Supporting Materials for Just-In-Time Reactive Synthesis

Shahar Maoz
Tel Aviv University, Israel

Ilia Shevrin
Tel Aviv University, Israel

ABSTRACT

This document provides supporting materials for the ASE'20 paper by the authors titled "Just-In-Time Reactive Synthesis" [6]. We provide additional evaluation of JITS over several benchmarks, including specifications for the Cinderella-Stepmother game and several arbiter variants, recently used in the SYNTCOMP competition [4].

1 ADDITIONAL EVALUATION

We evaluate JITS using two additional sets of benchmark specifications from the literature. All specifications used in our evaluation, the raw data we collected, and the code to reproduce our experiments, are available in supporting materials [1].

1.1 Additional Corpus of Specifications

1.1.1 Cinderella-Stepmother. We consider specifications of the safety only Cinderella-Stepmother game, which was proposed as a challenge problem for the synthesis community in [2]. The interaction between Cinderella and her stepmother is modeled as a safety game, in which the environment (i.e., the stepmother), fills k water units into n buckets with capacity b, and the system (i.e., Cinderella), must keep the buckets from overflowing by emptying c adjacent buckets of her choice. Bodlaender et al. [3] analyzed for which values of n, c, k, and b Cinderella can win.

We use only realizable configurations of these parameters, specifically ones for which the ratio b/k is smallest for given n and c. For these, we fixed k to be 5 and changed n to get one set of specifications, and fixed n to be 6 and changed k to get another set of specifications.

For this setup, we generated the desired configurations for Slugs via the tool available at [7], and manually wrote the counterpart for Spectra [5].

1.1.2 Arbiter. We consider specifications of an arbiter whose goal is to eventually grant environment requests, subject to varying restrictions on the system or on the environment. This family of specifications was used as a benchmark in the SYNTCOMP competition [4], in TLSF format, for n requests.

The arbiter kinds that we used are described as follows:

- Simple arbiter, with n justice guarantees defined using response patterns.
- Round-robin arbiter, with additional n environment assumptions on each request; that it stays turned on until granted, and that it immediately turns off once granted.
- Prioritized arbiter, with an additional master request, and a guarantee that a master request has to be granted before any other request.
- Full arbiter, with additional n guarantees that there are no spurious grants, i.e., that each grant is turned on only in resonse to a corresponding request.

Table 1: Comparison of construction vs. realizability running times for Slugs and Spectra: Cinderella-Stepmother

Specification				Synthesis Time (sec)			
				Slugs		Spectra	
n	c	b	k	Real.	Constr.	Real.	Constr.
5	2	9	5	0.27	0.01	0.23	0.30
6	2	11	5	1.81	0.13	0.57	0.68
6	2	14	6	1.38	0.07	0.79	1.13
6	2	19	8	4.22	0.28	2.65	2.88
6	2	24	10	38.32	0.91	8.15	8.92
6	2	36	15	283.55	17.27	54.16	49.10
7	2	11	5	39.67	1.11	16.81	16.94
8	2	13	5	39.68	3.21	671.29	1524.76
9	2	13	5	4045.59	38.07	1416.56	11.17

Each such arbiter specification kind (except round-robin) had two variants. One where the requests and grants are modeled using Boolean arrays of size n, and an additional guarantee of mutual exclusion of the grants. Another, which is identified by the suffix "enc", where the requests and grants are modeled using Boolean arrays of size $\log n$. Here, there is only one request and one grant per round, and their indexes are encoded in the corresponding array. Naturally, a guarantee for mutual exclusion is redundant in this setting.

For this setup, first, we manually translated the specification from the original TLSF format to Spectra, and then automatically generated a corresponding Slugs version using a script. For the prioritized and full versions of the arbiter, the translation to Spectra involved generating custom patterns for LTL formulas lying outside the GR(1) fragment. In these cases, we did not generate a corresponding Slugs version.

In most specifications, we did not use the same values for n as were used in the SYNTCOMP competition, but chose higher values in order to benefit from Spectra's performance during realizability checking, and from JITS performance during synthesis and system execution. Specifically, for the simple, round-robin and prioritized arbiters we took 5, 10, 20, and 30, and for the full arbiter we took 4, 6, 8, and 10, as we observed that this kind of artiber specification is synthesized much slower than the rest.

