Impact of censored sampling on the performance of restart strategies

Matteo Gagliolo[1,2] and Jürgen Schmidhuber[1,3]

[1] IDSIA, Galleria 2, 6928 Manno (Lugano), Switzerland
[2] University of Lugano, Faculty of Informatics,
Via Buffi 13, 6904 Lugano, Switzerland
[3] TU Munich, Boltzmannstr. 3,
85748 Garching, München, Germany

Abstract. Algorithm selection, algorithm portfolios, and randomized restarts, can profit from a probabilistic model of algorithm run-time, to be estimated from data gathered by solving a set of experiments. Censored sampling offers a principled way of reducing this initial training time. We study the trade-off between training time and model precision by varying the censoring threshold, and analyzing the consequent impact on the performance of an optimal restart strategy, based on an estimated model of runtime distribution.
We present experiments with a SAT solver on a graph-coloring benchmark. Due to the "heavy-tailed" runtime distribution, a modest censoring can already reduce training time by a few orders of magnitudes. The nature of the optimization process underlying the restart strategy renders its performance surprisingly robust, also to more aggressive censoring.

1 Introduction

The interest in algorithm performance modeling is twofold. An accurate model can provide useful insights for analyzing algorithm behavior [1]; it can as well be used to automate selection [2], or, more generally, allocation of computational resources among different algorithms. In the context of computational performance analysis of solvers for constraint satisfaction, an interest has been recently growing around the existence, in some structured domains, of a second phase transition, in the under-constrained region [3], where, for some problems, the run-time distribution exhibits "heavy tails", i.e., is Pareto for both very large and very small values of time [4]. In this case, the probability mass of the algorithm's runtime distribution can effectively be shifted towards lower time values, by simply running multiple copies of the same algorithm in parallel (algorithm *portfolios* [5]), or by repeatedly restarting a single algorithm, each time with a different initialization or random seed [6], as this allows to avoid the "unlucky" long runs, and profit from the very short ones.

Both strategies can be rendered more efficient if a model of run-time distribution (RTD) is available for the algorithm/problem combination at hand.[4] In the context of

[4] In the following, for the sake of readability, we will often refer to the RTD of a problem instance, meaning the RTD of different runs of the randomized algorithm of interest on that

algorithm portfolios, models can be used to allocate resources to different competing algorithms, or to evaluate the optimal number of components for a homogeneous portfolio [7]. In the case of restarts, it has been shown that the knowledge of the RTD allows to determine an optimal strategy, based on a constant cutoff [6].

What makes these approaches problematic is the huge computation time required, not for training the model, but for gathering the training data itself, as one might need to solve a large number of problems, in order to obtain a reliable sample of run-time values. One way of countering this is offered by *censored sampling*, a technique commonly used for lifetime distribution estimation (see, e.g., [8]), which allows to bound the duration of each training experiment, and still exploit the information conveyed by runs that reach the censoring threshold.

This obviously has a cost, to be paid in terms of the precision of the obtained model. This cost can in principle be measured according to traditional statistical goodness-of-fit tests, but if the sole purpose of the model is to set up a portfolio, or a restart strategy, in order to gain on future performance, then the only quantity of practical interest is the loss in performance induced by the censored sampling.

The main objective of this work is to analyze this trade-off between training time and efficiency, in the case of optimal restarts. To this aim we present experiments with a randomized version [4] of a well known complete SAT solver [9], on a benchmark of graph coloring problems from SATLIB [10], on which heavy-tailed behavior can be observed.

In the following, after some additional references (Sect. 2), censored sampling (Sect. 3) and restart strategies (Sect. 4) are briefly introduced, followed by a description (Sect. 5) and discussion (Sect. 6) of experimental results.

2 Previous work

As an extensive review of the literature on modeling run-time distribution is beyond the scope of this paper, we will limit to a few examples. The behavior of complete SAT solvers on solvable and unsolvable instances near phase transition have been shown to be approximable by Weibull and lognormal distributions respectively [11]. Heavy-tailed behavior is observed for backtracking search on structured underconstrained problems in [3,4], but also in many other problem domains, such as computer networks [12]. Interesting hypothesis on the mechanism behind it are explored in [1]. The performance of local search SAT solvers is analyzed in [13,14], and modeled in [15] using a mixture of exponential distributions.

The literature on algorithm portfolios [5,7], anytime algorithms [16,17], and restart strategies [18,6,19,20,12] provides many examples of the application of performance modeling to resource allocation. In [21] we presented a dynamic approach, in which a conditional model of performance is updated and progressively exploited during training. See also [22] for more references.

---

instance; and the RTD of a problem set, meaning the RTD of different runs of the randomized algorithm of interest, each run on a different instance, uniform randomly picked without replacement from the set.

3 Censored sampling

Consider a "Las-Vegas" algorithm [6] running k times on instances of a family of problems, on which we believe that the algorithm will display a similar behavior. Be $\mathcal{T} = \{t_1, t_2, ..., t_k\}$ the set of outcomes of the experiments. In order to model the probability density function (pdf) of solution time t, we can choose a parametric function $g(t|\theta)$, with parameter θ, and then express the likelihood of $\mathcal{T}$ given θ, as

$$\mathfrak{L}(\mathcal{T}|\theta) = \prod_{i=1}^{k} \mathfrak{L}(t_i|\theta) = \prod_{i=1}^{k} g(t_i|\theta) \tag{1}$$

We can then search the value of θ that maximizes (1), or, in a Bayesian approach, introduce a prior $p(\theta)$ on the parameter, and maximize the posterior $p(\theta|\mathcal{T}) \propto \mathfrak{L}(\mathcal{T}|\theta)p(\theta)$.

Type I censored sampling (see, e.g., [8]) consists in stopping experimental runs that exceed a cutoff time t_c, and replacing the corresponding multiplicative term in (1) with

$$\mathfrak{L}_c(t_c|\theta) = \int_{t_c}^{\infty} g(\tau|\theta)d\tau = [1 - G(t_c|\theta)] \tag{2}$$

where $G(t|\theta) = \int_0^t g(\tau|\theta)d\tau$ is the conditional cumulative distribution function (CDF) corresponding to g. This allows to limit the computational cost of running the experiments, while exploiting the information carried by unsuccessful runs. In *Type II censored sampling*, k experiments are run in parallel, and stopped after a desired number u of uncensored samples is obtained. In this way the fraction of censored samples $c = (k - u)/k$ is set in advance, while the censoring threshold t_c for the remaining $k - u$ runs is determined by the outcome of the experiments, as the ending time of the u-th fastest experiment.

