Proactive and Adaptive Energy-Aware Programming with Mixed Typing — Technical

1. Technical Summary

We provide some notes about reproducing our results, experimental results and raw data, our formal system, and its proof.

2. Extended Evaluation

2.1 Reproducing Experiments

In addition to the methodology presented in the main paper, we remark on a few notes for those wishing to reproduce or continue our experiments.

Our compiler and runtime may be downloaded at the compiler repository[1]. In addition, we provide the full set of our modified benchmarks, as well as all scripts used to run, record, analyze and plot data at the benchmark google drive link[2]. We recommend consulting the companion artifact documentation for examining the benchmarks, located at /doc/artifact.pdf of the compiler repository.

We measured energy consumed for System A benchmarks using jRAPL, which requires a compatible Intel processors (details in the paper). Measuring energy using jRAPL is a straight-forward process and is detailed at jRAPL's homepage [1]; additionally, curious readers may view the System A benchmarks to observe its usage.

We measured the energy consumed for System B and C using the Watts Up? Pro power monitor [2] as neither the Pi nor Android support a RAPL-like interface. We plugged the devices into the power monitor and used a recording library [3] which records the power consumed of the entire device and saves the logs to a local LINUX system. Our repository contains scripts for reading and analyzing these logs. We suggest giving the power monitor roughly 15 - 20 minutes to settle before beginning any recordings, and allowing ample time (roughly 30 seconds) between runs for the system to return to a resting energy state.

2.2 Extended Results

We extend the evaluation discussed in the paper by including the full set of our analytical results, and well as the raw data from our experiments. In all data presented, we shorten `full_throttle` and `energy_saver` to `full` and `saver` respectively.

We present the battery-exception results (E1) for System A, B, and C in Figures 1, 2, and 3 respectively. Here we show the energy difference between the ent and silent `energy_saver` boot mode contexts when accessing `full_throttle` and `managed` workload mode objects. Recall that an `EnergyException` will be thrown under these scenarios, which we respond to by reducing quality of service for the ENT cases.

We present the battery-casing results (E2) for System A, B and C in Figures 4, 5, and 6 respectively. Here we show the energy saved by adjusting quality of service for the `managed` and `energy_saver` boot mode runs against the energy consumed by the `full_throttle` boot mode runs for all input sizes.

We show the raw data collected from the battery-exception experiments in Figures 7, 8, and 9 and battery-casing experiments in Figures 10, 11, and 12. We show the average energy consumed for an individual benchmark run — recall this represents 10 runs — along with the standard deviation.

Lastly, our temperature-casing (E3) runs raw data are contained in the `/dat/tcasing_*temps.dat` files in our benchmark repository due to being too large to fit within the supplemental material.

[1] https://github.com/pl-ent-lang/ent

[2] https://drive.google.com/open?id=0BzP8QC30IDp6WG9OWWd0ME5UQTA

name	workload mode	saver boot silent (J)	saver boot ent (J)	difference (J)	energy saved (%)
sunflow	full	444.37	251.77	192.59	43.34%
sunflow	managed	368.93	226.25	142.68	38.67%
jspider	full	821.71	344.75	476.96	58.05%
jspider	managed	780.56	730.62	49.95	6.4%
crypto	full	814.11	413.39	400.73	49.22%
crypto	managed	408.53	213.25	195.28	47.8%
findbugs	full	13815.0	10368.56	3446.44	24.95%
findbugs	managed	3825.38	2888.03	937.35	24.5%
pagerank	full	3294.4	2742.7	551.7	16.75%
pagerank	managed	1983.14	1625.01	358.13	18.06%
batik	full	10.0	7.76	2.24	22.36%
batik	managed	4.5	4.45	0.05	1.13%

Fig. 1: ENT System A Battery-Exception Results

name	workload mode	saver boot silent (J)	saver boot ent (J)	difference (J)	energy saved (%)
sunflow	full	519.37	274.73	244.64	47.1%
sunflow	managed	431.59	242.43	189.16	43.83%
crypto	full	3151.63	1477.93	1673.7	53.11%
crypto	managed	2098.33	991.49	1106.84	52.75%
camera	full	370.64	380.1	-9.46	-2.55%
camera	managed	376.9	356.27	20.63	5.47%
video	full	458.37	424.79	33.58	7.33%
video	managed	411.29	394.09	17.2	4.18%
javaboy	full	291.33	289.25	2.07	0.71%
javaboy	managed	290.13	286.96	3.16	1.09%

Fig. 2: ENT System B Battery-Exception Results

name	workload mode	saver boot silent (J)	saver boot ent (J)	difference (J)	energy saved (%)
newpipe	full	1895.99	1623.73	272.26	14.36%
newpipe	managed	838.73	724.76	113.97	13.59%
duckduckgo	full	1405.15	1274.0	131.15	9.33%
duckduckgo	managed	940.14	878.8	61.34	6.52%
soundrecorder	full	474.07	458.35	15.72	3.32%
soundrecorder	managed	379.1	370.36	8.74	2.31%
materiallife	full	276.22	232.32	43.9	15.89%
materiallife	managed	254.51	248.69	5.82	2.29%

Fig. 3: ENT System C Battery-Exception Results

name	workload	full boot (J)	managed boot saved (J)	managed boot saved (%)	saver boot saved (J)	saver boot saved (%)
sunflow	full	723.63	277.57	38.36%	472.12	65.24%
sunflow	managed	571.9	203.17	35.52%	344.39	60.22%
sunflow	saver	353.88	100.08	28.28%	172.3	48.69%
jspider	full	1085.37	285.1	26.27%	777.22	71.61%
jspider	managed	785.06	14.71	1.87%	69.11	8.8%
jspider	saver	56.06	22.27	39.73%	23.06	41.14%
crypto	full	1430.48	604.87	42.28%	1008.59	70.51%
crypto	managed	722.37	308.87	42.76%	510.83	70.72%
crypto	saver	360.49	151.79	42.11%	256.8	71.24%
findbugs	full	14087.15	-27.48	-0.2%	3483.28	24.73%
findbugs	managed	3950.41	133.61	3.38%	888.06	22.48%
findbugs	saver	1241.35	20.63	1.66%	252.92	20.37%
pagerank	full	3874.88	605.55	15.63%	1180.68	30.47%
pagerank	managed	2374.41	390.57	16.45%	707.54	29.8%
pagerank	saver	228.3	34.1	14.94%	62.84	27.52%
batik	full	10.76	1.92	17.83%	2.21	20.49%
batik	managed	6.32	1.97	31.26%	2.05	32.48%
batik	saver	2.28	1.2	52.62%	1.43	62.94%

Fig. 4: ENT System A Battery-Casing Results

name	workload	full boot (J)	managed boot saved (J)	managed boot saved (%)	saver boot saved (J)	saver boot saved (%)
sunflow	full	886.34	374.37	42.24%	615.9	69.49%
sunflow	managed	711.57	287.08	40.34%	471.5	66.26%
sunflow	saver	435.44	152.02	34.91%	244.52	56.15%
crypto	full	5696.43	2561.5	44.97%	4238.66	74.41%
crypto	managed	3817.8	1735.17	45.45%	2840.32	74.4%
crypto	saver	1901.03	862.89	45.39%	1413.08	74.33%
camera	full	344.07	10.41	3.03%	21.97	6.39%
camera	managed	331.56	19.92	6.01%	30.57	9.22%
camera	saver	307.44	7.93	2.58%	12.35	4.02%
video	full	386.29	36.32	9.4%	75.83	19.63%
video	managed	319.55	15.18	4.75%	29.32	9.18%
video	saver	308.33	12.98	4.21%	24.29	7.88%
javaboy	full	292.63	1.8	0.62%	3.93	1.34%
javaboy	managed	292.75	1.02	0.35%	3.81	1.3%
javaboy	saver	298.39	2.65	0.89%	8.5	2.85%

Fig. 5: ENT System B Battery-Casing Results

name	workload	full boot (J)	managed boot saved (J)	managed boot saved (%)	saver boot saved (J)	saver boot saved (%)
newpipe	full	1983.65	82.77	4.17%	353.03	17.8%
newpipe	managed	854.36	19.02	2.23%	127.11	14.88%
newpipe	saver	338.47	7.43	2.2%	27.08	8.0%
duckduckgo	full	1464.74	31.82	2.17%	319.59	21.82%
duckduckgo	managed	957.73	28.45	2.97%	188.91	19.72%
duckduckgo	saver	475.31	13.55	2.85%	75.65	15.92%
soundrecorder	full	457.32	11.86	2.59%	39.56	8.65%
soundrecorder	managed	365.34	4.98	1.36%	21.73	5.95%
soundrecorder	saver	274.33	4.95	1.8%	15.42	5.62%
materiallife	full	292.73	19.9	6.8%	69.23	23.65%
materiallife	managed	260.23	10.01	3.85%	44.52	17.11%
materiallife	saver	283.85	42.06	14.82%	74.41	26.21%

