User Guide

Zongxu Mu
Department of Computer Science
University of British Columbia
zongxumu@cs.ubc.ca

31st October 2015

In this document, we introduce an automated tool – the Empirical Scaling Analyser (ESA) – that can perform core elements of the analysis described in [2]. To use ESA, a user needs to prepare an input file of running time data of an algorithm (referred to as target algorithm hereafter), as well as other optional files, including a configuration file, a file specifying the parametric models to be fitted, a LaTeX template and a gnuplot template. Details of these input files will be given in Section 1. Note that ESA is not limited to fitting and assessing a single scaling model, but can deal with multiple models simultaneously. In other words, once data collection is finished, a user can put all running time data into a file, feed it into ESA and obtain the results from the scaling analysis using several parametric models. Results are presented in a technical report, which contains easy-to-read tables and figures for the scaling of the target algorithm. The details of the output report are described in Section 2.

We believe the tool is useful for other researchers who want to study the empirical time complexity of other algorithms, and can thus promote the use of such analysis for other problems and algorithms. The tool is available as an easy-to-use online service at `www.cs.ubc.ca/labs/beta/Projects/ESA/esa-online.html` and as a command-line tool with additional functionality (see Section 5 on how to run ESA). Here, we describe all features available in the command-line version.

1 Input

To perform scaling analysis, ESA requires input data containing the sizes of the instances studied and the running times the target algorithm requires for solving these instances. The input running time data need to follow the following formatting rules:

- the input file contains lines of details of the instances, one instance per line;

- in each line, the following three pieces of information are provided in order and are separated by “,”:

    - instance name (e.g., file name) and other optional information (this field is for the user’s reference only; ESA does not use this field in the scaling

```
# instance name , size , datum ( running time)
portgen -500 -1000. tsp ,500 ,2.3
portgen -500 -100. tsp ,500 ,2.58
portgen -500 -101. tsp ,500 ,2.36
portgen -500 -102. tsp ,500 ,2.51
portgen -500 -103. tsp ,500 ,2.63
portgen -500 -104. tsp ,500 ,2.84
portgen -500 -105. tsp ,500 ,2.62
portgen -500 -106. tsp ,500 ,3
...
portgen -600 -1000. tsp ,600 ,3.42
...
portgen -4500 -10. tsp ,4500 ,727.68
portgen -4500 -11. tsp ,4500 ,inf
...
#instances ,4000 ,100
#instances ,4500 ,100
```

Figure 1: Excerpt of an input file for ESA, where deleted lines are represented by "...".

    analysis);

- instance size;
- running time required to solve the instance, which itself may be a statistic for multiple runs of the target algorithm solving the instance and may be "inf" for time-out or crashed runs.

Besides, the user may specify the number of instances for some sizes. If there are not enough entries for one size, ESA will treat the missing entries as instances with unknown running time. An example for such data is described in [1], in the context of analysing the scaling behaviour of EAX and LKH, where the running times of some instances are unknown because no optimal solution has been found in previous runs of Concorde. An excerpt of an input file for ESA is in Figure 1.

ESA also takes as input a configuration file, containing details on the target algorithm (algName), the instance distribution (instName), the number of bootstrap samples (numTrainingData), etc. The file contains lines of configurations, one configuration per line. Each configuration follows the "`name : value`" format. An example of a configuration file is shown in Figure 2.

There are a number of other files that a user may supply (if not supplied, ESA will use the default file(s) distributed with the code), including:

- a file specifying the models to be fit

- a LaTeX template specifying the content and format of the output report

- gnuplot templates specifying the format of the plots

```
fileName : runtimes.csv
algName : WalkSAT/SKC
instName : random 3-SAT instances at phase transition
modelFileName : models.txt
numTrainingData : 7
alpha : 95
numBootstrapSamples : 1000
statistic : median
latexTemplate : template-AutoScaling.tex
modelPlotTemplate : template_plotModels.plt
residuePlotTemplate : template_plotResidues.plt
```

Figure 2: Example of a configuration file for ESA.

```
Exp,2,@@a@@\times @@b@@^{x},@@a@@*@@b@@**x,@@a@@*@@b@@
    **x,1e-4,1.01
Poly,2,@@a@@\times x^{@@b@@},@@a@@*x**@@b@@,@@a@@*x**
    @@b@@,1e-8,1
```

Figure 3: An example of model specification for ESA.

