Evaluation Report on Six Experiments

Anne Anonymous

***Abstract*—This is the evaluation report on six experiments, namely 1FlipHC, 1FlipHCrs, 2FlipHC, 2FlipHCrs, mFlipHC, and mFlipHCrs on 100 benchmark instances. This report has been generated with the version 0.8.4 of the Evaluator Component of the Optimization Benchmarking Tool Suite.**

I. Instance Information

Experiments were conducted on 100 benchmark instances, which can be distinguished by two features.

The benchmark instances are characterized by two features:

- n (ten values, ranging from 20 to 250)
- k (ten values, ranging from 91 to 1065)

In Figure 2 we illustrate the relative amount of benchmark instances per feature value over all 100 benchmark instances. The slices in the pie charts are the bigger, the more benchmark instances have the associated feature value in comparison to the other values. The more similar the pie sizes are, the more evenly are the benchmark instances distributed over the benchmark feature values, which may be a good idea for fair experimentation.

II. Performance Comparisons

A. Estimated Cumulative Distribution Function

We analyze the estimated cumulative distribution function (*ECDF*) [1], [2], [3] computed based on $\frac{F}{k}$ over $\log_{10} FEs$. The $\mathrm{ECDF}\left(FEs, \frac{F}{k} \leq 0\right)$ represents the fraction of runs which reach a value of $\frac{F}{k}$ less than or equal to 0 for a given ellapsed runtime measured in *FEs*. The *ECDF* is always computed over the runs of an experiment for a given benchmark instance. If runs for multiple instances are available, we aggregate the results by computing their arithmetic mean. The x-axis does not represent the values of *FEs* directly, but instead $\log_{10} FEs$. The *ECDF* is always between 0 and 1 — and the higher it is, the better.

The corresponding plot is illustrated in Figure 1.

B. Estimated Cumulative Distribution Function

We analyze the estimated cumulative distribution function (*ECDF*) [1], [2], [3] computed based on $\frac{F}{k}$ over $\log_{10} RT$. The $\mathrm{ECDF}\left(RT, \frac{F}{k} \leq 0.01\right)$ represents the fraction of runs which reach a value of $\frac{F}{k}$ less than or equal to 0.01 for a given ellapsed runtime measured in *RT*. The *ECDF* is always computed over the runs of an experiment for a given benchmark instance. If runs for multiple instances are available, we aggregate the results by computing their arithmetic mean. The x-axis does not represent the values of *RT* directly, but instead $\log_{10} RT$. The *ECDF* is always between 0 and 1 — and the higher it is, the better. The instance run sets belonging to instances with the same value of the feature n grouped together.

The corresponding plots are illustrated in Figure 3.

C. Median of Medians

We analyze the median of medians (*med med*) of F over $\log_{10}\left(\frac{FEs}{n}\right)$. The $\mathrm{med\ med}(FEs, F)$ represents the median of the F for a given ellapsed runtime measured in *FEs*. The median is always computed over the runs of an experiment for a given benchmark instance. If runs for multiple instances are available, we aggregate these medians by computing their median. The x-axis does not represent the values of *FEs* directly, but instead $\log_{10}\left(\frac{FEs}{n}\right)$. The instance run sets belonging to instances with the same value of the feature k grouped together.

The corresponding plots are illustrated in Figure 4.

D. Median of Standard Deviations

We analyze the median of standard deviations (*med stddev*) computed based on $\frac{F}{k}$ over $\log_{10} RT$. The $\mathrm{med\ stddev}\left(RT, \frac{F}{k}\right)$ represents the standard deviation of the $\frac{F}{k}$ for a given ellapsed runtime measured in *RT*. The standard deviation is always computed over the runs of an experiment for a given benchmark instance. If runs for multiple instances are available, we aggregate these standard deviations by computing their median. The x-axis does not represent the values of *RT* directly, but instead $\log_{10} RT$. The instance run sets belonging to instances with the same value of the feature n grouped together.

The corresponding plots are illustrated in Figure 5.

References

[1] H. H. Hoos and T. Stützle, “Evaluating las vegas algorithms — pitfalls and remedies,” in *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI’98)*, G. F. Cooper and S. Moral, Eds. Madison, WI, USA: San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., Jul. 24–26, 1998, pp. 238–245. [Online]. Available: http://www.intellektik.informatik.tu-darmstadt.de/TR/1998/98-02.ps.Z

[2] D. A. D. Tompkins and H. H. Hoos, “Ubcsat: An implementation and experimentation environment for sls algorithms for sat and max-sat,” in *Revised Selected Papers from the Seventh International Conference on Theory and Applications of Satisfiability Testing (SAT’04)*, ser. Lecture Notes in Computer Science (LNCS), H. H. Hoos and D. G. Mitchell, Eds., vol. 3542. Vancouver, BC, Canada: Berlin, Germany: Springer-Verlag GmbH, May 10–13, 2004, pp. 306–320. [Online]. Available: http://ubcsat.dtompkins.com/downloads/sat04proc-ubcsat.pdf?attredirects=0

[3] N. Hansen, A. Auger, S. Finck, and R. Ros, “Real-parameter black-box optimization benchmarking: Experimental setup,” Orsay, France: Université Paris Sud, Institut National de Recherche en Informatique et en Automatique (INRIA) Futurs, Équipe TAO, Tech. Rep., Mar. 24, 2012. [Online]. Available: http://coco.lri.fr/BBOB-downloads/download11.05/bbobdocexperiment.pdf