Efficient Representation of Numerical Optimization Problems for SNARKs

Sebastian Angel, *University of Pennsylvania and Microsoft Research;*
Andrew J. Blumberg, *Columbia University;* Eleftherios Ioannidis
and Jess Woods, *University of Pennsylvania*

https://www.usenix.org/conference/usenixsecurity22/presentation/angel

This artifact appendix is included in the Artifact Appendices to the Proceedings of the 31st USENIX Security Symposium and appends to the paper of the same name that appears in the Proceedings of the 31st USENIX Security Symposium.

August 10–12, 2022 • Boston, MA, USA

978-1-939133-31-1

Open access to the Artifact Appendices to the Proceedings of the 31st USENIX Security Symposium is sponsored by USENIX.

ARTIFACT EVALUATED usenix ASSOCIATION AVAILABLE
ARTIFACT EVALUATED usenix ASSOCIATION FUNCTIONAL
ARTIFACT EVALUATED usenix ASSOCIATION REPRODUCED

A Artifact Appendix

A.1 Abstract

We present our Otti USENIX '22 artifact. It is a docker container that orchestrates the components of Otti to a single interface. To build the docker container and execute the script that reproduces our results, see README.md in our repository eniac/otti. The docker container is composed of 1. the Otti compiler from eniac/otti (Note: Otti was built on top of the Haskell CirC compiler, and later ported to the Rust CirC compiler. Both are included.) 2. The Spartan zkSNARK backend from microsoft/Spartan 3. The compatibility interface between compiler and Spartan in elefthei/spartan-zkinterface. We also fetch their dependencies, which are broadly Haskell, Python, and Rust's build tools, the lpsolve CLI, csdp, scikit-learn, the flatbuffer library, the Z3 model checker and more small, standard libraries in Haskell and Rust.

We also include in our repository representative datasets of linear programming (LP) [1], semi-definite programming (SDP) [3], and the datasets for stochastic-gradient descent (SGD), accessible by installing the PMLB [4] Python library. Our docker container includes scripts to run Otti end-to-end – generate C files from datasets, execute and compile C files to R1CS, and finally prove and verify their correct execution with Spartan.

A.2 Artifact check-list (meta-information)

- **Algorithm:** Otti uses optimization certificates to produce nondetermanistic checkers for zkSNARKS, as detailed in the paper.
- **Compilation:** Otti has a compiler which is included in the container.
- **Transformations:** Otti has transformations from model files for LP, SDP, SGD to C files which are also included as Python scripts.
- **Binary:** Binaries for LP solve [2] for x86_64 UNIX machines are included in the container. As we are not certain regarding compatibility with Apple M1, we would recommend running the container on a x86_64 architecture.
- **Data set:** We use the NETLIB [1], SDPLIB 1.2 [3], and PMLB [4] datasets which are publicly available and relatively small – in the order of a few MB. Representative examples from these datasets are included in the repository and you can refer to our results in the paper for the complete list.
- **Run-time environment:** Docker community edition is required, platform independent.
- **Hardware:** For running large datasets, a computer with > 256GB RAM is required. Small datasets can be run on personal computers.
- **Run-time state:** No
- **Execution:** Execution time varies from small to large datasets and the available memory in the machine. Small ones are really fast and finish in a few minutes but larger ones can take hours.
- **Security, privacy, and ethical concerns:** No
- **Metrics:** Execution time, prover time, verifier time, proof size, number of constraints.
- **Output:** Our result is a total runtime measurement and a "Verification Successful" message that confirms end-to-end execution was proven to the verifier
- **Experiments:** Docker container takes care of setup. Variation should be small (5-10%) in runtimes depending on the machine. Variation in constraint and proof sizes should be 0.
- **How much disk space required (approximately)?:** The docker container requires a substantial amount of disk space, between 20GB-30GB.
- **How much time is needed to prepare workflow (approximately)?:** The docker container builds in about an hour.
- **How much time is needed to complete experiments (approximately)?:** Smaller examples can be run immediately and take a couple of minutes, larger examples must be downloaded, but should not take more than an hour or so.
- **Publicly available (explicitly provide evolving version reference)?:** https://github.com/eniac/otti
- **Code licenses (if publicly available)?:** MIT license
- **Data licenses (if publicly available)?:** [1, 3] are very old and no licensing information was found, [4] is under MIT license.
- **Workflow frameworks used?:** No
- **Archived (explicitly provide DOI or stable reference)?:** https://github.com/eniac/otti/releases/tag/v1.0

A.3 Description

How to access Clone repository from GitHub: https://github.com/eniac/otti/releases/tag/v1.0

Hardware dependencies X86_64 machine with a sufficient amount of RAM memory (> 200GB) if evaluating large datasets.

Software dependencies Docker community, latest version.

Data sets See [1, 3, 4].

Models N/A

Security, privacy, and ethical concerns N/A

A.4 Installation

Cloning To clone the repository and its submodules run `git clone -recursive https://github.com/eniac/otti.git`

Building First, make sure you have installed Docker CE: https://docs.docker.com/get-docker/ Then build the Otti container: `docker build -t otti .` Then run the container with 200GB of memory and get terminal access: `docker run -m 200g -it otti`

Reproducing experimental results After connecting to the Docker container, run the following script to reproduce the experimental results from Otti: `./run.py [-lp | -sdp | -sgd] [-small | -full | -custom datasets/<path to dataset>]`

One of the `-lp | -sdp | -sgd` options is required. Then either execute with the `-small` or `-full` flag, or the `-custom` flag with an explicit path to a dataset file.

Running the small suite A subset of each dataset that can be reproduced on a personal computer with x86_64 architecture and >= 12GB of RAM. These datasets are expected to take less than 1 hour.