The first of these is needed, because ESA supports customised models, as long as the models are supported by python (including the math and the numpy packages) and gnuplot. This file contains lines of models, one model per line. Each contains the following items, separated by ";":

- Model name (e.g., Exponential)

- Number of parameters (e.g., 2)

- LaTeXexpression of the model

- Python expression of the model

- Gnuplot expression of the model

- Default values of the parameters, separated by ","

For all expressions of the models, x represents the size, and the parameters should be $a, b, \ldots$, and should be surrounded by "@@". For example, the specification in Figure 3, which is also the default model specification, tells ESA to fit an exponential and a polynomial model of the form $a \cdot b^x$ and $a \cdot x^b$ respectively:

For the LaTeX template, ESA will use the default template, if no customised template is found. In the template, dynamic values should be surrounded by "@@". For instance, the name of the algorithm (which is defined in the configuration file) is a dynamic value. Wherever mentioned in the template file, the user should use "@@alg-Name@@", and ESA will instantiate it to be the real name of the algorithm when

3

n	500	600	700	800
# instances	1000	1000	1000	1000
# running times	1000	1000	1000	1000
mean	19.33	31.55	56.8	89.35
coefficient of variation	1.164	1.418	1.303	1.455
Q(0.1)	0.43	2.21	6.73	10.62
Q(0.25)	3.75	8.75	15.3	22.31
median	12.22	18.64	33.15	48.95
Q(0.75)	25.01	37.64	68.29	102.5
Q(0.9)	47.33	70.35	130.4	195.5

n	900	1000	1100	1200
# instances	1000	1000	1000	1000
# running times	1000	1000	1000	1000
mean	139.7	201.2	314.6	385.4
coefficient of variation	1.734	1.759	1.851	1.713
Q(0.1)	17.23	23.28	28.8	38.76
Q(0.25)	32.72	43.23	60.84	78.66
median	70.64	98.56	145.7	177.2
Q(0.75)	142.7	216.1	341.3	409.2
Q(0.9)	302.7	429.4	693.2	846.3

n	1300	1400	1500
# instances	1000	1000	1000
# running times	1000	1000	1000
mean	548.7	749	1072
coefficient of variation	1.859	2.227	2.109
Q(0.1)	53.27	73.78	93.04
Q(0.25)	112.2	153.3	210.1
median	271.7	344.3	483.5
Q(0.75)	583.6	783.6	1136
Q(0.9)	1190	1517	2277

Table 1: Example output of ESA – statistics of running times for support data.

generating the report. Users can also specify the formats of the plots via the template gnuplot script. For instance, users may choose whether to use a log-log plot or a semi-log plot via the template gnuplot script. Default templates are available for download together with the source code (see Section 5 for details).

2 Output

ESA automatically generates a technical report containing detailed empirical scaling analysis results and interpretation. This report contains tables and figures that users can easily read, including:

- two tables of statistics of running times, one for support data and the other for challenge, as illustrated in Tables 1 & 2, respectively;

- a table of fitted models and corresponding RMSE values, as illustrated in Table 3;

- a figure of running times, fitted models and corresponding bootstrap confidence intervals of each model, as illustrated in Figure 4;

n	2000	2500	3000
# instances	1000	100	100
# running times	1000	100	100
mean	5402	∞	∞
coefficient of variation	2.624	N/A	N/A
Q(0.1)	307	671.5	3538
Q(0.25)	765.5	1694	8188
median	1969	6149	1.84×10^4
Q(0.75)	5207	1.766×10^4	4.118×10^4
Q(0.9)	1.137×10^4	4.611×10^4	9.048×10^4

n	3500	4000	4500
# instances	100	100	100
# running times	100	100	100
mean	∞	∞	∞
coefficient of variation	N/A	N/A	N/A
Q(0.1)	6060	2.096×10^4	3.041×10^4
Q(0.25)	1.226×10^4	4.125×10^4	1.22×10^5
median	3.246×10^4	1.312×10^5	2.633×10^5
Q(0.75)	9.717×10^4	3.938×10^5	∞
Q(0.9)	2.76×10^5	∞	∞

Table 2: Example output of ESA – statistics of running times for challenge data.