4 Optimal restart strategies

A restart strategy consists in executing a sequence of runs of a randomized algorithm, in order to solve a same problem instance, stopping each run k after a time $T(k)$ if no solution is found, and restarting the algorithm with a different random seed; it can be operationally defined by a function $T : \mathbb{N} \to \mathbb{R}^+$ producing the sequence of thresholds $T(k)$ employed. Luby et. al [6] proved that the optimal restart strategy is *uniform*, i. e., one in which a constant $T(k) = T$ is used to bound each run. They show that, in this case, the expected value of the total run-time t_T, i. e., the sum of runtimes of the successful run, and all previous unsuccessful runs, can be evaluated as

$$E(t_T) = \frac{T - \int_0^T F(\tau)d\tau}{F(T)} \tag{3}$$

where $F(t)$ is the cumulative distribution function (CDF) of the run-time t for an unbounded run of the algorithm, i. e., a function quantifying the probability that the problem is solved before time t. If such distribution is known, an optimal cutoff time T^* can be evaluated minimizing (3). Otherwise, they suggest a universal non-uniform restart

strategy U, with cutoff sequence $\{1,1,2,1,1,2,4,1,1,2,1,1,2,4,8,1,...\}$,[5] proving that its computational performance t_U is, with high probability, within a logarithmic factor worse than the *expected* total run-time $E(t_{T^*})$ of the optimal strategy.

5 Experiments

A model $\hat{F}$ of the RTD on a set of problems can be obtained from censored and uncensored runtime samples; the performance of the corresponding sub-optimal uniform strategy $\hat{T}$, evaluated minimizing (3) with $\hat{F}$ in place of the real F, will depend on the precision of the estimated $\hat{F}$, which will in turn depend on the number of samples used to estimate it, and the amount of censoring. If we fix the number of samples, and vary the fraction of censored samples, we expect to observe a trade-off between the time spent running the training experiments, from whose outcomes $\hat{F}$ is estimated, and the performance of the corresponding $\hat{T}$. It is precisely this trade-off that we intend to analyze here.[6]

In order to do so, we set up a simple learning scheme. Given a set of problems, and a randomized solver s, we first randomly pick a subset of n problems. For each problem, we start r runs of the algorithm s, differing only for the random seed, for a total of $k = nr$ parallel runs. We control the duration of these "training" experiments with Type II censored sampling (see Sect 3), fixing a censoring fraction $c \in [0,1)$ in advance: as the first $\lfloor (1-c)k \rfloor$ runs terminate, we stop also the remaining $\lceil ck \rceil$ runs. The gathered runtime samples are then used to train a model $\hat{F}$ of the RTD, from which a uniform strategy $\hat{T}$ is evaluated, by minimizing (3) numerically. The performance of $\hat{T}$ is then tested on the remaining problems of the set. Varying c, we can measure the corresponding variations in training time, and in the performance of $\hat{T}$ on the test set.

The experiments were conducted using Satz-Rand [4], a version of Satz [9] in which random noise influences the choice of the branching variable. Satz is a modified version of the complete DPLL procedure, in which the choice of the variable on which to branch next follows an heuristic ordering, based on first and second level unit propagation. Satz-Rand differs in that, after the list is formed, the next variable to branch on is randomly picked among the top h fraction of the list. We present results with the heuristic starting from the most constrained variables, as suggested also in [9], and noise parameter set to 0.4.

The benchmark used (obtained from SATLIB [10]) consists of different sets of SAT-encoded "morphed" graph-coloring problems [23] (100 vertices, 400 edges, 5 colors, resulting in 500 variables and 3100 clauses when encoded as a CNF3 SAT problem): each graph is composed of the set of common edges among two random graphs, plus

[5] The sequence is composed of powers of 2: when 2^{j-1} is used twice, 2^j is the next. More precisely, $k = 1,2,...$, $T(k) := 2^{j-1}$ if $k = 2^j - 1$; $T(k) := T(k - 2^{j-1} + 1)$ if $2^{j-1} \leq k < 2^j - 1$

[6] Note that a given problem set might contain instances which follow sensibly different RTDs: in this case, the obtained model would capture the overall behavior of the algorithm on the set, and the corresponding restart strategy would be suboptimal for each single instance. We ignore this issue here, as we are only interested in comparing among different $\hat{F}$, and not with the real F.

fractions $p \in [0, 1]$ and $1 - p$ of the remaining edges for each graph, chosen as to form regular ring lattices. Each of the 9 problem sets contains 100 instances, generated with a logarithmic grid of 9 different values for the parameter p, from 2^0 to 2^{-8}, to which we henceforth refer with labels $0, ..., 8$. This benchmark is particularly interesting, as the parameter p controls the amount of structure in the problem, and the heavy-tailed behavior of Satz-Rand varies accordingly on the different sets.

For each problem set, we repeated the simple scheme described above, running $r = 20$ copies of Satz-Rand on each of $n = 50$ randomly picked training instances, and evaluating $\hat{T}$ on the remaining 50. The process was repeated for 10 different levels of the censored fraction c during training, from $c = 0$ to $c = 0.9$. For practical reasons, experiments with $c = 0$ were actually run with a very high censoring threshold[7] (10^6): only a few runs of Satz-Rand exceeded this value.

As for the model, we tried different alternatives, including Weibull,[8] lognormal, and the novel double and right-hand Pareto lognormal, introduced in [24] to model heavy tailed distributions. The double Pareto-lognormal distribution (DPLN) describes the distribution of the product of two independent random variables, one with lognormal distribution, one with Double Pareto distribution, whose pdf can be written as $\gamma t^{\beta-1}$ for $t < 1$ (left tail), and $\gamma t^{-\alpha-1}$ for $t > 1$, $\gamma = \alpha\beta/(\alpha + \beta)$. The advantage of DPLN is that it can adapt to both lognormal and heavy tailed distributions. We also tested various mixtures of two distributions, with pdf of the form $f(t|\theta) = wf_1(t|\theta_1) + (1 - w)f_2(t|\theta_2)$, with parameter $\theta = (w, \theta_1, \theta_2)$, $w \in [0, 1]$.

The models were trained by maximum likelihood, as described in Sect. 3. To compare with a non-parametric approach, we repeated the experiments using the Kaplan-Meier estimator [25], which can also account for censored samples. Among the parametric models, we obtained the best results with a mixture including one lognormal and either one double-Pareto lognormal, or only the heavy-tailed component, the Double Pareto distribution described above.