2 RESULTS: REALIZABILITY VS. CONSTRUCTION

Tables 1 and 2 present a comparison between realizability and controller construction times, in seconds, for Spectra and Slugs. The results for the Cinderella-Stepmother specifications show that in the majority of cases, construction time is very close to realizability

Table 2: Comparison of construction vs. realizability running times for Slugs and Spectra: Arbiter

Specification	Synthesis Time (sec)			
	Slugs		Spectra	
	Real.	Constr.	Real.	Constr.
Simple5	0.01	<0.01	<0.01	0.03
Simple10	0.01	0.06	<0.01	0.08
Simple20	0.07	228.42	0.01	0.20
Simple30	0.50	xx	0.02	1.21
SimpleEnc5	0.01	0.01	<0.01	0.07
SimpleEnc10	0.06	0.36	0.01	0.27
SimpleEnc20	1.00	–	0.10	3.52
SimpleEnc30	5.10	xx	0.14	43.08
RoundRobin5	0.02	<0.01	0.01	0.06
RoundRobin10	0.11	0.11	0.04	0.18
RoundRobin20	1.22	634.60	0.23	1.64
RoundRobin30	6.39	xx	0.90	5.40
Prioritized5			0.01	0.05
Prioritized10			0.02	0.10
Prioritized20			0.07	0.40
Prioritized30			0.13	1.30
PrioritizedEnc5			0.01	0.09
PrioritizedEnc10			0.05	0.52
PrioritizedEnc20			0.49	14.80
PrioritizedEnc30			0.69	55.37
Full4			0.05	0.16
Full6			0.63	1.10
Full8			8.53	9.58
Full10			76.27	202.66
FullEnc4			0.12	0.51
FullEnc6			1.53	4.08
FullEnc8			23.88	18.97
FullEnc10			52.64	241.51

time. The results for the Arbiter specifications show a similar pattern. As the array size grows, construction takes considerably more time than realizability checking for both Spectra and Slugs. For the largest arbiter specifications, where $n = 30$, Slugs responded with an out-of-memory error during construction. For the simple arbiter with $n = 20$, Slugs timed out after two hours during construction.

3 RESULTS: SYNTHESIS

Tables 3 and 4 present the results for synthesis for the Cinderella-Stepmother and Arbiter benchmarks.

Cinderella-Stepmother. In contrast to the parametric AMBA and GenBuf specifications, these results do not show a clear monotonic trend as the values of n, b, and k grow.

In terms of construction time, Slugs achieves better results than JITS and Spectra for small values of n and b/k, but JITS eventually surpasses Slugs as these values grow. In terms of active nodes, except a single configuration ($n = 6$ and $b/k = 24/10$), JITS is in all cases better than all other tools. In terms of space on disk, there are several configurations where Slugs' output was smaller in size than JITS' output ($n = 6$ and $b/k = 11/5$; $n = 6$ and $b/k = 24/10$; $n = 6$ and $b/k = 36/10$). In all other cases, JITS' output was smaller. Specifically, JITS shows better scalability as the number of buckets n grows. The advantage of JITS is less evident as the values of bucket capacity b and added water units k grow.

Arbiter. The results show similar patterns to other parametric specifications, such as AMBA and GenBuf. In terms of construction time, the results show that in all arbiter kinds, while Slugs and Spectra grow quite fast with the number of requests n, JITS storing time grows slower. Similarily, in terms of active nodes and space on disk, the results show that in most arbiter kinds, while Slugs and Spectra grow very fast with the size of the specification, JITS growth is much slower.

The only exception is the full non-encoded request arbiter, where Spectra performs better than both other tools for all n values that we have examined. Aside from this case, in absolute terms, only in the smallest specification for each arbiter kind, where $n = 5$, Slugs or Spectra outperformed JITS in any of the measures.

4 RESULTS: SYSTEM EXECUTION

Tables 5 and 6 present the results for system execution on specifications from the Cinderella-Stepmother and Arbiter benchmarks. We do not show results of JITS with bookkeeping for the Cinderella-Stepmother benchmarks since these are safety only specifications and are not affected by justice guarantee manipulations.