		Model	RMSE (support)	RMSE (challenge)
	Exp. Model	4.0388×1.0032^x	7.7847	2.7852×10^6
Concorde	RootExp. Model	$\mathbf{0.083457 \times 1.2503^{\sqrt{x}}}$	**7.0439**	**9169.4**
	Poly. Model	$1.6989 \times 10^{-10} \times x^{3.9176}$	9.9327	1.038×10^5

Table 3: Example output of ESA – fitted models and corresponding RMSE values.
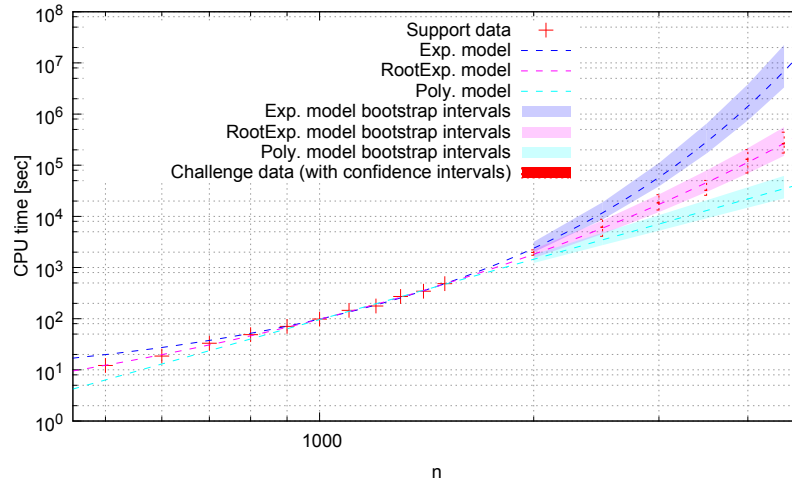

Figure 4: Example output of ESA – a figure of running times, fitted models and corresponding bootstrap confidence intervals of each model.
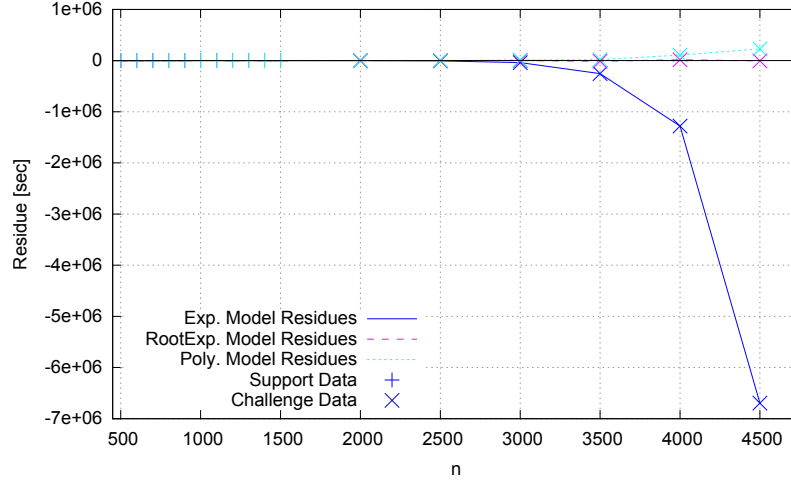
Figure 5: Example output of ESA – a figure of residues of the fitted models.

Solver	Model	Confidence interval of a	Confidence interval of b
	Exp.	$[2.6108, 5.2975]$	$[1.003, 1.0036]$
Concorde	RootExp.	$[0.037056, 0.15111]$	$[1.2287, 1.2793]$
	Poly.	$\left[6.1872 \times 10^{-12}, 1.7351 \times 10^{-9}\right]$	$[3.5859, 4.3713]$

Table 4: Example output of ESA – bootstrap confidence intervals for all model parameters.