We present the results for this latter mixture, and for the Kaplan-Meier estimator. All quantities reported are upper 95% confidence bounds obtained from results of 10 runs. In Figg. 1, 2, right column, we present the tradeoff between training cost, labeled `train`, and restart performances on the test set, for the two models, respectively labeled `logndp` and `kme`, at different values of the censoring fraction c. We also plot the cost of the universal strategy, labeled U, on the test set (the performance on the training set is similar, as both are composed of 50 randomly picked problem instances). For the test time, we can appreciate some degradation of performance only for very heavy censoring ($c = 0.8, 0.9$), for which the advantage in training time is negligible anyway. Note that this does not mean that the accuracy of the model is unaffected: to highlight this apparent contradiction, we also plotted, in left column, the value of a χ^2 statistic

[7] As we are also conducting experiments with parallel portfolios of heterogeneous algorithms, and thus we need a common measure of time, we modified the original code of Satz-Rand adding a counter, that is incremented at every loop in the code. The resulting time measure was consistent with the number of backtracks. All runtimes reported are in millions of loop cycles.

[8] It is interesting to see that the Weibull distribution, reported in [11] as having a good fit on satisfiable problems near the sat-unsat phase transition, has instead a very poor fit in this case.

of the parametric model `logndp`.[9] While for the uncensored estimate this is near or below the 95% acceptance threshold (indicated by the white bars in the plot), the value of the χ^2 test degrades rapidly even for low values of c.

In Figg. 3 to 6 we display, on the top row, the average of the resulting censoring threshold t_c, for different censoring fractions c. This, along with the training cost, allows to appreciate the tail behavior of Satz-Rand on the different problem set. On problem set 1, most runtimes have a similar value, and the remaining few are very large. t_c is greatly reduced by a modest $c = 0.1$, but further censoring does not decrease it much: the same obviously applies to training cost. On problem set 8, runtime values are spread along two orders of magnitude. Increasing c has a more gradual impact on t_c, and on training cost. This situation varies gradually for intermediate problem sets. Problem 0 is less interesting, as all runs of Satz-Rand end in a similar time, and heavy tailed behavior is not observed. The resulting plots are similar to problem 1, without the heavy tail effects. In the second row, we display the CDF $\hat{F}$ estimated by `dplogn`, on a single run, for different censoring levels ($c = 0.1$, $c = 0.5$, $c = 0.9$), compared with a better approximation of the real F of the set, the empirical Kaplan-Meier CDF evaluated on 200 uncensored runs for each problem (labeled `real`). To further investigate the tail behavior of Satz-Rand RTD, in the third row we display the log-log plot of its survival function $1 - F(x)$, where a Pareto tail ($\propto t^{-\alpha}$) is displayed as a straight line. In both cases, one can visually appreciate the degradation of the model, induced by censored sampling, especially for values of t larger than t_c, which are not seen by the model. So why the performance of the restart strategy is not affected? The fourth row of Figg. 3 to 6 seems to suggest an answer. It plots the expected cost (3) of a uniform restart strategy, against the restart threshold T, evaluated using `dplogn` at different levels of censoring. The comparison term `real` is the *actual* performance of a restart strategy T, evaluated *a-posteriori* on the same run: averaging this on multiple runs, one would obtain an estimate of the real $E(t_T)$ for the problem set. We can see that the estimated and real cost differ greatly, but have a similar minimum: this allows $\hat{T}$ to be efficient also with a poor $\hat{F}$, obtained from a heavily censored runtime sample.

Fig. 7 plots the CDF $\hat{F}$ obtained with the non-parametric Kaplan-Meier estimator. This simple model proved similar in performance to `dplogn`, also in the few cases where this latter failed to converge (see again Figg. 1, 2).

For what concerns the universal restart strategy U, its performance on the test set is consistently worse. Its advantage on the training set was predictable, as in our simple scheme 20 copies of Satz-Rand are run in parallel on each training problem, and obviously decreases with c: on sets 7, 8, training cost is actually lower for $c = 0.8, 0.9$. We expect to further reduce training cost, again with a low impact on test performance, by simply running less parallel copies.

[9] Measured as in [11], dividing the *uncensored* data into m bins (on a logarithmic scale), and comparing the number of samples o_i in each bin to the one e_i predicted by the fitted distribution, according to the formula $\chi^2 = \sum_i [(o_i - e_i)/e_i]^2$. A high value indicates a poor fit: the model passes the test with confidence α if χ^2 is lower than the $1 - \alpha$ quantile of the χ^2 distribution with $m - k$ degrees of freedom, k being the number of parameters in the model.

6 Discussion

There is only an apparent contradiction between the rapid degradation of the model, following the increase in censored data, and the stability of the performance of the estimated optimal restart strategy. Traditional statistical tests are in fact intended to measure the fit of a pdf along the whole spectrum of possible values. The formula for the restart performance (3) is instead based on the *cumulative distribution* function, which is the integral of the pdf; and on its further integration up to T, which is usually small. This means that the actual shape of a large portion of the distribution is irrelevant, as long as its mass does not vary; while for values lower than T, the integration involved in the CDF acts as a "denoising" filter, making (3) more robust to loss of fit of the model.

From a more practical point of view, our experiments show that even a sub-optimal restart strategy can have a relevant advantage over the universal. This advantage can be obtained for an additional training effort, which can be greatly reduced by censored sampling. On a larger test set, this initial training effort would pay off quite rapidly. Note also that there is no reason to stop the training process, as it is relatively cheap compared to problem solving, and each restart can also be interpreted as a censored sample.

We expect analogous results with any heavy-tailed randomized algorithm. This motivates us to further investigate methods to merge training and exploitation of models of algorithm performance, as in [22]. Current research is aimed at a companion analysis of the effect of censoring in the case of algorithm portfolios, and the combination of universal and estimated optimal restart strategies, the latter based on censored and uncensored samples gathered on a sequence of problems, the former limiting cost in the initial phase, when the model is still unreliable, in a *life-long learning* fashion.

References

1. Gomes, C.P., Fernandez, C., Selman, B., Bessiere, C.: Statistical regimes across constrainedness regions. Constraints **10**(4) (2005) 317–337
2. Rice, J.R.: The algorithm selection problem. In Rubinoff, M., Yovits, M.C., eds.: Advances in computers. Volume 15. Academic Press, New York (1976) 65–118
3. Hogg, T., Williams, C.P.: The hardest constraint problems: a double phase transition. Artif. Intell. **69**(1-2) (1994) 359–377
4. Gomes, C.P., Selman, B., Crato, N., Kautz, H.: Heavy-tailed phenomena in satisfiability and constraint satisfaction problems. J. Autom. Reason. **24**(1-2) (2000) 67–100
5. Huberman, B.A., Lukose, R.M., Hogg, T.: An economic approach to hard computational problems. Science **275** (1997) 51–54
6. Luby, M., Sinclair, A., Zuckerman, D.: Optimal speedup of las vegas algorithms. Inf. Process. Lett. **47**(4) (1993) 173–180
7. Gomes, C.P., Selman, B.: Algorithm portfolios. Artificial Intelligence **126**(1–2) (2001) 43–62
8. Nelson, W.: Applied Life Data Analysis. John Wiley, New York (1982)
9. Li, C.M., Anbulagan: Heuristics based on unit propagation for satisfiability problems. In: IJCAI97. (1997) 366–371
10. Hoos, H.H., Stützle, T.: SATLIB: An Online Resource for Research on SAT. In I.P.Gent, H.v.Maaren, T.W., ed.: SAT 2000, IOS press (2000) 283–292 `http://www.satlib.org`.