Cinderella-Stepmother. In terms of load time, active nodes, and single step time, the results show clear advantage for JITS. In fact, as n, b, and k grow, many of the configurations result in an out-of-memory error during loading in Spectra, while JITS manages to handle them with minor memory overhead. Specifically, the growth in active nodes is more evident as the number of buckets n grows while k remains intact.

Arbiter. The results do not show a clear advantage to any approach. For the encoded request specifications, there is an advantage to JITS w.r.t. load time, memory, and step time. For the non-encoded request specifications, JITS performs better than Spectra as number of master grows w.r.t. load time. The results are inconclusive w.r.t. memory usage and single step time. In most of the cases, JITS performs better without the bookkeeping extension.

REFERENCES

[1] [n.d.]. JITS Supporting Materials Website. http://smlab.cs.tau.ac.il/syntech/jits/.

[2] Tewodros A. Beyene, Swarat Chaudhuri, Corneliu Popeea, and Andrey Rybalchenko. 2014. A constraint-based approach to solving games on infinite graphs. In *The 41st Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '14, San Diego, CA, USA, January 20-21, 2014*, Suresh Jagannathan and Peter Sewell (Eds.). ACM, 221–234. https://doi.org/10.1145/2535838.2535860

[3] Marijke H. L. Bodlaender, Cor A. J. Hurkens, Vincent J. J. Kusters, Frank Staals, Gerhard J. Woeginger, and Hans Zantema. 2012. Cinderella versus the Wicked Stepmother. In *Theoretical Computer Science - 7th IFIP TC 1/WG 2.2 International Conference, TCS 2012, Amsterdam, The Netherlands, September 26-28, 2012. Proceedings (Lecture Notes in Computer Science)*, Jos C. M. Baeten, Thomas Ball, and Frank S. de Boer (Eds.), Vol. 7604. Springer, 57–71. https://doi.org/10.1007/978-3-642-33475-7_5

[4] Swen Jacobs, Roderick Bloem, Maximilien Colange, Peter Faymonville, Bernd Finkbeiner, Ayrat Khalimov, Felix Klein, Michael Luttenberger, Philipp J. Meyer, Thibaud Michaud, Mouhammad Sakr, Salomon Sickert, Leander Tentrup, and Adam Walker. 2019. The 5th Reactive Synthesis Competition (SYNTCOMP 2018):

Table 3: Comparison of Slugs (static), Spectra (static), and JITS: Cinderella-Stepmother during synthesis

Specification				Construction Time (sec)			Memory (# AN in thousands)			Size on Disk (Mb)		
				Slugs	Spectra	JITS	Slugs	Spectra	JITS	Slugs	Spectra	JITS
n	c	b	k	Constr.	Constr.	Stor.						
5	2	9	5	**0.01**	0.30	0.20	8.06	22.63	**2.72**	0.13	0.91	**0.05**
6	2	11	5	**0.13**	0.68	0.15	32.87	47.58	**19.27**	**0.62**	2.02	0.73
6	2	14	6	**0.07**	1.13	0.54	24.77	70.99	**5.31**	0.46	2.87	**0.14**
6	2	19	8	**0.28**	2.88	1.23	61.24	185.11	**7.96**	1.19	7.45	**0.21**
6	2	24	10	**0.91**	8.92	2.01	**137.65**	293.78	216.67	**2.78**	12.12	9.39
6	2	36	15	17.27	49.10	**1.61**	202.15	925.65	**159.07**	**4.14**	38.42	7.19
7	2	11	5	1.11	16.94	**0.32**	146.65	198.47	**31.18**	2.98	7.96	**1.16**
8	2	13	5	3.21	1524.76	**0.81**	210.13	2208.43	**70.89**	4.44	89.43	**2.72**
9	2	13	5	38.07	11.17	**1.27**	1718.87	2595.35	**136.64**	40.52	103.99	**4.85**

Table 4: Comparison of Slugs (static), Spectra (static), and JITS: Arbiter during synthesis