- a figure of residues of the fitted models, as illustrated in Figure 5;

- a table of bootstrap confidence intervals for all model parameters, as illustrated in Table 4;

- two tables of bootstrap confidence intervals for observed and predicted running times, one for support data and the other for challenge, as illustrated in Tables 5 & 6.

A snapshot of the reported generated by ESA using the default LATEX template is shown in Figure 6.

3   Automated Interpretation of Scaling Results

In addition, ESA generates automated interpretations for scaling analysis results. It evaluates how well a model fits the given data based on the percentage of challenge sizes for which the model predicts the corresponding running times reasonably accurately. If a model predicts well for most challenge sizes, then the model should be accepted as a good fit. Technically, the evaluation is based on the percentage of challenge points that lie within the predicted bootstrap confidence intervals of the model. It especially emphasises the challenge points for larger input sizes, as those provide more

Solver	n	Predicted confidence intervals	Observed median run-time	
		Exp. model	Point estimates	Confidence intervals
	500	$[15.43, 23.47]$	12.22	$[11.01, 13.33]$
	600	$[22, 31.52]$	18.64	$[16.99, 20.16]$
	700	$[\mathbf{31.38}, \mathbf{42.57}]$#	33.15	$[30.38, 35.9]$
	800	$[\mathbf{44.68}, \mathbf{57.59}]$#	48.95	$[44.05, 53.15]$
	900	$[\mathbf{63.49}, \mathbf{77.63}]$*	70.64	$[64.34, 77.28]$
Concorde	1000	$[\mathbf{90.18}, \mathbf{104.8}]$#	98.56	$[90.16, 107.8]$
	1100	$[\mathbf{127.6}, \mathbf{141.9}]$	145.7	$[130.9, 158.6]$
	1200	$[\mathbf{178.6}, \mathbf{193}]$	177.2	$[165.8, 196.1]$
	1300	$[\mathbf{244.7}, \mathbf{266.8}]$	271.7	$[244.9, 298.3]$
	1400	$[\mathbf{332.7}, \mathbf{375.7}]$#	344.3	$[318.9, 383.3]$
	1500	$[\mathbf{448.8}, \mathbf{534.7}]$#	483.5	$[432.8, 532.9]$

Solver	n	Predicted confidence intervals	Observed median run-time	
		RootExp. model	Point estimates	Confidence intervals
	500	$[\mathbf{9.176}, \mathbf{15.31}]$*	12.22	$[11.01, 13.33]$
	600	$[\mathbf{15.56}, \mathbf{23.81}]$*	18.64	$[16.99, 20.16]$
	700	$[\mathbf{25.21}, \mathbf{35.83}]$#	33.15	$[30.38, 35.9]$
	800	$[\mathbf{39.43}, \mathbf{52.46}]$#	48.95	$[44.05, 53.15]$
	900	$[\mathbf{60.23}, \mathbf{74.99}]$#	70.64	$[64.34, 77.28]$
Concorde	1000	$[\mathbf{89.56}, \mathbf{105.2}]$#	98.56	$[90.16, 107.8]$
	1100	$[\mathbf{130.4}, \mathbf{145.2}]$	145.7	$[130.9, 158.6]$
	1200	$[\mathbf{184.4}, \mathbf{199.1}]$	177.2	$[165.8, 196.1]$
	1300	$[\mathbf{252}, \mathbf{274}]$#	271.7	$[244.9, 298.3]$
	1400	$[\mathbf{336.6}, \mathbf{380.4}]$#	344.3	$[318.9, 383.3]$
	1500	$[\mathbf{442}, \mathbf{522.8}]$#	483.5	$[432.8, 532.9]$