11. Frost, D., Rish, I., Vila, L.: Summarizing csp hardness with continuous probability distributions. In: AAAI/IAAI. (1997) 327–333
12. van Moorsel, A.P.A., Wolter, K.: Analysis and algorithms for restart. In: QEST '04: Proceedings of the The Quantitative Evaluation of Systems, First International Conference on (QEST'04), Washington, DC, USA, IEEE Computer Society (2004) 195–204
13. Hoos, H.H.: On the run-time behaviour of stochastic local search algorithms for sat. In: Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI'99), Orlando, Florida (1999) 661–666
14. Hoos, H.H., Stützle, T.: Local search algorithms for SAT: An empirical evaluation. Journal of Automated Reasoning **24**(4) (2000) 421–481
15. Hoos, H.H.: A mixture-model for the behaviour of sls algorithms for sat. In: Eighteenth national conference on Artificial intelligence, Menlo Park, CA, USA, American Association for Artificial Intelligence (2002) 661–667
16. Boddy, M., Dean, T.L.: Deliberation scheduling for problem solving in time-constrained environments. Artificial Intelligence **67**(2) (1994) 245–285
17. Hansen, E.A., Zilberstein, S.: Monitoring and control of anytime algorithms: A dynamic programming approach. Artificial Intelligence **126**(1–2) (2001) 139–157
18. Alt, H., Guibas, L., Mehlhorn, K., Karp, R., Widgerson, A.: A method for obtaining randomized algorithms with small tail probalities. Research Report MPI-I-92-110, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany (1992)
19. Gomes, C.P., Selman, B., Kautz, H.: Boosting combinatorial search through randomization. In: AAAI '98/IAAI '98: Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence, Menlo Park, CA, USA, American Association for Artificial Intelligence (1998) 431–437
20. Kautz, H., Horvitz, E., Ruan, Y., Gomes, C., Selman, B.: Dynamic restart policies (2002)
21. Gagliolo, M., Schmidhuber, J.: A neural network model for inter-problem adaptive online time allocation. In Duch, W., et al., eds.: Artificial Neural Networks: Formal Models and Their Applications - ICANN 2005, Part 2, Springer (2005) 7–12
22. Gagliolo, M., Schmidhuber, J.: Dynamic algorithm portfolios. In: Ninth International Symposium on Artificial Intelligence and Mathematics, Fort Lauderdale, Florida, January 2006. (2006)
23. Gent, I., Hoos, H.H., Prosser, P., Walsh, T.: Morphing: Combining structure and randomness. In: Proc. of AAAI-99. (1999) 654–660 — Benchmark avaliable at `http://www.intellektik.informatik.tu-darmstadt.de/SATLIB/Benchmarks/SAT/SW-GCP/descr.html`.
24. Reed, W.: The double pareto-lognormal distribution - a new parametric model for size distribution (2001) — Available at http://www.math.uvic.ca/faculty/reed/index.html.
25. Kaplan, E., Meyer, P.: Nonparametric estimation from incomplete samples. J. of the ASA **73** (1958) 457–481