Specification	Construction Time (sec)			Memory (# AN in thousands)			Size on Disk (Mb)		
	Slugs	Spectra	JITS	Slugs	Spectra	JITS	Slugs	Spectra	JITS
	Constr.	Constr.	Stor.						
Simple5	**<0.01**	0.03	0.03	0.51	**0.34**	1.93	0.03	**0.02**	0.06
Simple10	0.06	0.08	**0.04**	1.01	1.95	**1.00**	1.45	0.07	**0.05**
Simple20	228.42	0.20	**0.10**	3.51	7.18	**2.36**	3343.86	0.23	**0.08**
Simple30	xx	1.21	**0.36**	n/a	17.28	**5.32**	n/a	0.55	**0.15**
SimpleEnc5	**0.01**	0.07	0.03	**0.93**	2.12	2.04	**0.07**	0.11	0.10
SimpleEnc10	0.36	0.27	**0.06**	6.52	16.53	**4.21**	5.22	0.59	**0.11**
SimpleEnc20	–	3.52	**0.50**	n/a	131.00	**19.52**	n/a	4.68	**0.32**
SimpleEnc30	xx	43.08	**1.60**	n/a	550.29	**26.19**	n/a	20.87	**0.89**
RoundRobin5	**<0.01**	0.06	0.04	**1.33**	1.83	1.36	**0.03**	0.07	**0.03**
RoundRobin10	0.11	0.18	**0.08**	8.36	8.21	**4.04**	2.22	0.24	**0.07**
RoundRobin20	634.60	1.64	**0.39**	56.39	26.91	**17.80**	6588.44	0.74	**0.51**
RoundRobin30	xx	5.40	**2.52**	n/a	81.13	**42.33**	n/a	1.68	**0.60**
Prioritized5		0.05	**0.04**		1.35	**0.67**		0.06	**0.02**
Prioritized10		0.10	**0.04**		3.08	**1.17**		0.09	**0.02**
Prioritized20		0.40	**0.14**		9.22	**2.96**		0.26	**0.07**
Prioritized30		1.30	**0.50**		24.63	**4.95**		0.82	**0.13**
PrioritizedEnc5		0.09	**0.04**		6.31	**2.11**		0.22	**0.06**
PrioritizedEnc10		0.52	**0.14**		27.84	**6.86**		0.97	**0.14**
PrioritizedEnc20		14.80	**0.51**		296.07	**28.27**		11.64	**0.90**
PrioritizedEnc30		55.37	**2.15**		813.35	**32.43**		31.82	**1.12**
Full4		0.16	**0.09**		**2.78**	3.06		**0.10**	**0.10**
Full6		1.10	**0.62**		**8.99**	17.83		**0.26**	0.74
Full8		9.58	**4.83**		**14.38**	27.74		**0.43**	0.96
Full10		202.66	**91.79**		**52.32**	297.35		**1.74**	11.22
FullEnc4		0.51	**0.10**		18.90	**3.81**		0.73	**0.18**
FullEnc6		4.08	**0.31**		117.83	**14.96**		4.70	**0.67**
FullEnc8		18.97	**2.01**		293.39	**17.33**		11.05	**0.85**
FullEnc10		241.51	**3.04**		1142.17	**37.36**		43.43	**1.88**

Benchmarks, Participants & Results. *CoRR* abs/1904.07736 (2019). arXiv:1904.07736 http://arxiv.org/abs/1904.07736

[5] Shahar Maoz and Jan Oliver Ringert. 2019. Spectra: A Specification Language for Reactive Systems. *CoRR* abs/1904.06668 (2019). arXiv:1904.06668 http://arxiv.org/abs/1904.06668

[6] Shahar Maoz and Ilia Shevrin. 2020. Just-In-Time Reactive Synthesis. In *ASE*. To appear.

[7] WWW 2017. Slugs website. https://github.com/VerifiableRobotics/slugs.

