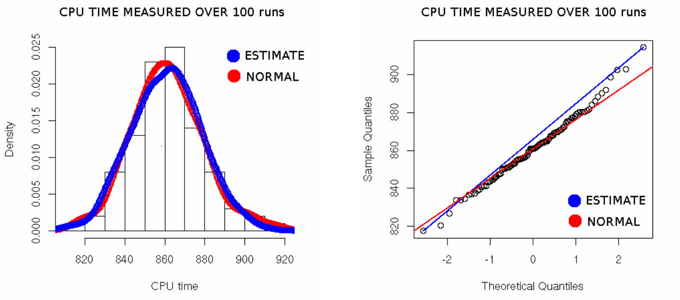
Factors that we disregarded

- Memory consumption
  - Difficult to define precisely
  - Difficult to measure precisely
- Correctness of the solution
  - Solving QBFs is a PSPACE-complete problem
  - The witness is not guaranteed to be compact
  - At the time, none of the solvers output a reliable witness

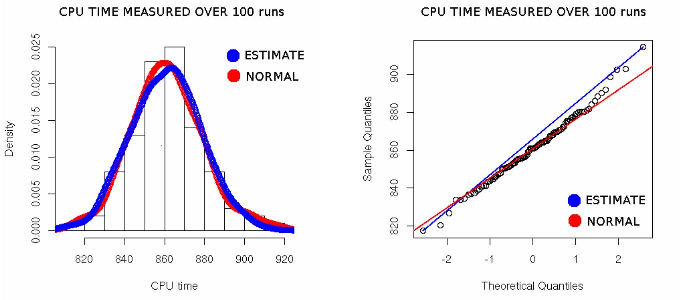
Factors that we disregarded

- Memory consumption
  - Difficult to define precisely
  - Difficult to measure precisely
- Correctness of the solution
  - Solving QBFs is a PSPACE-complete problem
  - The witness is not guaranteed to be compact
  - At the time, none of the solvers output a reliable witness
- Quality of the solution
  - No witness to check for quality
  - Checking could be expensive
- Noise in CPU time measures

What about CPU time?



What about CPU time?



Noise does affects the CPU time measures of systems (statistical methods can deal with this phenomenon)

Aggregation procedures: systems contests

CASC In the CADE ATP systems comparison

- solvers are ranked according to the number of times that RESULT is one of {SAT,UNSAT}, and
- ties are broken using average CPUTIME.

QBFEVAL (before 2006) Same as CASC, but ties are broken using total CPUTIME.

SATCOMP The 2005 SAT competition assigned two purses to each instance

- a solution purse, distributed uniformly, and
- a speed purse, distributed proportionally (w.r.t. speed)

among all the solvers that solve it.
A series purse is distributed to all the solvers that solve at least one instance in a series.

Aggregation procedures: voting systems

Borda count Given n solvers, instance i ranks solver s in position $p_{s,i}$ ($1 \leq p_{s,i} \leq n$). The score of s is $S_{s,i} = n - p_{s,i}$.

Range voting Similar to Borda count, whereas an arbitrary scale is used to associate a weight w_p with each of the n positions.

Schulze's method it is a Condorcet method that computes the Schwartz set to determine a winner. We use an extension of the single overall winner procedure, in order to make it capable of generating an overall ranking.

YASM: the formula

$$\underbrace{S_{s,i}}_{\text{Score}} = \underbrace{k_{s,i}}_{\text{Borda weight}} \cdot \underbrace{(1 + H_i)}_{\text{Instance hardness}} \cdot \underbrace{\frac{\overbrace{L}^{\text{time limit}} - \overbrace{T_{s,i}}^{\text{cputime of } s \text{ on instance } i}}{L - M_i}}_{\text{Solver speed}}$$

$$H_i = 1 - \frac{\text{\# solvers that solved } i}{\text{\# solvers that didn't solve } i} \qquad M_i = \min_s \{T_{s,i}\}$$

YASM: rationale

What makes for a good solver?

The ability to solve:

- many instances within the time limit ($L - T_{s,i}$)
- preferably hard ones ($1 + H_i$)
- in a relatively short time ($\frac{L-T_{s,i}}{L-M_i}$)

Why the Borda weight $k_{s,i}$?

It helps to stabilize YASM against bias in the test set!