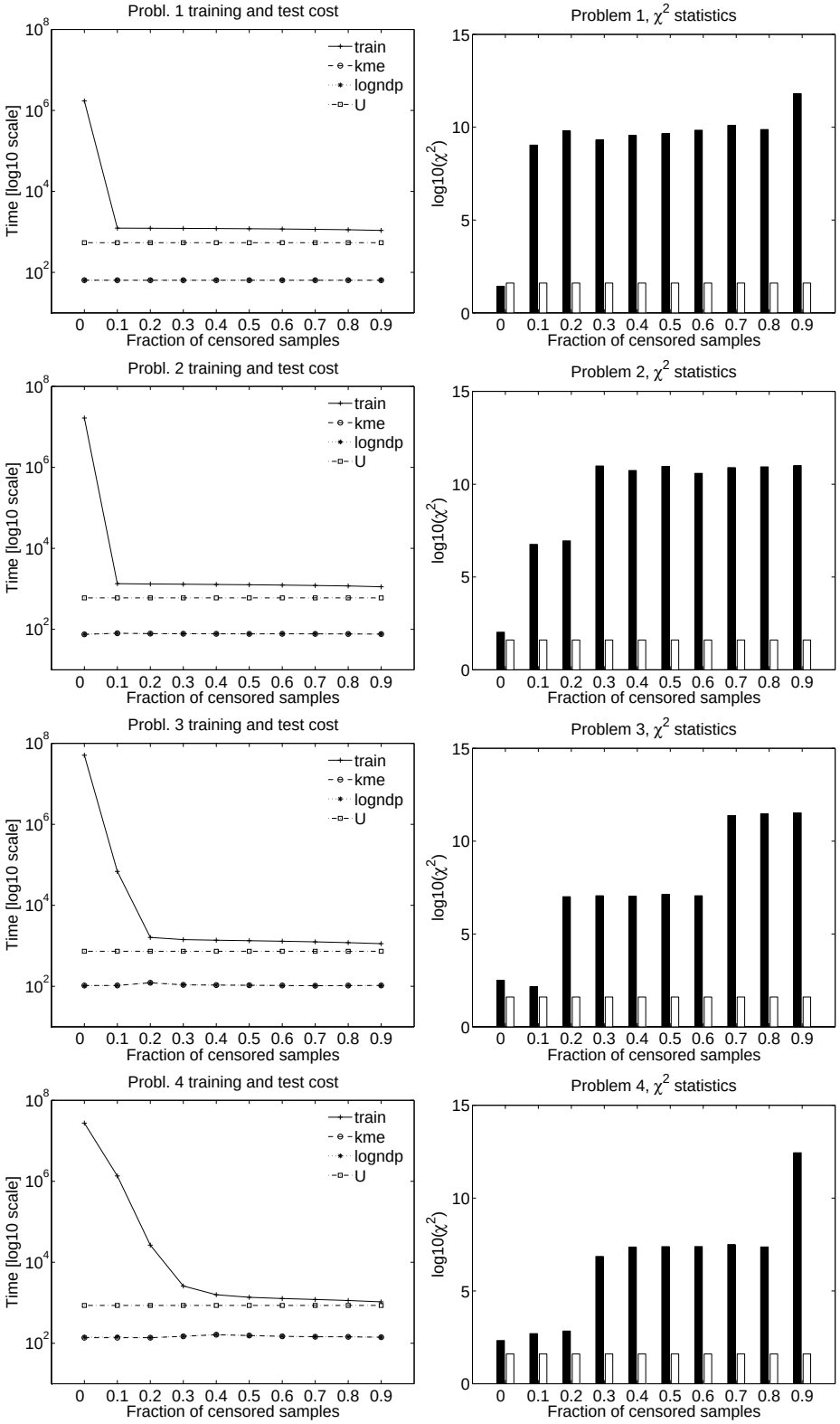

Fig. 1. Problems 1 to 4. Left: the trade-off between training cost (`train`) and test performances of the parametric mixture lognormal-double Pareto (`logndp`), and the non-parametric Kaplan-Meier estimator (`kme`), for different censoring fractions c. `U` labels the performance t_U of the universal strategy on the test set. Right: $\log_{10}$ of the χ^2 statistics for `logndp` (black), compared to $\log_{10}$ of the acceptance threshold (white).

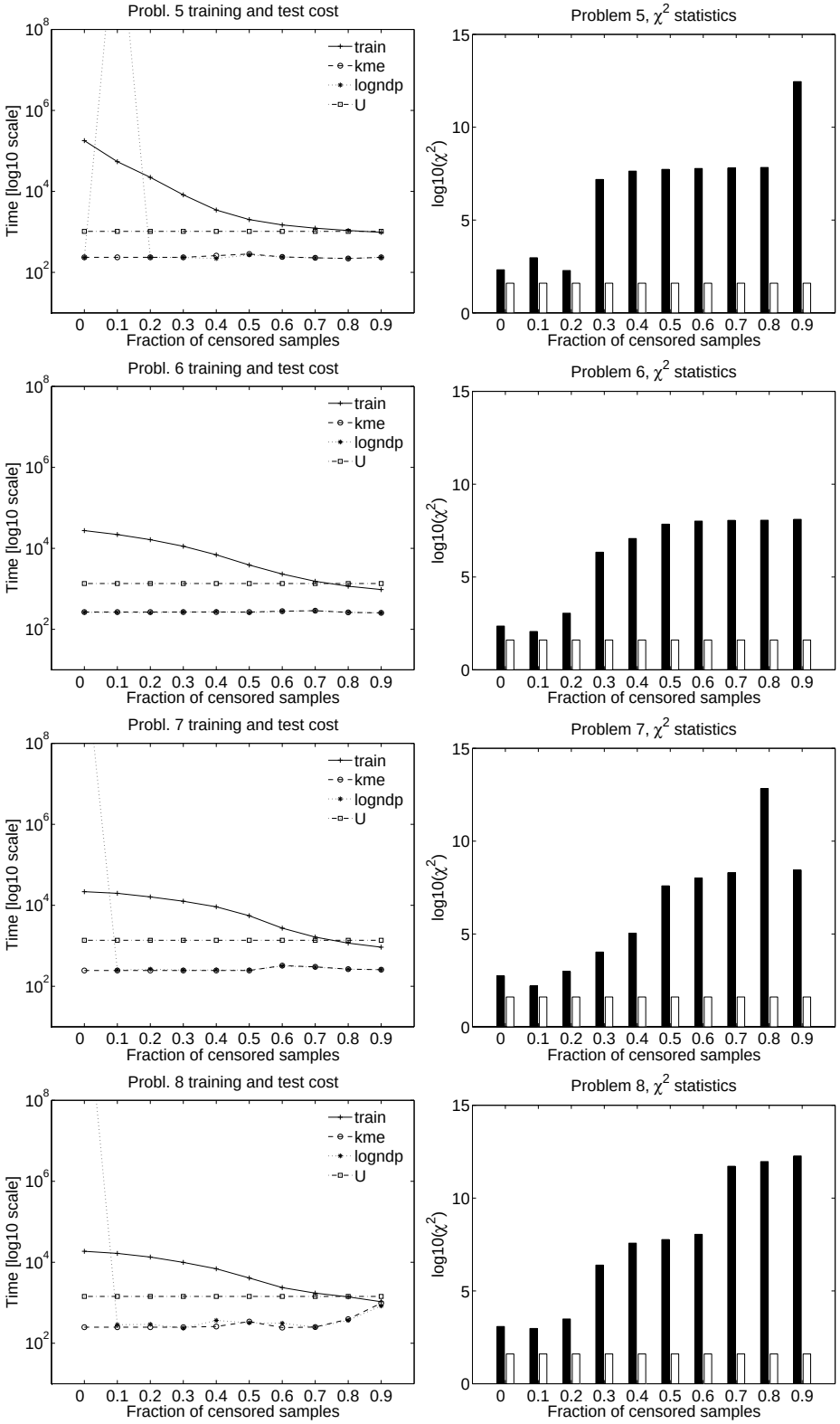

Fig. 2. Problems 5 to 8. Left: the trade-off between training cost (`train`) and test performances of the parametric mixture lognormal-double Pareto (`logndp`), and the non-parametric Kaplan-Meier estimator (`kme`), for different censoring fractions c. `U` labels the performance t_U of the universal strategy on the test set. Right: $\log_{10}$ of the χ^2 statistics for `logndp` (black), compared to $\log_{10}$ of the acceptance threshold (white).