Fig. 6: ENT System C Battery-Casing Results

name	workload	full boot (J)	full deviation (J)	silent full boot (J)	silent full deviation (J)
sunflow	full	452.45	3.61	451.88	3.46
sunflow	managed	374.31	2.79	375.88	4.18
sunflow	saver	259.01	1.26	258.25	2.42
jspider	full	781.3	8.33	786.26	20.76
jspider	managed	766.71	11.73	761.03	12.59
jspider	saver	33.71	2.43	33.01	1.7
crypto	full	805.47	7.37	813.29	8.84
crypto	managed	406.22	2.72	410.69	4.97
crypto	saver	200.43	2.09	204.04	3.11
findbugs	full	13890.65	273.78	13796.99	115.82
findbugs	managed	3798.47	94.76	3750.04	52.65
findbugs	saver	1223.92	84.96	1206.47	83.28
pagerank	full	3302.77	22.73	3294.89	36.88
pagerank	managed	1981.25	17.27	1980.22	16.96
pagerank	saver	194.79	1.66	194.67	1.33
batik	full	7.84	3.58	7.85	3.37
batik	managed	4.26	1.1	4.9	1.74
batik	saver	1.1	0.28	1.17	0.36
name	**workload**	**managed boot (J)**	**managed deviation (J)**	**silent managed boot (J)**	**silent managed deviation (J)**
sunflow	full	253.44	2.04	456.2	4.56
sunflow	managed	373.98	4.21	376.25	4.1
sunflow	saver	260.45	3.05	258.31	2.05
jspider	full	321.11	5.55	812.01	8.57
jspider	managed	765.3	9.55	764.29	9.73
jspider	saver	32.84	2.4	33.49	2.1
crypto	full	403.5	7.43	823.58	7.53
crypto	managed	413.97	5.44	402.64	4.78
crypto	saver	204.48	2.24	207.6	2.36
findbugs	full	10230.83	83.99	13768.63	167.02
findbugs	managed	3752.77	41.02	3730.49	35.13
findbugs	saver	1260.91	73.36	1225.2	92.56
pagerank	full	2692.96	7.93	3279.76	14.93
pagerank	managed	2009.02	15.72	1979.69	10.88
pagerank	saver	198.22	2.89	193.89	0.77
batik	full	8.2	2.31	8.58	3.71
batik	managed	4.57	1.26	4.76	1.49
batik	saver	1.16	0.37	1.25	0.49
name	**workload**	**saver boot (J)**	**saver deviation (J)**	**silent saver boot (J)**	**silent saver deviation (J)**
sunflow	full	251.77	1.15	444.37	1.86
sunflow	managed	226.25	1.3	368.93	1.71
sunflow	saver	257.33	3.62	255.27	1.78
jspider	full	344.75	21.3	821.71	8.6
jspider	managed	730.62	11.97	780.56	11.71
jspider	saver	32.75	1.98	33.62	1.82
crypto	full	413.39	5.21	814.11	15.37
crypto	managed	213.25	3.71	408.53	8.52
crypto	saver	207.65	2.36	206.08	2.48
findbugs	full	10368.56	153.2	13815.0	99.7
findbugs	managed	2888.03	55.57	3825.38	50.04
findbugs	saver	1219.77	74.73	1210.75	76.59
pagerank	full	2742.7	16.59	3294.4	12.47
pagerank	managed	1625.01	12.09	1983.14	23.39
pagerank	saver	199.06	1.98	194.95	0.91
batik	full	7.76	2.85	10.0	3.25
batik	managed	4.45	1.5	4.5	1.48
batik	saver	1.14	0.3	1.1	0.21

Fig. 7: ENT System A Battery-Exception Raw Data

name	workload	full boot (J)	full deviation (J)	silent full boot (J)	silent full deviation (J)
sunflow	full	519.57	2.67	510.3	2.25
sunflow	managed	432.23	2.0	420.08	2.86
sunflow	saver	290.21	1.7	298.26	1.73
crypto	full	3120.33	14.86	3134.51	13.64
crypto	managed	2085.0	11.0	2104.02	8.78
crypto	saver	1038.56	6.68	1055.18	5.07
camera	full	372.53	3.31	368.16	2.42
camera	managed	356.86	1.45	370.06	2.54
camera	saver	351.47	1.54	354.62	1.58
video	full	460.82	2.11	459.64	1.56
video	managed	414.97	2.01	410.79	1.91
video	saver	405.03	3.0	400.82	2.69
javaboy	full	287.69	0.57	291.26	0.92
javaboy	managed	287.89	0.61	291.46	1.17
javaboy	saver	306.07	1.16	306.03	0.42
name	**workload**	**managed boot (J)**	**managed deviation (J)**	**silent managed boot (J)**	**silent managed deviation (J)**
sunflow	full	275.57	1.05	513.73	1.75
sunflow	managed	433.23	2.51	427.21	1.61
sunflow	saver	287.21	2.0	288.67	2.12
crypto	full	1467.26	9.81	3141.65	16.49
crypto	managed	2084.73	9.6	2089.72	12.69
crypto	saver	1041.08	5.44	1048.52	7.81
camera	full	372.85	4.37	371.73	3.7
camera	managed	356.6	1.81	373.19	4.43
camera	saver	354.86	1.01	359.9	2.2
video	full	423.08	1.76	459.71	2.33
video	managed	414.29	1.43	412.1	2.31
video	saver	400.17	2.42	398.54	1.15
javaboy	full	289.43	1.16	291.27	0.73
javaboy	managed	287.08	1.03	289.16	0.5
javaboy	saver	305.87	0.27	305.88	0.78
name	**workload**	**saver boot (J)**	**saver deviation (J)**	**silent saver boot (J)**	**silent saver deviation (J)**
sunflow	full	274.73	1.33	519.37	3.18
sunflow	managed	242.43	1.56	431.59	1.9
sunflow	saver	294.15	1.49	288.37	1.85
crypto	full	1477.93	13.29	3151.63	20.49
crypto	managed	991.49	6.21	2098.33	7.02
crypto	saver	1042.72	5.11	1056.78	6.56
camera	full	380.1	6.25	370.64	3.47
camera	managed	356.27	2.4	376.9	3.55
camera	saver	351.4	1.7	358.97	2.59
video	full	424.79	2.43	458.37	2.21
video	managed	394.09	2.47	411.29	1.27
video	saver	401.12	2.13	400.83	4.98
javaboy	full	289.25	1.05	291.33	0.52
javaboy	managed	286.96	0.35	290.13	0.42
javaboy	saver	305.51	0.79	306.54	0.44

Fig. 8: ENT System B Battery-Exception Raw Data

name	workload	full boot (J)	full deviation (J)	silent full boot (J)	silent full deviation (J)
NewPipe	full	1780.82	197.48	1821.91	199.54
NewPipe	managed	863.01	30.52	863.65	35.24
NewPipe	saver	353.75	35.36	365.24	3.81
duckduckgo	full	1472.49	116.73	1396.84	7.81
duckduckgo	managed	948.25	5.59	914.68	23.27
duckduckgo	saver	457.76	6.99	454.29	10.69
SoundRecorder	full	460.68	1.7	472.79	3.36
SoundRecorder	managed	378.16	5.38	382.98	2.59
SoundRecorder	saver	288.17	2.97	288.63	7.0
MaterialLife	full	274.38	1.84	275.76	1.97
MaterialLife	managed	254.28	4.1	253.64	3.65
MaterialLife	saver	252.76	4.33	255.75	9.77
name	**workload**	**managed boot (J)**	**managed deviation (J)**	**silent managed boot (J)**	**silent managed deviation (J)**
NewPipe	full	1608.53	10.06	1920.98	6.82
NewPipe	managed	853.98	3.12	845.2	6.45
NewPipe	saver	363.2	13.29	367.43	4.36
duckduckgo	full	1358.41	12.45	1396.52	14.35
duckduckgo	managed	918.93	18.69	933.49	16.21
duckduckgo	saver	457.15	4.08	450.07	5.16
SoundRecorder	full	442.95	8.12	473.68	1.21
SoundRecorder	managed	372.37	4.59	384.35	1.64
SoundRecorder	saver	281.42	2.18	289.08	1.62
MaterialLife	full	232.75	1.86	275.75	2.0
MaterialLife	managed	250.25	5.07	252.37	3.48
MaterialLife	saver	252.04	3.66	253.78	3.74
name	**workload**	**saver boot (J)**	**saver deviation (J)**	**silent saver boot (J)**	**silent saver deviation (J)**
NewPipe	full	1623.73	3.93	1895.99	21.94
NewPipe	managed	724.76	3.53	838.73	7.34
NewPipe	saver	365.77	3.07	365.62	2.43
duckduckgo	full	1274.0	8.23	1405.15	22.4
duckduckgo	managed	878.8	11.42	940.14	11.19
duckduckgo	saver	465.73	8.69	449.32	8.1
SoundRecorder	full	458.35	2.17	474.07	3.58
SoundRecorder	managed	370.36	1.8	379.1	2.87
SoundRecorder	saver	280.39	1.09	291.08	1.5
MaterialLife	full	232.32	2.79	276.22	1.24
MaterialLife	managed	248.69	10.64	254.51	4.56
MaterialLife	saver	248.65	4.85	253.51	2.9