Measures to compare scoring methods

Fidelity How much a scoring method reflects the true relative merits of the competitors

Stability with respect to

- decreasing time limit (DTL-stability)
- decreasing test set cardinality (RDT-stability)
- biased test set (SBT-stability)

Homogeneity

- Degree of **(dis)agreement** between different aggregation procedures.

Homogeneity

- Degree of **(dis)agreement** between different aggregation procedures.
- Verify that the aggregation procedures considered
  - do not produce exactly the same solver rankings
  - do not yield antithetic solver rankings

Homogeneity

- Degree of **(dis)agreement** between different aggregation procedures.
- Verify that the aggregation procedures considered
  - do not produce exactly the same solver rankings
  - do not yield antithetic solver rankings
- Kendall rank correlation coefficient τ as measure of homogeneity.

Homogeneity

	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								–

r.v. = range voting

Fidelity

- Given a synthesized set of raw data, evaluates whether an aggregation procedure distorts the results.

Fidelity

- Given a synthesized set of raw data, evaluates whether an aggregation procedure distorts the results.
- Several samples of table RUNS filled with random results:
  - RESULT is assigned to SAT/UNSAT, TIME or FAIL with equal probability
  - a value of CPUTIME is chosen uniformly at random in the interval [0;1]

Fidelity

- Given a synthesized set of raw data, evaluates whether an aggregation procedure distorts the results.
- Several samples of table RUNS filled with random results:
  - RESULT is assigned to SAT/UNSAT, TIME or FAIL with equal probability
  - a value of CPUTIME is chosen uniformly at random in the interval [0;1]
- A high-fidelity aggregation procedure:
  - computes approximately the same scores for each solver
  - produces a final ranking where scores have a small variance-to-mean ratio

Fidelity

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	–	–	–	–	–	–	–

r.v. = range voting

RDT-stability

- Stability on a **Randomized Decreasing Test set** aims to measure how much an aggregation procedure is sensitive to perturbations that diminish the size of the original test set.

RDT-stability

- Stability on a **Randomized Decreasing Test set** aims to measure how much an aggregation procedure is sensitive to perturbations that diminish the size of the original test set.

INSTANCE_1
INSTANCE_2
INSTANCE_3
INSTANCE_4
INSTANCE_5
INSTANCE_6
INSTANCE_7
INSTANCE_8
INSTANCE_9
INSTANCE_10
INSTANCE_11
INSTANCE_12
INSTANCE_13
INSTANCE_14
INSTANCE_15

RDT-stability

- Stability on a **Randomized Decreasing Test set** aims to measure how much an aggregation procedure is sensitive to perturbations that diminish the size of the original test set.

INSTANCE_1
INSTANCE_3
INSTANCE_6
INSTANCE_7
INSTANCE_8
INSTANCE_9
INSTANCE_11
INSTANCE_12
INSTANCE_14
INSTANCE_15

RDT-stability

- Stability on a **Randomized Decreasing Test set** aims to measure how much an aggregation procedure is sensitive to perturbations that diminish the size of the original test set.

INSTANCE_1
INSTANCE_3
INSTANCE_6
INSTANCE_7
INSTANCE_8
INSTANCE_9
INSTANCE_11
INSTANCE_12
INSTANCE_14
INSTANCE_15

→

RDT-stability

- Stability on a **Randomized Decreasing Test set** aims to measure how much an aggregation procedure is sensitive to perturbations that diminish the size of the original test set.

INSTANCE_1

INSTANCE_3

INSTANCE_6
INSTANCE_7
INSTANCE_8 → RANKING_A
INSTANCE_9

INSTANCE_11
INSTANCE_12

INSTANCE_14
INSTANCE_15

RDT-stability

- Stability on a **Randomized Decreasing Test set** aims to measure how much an aggregation procedure is sensitive to perturbations that diminish the size of the original test set.

INSTANCE_1
INSTANCE_2
INSTANCE_3
INSTANCE_4

INSTANCE_6
INSTANCE_7

$\rightarrow$ RANKING_A

INSTANCE_10
INSTANCE_11
INSTANCE_12
INSTANCE_13

RDT-stability

- Stability on a **Randomized Decreasing Test set** aims to measure how much an aggregation procedure is sensitive to perturbations that diminish the size of the original test set.

INSTANCE_1		
INSTANCE_2		
INSTANCE_3		
INSTANCE_4		
INSTANCE_6		
INSTANCE_7		
	→	RANKING_A
		RANKING_B
INSTANCE_10		
INSTANCE_11		
INSTANCE_12		
INSTANCE_13		

RDT-stability

- Stability on a **Randomized Decreasing Test set** aims to measure how much an aggregation procedure is sensitive to perturbations that diminish the size of the original test set.