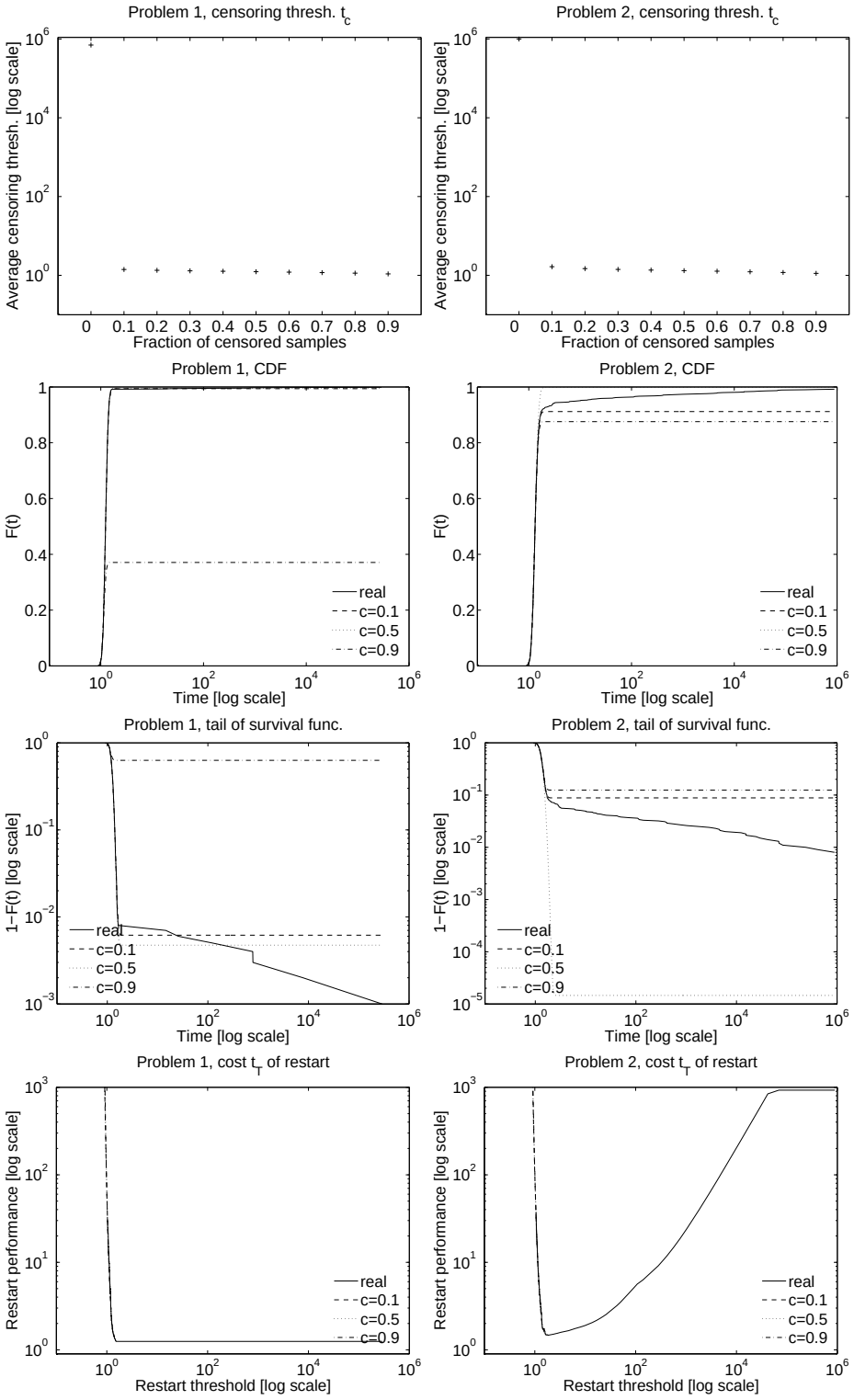

Fig. 3. Problems 1 (left column) and 2 (right column). From top to bottom: average censoring threshold t_c for different fractions of censoring; CDF; tail of the survival function; estimated expected cost of restart for different censoring levels ($c = 0.1$, $c = 0.5$, $c = 0.9$), compared with real cost, evaluated *a posteriori*. Last three rows refer to a single run. Note the similar minima in last row.

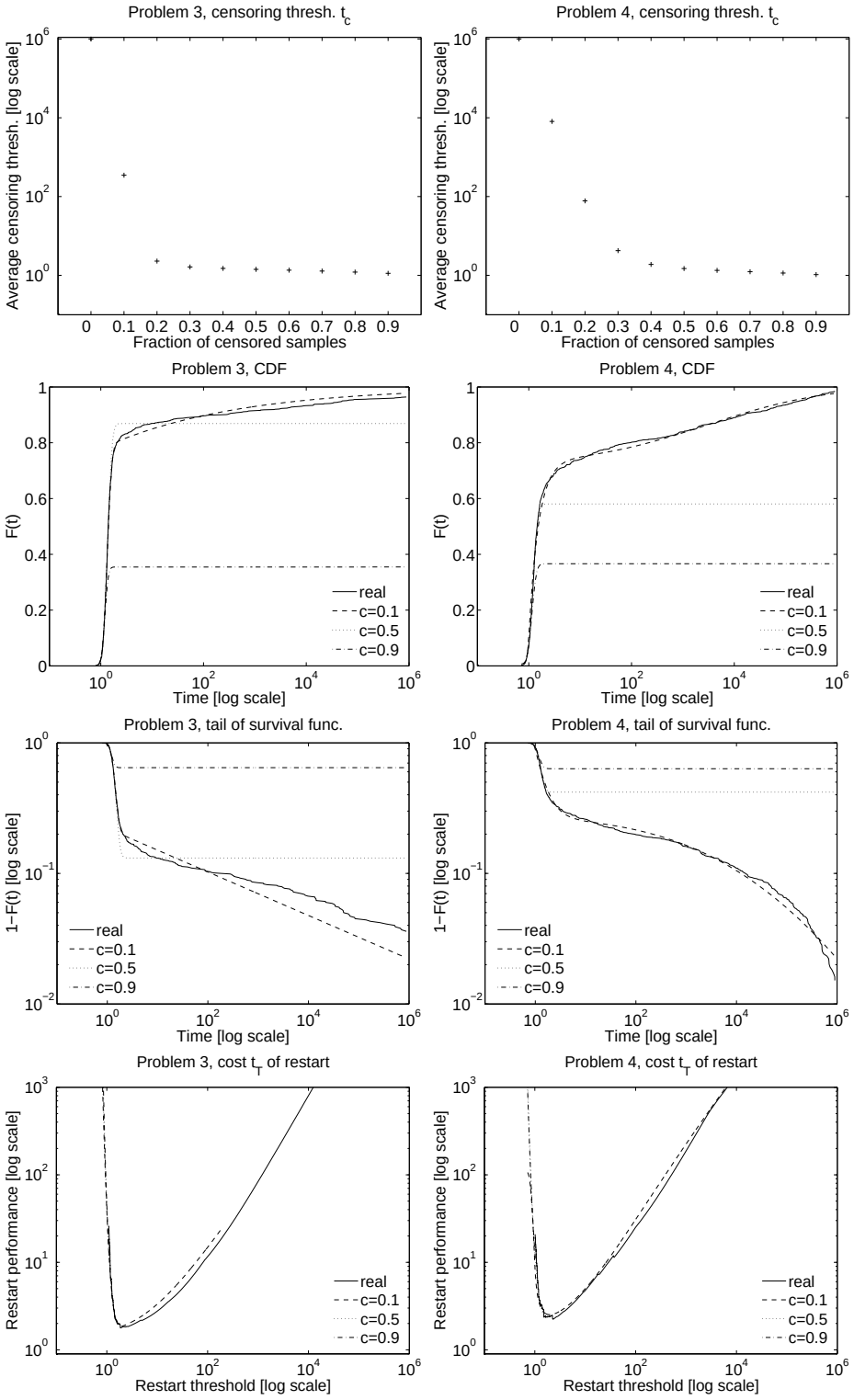

Fig. 4. Problems 3 (left column) and 4 (right column).From top to bottom: average censoring threshold t_c for different fractions of censoring; CDF; tail of the survival function; estimated expected cost of restart for different censoring levels ($c = 0.1$, $c = 0.5$, $c = 0.9$), compared with real cost, evaluated *a posteriori*. Last three rows refer to a single run. Note the similar minima in last row.