Fig. 9: ENT System C Battery-Exception Raw Data

name	workload	full boot (J)	deviation	managed boot (J)	deviation	saver boot (J)	deviation
sunflow	full	723.63	2.0	446.06	1.51	251.51	1.16
sunflow	managed	571.9	2.06	368.74	0.93	227.51	1.69
sunflow	saver	353.88	2.03	253.8	1.51	181.58	1.72
jspider	full	1085.37	20.93	800.26	10.65	308.15	5.73
jspider	managed	785.06	11.09	770.36	15.51	715.96	8.83
jspider	saver	56.06	1.36	33.79	2.56	33.0	1.4
crypto	full	1430.48	28.66	825.61	10.23	421.89	7.68
crypto	managed	722.37	3.06	413.5	4.82	211.54	3.15
crypto	saver	360.49	4.55	208.7	7.71	103.69	1.53
findbugs	full	14087.15	146.33	14114.63	95.89	10603.87	79.39
findbugs	managed	3950.41	76.12	3816.8	56.53	3062.35	41.0
findbugs	saver	1241.35	92.08	1220.72	95.0	988.43	94.74
pagerank	full	3874.88	7.85	3269.33	27.7	2694.2	5.54
pagerank	managed	2374.41	20.35	1983.84	16.85	1666.87	28.32
pagerank	saver	228.3	1.36	194.2	2.19	165.46	1.86
batik	full	10.76	2.98	8.84	2.08	8.56	2.97
batik	managed	6.32	1.52	4.34	1.11	4.27	1.78
batik	saver	2.28	0.41	1.08	0.22	0.84	0.19

Fig. 10: ENT System A Battery-Casing Raw Data

name	workload	full boot (J)	deviation	managed boot (J)	deviation	saver boot (J)	deviation
sunflow	full	886.34	3.65	511.97	2.34	270.44	2.07
sunflow	managed	711.57	2.7	424.49	1.63	240.07	1.97
sunflow	saver	435.44	2.35	283.42	1.55	190.92	1.62
crypto	full	5696.43	20.45	3134.93	25.48	1457.77	6.58
crypto	managed	3817.8	16.27	2082.63	11.79	977.48	6.12
crypto	saver	1901.03	10.4	1038.14	6.21	487.95	4.35
camera	full	344.07	2.67	333.66	2.6	322.1	4.61
camera	managed	331.56	4.95	311.64	2.64	300.99	1.81
camera	saver	307.44	1.84	299.51	1.75	295.09	2.65
video	full	386.29	3.42	349.97	1.77	310.46	1.49
video	managed	319.55	0.98	304.37	1.69	290.23	1.15
video	saver	308.33	1.1	295.35	1.57	284.04	1.7
javaboy	full	292.63	1.43	290.83	0.87	288.7	0.65
javaboy	managed	292.75	1.24	291.74	0.98	288.95	0.82
javaboy	saver	298.39	0.45	295.75	0.8	289.89	0.82

Fig. 11: ENT System B Battery-Casing Raw Data

name	workload	full boot (J)	deviation	managed boot (J)	deviation	saver boot (J)	deviation
NewPipe	full	1983.65	131.85	1900.88	2.4	1630.62	6.95
NewPipe	managed	854.36	2.48	835.34	3.07	727.25	3.22
NewPipe	saver	338.47	2.14	331.04	2.69	311.39	1.91
duckduckgo	full	1464.74	11.77	1432.92	9.04	1145.15	5.15
duckduckgo	managed	957.73	9.5	929.28	9.53	768.82	8.01
duckduckgo	saver	475.31	4.48	461.76	4.6	399.66	6.33
SoundRecorder	full	457.32	2.43	445.46	3.97	417.76	5.27
SoundRecorder	managed	365.34	1.29	360.36	2.17	343.61	3.38
SoundRecorder	saver	274.33	2.83	269.38	1.84	258.91	2.61
MaterialLife	full	292.73	5.64	272.83	3.04	223.5	5.16
MaterialLife	managed	260.23	6.41	250.22	4.65	215.71	4.17
MaterialLife	saver	283.85	13.72	241.79	5.61	209.44	3.3

Fig. 12: ENT System C Battery-Casing Raw Data

P	$::=$	$D\ \overline{C}$	*program*
D	$::=$	$\overline{\mathtt{m} \leq \mathtt{m}}$	*mode declaration*
C	$::=$	$\textbf{class}\ \mathtt{c}\ \Delta\ \textbf{extends}\ \mathtt{c}\{\overline{F}\ \overline{M}\ A\ \}$	*class*
F	$::=$	$T\ \mathtt{fd} = e$	*field*
M	$::=$	$T\ \mathtt{md}(\overline{T\ \mathtt{x}})\{e\}$	*method*
A	$::=$	e	*attributor*
e	$::=$	$\mathtt{x} \mid e.\mathtt{fd} \mid \textbf{new}\ \mathtt{c}\langle\iota\rangle \mid e.\mathtt{md}(\overline{e}) \mid (T)e$	*expression*
	$\mid$	$\textbf{snapshot}\ e\ [\eta, \eta] \mid \textbf{mcase}\langle T\rangle\{\overline{\mathtt{m} : e}\} \mid e \triangleright \eta$	
$\mathtt{m}$	$\in$	**MCONST**	*mode name*
$\mathtt{c}$	$\in$	$\textbf{CN} \cup \{\mathtt{Object}, \mathtt{Main}\}$	*class name*
$\mathtt{md}$	$\in$	$\textbf{MN} \cup \{\mathtt{main}\}$	*method name*
$\mathtt{x}$	$\in$	**VAR**	*variable name*
T	$::=$	$\mathtt{c}\langle\iota\rangle \mid \textbf{mcase}\langle T\rangle$	*programmer type*
ι	$::=$	$\overline{\eta} \mid ?, \overline{\eta}$	*object mode parameter list*
η	$::=$	$\mathtt{m} \mid \mathtt{mt} \mid \top \mid \bot$	*static mode*
$\mathtt{mt}$			*mode type variable*
$?$			*dynamic mode type*
ω	$::=$	$\eta \leq \mathtt{mt} \leq \eta'$	*constrained mode*
Δ	$::=$	$? \to \omega, \Omega \mid \Omega$	*class mode parameter list*
Ω	$::=$	$\overline{\omega}$	*constrained mode list*

Fig. 13: Abstract Syntax: Terms and Types

μ	$::=$	$\eta \mid ?$	*mode*
τ	$::=$	$T \mid \exists\omega.\tau \mid \mathtt{modev}$	*type*
Γ	$::=$	$\overline{\mathtt{x} : \tau}$	*typing environment*
$\mathtt{K}$	$::=$	$\overline{\eta \leq \eta'}$	*constraints*

Fig. 14: Type System Elements

$$\text{(WF-Class)}\ \frac{\textbf{class}\ \mathtt{c}\ \Delta\ \textbf{extends}\ \mathtt{c}'\ \cdots \in P \qquad \mathtt{K} \looparrowright \mathtt{cons}(\Delta)'\{\overline{\eta}/\mathtt{param}(\Delta)\}}{\mathtt{K} \vdash_{\mathtt{wft}} \mathtt{c}\langle\overline{\eta}\rangle}$$

$$\text{(WF-ClassDyn)}\ \frac{\begin{array}{c}\textbf{class}\ \mathtt{c}\ ? \to \omega, \Omega\ \textbf{extends}\ \mathtt{c}' \cdots \in P \\ \mathtt{K} \looparrowright \mathtt{cons}(\Omega)\{\overline{\eta}/\mathtt{param}(\Omega)\}\end{array}}{\mathtt{K} \vdash_{\mathtt{wft}} \mathtt{c}\langle ?, \overline{\eta}\rangle}$$

$$\text{(WF-Top)}\ \mathtt{K} \vdash_{\mathtt{wft}} \mathtt{Object}\langle\eta\rangle$$

$$\text{(WF-MCase)}\ \frac{\mathtt{K} \vdash_{\mathtt{wft}} T}{\mathtt{K} \vdash_{\mathtt{wft}} \textbf{mcase}\langle T\rangle}$$