Solver	n	Predicted confidence intervals	Observed median run-time	
		Poly. model	Point estimates	Confidence intervals
	500	$[4.206, 8.58]$	12.22	$[11.01, 13.33]$
	600	$[9.398, 16.47]$	18.64	$[16.99, 20.16]$
	700	$[18.41, 28.62]$	33.15	$[30.38, 35.9]$
	800	$[\mathbf{32.91}, \mathbf{46.48}]$	48.95	$[44.05, 53.15]$
	900	$[\mathbf{55.04}, \mathbf{71.24}]$#	70.64	$[64.34, 77.28]$
Concorde	1000	$[\mathbf{86.98}, \mathbf{104.3}]$#	98.56	$[90.16, 107.8]$
	1100	$[\mathbf{131.4}, \mathbf{147.6}]$#	145.7	$[130.9, 158.6]$
	1200	$[\mathbf{188.5}, \mathbf{204.4}]$	177.2	$[165.8, 196.1]$
	1300	$[\mathbf{257.9}, \mathbf{280.1}]$#	271.7	$[244.9, 298.3]$
	1400	$[\mathbf{339.3}, \mathbf{384.2}]$#	344.3	$[318.9, 383.3]$
	1500	$[\mathbf{435.4}, \mathbf{516.3}]$#	483.5	$[432.8, 532.9]$

Table 5: Example output of ESA – bootstrap confidence intervals for observed and predicted running times for support data.

Solver	n	Predicted confidence intervals Exp. model	Observed median run-time Point estimates	Confidence intervals
Concorde	2000	$[\mathbf{1988}, \mathbf{3179}]$	1969	$[1739, 2222]$
	2500	$[\mathbf{8718}, \mathbf{1.884 \times 10^4}]$	6149	$[4084, 8812]$
	3000	$[3.853 \times 10^4, 1.103 \times 10^5]$	1.84×10^4	$[1.332 \times 10^4, 2.669 \times 10^4]$
	3500	$[1.698 \times 10^5, 6.479 \times 10^5]$	3.246×10^4	$[2.581 \times 10^4, 5.038 \times 10^4]$
	4000	$[7.5 \times 10^5, 3.809 \times 10^6]$	1.312×10^5	$[7.073 \times 10^4, 2.024 \times 10^5]$
	4500	$[3.301 \times 10^6, 2.245 \times 10^7]$	2.633×10^5	$[1.73 \times 10^5, 4.419 \times 10^5]$

Solver	n	Predicted confidence intervals RootExp. model	Observed median run-time Point estimates	Confidence intervals
Concorde	2000	$[\mathbf{1528}, \mathbf{2269}]$*	1969	$[1739, 2222]$
	2500	$[\mathbf{4536}, \mathbf{8335}]$#	6149	$[4084, 8812]$
	3000	$[\mathbf{1.212 \times 10^4}, \mathbf{2.694 \times 10^4}]$*	1.84×10^4	$[1.332 \times 10^4, 2.669 \times 10^4]$
	3500	$[\mathbf{3.001 \times 10^4}, \mathbf{7.925 \times 10^4}]$#	3.246×10^4	$[2.581 \times 10^4, 5.038 \times 10^4]$
	4000	$[\mathbf{6.95 \times 10^4}, \mathbf{2.163 \times 10^5}]$*	1.312×10^5	$[7.073 \times 10^4, 2.024 \times 10^5]$
	4500	$[\mathbf{1.528 \times 10^5}, \mathbf{5.563 \times 10^5}]$*	2.633×10^5	$[1.73 \times 10^5, 4.419 \times 10^5]$

Solver	n	Predicted confidence intervals Poly. model	Observed median run-time Point estimates	Confidence intervals
Concorde	2000	$[\mathbf{1228}, \mathbf{1795}]$	1969	$[1739, 2222]$
	2500	$[\mathbf{2737}, \mathbf{4771}]$	6149	$[4084, 8812]$
	3000	$[5252, 1.057 \times 10^4]$	1.84×10^4	$[1.332 \times 10^4, 2.669 \times 10^4]$
	3500	$[9149, 2.069 \times 10^4]$	3.246×10^4	$[2.581 \times 10^4, 5.038 \times 10^4]$
	4000	$[1.477 \times 10^4, 3.708 \times 10^4]$	1.312×10^5	$[7.073 \times 10^4, 2.024 \times 10^5]$
	4500	$[2.248 \times 10^4, 6.205 \times 10^4]$	2.633×10^5	$[1.73 \times 10^5, 4.419 \times 10^5]$