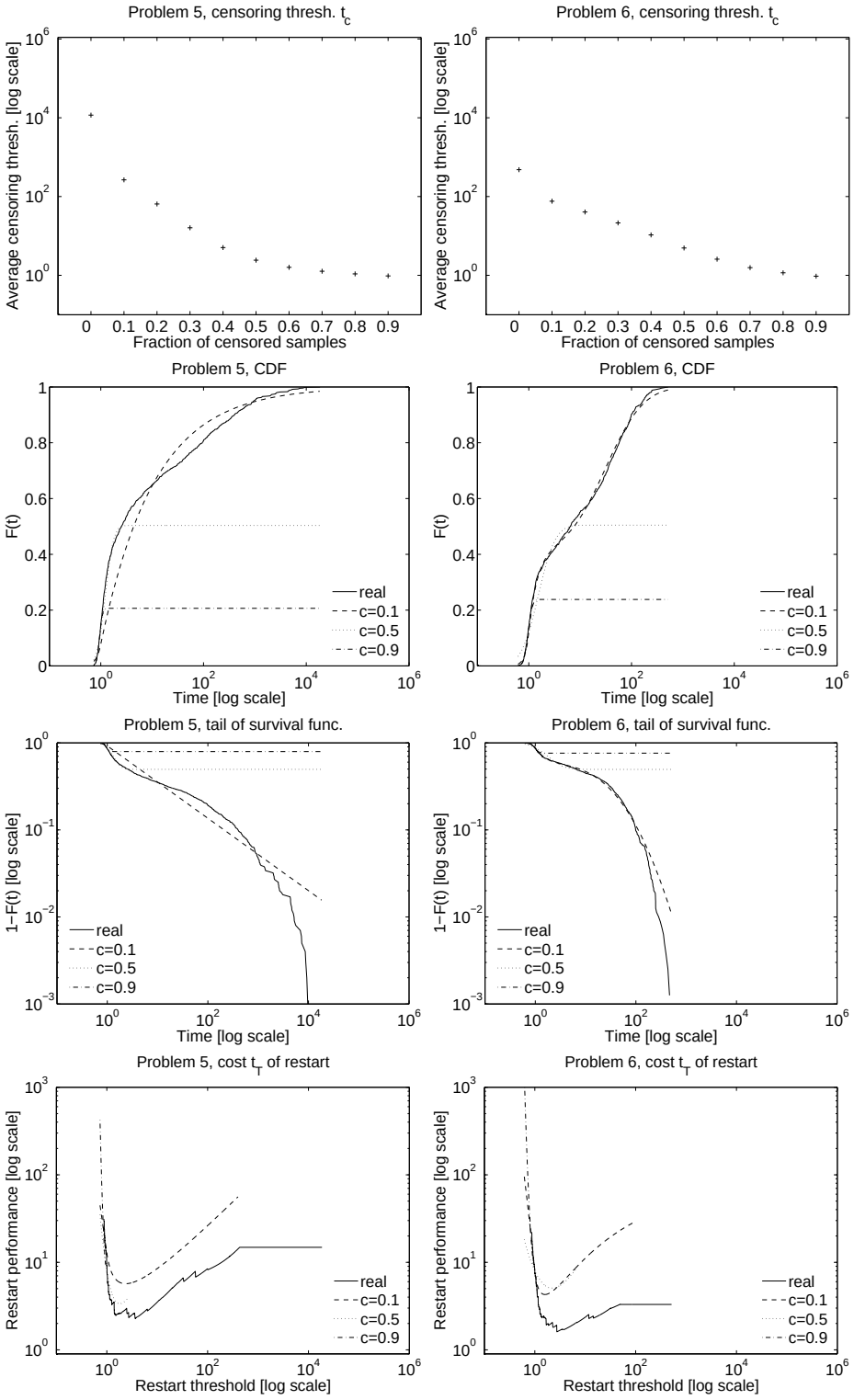

Fig. 5. Problems 5 (left column) and 6 (right column). From top to bottom: average censoring threshold t_c for different fractions of censoring; CDF; tail of the survival function; estimated expected cost of restart for different censoring levels ($c = 0.1$, $c = 0.5$, $c = 0.9$), compared with real cost, evaluated *a posteriori*. Last three rows refer to a single run. Note the similar minima in last row.