Table 5: Comparison of Spectra (static) and JITS: Cinderella-Stepmother during system execution

				Load Time (sec)		Memory (# AN in thousands)		Single Step Time (ms)	
n	c	b	k	Spectra	JITS	Spectra	JITS	Spectra	JITS
5	2	9	5	2.10	**1.25**	463.75	**149.80**	16.98	**3.73**
6	2	11	5	4.39	**3.03**	2986.17	**272.73**	31.76	**5.23**
6	2	14	6	11.04	**5.46**	2599.80	**298.83**	14.32	**5.64**
6	2	19	8	xx	**2.91**	n/a	**433.03**	n/a	**8.62**
6	2	24	10	xx	**4.65**	n/a	**483.26**	n/a	**9.41**
6	2	36	15	xx	**19.23**	n/a	**554.42**	n/a	**12.31**
7	2	11	5	60.61	**8.94**	12659.06	**412.65**	99.58	**7.64**
8	2	13	5	xx	**19.60**	n/a	**492.48**	n/a	**9.17**
9	2	13	5	xx	**142.11**	n/a	**837.19**	n/a	**14.91**

Table 6: Comparison of Spectra (static) and JITS: Arbiter during system execution

	Load Time (sec)		Memory (# AN in thousands)			Single Step Time (ms)		
Specification	Spectra	JITS	Spectra	JITS	JITS w/ Bk.	Spectra	JITS	JITS w/ Bk.
Simple5	**1.04**	1.15	22.91	**21.41**	28.94	0.74	**0.58**	0.62
Simple10	**1.14**	1.17	**65.15**	122.18	227.96	2.35	**2.23**	2.29
Simple20	1.54	**1.23**	**244.44**	476.36	797.76	**7.86**	7.99	8.43
Simple30	2.43	**1.90**	**468.39**	1326.10	1855.24	**16.58**	18.65	19.96
SimpleEnc5	**1.15**	1.20	29.50	**15.98**	18.88	0.53	**0.42**	0.44
SimpleEnc10	1.86	**1.23**	169.11	**79.91**	131.83	1.20	**0.95**	1.00
SimpleEnc20	17.29	**1.78**	744.54	555.45	**386.95**	2.77	**2.21**	2.28
SimpleEnc30	121.67	**4.77**	1282.57	**383.76**	604.67	4.28	**3.36**	3.68
RoundRobin5	**1.12**	1.13	14.83	**12.21**	14.34	0.61	**0.59**	0.62
RoundRobin10	1.45	**1.39**	63.39	**57.78**	75.16	**1.89**	1.93	2.01
RoundRobin20	**3.04**	4.42	219.74	**211.99**	269.09	**5.81**	6.07	6.21
RoundRobin30	11.85	**18.53**	**442.87**	485.67	611.31	**12.46**	14.49	14.38
Prioritized5	**1.10**	**1.10**	31.31	**25.32**	28.37	1.11	**0.81**	0.84
Prioritized10	1.15	**1.11**	**65.83**	144.52	187.02	2.77	**2.49**	2.52
Prioritized20	1.63	**1.25**	**224.36**	545.59	669.91	**8.27**	8.40	8.88
Prioritized30	3.39	**2.13**	**426.54**	943.18	1132.34	**14.94**	17.17	18.68
PrioritizedEnc5	1.34	**1.15**	47.22	**17.05**	19.06	0.73	**0.48**	0.50
PrioritizedEnc10	2.53	**1.33**	312.46	**95.49**	146.93	1.69	1.18	**1.25**
PrioritizedEnc20	63.56	**3.14**	1504.52	**265.66**	410.18	3.64	**2.53**	2.73
PrioritizedEnc30	277.12	**5.91**	2074.00	**403.26**	614.47	5.87	**3.70**	4.11
Full4	**1.15**	1.28	44.83	**34.97**	46.19	1.44	1.25	**1.17**
Full6	**1.53**	5.69	**86.19**	144.25	197.84	2.77	2.61	**2.60**
Full8	**3.18**	32.44	**148.44**	328.05	445.10	4.37	**4.33**	4.41
Full10	187.38	**98.42**	588.73	**453.30**	634.49	6.72	**6.08**	6.53
FullEnc4	2.32	**1.33**	56.54	**19.18**	23.80	1.21	**0.94**	0.99
FullEnc6	64.96	**2.63**	305.72	**67.88**	98.71	2.06	**1.74**	1.76
FullEnc8	127.10	**3.77**	630.83	357.28	**260.61**	3.55	**2.81**	2.90
FullEnc10	2078.51	**11.38**	2285.84	554.58	**434.37**	5.98	**4.29**	4.47