INSTANCE_2		
INSTANCE_3		
INSTANCE_4		
INSTANCE_5		
INSTANCE_6		
INSTANCE_8	$\longrightarrow$	RANKING_A
INSTANCE_9		RANKING_B
INSTANCE_10		
INSTANCE_14		
INSTANCE_15		

RDT-stability

- Stability on a **Randomized Decreasing Test set** aims to measure how much an aggregation procedure is sensitive to perturbations that diminish the size of the original test set.

INSTANCE_2
INSTANCE_3
INSTANCE_4
INSTANCE_5
INSTANCE_6
INSTANCE_8
INSTANCE_9
INSTANCE_10
INSTANCE_14
INSTANCE_15

$\longrightarrow$

RANKING_A
RANKING_B
RANKING_C

RDT-stability

- Stability on a **Randomized Decreasing Test set** aims to measure how much an aggregation procedure is sensitive to perturbations that diminish the size of the original test set.

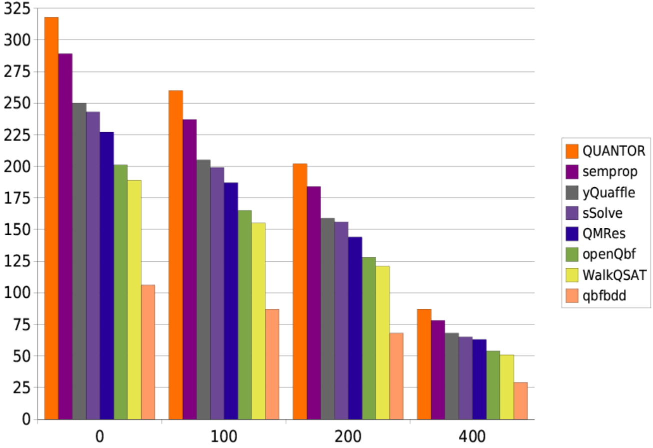
INSTANCE_2			
INSTANCE_3			
INSTANCE_4			
INSTANCE_5			
INSTANCE_6			
		RANKING_A	
INSTANCE_8	→	RANKING_B	→
INSTANCE_9		RANKING_C	
INSTANCE_10			
INSTANCE_14			
INSTANCE_15			

RDT-stability

- Stability on a **Randomized Decreasing Test set** aims to measure how much an aggregation procedure is sensitive to perturbations that diminish the size of the original test set.

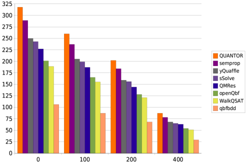
Instances		Rankings		Median
INSTANCE_2				
INSTANCE_3				
INSTANCE_4				
INSTANCE_5				
INSTANCE_6				
		RANKING_A		
INSTANCE_8	→	RANKING_B	→	RANKING_MEDIAN
INSTANCE_9		RANKING_C		
INSTANCE_10				
INSTANCE_14				
INSTANCE_15				

RDT-stability

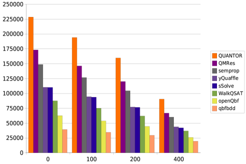


RDT-stability of CASC aggregation procedure

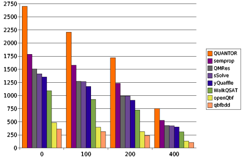
RDT-stability



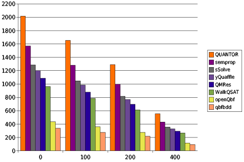
CASC



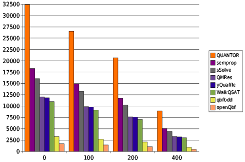
SAT



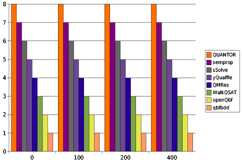
YASMv2



Borda



r.v.



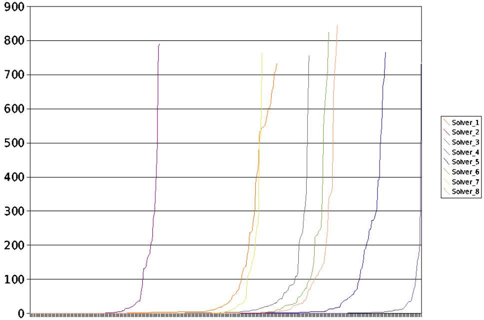
Schulze

DTL-stability

- Stability on a **Decreasing Time Limit** aims to measure how much an aggregation procedure is sensitive to perturbations that diminish the maximum amount of CPU time granted to the solvers.