Fig. 15: Type Well-Formedness

$$\text{(WF-Empty)}\ P \vdash_{\mathtt{wfe}} \epsilon$$

$$\text{(WF-ESpec)}\ \frac{P \vdash_{\mathtt{wfe}} \Omega \qquad \eta \leq \eta'}{P \vdash_{\mathtt{wfe}} \Omega, \eta \leq \mathtt{mt} \leq \eta'}$$

$$\text{(WF-TSpec)}\ \frac{P \vdash_{\mathtt{wfe}} \omega, \Omega}{P \vdash_{\mathtt{wfe}} ? \to \omega, \Omega}$$

Fig. 16: Environment Well-Formedness

$$\text{(FD-Object)}\ \mathtt{fields}(\mathtt{Object}\langle\eta\rangle) = \bullet$$

$$\text{(FD-Class)}\ \frac{\begin{array}{c}\textbf{class}\ \mathtt{c}\ \Delta\ \textbf{extends}\ \mathtt{c}'\{\ \overline{T\ \mathtt{fd} = e}\ \overline{M}\ A\} \\ \mathtt{param}(\Delta) = \iota' \qquad \mathtt{fields}(\mathtt{c}'\langle\iota\rangle) = \overline{T_0}\ \overline{\mathtt{fd}_0}\end{array}}{\mathtt{fields}(\mathtt{c}\langle\iota\rangle) = \overline{T_0}\ \overline{\mathtt{fd}_0}, \overline{T\{\iota/\iota'\}}\ \overline{\mathtt{fd}}}$$

$$\text{(MT-Class)}\ \frac{\begin{array}{c}\textbf{class}\ \mathtt{c}\ \Delta\ \textbf{extends}\ \mathtt{c}'\{\overline{F}\ \overline{M}\ A\} \\ T\ \mathtt{md}(\overline{T\ \mathtt{x}})\ \{\ e\ \} \in \overline{M} \qquad \mathtt{param}(\Delta) = \iota'\end{array}}{\mathtt{mtype}(\mathtt{md}, \mathtt{c}\langle\iota\rangle) = \overline{T}\{\iota/\iota'\} \to T\{\iota/\iota'\}}$$

$$\text{(MT-Super)}\ \frac{\textbf{class}\ \mathtt{c}\ \Delta\ \textbf{extends}\ \mathtt{c}'\{\overline{F}\ \overline{M}\ A\} \qquad \mathtt{md} \notin \overline{M}}{\mathtt{mtype}(\mathtt{md}, \mathtt{c}\langle\iota\rangle) = \mathtt{mtype}(\mathtt{md}, \mathtt{c}'\langle\iota\rangle)}$$

$$\text{(MB-Class)}\ \frac{\begin{array}{c}\textbf{class}\ \mathtt{c}\ \Delta\ \textbf{extends}\ \mathtt{c}'\{\overline{F}\ \overline{M}\ A\} \\ \mathtt{param}(\Delta) = \iota' \qquad T\ \mathtt{md}(\overline{T\ \mathtt{x}})\{e\} \in \overline{M}\end{array}}{\mathtt{mbody}(\mathtt{md}, \mathtt{c}\langle\iota\rangle) = \overline{\mathtt{x}}.e\{\iota/\iota'\}}$$

$$\text{(MB-Super)}\ \frac{\textbf{class}\ \mathtt{c}\ \Delta\ \textbf{extends}\ \mathtt{c}'\{\overline{F}\ \overline{M}\ A\} \qquad \mathtt{md} \notin \overline{M}}{\mathtt{mbody}(\mathtt{md}, \mathtt{c}\langle\iota\rangle) = \mathtt{mbody}(\mathtt{md}, \mathtt{c}'\langle\iota\rangle)}$$

$$\text{(AB-Class)}\ \frac{\begin{array}{c}\textbf{class}\ \mathtt{c}\ ? \to \omega, \Omega\ \textbf{extends}\ \mathtt{c}'\{\overline{F}\ \overline{M}\ A\} \\ \mathtt{param}(\Omega) = \iota' \qquad A = e\end{array}}{\mathtt{abody}(\mathtt{c}\langle ?, \iota\rangle) = e\{\iota/\iota'\}}$$

Fig. 17: FJ Functions

$$\text{(T-Program)}\ \frac{D \text{ form a latice} \qquad \overline{C}\ \text{OK}}{D\ \overline{C}\ \text{OK}}$$

$$\text{(T-Class)}\ \frac{\begin{array}{c}\overline{M}\ \text{OK IN}\ \mathtt{c}, \Omega \qquad \overline{F} = \overline{T}\ \overline{\mathtt{fd}} = \overline{e} \\ \emptyset; \mathtt{cons}(\Omega) \vdash \overline{e} : \overline{T} \qquad \textbf{class}\ \mathtt{c}'\ \Omega\ \textbf{extends}\ \mathtt{c}''\{\ldots\}\ \text{FJ OK}\end{array}}{\textbf{class}\ \mathtt{c}\ \Omega\ \textbf{extends}\ \mathtt{c}'\{\overline{F}\ \overline{M}\}\ \text{OK}}$$

$$\text{(T-ClassDyn)}\ \frac{\begin{array}{c}\Delta = ? \to \omega, \Omega \\ \overline{M}\ \text{OK IN}\ \mathtt{c}, \Delta \qquad A\ \text{OK IN}\ \mathtt{c}, \Delta \qquad \overline{F} = \overline{T}\ \overline{\mathtt{fd}} = \overline{e} \\ \emptyset; \mathtt{cons}(\Omega) \vdash \overline{e} : \overline{T} \qquad \textbf{class}\ \mathtt{c}'\ \Delta\ \textbf{extends}\ \mathtt{c}''\{\ldots\}\ \text{FJ OK}\end{array}}{\textbf{class}\ \mathtt{c}\ \Delta\ \textbf{extends}\ \mathtt{c}'\{\overline{F}\ \overline{M}\ A\}\ \text{OK}}$$

$$\text{(T-Attributor)}\ \frac{\begin{array}{c}\iota = \mathtt{param}(\Omega) \qquad \mathtt{K} = \mathtt{cons}(\Omega) \\ A = e \qquad \mathtt{K}; \textbf{this} : \mathtt{c}\langle ?, \iota\rangle \vdash e : \mathtt{modev} \qquad \mathtt{K} \vdash_{\mathtt{wft}} \mathtt{c}\langle ?, \iota\rangle\end{array}}{A\ \text{OK IN}\ \mathtt{c}, \Omega}$$

$$\text{(T-Method)}\ \frac{\begin{array}{c}\iota = \mathtt{param}(\Delta) \qquad \mathtt{K} = \mathtt{cons}(\Delta) \\ \overline{x} : \overline{T}; \textbf{this} : \mathtt{c}\langle\iota\rangle; \mathtt{K} \vdash e : T \qquad \mathtt{K} \vdash_{\mathtt{wft}} \mathtt{c}\langle\iota\rangle\end{array}}{T\ \mathtt{md}(\overline{T}\ \overline{x})\{\ e\ \}\ \text{OK IN}\ \mathtt{c}\ \Delta}$$