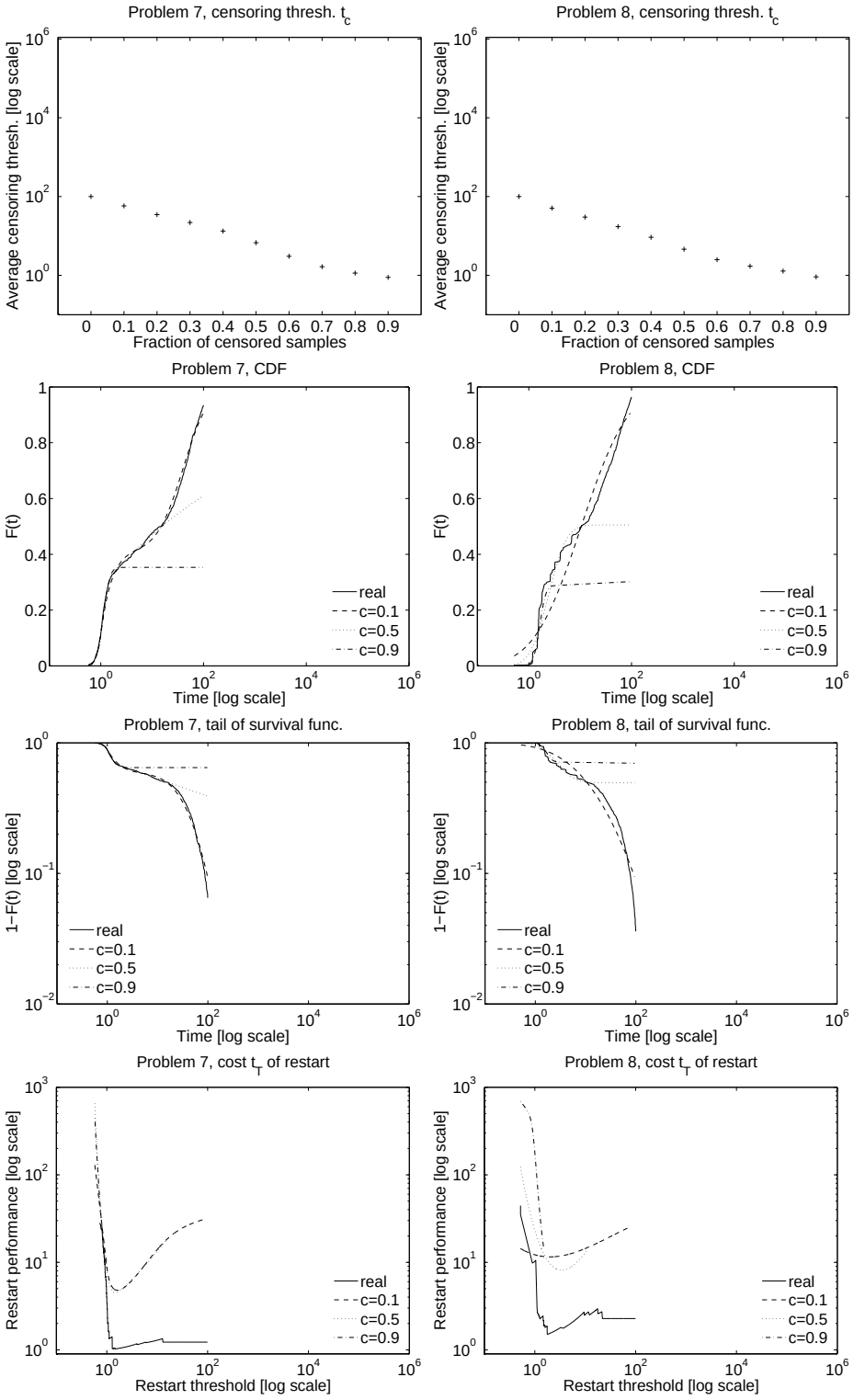

Fig. 6. Problems 7 (left column) and 8 (right column). From top to bottom: average censoring threshold t_c for different fractions of censoring; CDF; tail of the survival function; estimated expected cost of restart for different censoring levels ($c = 0.1$, $c = 0.5$, $c = 0.9$), compared with real cost, evaluated *a posteriori*. Last three rows refer to a single run. Note the similar minima in last row.

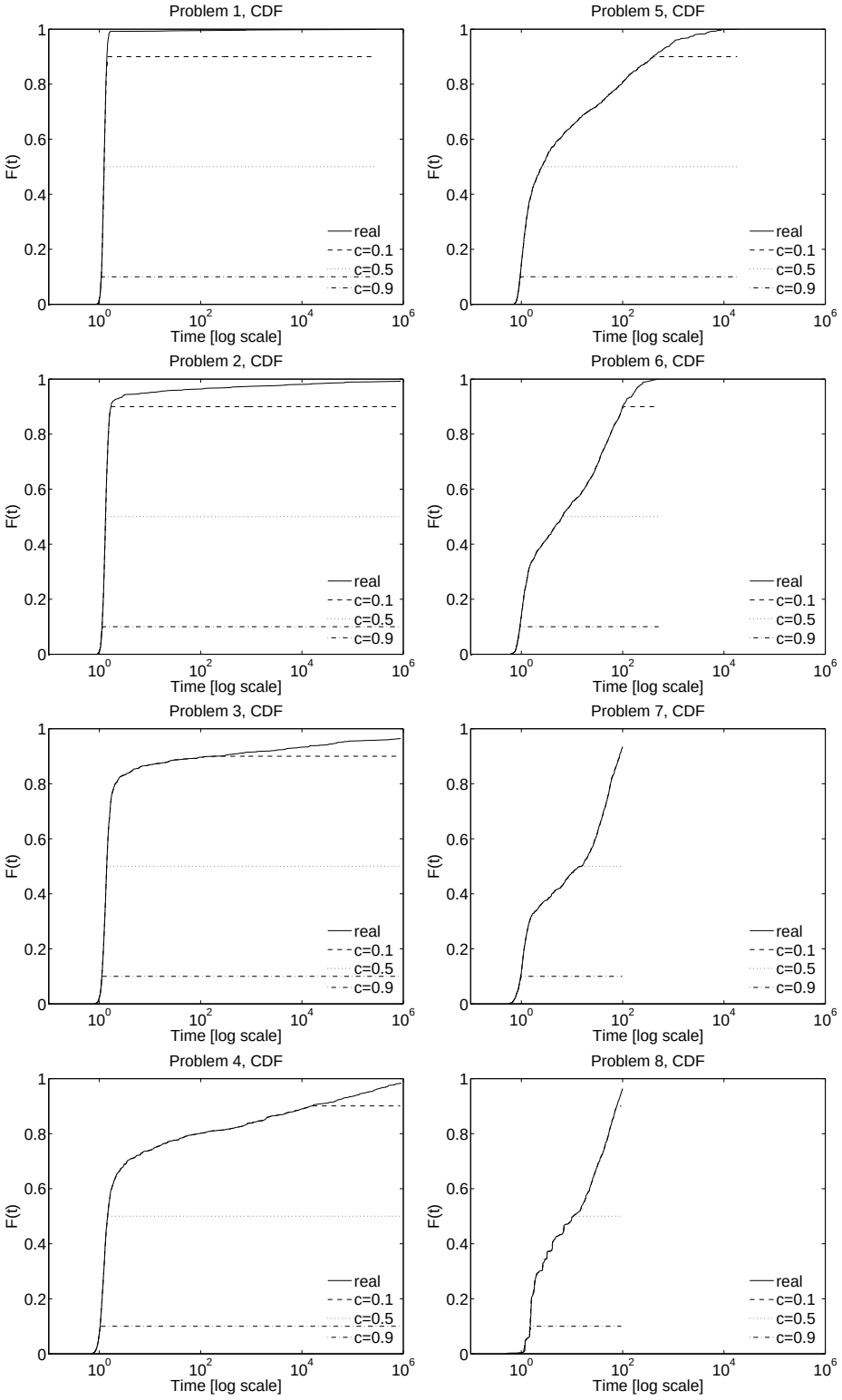

Fig. 7. Problems 1 to 8, Kaplan-Meier estimator.