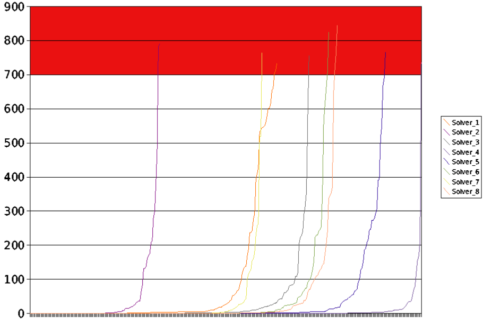
DTL-stability

- Stability on a **Decreasing Time Limit** aims to measure how much an aggregation procedure is sensitive to perturbations that diminish the maximum amount of CPU time granted to the solvers.



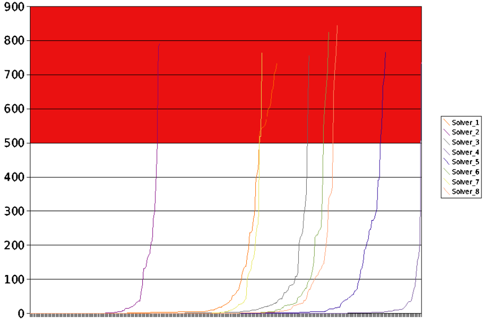
DTL-stability

- Stability on a **Decreasing Time Limit** aims to measure how much an aggregation procedure is sensitive to perturbations that diminish the maximum amount of CPU time granted to the solvers.



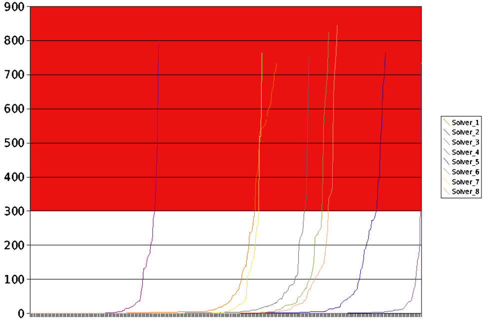
DTL-stability

- Stability on a **Decreasing Time Limit** aims to measure how much an aggregation procedure is sensitive to perturbations that diminish the maximum amount of CPU time granted to the solvers.

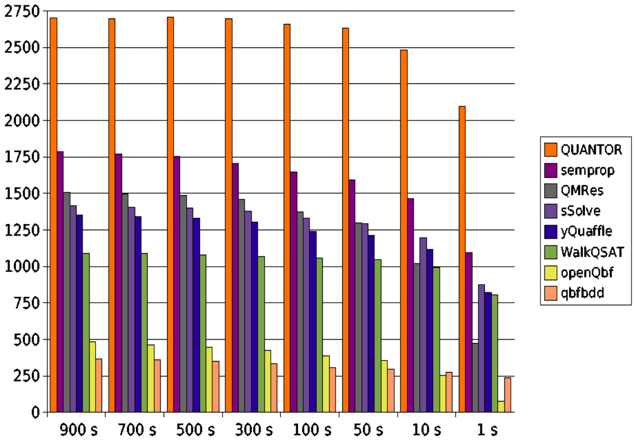


DTL-stability

- Stability on a **Decreasing Time Limit** aims to measure how much an aggregation procedure is sensitive to perturbations that diminish the maximum amount of CPU time granted to the solvers.

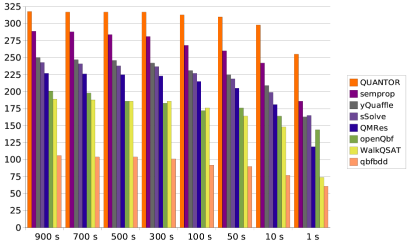


DTL-stability

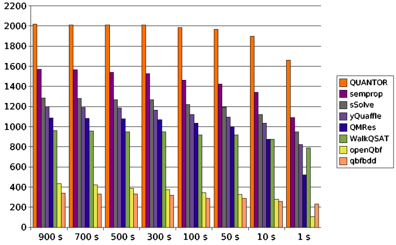


YASMv2

DTL-stability



CASC



Borda

SBT-stability

- Stability on a **Solver Biased Test set** aims to measure how much an aggregation procedure is sensitive to a test set that is biased in favor of a given solver.