Running the full suite A subset of each MPS dataset that can be reproduced on a large machine with x86_64 architecture and > 200GB RAM. These datasets can take several hours, on the order of 2-3 days to terminate. If your computer does not have sufficient RAM memory or more applications have reserved memory, this might be killed by the OS. This is a well-known limitation of the compiler that consumes large amounts of memory.

Running individual files in `datasets/*` Our script will generate a C file from the dataset file including non-deterministic checks. We compile it with the Otti compiler, prove and verify it and print Verification successful and the total runtime. of each stage. Note that running indiviudal SGD datasets not from PLMB is not supported at this time.

A.5 Experiment workflow

Our experiment runs a script around the components of Otti to compile publicly available datasets to zkSNARKS and then verifies them, printing "Verification successful" upon completion. We also output profiling information such as runtime and zkSNARK proof size.

A.6 Evaluation and expected results

In Otti we evaluate the practicality of compiling numerical optimization problems to zkSNARKs. We evaluate Otti in linear programming, semi-definite programming, and stochastic optimization problems. We apply this technique to publicly available datasets [1, 3, 4], and show the following results.

A.6.1 Semidefinite programming results

Dataset	Prover (ms)	Verifier (ms)	Proof (KB)	Solver (ms)	R1CS constraints
truss1	5140	768	79.20	197	3,007,933
hinf1	7166	1209	79.88	215	4,703,942
hinf2	10607	1187	79.88	313	6,536,398
hinf3	7795	1038	79.88	362	6,536,398
hinf4	9008	1211	79.88	193	6,536,398
hinf5	7748	1248	79.88	238	6,536,398
hinf6	7051	912	79.88	294	6,536,398
hinf7	7432	1058	79.88	343	6,536,398
hinf8	7241	1105	79.88	321	6,536,398
hinf9	7546	1153	79.88	301	6,536,398
control1	7398	1069	79.88	181	6,968,254

A.6.2 Linear programming evaluation results

Dataset	Prover (ms)	Verifier (ms)	Proof (KB)	Solver (ms)	R1CS constraints
afiro	318	73	19.82	41	36,811
sc50a	320	78	19.82	42	54,066
sc50b	336	77	19.82	40	55,085
adlittle	609	117	29.33	45	180,747
sc105	473	104	20.51	45	113,282
scagr7	595	111	29.33	47	229,061
israel	1072	128	47.02	56	511,156
agg	2486	511	47.71	56	1,069,523
sc205	665	121	29.33	52	220,520
brandy	1631	227	47.02	61	815,356
beaconfd	2499	337	47.71	56	1,149,169
agg2	2237	313	47.71	79	1,887,762
agg3	2401	383	47.71	71	1,891,690
lotfi	1014	183	30.01	56	326,102
scorpion	1645	208	47.02	62	731,137
sctap1	1007	180	47.71	61	414,101
scfxm1	1831	254	47.02	105	965,504
bandm	2499	467	47.02	103	1,093,340
scagr25	1637	268	47.71	111	823,136
degen2	1534	223	47.71	308	626,407
scsd1	1636	216	47.02	54	1,034,359
fffff800	2431	330	47.71	197	1,479,725
scfxm2	2426	354	47.02	304	1,932,500
scrs8	2512	363	47.71	117	1,601,971
bnl1	4077	558	81.10	236	2,324,544
scsd6	2372	422	47.71	100	1,845,814
modszk1	2449	369	47.71	185	1,805,821
scsd8	4767	567	81.10	477	3,607,188

A.6.3 Stochastic gradient descent results

Dataset	Prover (ms)	Verifier (ms)	Proof (KB)	Solver (ms)	R1CS constraints
confidence	0.117	0.038	14.08	2.35	13,027
haberman	0.215	0.052	19.36	7.84	60,237
iris	0.293	0.076	11.47	4.13	4,730
new_thyroid	0.296	0.058	14.75	2.96	25,810
krkopt	0.997	0.125	29.31	39.70	399,555
diabetes	0.484	0.071	28.64	32.14	212,501
glass	0.104	0.027	11.47	3.14	7,571
labor	0.186	0.047	14.75	3.19	22,763
letter	1.01	0.164	29.31	27.97	374,655
lymphography	0.284	0.055	14.75	4.37	31,823
collins	0.323	0.08	14.75	4.23	31,733
allbp	0.301	0.055	20.03	11.35	103,451
dermatology	0.517	0.106	19.36	6.90	55,877
kddcup	0.904	0.147	28.64	67.80	198,840
molecular_biology_promoters	0.707	0.263	19.36	7.19	41,343
mfeat_karhunen	0.488	0.073	28.64	13.80	162,352
analcatdata_authorship	0.586	0.095	28.64	8.70	231,455
clean1	6.423	0.535	79.20	14.47	3,473,740
clean1 (50%)	4.675	0.607	79.20	14.47	2,262,837
clean2 (50%)	18.234	1.337	79.88	477.17	6,773,944
GE1000 (50%)	4.842	0.356	45.92	310.64	571,558

A.7 Version

Based on the LaTeX template for Artifact Evaluation V20220119.

References

[1] LP/data index. https://ampl.com/netlib/lp/data/, 2013.

[2] lpsolve: Mixed integer linear programming (MILP) solver. http://lpsolve.sourceforge.net/5.5, 2021.

[3] B. Borchers. SDPLIB 1.2, a library of semidefinite programming test problems. *Optimization Methods and Software*, 11(1-4), 1999.

[4] J. D. Romano, T. T. Le, W. La Cava, J. T. Gregg, D. J. Goldberg, P. Chakraborty, N. L. Ray, D. Himmelstein, W. Fu, and J. H. Moore. PMLB v1.0: an open-source dataset collection for benchmarking machine learning methods. *Bioinformatics*, 38(3):878–880, 10 2021.