Table 6: Example output of ESA – bootstrap confidence intervals for observed and predicted running times for support data.
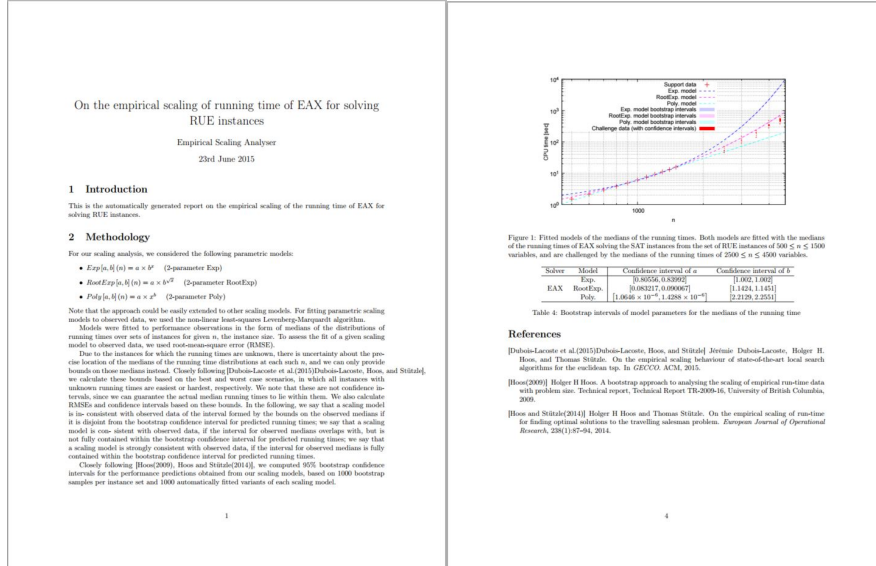

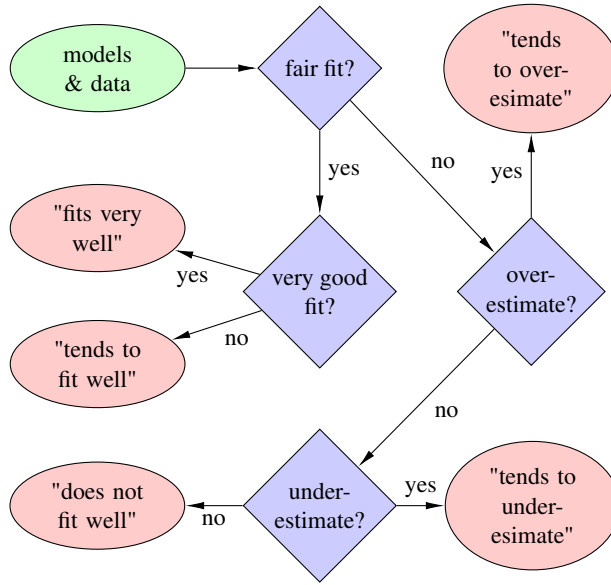Figure 6: Snapshot of the technical report generated by ESA.

Figure 7: Flow diagram on how ESA automatically interprets the fitting results. Detailed definitions of the conditions are given in the main text.

information regarding whether the model predicts well. The detailed criteria, which we design based on extensive experiments with SAT and TSP algorithms, are as follows:

- fair fit: the model predicts well for a fair percentage of the challenge sizes, more precisely, $> 70\%$ of the challenge points or $> 70\%$ of the larger half of the challenge points are within the predicted bootstrap intervals;

- very good fit: the model predicts well for almost all challenge sizes, more precisely, $> 95\%$ of the challenge points are within the predicted bootstrap intervals;

- over-/under-estimate: the model over-/under-estimates the running times of a significant percentage of the challenge sizes, more precisely, $> 75\%$ of the challenge points or $> 75\%$ of the larger half of the challenge points are below/above the predicted bootstrap intervals.

These criteria are combined into the fully automated interpretation procedure illustrated in Figure 7. Note that when medians (or other quantiles) are not definitely known (due to instances with unknown running times), we compare the intervals of the medians against the predicted bootstrap intervals. To be more precise, for instance, we say a challenge point is below the predicted bootstrap interval, if the upper bound of the median is smaller than the lower bound of the bootstrap interval.