Fig. 18: Class Typing

$$\text{(T-Var)}\ \Gamma;\mathtt{K} \vdash \mathtt{x} : \Gamma(\mathtt{x})$$

$$\text{(T-New)}\ \frac{\begin{array}{c}\iota = ?, \iota' \text{ iff } \textbf{class } \mathtt{c}\ \Delta \cdots \in P \text{ and } \mathtt{cmode}(\Delta) = ? \\ \iota \neq ?, \iota' \text{ iff } \textbf{class } \mathtt{c}\ \Delta \cdots \in P \text{ and } \mathtt{cmode}(\Delta) \neq ? \\ \mathtt{K} \looparrowright \mathtt{cons}(\Delta)\end{array}}{\Gamma;\mathtt{K} \vdash\ \textbf{new } \mathtt{c}\langle\iota\rangle : \mathtt{c}\langle\iota\rangle}$$

$$\text{(T-Cast)}\ \frac{\Gamma;\mathtt{K} \vdash e : T'}{\Gamma;\mathtt{K} \vdash (T)e : T}$$

$$\text{(T-Msg)}\ \frac{\begin{array}{c}\Gamma;\mathtt{K} \vdash\ e : \mathtt{c}\langle\iota\rangle \\ \mathtt{mtype}(\mathtt{md}, \mathtt{c}\langle\iota\rangle) = \overline{T} \rightarrow T \qquad \Gamma;\mathtt{K} \vdash \overline{e} : \overline{T} \qquad \Gamma;\mathtt{K} \vdash\ \textbf{this} : T_0 \\ \mathtt{K} \looparrowright \{\mathtt{omode}(\mathtt{c}\langle\iota\rangle) \leq \mathtt{omode}(T_0)\} \qquad \mathtt{omode}(\mathtt{c}\langle\iota\rangle) \neq ?\end{array}}{\Gamma;\mathtt{K} \vdash e.\mathtt{md}(\overline{e}) : T}$$

$$\text{(T-Field)}\ \frac{\Gamma;\mathtt{K} \vdash e : \mathtt{c}\langle\iota\rangle \qquad \mathtt{fields}(\mathtt{c}\langle\iota\rangle) = \overline{T}\ \overline{\mathtt{fd}}}{\Gamma;\mathtt{K} \vdash e.\mathtt{fd}_i : T_i}$$

$$\text{(T-Snapshot)}\ \frac{\Gamma;\mathtt{K} \vdash e : \mathtt{c}\langle ?, \iota\rangle \qquad \omega = \eta_1 \leq \mathtt{mt} \leq \eta_2}{\Gamma;\mathtt{K} \vdash \textbf{snapshot } e\,[\eta_1, \eta_2] : \exists\omega.\mathtt{c}\langle\mathtt{mt}, \iota\rangle}$$

$$\text{(T-MCase)}\ \frac{\overline{\mathtt{m}} = \mathtt{modes}(P) \qquad \Gamma;\mathtt{K} \vdash e_i : T \text{ for all } i}{\Gamma;\mathtt{K} \vdash \textbf{mcase}\langle T\rangle\{\overline{\mathtt{m} : e}\} : \textbf{mcase}\langle T\rangle}$$

$$\text{(T-ElimCase)}\ \frac{\Gamma;\mathtt{K} \vdash e : \textbf{mcase}\langle T\rangle \qquad \eta \in \mathtt{modes}(P) \text{ or } \eta \text{ appears in } \mathtt{K}}{\Gamma;\mathtt{K} \vdash e \rhd \eta : T}$$

$$\text{(T-ModeValue)}\ \frac{\mathtt{m} \in \mathtt{modes}(P)}{\Gamma;\mathtt{K} \vdash \mathtt{m} : \mathtt{modev}}$$

$$\text{(T-Sub)}\ \frac{\Gamma;\mathtt{K} \vdash e : \tau \qquad \mathtt{K} \vdash \tau <: \tau'}{\Gamma;\mathtt{K} \vdash e : \tau'}$$

Fig. 19: Expression Typing

$$\text{(S-Mcase)}\ \frac{\mathtt{K} \vdash \tau <: \tau'}{\mathtt{K} \vdash \textbf{mcase}\langle\tau\rangle <: \textbf{mcase}\langle\tau'\rangle}$$

$$\text{(S-ExistOpen)}\ \frac{\begin{array}{c}\omega = \eta_1 \leq \mathtt{mt} \leq \eta_2 \qquad \mathtt{K}' = \mathtt{K} \cup \{\eta_1 \leq \mathtt{mt}, \mathtt{mt} \leq \eta_2\} \\ \mathtt{mt} \notin \mathtt{K} \qquad \mathtt{K}' \vdash \tau <: \tau'\end{array}}{\mathtt{K} \vdash \exists\omega.\tau <: \tau'}$$

$$\text{(S-ExistAbstract)}\ \frac{\begin{array}{c}\omega = \eta_1 \leq \mathtt{mt} \leq \eta_2 \\ \mathtt{omode}(\tau) = \eta \qquad \mathtt{K} \vdash \tau <: \tau'\{\eta/\mathtt{mt}\}\end{array}}{\mathtt{K} \vdash \tau <: \exists\omega.\tau'}$$

$$\text{(S-Class)}\ \frac{\textbf{class } \mathtt{c}\ \Delta\ \textbf{extends } \mathtt{c}' \cdots \in P \qquad \mathtt{K} \looparrowright \mathtt{cons}(\Delta)}{\mathtt{K} \vdash \mathtt{c}\langle\iota\rangle <: \mathtt{c}'\langle\iota\rangle}$$

Fig. 20: Subtyping (reflexivity and transitivity rules are omitted.)

$$\text{(M-Sub)}\ \frac{\{\eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta', \eta \leq \eta'\} \in \mathtt{K}}{\mathtt{K} \looparrowright \{\eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta', \eta \leq \eta'\}}$$

Fig. 21: Submoding

$$\begin{array}{lcll}
e & ::= & \ldots & \textit{runtime expressions} \\
 & | & \textbf{check}(e, \mathtt{m}, \mathtt{m}', e) & \\
 & | & \mathtt{obj}(\alpha, \mathtt{c}\langle\iota\rangle, \overline{e}) & \\
 & | & \mathtt{cl}(\mathtt{m}, e) & \\
\mathbf{E} & ::= & \odot & \textit{evaluation context} \\
 & | & \mathbf{E}.\mathtt{md}(\overline{e}) & \\
 & | & o.\mathtt{md}(\ldots, o, \mathbf{E}, e, \ldots) & \\
 & | & (T)\mathbf{E} \mid \mathbf{E}.\mathtt{fd} & \\
 & | & \textbf{snapshot } \mathbf{E}\,[\mathtt{m}_1, \mathtt{m}_2] & \\
 & | & \{\ldots \mathtt{m} : v; \mathtt{m} : \mathbf{E}; \mathtt{m} : e \ldots\} \mid \mathbf{E} \rhd \mu & \\
 & | & \textbf{check}(\mathbf{E}, \mathtt{m}_1, \mathtt{m}_2, e) & \\
 & | & \textbf{check}(\mathtt{m}', \mathtt{m}_1, \mathtt{m}_2, \mathbf{E}) & \\
 & | & \mathtt{obj}(\alpha, \mathtt{c}\langle\iota\rangle, \ldots v, \mathbf{E}, e \ldots) &
\end{array}$$

Fig. 22: Run-Time Elements

$$\text{(T-Obj)}\ \frac{\Gamma;\mathtt{K} \vdash \overline{e} : \overline{T} \qquad \mathtt{fields}(\mathtt{c}\langle\iota\rangle) = \overline{T}\ \overline{\mathtt{fd}}}{\Gamma;\mathtt{K} \vdash \mathtt{obj}(\alpha, \mathtt{c}\langle\iota\rangle, \overline{e}) : \mathtt{c}\langle\iota\rangle}$$

$$\text{(T-Check)}\ \frac{\Gamma;\mathtt{K} \vdash e_1 : \mathtt{modev} \qquad \mathtt{mt} \text{ fresh} \qquad \Gamma;\mathtt{K} \vdash e_2 : \mathtt{c}\langle ?, \iota\rangle}{\Gamma;\mathtt{K} \vdash \textbf{check}(e_1, \mathtt{m}_1, \mathtt{m}_2, e_2) : \exists\mathtt{m}_1 \leq \mathtt{mt} \leq \mathtt{m}_2.\mathtt{c}\langle\mathtt{mt}, \iota\rangle}$$

$$\text{(T-Closure)}\ \frac{\Gamma, \textbf{this} : T; \mathtt{K} \vdash e : \tau \qquad \mathtt{omode}(T) = \mathtt{m}}{\Gamma;\mathtt{K} \vdash \mathtt{cl}(\mathtt{m}, e) : \tau}$$