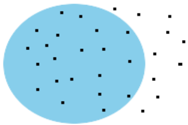
SBT-stability

- Stability on a **Solver Biased Test set** aims to measure how much an aggregation procedure is sensitive to a test set that is biased in favor of a given solver.

- Test set instances
- Solved by SOLVER_1
- Solved by SOLVER_2
- Solved by SOLVER_3

SBT-stability

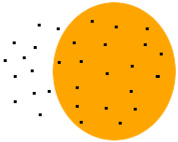
- Stability on a **Solver Biased Test set** aims to measure how much an aggregation procedure is sensitive to a test set that is biased in favor of a given solver.



- Test set instances
- Solved by SOLVER_1
- Solved by SOLVER_2
- Solved by SOLVER_3

SBT-stability

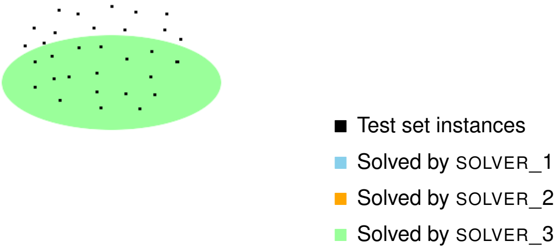
- Stability on a **Solver Biased Test set** aims to measure how much an aggregation procedure is sensitive to a test set that is biased in favor of a given solver.



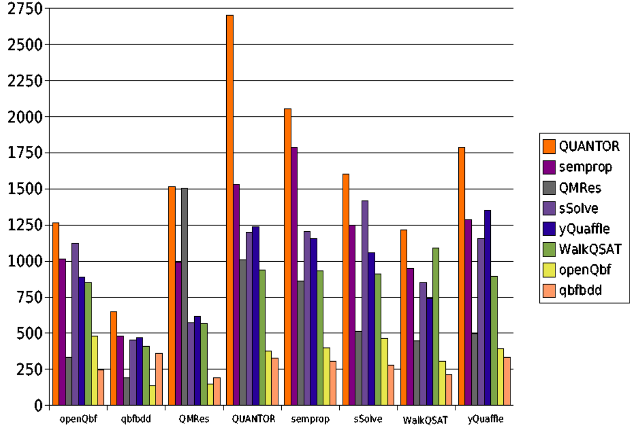
- Test set instances
- Solved by SOLVER_1
- Solved by SOLVER_2
- Solved by SOLVER_3

SBT-stability

- Stability on a **Solver Biased Test set** aims to measure how much an aggregation procedure is sensitive to a test set that is biased in favor of a given solver.



SBT-stability



YASMv2

SBT-stability

	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

Kendall τ between rankings on biased test sets (rows) vs. the original one (columns)

Null and alternative hypotheses

- We are interested in statistically significant differences in the (average) performances of the solvers

Null and alternative hypotheses

- We are interested in statistically significant differences in the (average) performances of the solvers
- Given any two solvers A and B we state the

  null hypothesis (H_0), i.e., there are no significant differences in the performances of A with respect to the performances of B; and the

  alternative hypothesis (H_1), i.e., there are significant differences in the performances of A with respect to the performances of B.

Two fundamental issues

Let X_A and X_B be the vectors of run times for solvers A and B

1. How do we consider TIME and FAIL values in X_A and X_B?
2. Which assumptions, if any, can be made about the underlying distributions of X_A and X_B?

Data models

FAT (Failure as time limit) FAIL is replaced by TIME

- Consistently overestimates the performances of the solvers, but
- it allows the paired comparison of the values in X_A and in X_B.

Data models

FAT (Failure as time limit) FAIL is replaced by TIME

- Consistently overestimates the performances of the solvers, but
- it allows the paired comparison of the values in X_A and in X_B.

TAF (Time limit as failure) TIME is replaced by FAIL and both are considered “missing values”

- Overestimation does not occur, but
- X_A and X_B may not be equal in length, so their paired comparison is not generally possible.

Parametric or distribution-free?

For each solver A

- we check X_A under FAT and TAF models using
- the Shapiro-Wilk test of the null hypothesis that the samples come from a normally distributed population.

Parametric or distribution-free?

For each solver A

- we check X_A under FAT and TAF models using
- the Shapiro-Wilk test of the null hypothesis that the samples come from a normally distributed population.