4   Implementation

ESA is implemented in python 2.7 and calls gnuplot to generate plots. All provided gnuplot scripts are prepared for gnuplot version 4.6, but only minimal modifications, if any, will be needed to use with another gnuplot version. The online service has a clear user interface implemented with HTML/CSS, and the back-end service was realised using python CGI.

ESA was designed to work with customisable LaTeX and gnuplot templates (see Sec. 1 for details). There is a special LaTeX syntax that can be used as "variables" in the LaTeX template, which will be replaced by actual content when ESA runs. The LaTeX template can also be compiled "as-is"to make it easier for users to adapt it to their specific needs. We chose gnuplot for plot generation so that users can easily supply a template for customised figure formatting. ESA also supports user-defined models for scaling analysis. To achieve this, ESA defines functions on-the-fly from user-supplied strings.

5   Downloading and Running ESA

5.1   Downloading and Running ESA from the Command Line

ESA can be downloaded from the project page online at `www.cs.ubc.ca/labs/ beta/Projects/ESA/`. After unzipping the compressed file, there will be a directory named ESA, which contains runESA.sh, the source code and other support files. A user needs to have python and gnuplot installed in order to run ESA. We used python 2.7 and gnuplot 4.6 in our environment, but expect ESA to work for other versions with minimal modifications, if any.

To run ESA from command line, a user should follow the following steps:

1. Create a directory for input and output files.

2. Put the primary input file for running time data into the directory.

3. Create files for models and LaTeX and gnuplot templates (optional; if not provided, ESA will use the default files distributed with the source code).

4. Create a configuration file named configurations.txt within the directory, telling ESA the details it requires, including names of the algorithm and the instance set/distribution, file names of running time data and, if used, the file names of model specifications and LaTeX and gnuplot templates.

5. Run the script in the ESA directory by `./runESA.sh <directory name>`, and ESA will run according to the specifications in <directory name>/configurations.txt.

5.2   Running ESA as a Web Service

ESA is also available online as a web service, which supports the essential but not all features of the command-line version. In particular:

Empirical Scaling Analyser (ESA)

Introduction

Empirical Scaling Analyser (ESA) is a tool that takes a file of solver runtimes (sample) as input, automatically fits and evaluates parametric models, and generate a PDF report for the analysis results (sample). The models are fitted using standard numerical methods, and are challenged by extrapolation using a bootstrap approach. For detailed information about the methodology, please refer to the papers below.

Upload your runtime file below to give it a try!

The On-line Tool

Runtime file: [   ] Browse...

Models to fit: ☑ exponential: a*b^n
☐ square-root exponential: a*b^sqrt(n)
☑ polynomial: a*n^b

Algorithm name: WalkSAT/SKC

Instance name: phase-transition random 3-SAT

support sizes: 0   A value of 0 instructs ESA to use ~60% of the data as support

bootstrap samples: 1000

Statistic: median ▼

Upload and Run!

For the default settings and the sample runtime file, ESA should complete within 10 minutes. The time required is roughly linear in the number of bootstrap samples, and the time required to fit the models is the most important factor affecting ESA's processing time.

Figure 8: The user interface of ESA as a web service.

- it only supports three pre-defined models, include:

    - exponential: $a \cdot b^n$,
    - root-exponential: $a \cdot b^{\sqrt{n}}$, and
    - polynomial: $a \cdot n^b$;

- it uses the default LaTeX and gnuplot templates for report generation;

- it only supports a limited number of statistics, include median, mean, as well as 75th, 90th and 95th percentiles.

To run ESA as a web service, visit `www.cs.ubc.ca/labs/beta/Projects/ESA/esa-online.html`, upload the file of running time data and fill in the other details of the form. After submission, ESA will run in the back and redirect the user to the output technical report. The user interface is shown in Figure 8.

References

[1] Jérémie Dubois-Lacoste, Holger H Hoos, and Thomas Stützle. On the empirical scaling behaviour of state-of-the-art local search algorithms for the Euclidean TSP. In *Proceedings of GECCO 2015*, pages 377–384. ACM, 2015.

[2] Holger H Hoos. A bootstrap approach to analysing the scaling of empirical run-
time data with problem size. Technical report, TR-2009-16, Department of Com-
puter Science, University of British Columbia, 2009.