Fig. 23: Auxiliary Run-time Expression Typing

(R-New)	**new** $\texttt{c}\langle\iota\rangle$	$\xRightarrow{\texttt{m}}$	$\texttt{obj}(\alpha, \texttt{c}\langle\iota\rangle, \texttt{init}(P, \texttt{c}))$	if α fresh
(R-Cast)	$(\tau_0)\,o$	$\xRightarrow{\texttt{m}}$	o	if $\emptyset \vdash \tau <: \tau_0$
(R-Msg)	$o.\texttt{md}(\overline{v}')$	$\xRightarrow{\texttt{m}}$	$\texttt{cl}(\mu, e\{\overline{v}'/\overline{\texttt{x}}\}\{o/\textbf{this}\})$	if $\emptyset \looparrowright \mu \leq \texttt{m}$
(R-Field)	$o.\texttt{fd}_i$	$\xRightarrow{\texttt{m}}$	v_i	
(R-Snapshot)	**snapshot** $o\ [\texttt{m}_1, \texttt{m}_2]$	$\xRightarrow{\texttt{m}}$	$\textbf{check}(\texttt{abody}(T)\{o/\textbf{this}\}, \texttt{m}_1, \texttt{m}_2, o)$	if $\mu = ?$
(R-Check)	$\textbf{check}(\texttt{m}', \texttt{m}_1, \texttt{m}_2, o)$	$\xRightarrow{\texttt{m}}$	$\texttt{obj}(\alpha', \texttt{c}\langle\texttt{m}', \iota\rangle, \overline{v})$	if $\emptyset \looparrowright \{\texttt{m}_1 \leq \texttt{m}', \texttt{m}' \leq \texttt{m}_2\}$, α' fresh
(R-McaseProj)	$\textbf{mcase}\langle T'\rangle\{\overline{\texttt{m} : v}\} \rhd \texttt{m}_j$	$\xRightarrow{\texttt{m}}$	v_j	
(R-Closure1)	$\texttt{cl}(\texttt{m}', e)$	$\xRightarrow{\texttt{m}}$	$\texttt{cl}(\texttt{m}', e')$	if $e \xRightarrow{\texttt{m}'} e'$
(R-Closure2)	$\texttt{cl}(\texttt{m}', v)$	$\xRightarrow{\texttt{m}}$	v	
(R-Context)	$\mathbf{E}[\,e\,]$	$\xRightarrow{\texttt{m}}$	$\mathbf{E}[\,e'\,]$	if $e \xRightarrow{\texttt{m}} e'$

For all rules: $o = \texttt{obj}(\alpha, \texttt{c}\langle\mu, \iota\rangle, \overline{v}), \texttt{mbody}(\texttt{md}, \texttt{c}\langle\mu, \iota\rangle) = \overline{\texttt{x}}.e$.

Fig. 24: Reduction Rules

$\texttt{modes}(P)$	$\triangleq$	$\overline{\texttt{m} \leq \texttt{m}'}$	
$\texttt{omode}(\texttt{c}\langle\overline{\iota}\rangle)$	$\triangleq$	μ	if $\iota = \mu, \overline{\eta}$
$\texttt{param}(\overline{\eta \leq \texttt{mt} \leq \eta'})$	$\triangleq$	$\overline{\texttt{mt}}$	
$\texttt{param}(? \to \omega, \Omega)$	$\triangleq$	$\texttt{mt}, \texttt{param}(\Omega)$	if $\omega = \eta \leq \texttt{mt} \leq \eta'$
$\texttt{cmode}(\Omega)$	$\triangleq$	μ	if $\texttt{param}(\Omega) = \mu, \texttt{mt}$
$\texttt{init}(P, \texttt{c}\langle\iota\rangle)$	$\triangleq$	$\texttt{init}(\texttt{c}'\langle\iota\rangle) \cup \overline{e\{\iota/\texttt{eparam}(\Delta)\}}$	if **class** Δ $\texttt{c}$ **extends** $\texttt{c}'$ $\overline{T\ \texttt{fd} = e} \in P$
$\texttt{init}(P, \texttt{c}\langle\iota\rangle)$	$\triangleq$	ϵ	if $\texttt{c} = \texttt{Object}$
$\texttt{boot}(P)$	$\triangleq$	$\texttt{cl}(\top, \texttt{mbody}(\texttt{main}, \texttt{Main}\langle\top\rangle))$	
$\texttt{cons}(\eta \leq \texttt{mt} \leq \eta')$	$\triangleq$	$\bigcup\{\eta \leq \texttt{mt}, \texttt{mt} \leq \eta'\}$	
$\texttt{cons}(? \to \omega, \Omega)$	$\triangleq$	$\{\eta \leq \texttt{mt}, \texttt{mt} \leq \eta'\} \cup \texttt{cons}(\Omega)$	if $\omega = \eta \leq \texttt{mt} \leq \eta'$

We require $\overline{\texttt{m}}$ as a lattice. We use $\bot$ and $\top$ to represent the bottom and top of $\overline{\texttt{m}}$ respectively.
We define $\texttt{init}(P, \texttt{c})$ as $\texttt{init}(P, \texttt{c}') \cup \overline{e}$ if **class** $\texttt{c}$ **extends** $\texttt{c}'$ $\overline{\tau\ \texttt{fd} = e} \in P$ or ϵ if $\texttt{c} = \texttt{Object}$.

Fig. 25: Compile Functions

$\texttt{emode}(\texttt{m})$	$\triangleq$	$\texttt{m}$
$\texttt{emode}(\texttt{obj}(\texttt{c}\langle\iota\rangle, \overline{v},))$	$\triangleq$	$\texttt{omode}(\texttt{c}\langle\iota\rangle)$

Fig. 26: Runtime Functions

3. Proof

Lemma 1 (Weakening).

(1) If $\mathtt{K} \vdash_{\mathtt{wft}} \tau$ *and* $\mathtt{K} \looparrowright \{\eta \leq \eta'\}$ *then* $\mathtt{K}, \eta \leq \eta' \vdash_{\mathtt{wft}} \tau$.
(2) If $\mathtt{K} \vdash \tau <: \tau'$ *and* $\mathtt{K} \looparrowright \{\eta \leq \eta'\}$ *then* $\Gamma; \mathtt{K}, \eta \leq \eta' \vdash \tau <: \tau'$.
(3) If $\Gamma; \mathtt{K} \vdash e : \tau$, *and* $\mathtt{K} \looparrowright \{\eta \leq \eta'\}$, *then* $\Gamma; \mathtt{K}, \eta \leq \eta' \vdash e : \tau$.
(4) If $\Gamma; \mathtt{K} \vdash e : \tau$, *and* $\Gamma \vdash \mathtt{y} : \tau'$, *then* $\Gamma, \mathtt{y} : \tau'; \mathtt{K} \vdash e : \tau$.

Proof Each is proved by straightforward induction on the derivations of $\mathtt{K} \vdash_{\mathtt{wft}} \tau$, $\mathtt{K} \vdash \tau <: \tau'$, and $\Gamma; \mathtt{K} \vdash e : \tau$. ■

Lemma 2. *If* $\mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' \looparrowright \{\eta_2 \leq \eta_2'\}$, $\mathtt{K} \looparrowright \{\eta \leq \eta'', \eta'' \leq \eta'\}$, *and* $\mathtt{mt} \notin \mathtt{K}$, *then* $\mathtt{K}\{\eta''/\mathtt{mt}\} \looparrowright \{\eta_2\{\eta''/\mathtt{mt}\} \leq \eta_2'\{\eta''/\mathtt{mt}\}\}$.

Proof Trivial. ■

Lemma 3 (Mode Substitution Perserves Type Well-Formedness). *If* $\mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' \vdash_{\mathtt{wft}} \tau$, $\mathtt{K} \looparrowright \{\eta \leq \eta'', \eta'' \leq \eta'\}$, *and* $\mathtt{mt} \notin \mathtt{K}$, *then* $\mathtt{K}\{\eta''/\mathtt{mt}\} \vdash_{\mathtt{wft}} \tau\{\eta''/\mathtt{mt}\}$.

Proof By induction on the derivation of $\mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' \vdash_{\mathtt{wft}} \tau$.

Case WF-Top, WF-Mcase Trivial.

Case WF-Class
$\tau = \mathtt{c}\langle\overline{\eta}\rangle$ **class** $\mathtt{c}\ \Delta$ **extends** $\mathtt{c}' \cdots \in P$
$\mathtt{param}(\Delta) = \iota'$ $\mathtt{cons}(\Delta) = \mathtt{K}'$
$\mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' \looparrowright \mathtt{K}'\{\overline{\eta}/\iota'\}$

Lemma 2 gives us $\mathtt{K}\{\eta''/\mathtt{mt}\} \looparrowright \mathtt{K}'\{\overline{\eta}/\iota'\}\{\eta''/\mathtt{mt}\}$. Then, by WF-Class, $\mathtt{K}\{\eta''/\mathtt{mt}\} \vdash_{\mathtt{wft}} \mathtt{c}\langle\overline{\eta}\rangle\{\eta''/\mathtt{mt}\}$.

Case WF-ClassDyn Similar.