$\mathbf{X_A}$	**FAT**	**TAF**
OPENQBF	9.665×10^{27}	2.036×10^{24}
QBFBDD	2.768×10^{30}	7.051×10^{19}
QMRES	1.419×10^{27}	1.588×10^{28}
QUANTOR	8.334×10^{32}	6.926×10^{36}
SEMPROP	5.012×10^{29}	2.359×10^{31}
SSOLVE	9.513×10^{28}	1.359×10^{29}
WALKQSAT	1.148×10^{27}	6.414×10^{27}
YQUAFFLE	6.753×10^{28}	5.453×10^{30}

(Values: Shapiro-Wilk test p-values)

Parametric or distribution-free?

For each solver A

- we check X_A under FAT and TAF models using
- the Shapiro-Wilk test of the null hypothesis that the samples come from a normally distributed population.

X_A	FAT	TAF
OPENQBF	9.665×10^{27}	2.036×10^{24}
QBFBDD	2.768×10^{30}	7.051×10^{19}
QMRES	1.419×10^{27}	1.588×10^{28}
QUANTOR	8.334×10^{32}	6.926×10^{36}
SEMPROP	5.012×10^{29}	2.359×10^{31}
SSOLVE	9.513×10^{28}	1.359×10^{29}
WALKQSAT	1.148×10^{27}	6.414×10^{27}
YQUAFFLE	6.753×10^{28}	5.453×10^{30}

(Values: Shapiro-Wilk test p-values)

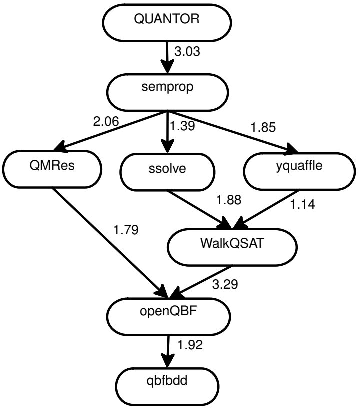
It is highly unlikely that the X_A's are normally distributed!

Wilcoxon signed rank (WSR) test

- A distribution-free alternative to correlated-samples **t**-test
- H_0 is that X_A and X_B do not differ significantly (on average)
- Its basic assumptions are
  - that the paired values of X_A and X_B are randomly and independently drawn;
  - that the dependent variable is intrinsically continuous; and
  - that the measures of X_A and X_B have the properties of at least an ordinal scale of measurement.

WSR test is ok with the FAT model, but not with the TAF one!

QBFEVAL'05 dataset and the WSR test



- Nodes correspond to solvers
- An edge from A to B means

$$\frac{\#\text{ of times } (X_A - X_B) > 0}{\#\text{ of times } (X_B - X_A) > 0} > 1$$

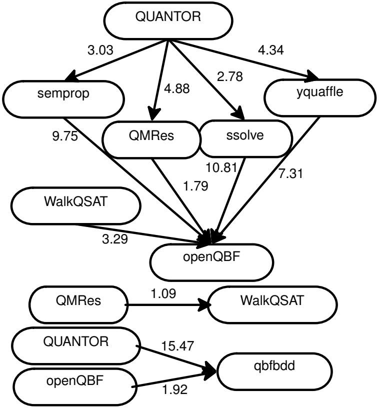
- A path between A and B means that WSR rejects H_0
  - Confidence level: 99%
  - Control: family-wise error rate

Mann-Whitney-Wilcoxon (MWW) test

- A distribution-free alternative to independent-samples **t**-test
- H_0 is that X_A and X_B do not differ substantially
- Its basic assumptions are
  - that X_A and X_B are randomly and independently drawn;
  - that the dependent variable is intrinsically continuous; and
  - that the measures of X_A and X_B have the properties of at least an ordinal scale of measurement.

MWW test is ok with the TAF model, and it gives an approximate, although conservative, picture.

QBFEVAL'05 dataset and the MWW test



- Nodes correspond to solvers
- An edge from A to B means

$$\frac{\#\text{ of times } (X_A - X_B) > 0}{\#\text{ of times } (X_B - X_A) > 0} > 1$$

under the FAT model.

- A path between A and B means that MWW rejects H_0
  - Confidence level: 99%
  - Control: family-wise error rate

under the TAF model.

Scoring methods, WSR and MWW (1/2)

- All the scoring methods produce rankings mostly compatible with WSR and MWW although
  - SAT conflicts with WSR on QMRES vs. SEMPROP, but
  - MWW finds the two incomparable.