■

Lemma 4 (Mode Substitution Perserves Subtyping). *If* $\mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' \vdash \tau <: \tau'$, $\mathtt{K} \looparrowright \{\eta \leq \eta'', \eta'' \leq \eta'\}$, *and* $\mathtt{mt} \notin \mathtt{K}$, *then* $\mathtt{K}\{\eta''/\mathtt{mt}\} \vdash \tau\{\eta''/\mathtt{mt}\} <: \tau'\{\eta''/\mathtt{mt}\}$.

Proof Induction on the derivation of $\mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' \vdash \tau <: \tau'$.

Case S-Mcase Easy.

Case S-ExistsOpen
$\tau = \exists\omega.\tau_1$ $\tau' = \tau_1'$
$\omega = \eta_1 \leq \mathtt{mt}_1 \leq \eta_2$ $\mathtt{mt}_1 \notin \mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta'$
$\mathtt{K}' = \mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' \cup \{\eta_1 \leq \mathtt{mt}_1, \mathtt{mt}_1 \leq \eta_2\}$
$\mathtt{K}' \vdash \tau_1 <: \tau_1'$

Since $\mathtt{mt}_1 \notin \mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta'$ we trivially have $\mathtt{mt}_1 \notin \mathtt{K}\{\eta''/\mathtt{mt}\}$. Let η_1' and η_2' stand for $\eta_1\{\eta''/\mathtt{mt}\}$ and $\eta_2\{\eta''/\mathtt{mt}\}$ resp. We have $\mathtt{K}'\{\eta''/\mathtt{mt}\} = \mathtt{K}\{\eta''/\mathtt{mt}\} \cup \{\eta_1' \leq \mathtt{mt}_1, \mathtt{mt}_1 \leq \eta_2'\}$ with $\omega\{\eta''/\mathtt{mt}\} = \eta_1' \leq \mathtt{mt}_1 \leq \eta_2'$. By the induction hypothesis,

$$\mathtt{K}'\{\eta''/\mathtt{mt}\} \vdash \tau_1\{\eta''/\mathtt{mt}\} <: \tau_1'\{\eta''/\mathtt{mt}\}.$$

Let ω' stand for $\eta_1' \leq \mathtt{mt}_1 \leq \eta_2'$. Then, by S-ExistOpen, we have

$$\mathtt{K}\{\eta''/\mathtt{mt}\} \vdash \exists\omega.\tau_1\{\eta''/\mathtt{mt}\} <: \tau_1\{\eta''/\mathtt{mt}\}.$$

Case S-ExistsAbstract
$\tau = \tau_1$ $\tau' = \exists\omega.\tau_1'$
$\omega = \eta_1 \leq \mathtt{mt}_1 \leq \eta_2$ $\mathtt{omode}(\tau_1) = \eta_\tau$
$\mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' \vdash \tau_1 <: \tau_1'\{\eta_\tau/\mathtt{mt}_1\}$

$\mathtt{omode}(\tau_1\{\eta''/\mathtt{mt}\}) = \eta_\tau\{\eta''/\mathtt{mt}\}$ is immediate. Let η_1' and η_2' stand for $\eta_1\{\eta''/\mathtt{mt}\}$ and $\eta_2\{\eta''/\mathtt{mt}\}$ resp. We may assume $\mathtt{mt} \neq \mathtt{mt}_1$ since $\mathtt{mt}_1$ is bound by ω.

By the inductive hypothesis,

$$\mathtt{K}\{\eta''/\mathtt{mt}\} \vdash \tau_1\{\eta''/\mathtt{mt}\} <: \tau_1'\{\eta_\tau\{\eta''/\mathtt{mt}\}/\mathtt{mt}_1\}.$$

Let ω' stand for $\eta_1' \leq \mathtt{mt}_1 \leq \eta_2'$. Then, by S-ExistAbstract, we have

$$\mathtt{K}\{\eta''/\mathtt{mt}\} \vdash \tau_1\{\eta''/\mathtt{mt}\} <: \exists\omega'.\tau_1'\{\eta''/\mathtt{mt}\}.$$

Case S-Class
$\tau = \mathtt{c}\langle\iota\rangle$ $\tau' = \mathtt{c}'\langle\iota\rangle$
class $\mathtt{c}\ \Delta$ **extends** $\mathtt{c}' \ \ldots \in P$
$\mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' = \mathtt{cons}(\Delta)$
$\mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' \vdash_{\mathtt{wft}} \mathtt{c}'\langle\iota\rangle$

By Lemma 3 we have $\mathtt{K}\{\eta''/\mathtt{mt}\} \vdash_{\mathtt{wft}} \mathtt{c}\langle\iota\{\eta''/\mathtt{mt}\}\rangle$ and $\mathtt{K}\{\eta''/\mathtt{mt}\} \vdash_{\mathtt{wft}} \mathtt{c}'\langle\iota\{\eta''/\mathtt{mt}\}\rangle$. Lemma 2 gives $\mathtt{K}\{\eta''/\mathtt{mt}\} \looparrowright \mathtt{cons}(\Delta)\{\eta''/\mathtt{mt}\}$.

Then, by S-Class, $\mathtt{K}\{\eta''/\mathtt{mt}\} \vdash \mathtt{c}\langle\iota\{\eta''/\mathtt{mt}\}\rangle <: \mathtt{c}'\langle\iota\{\eta''/\mathtt{mt}\}\rangle$.

■

Lemma 5. *If* $\mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' \vdash_{\mathtt{wft}} \mathtt{c}\langle\iota\rangle$, $\mathtt{K} \looparrowright \{\eta \leq \eta'', \eta'' \leq \eta'\}$, $\mathtt{mt} \notin \mathtt{K}$, *and* $\mathtt{mtype}(\mathtt{md}, \mathtt{c}\langle\iota\rangle) = \overline{T} \to T$ *then* $\mathtt{mtype}(\mathtt{md}, \mathtt{c}\langle\iota\rangle\{\eta''/\mathtt{mt}\}) = \overline{T\{\eta''/\mathtt{mt}\}} \to T\{\eta''/\mathtt{mt}\}$.

Proof Easy induction on the derivation of $\mathtt{mtype}(\mathtt{md}, \mathtt{c}\langle\iota\rangle) = \overline{T} \to T$.

Case MT-Class
$\mathtt{mtype}(\mathtt{md}, \mathtt{c}\langle\iota\rangle) = \overline{T_0}\{\iota/\iota'\} \to T_0\{\iota/\iota'\}$
$\overline{T} = \overline{T_0}\{\iota/\iota'\}$ $T = T_0\{\iota/\iota'\}$

By Lemma 3, we have $\mathtt{K}\{\eta''/\mathtt{mt}\} \vdash_{\mathtt{wft}} \mathtt{c}\langle\iota\{\eta''/\mathtt{mt}\}\rangle$. Let ι'' stand for $\iota\{\eta''/\mathtt{mt}\}$. By MT-Class, $\mathtt{mtype}(\mathtt{md}, \mathtt{c}\langle\iota''\rangle) = \overline{T_0}\{\iota''/\iota'\} \to T_0\{\iota''/\iota'\}$.

Since $\{\iota\{\eta''/\mathtt{mt}\}/\iota'\} = \{\iota/\iota'\}\{\eta''/\mathtt{mt}\}$, we have $\mathtt{mtype}(\mathtt{md}, \mathtt{c}\langle\iota''\rangle) = \overline{T_0}\{\iota/\iota'\}\{\eta''/\mathtt{mt}\} \to T_0\{\iota/\iota'\}\{\eta''/\mathtt{mt}\}$.

Case MT-Super Immediate from the inductive hypothesis.

■

Lemma 6. *If* $\mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' \vdash_{\mathtt{wft}} \mathtt{c}\langle\iota\rangle$, $\mathtt{K} \looparrowright \{\eta \leq \eta'', \eta'' \leq \eta'\}$, $\mathtt{mt} \notin \mathtt{K}$ *and* $\mathtt{fields}(T) = \overline{T}\ \overline{\mathtt{fd}}$ *then* $\mathtt{fields}(\mathtt{c}\langle\iota\rangle\{\eta''/\mathtt{mt}\}) = \overline{T\{\eta''/\mathtt{mt}\}}\ \overline{\mathtt{fd}}$.

Proof Similar, but with induction on the derivation of $\mathtt{fields}(\mathtt{md}, \mathtt{c}\langle\iota\rangle) = \overline{T}\ \overline{\mathtt{fd}}$. ■

Lemma 7 (Mode Substitution Preserves Typing). *If* $\Gamma; \mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' \vdash e : \tau$, $\mathtt{K} \looparrowright \{\eta \leq \eta'', \eta'' \leq \eta'\}$, *and* $\mathtt{mt} \notin \mathtt{K}$, *then* $\Gamma\{\eta''/\mathtt{mt}\}; \mathtt{K}\{\eta''/\mathtt{mt}\} \vdash e\{\eta''/\mathtt{mt}\} : \tau\{\eta''/\mathtt{mt}\}$.