Scoring methods, WSR and MWW (1/2)

- All the scoring methods produce rankings mostly compatible with WSR and MWW although
  - SAT conflicts with WSR on QMRES vs. SEMPROP, but
  - MWW finds the two incomparable.
- QMRES, SSOLVE and YQUAFFLE are
  - incomparable according to WSR, and
  - the solvers on which the rankings mostly differ.

Scoring methods, WSR and MWW (1/2)

- All the scoring methods produce rankings mostly compatible with WSR and MWW although
  - SAT conflicts with WSR on QMRES vs. SEMPROP, but
  - MWW finds the two incomparable.
- QMRES, SSOLVE and YQUAFFLE are
  - incomparable according to WSR, and
  - the solvers on which the rankings mostly differ.
- MWW finds also
  - SEMPROP to be incomparable w.r.t. QMRES, SSOLVE and YQUAFFLE, but
  - all the methods, except SAT, rank SEMPROP second best.

Scoring methods, WSR and MWW (2/2)

WSR and MWW rankings obtained by

- considering the DAGs induced by the two tests, and
- breaking ties in reverse order of edge labels.

	Borda	MWW	QBF/CASC	r.v.	SAT	Schulze	WSR
MWW	0.93	-	-	-	-	-	-
QBF/CASC	0.84	0.76	-	-	-	-	-
r.v.	0.86	0.79	0.69	-	-	-	-
SAT	0.71	0.64	0.69	0.71	-	-	-
Schulze	1.00	0.93	0.84	0.86	0.71	-	-
WSR	1.00	0.93	0.84	0.86	0.71	1.00	-
YASM	0.86	0.79	0.69	0.86	0.86	0.86	0.86

(Values: Kendall's τ between rankings)

Summing up

Lessons learned

- Empirical scoring can borrow a lot from voting theory and benefit from statistical testing
- Elaborate scoring methods are not necessarily better than simple ones
- Statistical testing provides insightful cross-validation of the empirical scoring results

Possible extensions

- Is there a better YASM than YASM?
- Are there other useful statistical techniques?

Measures to compare scoring methods

Fidelity How much a scoring method reflects the true relative merits of the competitors

Measures to compare scoring methods

Fidelity How much a scoring method reflects the true relative merits of the competitors

Stability with respect to

- decreasing time limit (DTL-stability)
- decreasing test set cardinality (RDT-stability)
- biased test set (SBT-stability)

Measures to compare scoring methods

Fidelity How much a scoring method reflects the true relative merits of the competitors

Stability with respect to

- decreasing time limit (DTL-stability)
- decreasing test set cardinality (RDT-stability)
- biased test set (SBT-stability)

SOTA distance Considering $M_i = \min_s\{T_{s,i}\}$ and given m instances, the distance of solver s from the state-of-the-art (SOTA) solver is

$$d_s = \sqrt{\sum_{i=1}^{m}(T_{s,i} - M_i)^2}$$

Fidelity ☺

Feed each scoring method with "white noise"

- RESULT equally likely to be either SAT, UNSAT, TIME, or FAIL
- CPUTIME distributed uniformly in [0,1]
- generate several sample datasets accordingly

Method	Median	Min	Max	IQ Range	F
QBF	183.00	170.00	192.00	13.00	88.54
CASC	183.00	170.00	192.00	13.00	88.54
SAT	83262.33	78532.74	119780.48	4263.94	65.56
YASM	1268.73	1198.43	1312.72	95.11	91.29
Borda	982.50	752.00	1176.00	194.50	63.95
r.v.	12104.00	5186.00	21504.00	8096.00	24.12

(Values: scoring statistics over 100 random datasets)
The fidelity index F is Min/Max$\times$100

SBT-Stability ☺

Given a scoring method

- obtain the ranking R using the entire dataset,
- consider the ranking R_s obtained by removing from the dataset all the instances that are not solved by s, and
- compare R and R_s using Kendall's τ.

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

SOTA distance ☹

Given a scoring method

- obtain the ranking *R* using the entire dataset,
- consider the ranking *S* induced by the SOTA-distance, and
- compare *R* and *S* using Kendall's τ.

	SOTA-distance
CASC	1.00
QBF	1.00
SAT	0.71
YASM	0.79
Borda	0.86
r.v.	0.71
Schulze	0.86