Proof Induction on the derivation of $\Gamma; \mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' \vdash e : \tau$.

Case T-Var Easy.

Case T-New
$e = \textbf{new}\ \mathtt{c}\langle\iota\rangle$ $\tau = \mathtt{c}\langle\iota\rangle$
$\mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' \looparrowright \mathtt{cons}(\Delta)$ $\mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' \vdash_{\mathtt{wft}} \mathtt{c}\langle\iota\rangle$

Using Lemmas 2 and 3 gives us $\mathtt{K}\{\eta''/\mathtt{mt}\} \looparrowright \mathtt{cons}(\Delta)\{\eta''/\mathtt{mt}\}$ and $\mathtt{K}\{\eta''/\mathtt{mt}\} \vdash_{\mathtt{wft}} \mathtt{c}\langle\iota\{\eta''/\mathtt{mt}\}\rangle$. Then, by T-New, we have $\Gamma\{\eta''/\mathtt{mt}\}; \mathtt{K}\{\eta''/\mathtt{mt}\} \vdash \textbf{new}\ \mathtt{c}\langle\iota\{\eta''/\mathtt{mt}\}\rangle : \mathtt{c}\langle\iota\{\eta''/\mathtt{mt}\}\rangle$.

Case T-Cast Easy.

Case T-Msg
$e = e_1.\mathtt{md}(\overline{e_1})$ $\tau = T$
$\mathtt{mtype}(\mathtt{md}, \mathtt{c}\langle\iota\rangle) = \overline{T} \to T$ $\mathtt{omode}(\mathtt{c}\langle\iota\rangle) \neq\ ?$
$\mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' \looparrowright \{\mathtt{omode}(\mathtt{c}\langle\iota\rangle) \leq \mathtt{omode}(T_0)\}$

By the induction hypothesis we have,

$$\Gamma\{\eta''/\mathtt{mt}\}; \mathtt{K}\{\eta''/\mathtt{mt}\} \vdash e_1\{\eta''/\mathtt{mt}\} : \mathtt{c}\langle\iota\{\eta''/\mathtt{mt}\}\rangle$$
$$\Gamma\{\eta''/\mathtt{mt}\}; \mathtt{K}\{\eta''/\mathtt{mt}\} \vdash \overline{e_1\{\eta''/\mathtt{mt}\}} : \overline{T\{\eta''/\mathtt{mt}\}}$$
$$\Gamma\{\eta''/\mathtt{mt}\}; \mathtt{K}\{\eta''/\mathtt{mt}\} \vdash \textbf{this}\{\eta''/\mathtt{mt}\} : T_0\{\eta''/\mathtt{mt}\}.$$

Now, by Lemma 5 we have $\mathtt{mtype}(\mathtt{md}, \mathtt{c}\langle\iota\{\eta''/\mathtt{mt}\}\rangle) = \overline{T\{\eta''/\mathtt{mt}\}} \to T\{\eta''/\mathtt{mt}\}$.

Using Lemma 2 gives us $\mathtt{K}\{\eta''/\mathtt{mt}\} \hookrightarrow \{\mathtt{omode}(\mathtt{c}\langle\iota\{\eta''/\mathtt{mt}\}\rangle) \leq \mathtt{omode}(T_0\{\eta''/\mathtt{mt}\})\}$.

Then, by T-Msg, we have

$$\Gamma\{\eta''/\mathtt{mt}\}; \mathtt{K}\{\eta''/\mathtt{mt}\} \vdash e_1\{\eta''/\mathtt{mt}\}.\mathtt{md}(\overline{e_1\{\eta''/\mathtt{mt}\}}) : T\{\eta''/\mathtt{mt}\}.$$

Case T-Field
$e = e_1.\mathtt{fd}_i$ $\quad \tau = T_i$
$\mathtt{fields}(\mathtt{c}\langle\iota\rangle) = \overline{T}\;\overline{\mathtt{fd}}$ $\quad \mathtt{omode}(\mathtt{c}\langle\iota\rangle) \neq ?$
$\mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' \hookrightarrow \{\mathtt{omode}(\mathtt{c}\langle\iota\rangle) \leq \mathtt{omode}(T_0)\}$

By the induction hypothesis we have,

$$\Gamma\{\eta''/\mathtt{mt}\}; \mathtt{K}\{\eta''/\mathtt{mt}\} \vdash e_1\{\eta/\mathtt{mt}\} : \mathtt{c}\langle\iota\{\eta''/\mathtt{mt}\}\rangle$$
$$\Gamma\{\eta''/\mathtt{mt}\}; \mathtt{K}\{\eta''/\mathtt{mt}\} \vdash \textbf{this} : T_0\{\eta/\mathtt{mt}\}.$$

Let ι'' stand for $\iota\{\eta''/\mathtt{mt}\}$. Now, by Lemma 6 we have $\mathtt{fields}(\mathtt{c}\langle\iota''\rangle) = \overline{T\{\eta''/\mathtt{mt}\}}\;\overline{\mathtt{fd}}$. Using Lemma 2 gives us $\mathtt{K}\{\eta''/\mathtt{mt}\} \hookrightarrow \{\mathtt{omode}(\mathtt{c}\langle\iota''\rangle) \leq \mathtt{omode}(T_0\{\eta''/\mathtt{mt}\})\}$.

Then, by T-Field, we have

$$\Gamma\{\eta''/\mathtt{mt}\}; \mathtt{K}\{\eta''/\mathtt{mt}\} \vdash e_1\{\eta''/\mathtt{mt}\}.\mathtt{fd}_i : T_i\{\eta''/\mathtt{mt}\}.$$

Case T-Snapshot
$e = \textbf{snapshot}\; e_1\;[\eta_1, \eta_2]$ $\quad \tau = \exists\omega.\mathtt{c}\langle\mathtt{mt}_1, \iota\rangle$
$\Gamma; \mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' \vdash e_1 : \mathtt{c}\langle ?, \iota\rangle$ $\quad \omega = \eta_1 \leq \mathtt{mt}_1 \leq \eta_2$

By the induction hypothesis,

$$\Gamma\{\eta''/\mathtt{mt}\}; \mathtt{K}\{\eta''/\mathtt{mt}\} \vdash e_1\{\eta''/\mathtt{mt}\} : \mathtt{c}\langle ?, \iota\{\eta''/\mathtt{mt}\}\rangle.$$

Let η_1', η_2', and ι'' stand for $\eta_1\{\eta''/\mathtt{mt}\}$, $\eta_2\{\eta''/\mathtt{mt}\}$, and $\iota\{\eta''/\mathtt{mt}\}$ resp. Now, we may assume $\mathtt{mt} \neq \mathtt{mt}_1$ since $\mathtt{mt}_1$ is bound by ω; hence, $\omega\{\eta''/\mathtt{mt}\} = \eta_1' \leq \mathtt{mt}_1 \leq \eta_2'$.

Then, by T-Snapshot,

$$\Gamma\{\eta''/\mathtt{mt}\}; \mathtt{K}\{\eta''/\mathtt{mt}\} \vdash \textbf{snapshot}\; e_1\{\eta''/\mathtt{mt}\}\;[\eta_1', \eta_2'] : \exists\omega\{\eta''/\mathtt{mt}\}.\mathtt{c}\langle\mathtt{mt}_1, \iota''\rangle.$$

Case T-MCase, T-ElimCase, T-ModeValue Easy.

Case T-Sub
$e = e_1$ $\quad \tau = \tau_1'$
$\Gamma; \mathtt{K}, \eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' \vdash e_1 : \tau_1$ $\quad \mathtt{K}\eta \leq \mathtt{mt}, \mathtt{mt} \leq \eta' \vdash \tau_1 <: \tau_1'$

By the induction hypothesis,

$$\Gamma\{\eta''/\mathtt{mt}\}; \mathtt{K}\{\eta''/\mathtt{mt}\} \vdash e_1\{\eta''/\mathtt{mt}\} : \tau_1\{\eta''/\mathtt{mt}\}.$$

Using Lemma 4 gives us $\mathtt{K}\{\eta''/\mathtt{mt}\} \vdash \tau_1\{\eta''/\mathtt{mt}\} <: \tau_1'\{\eta''/\mathtt{mt}\}$

Then, by T-Sub, we have

$$\Gamma\{\eta''/\mathtt{mt}\}; \mathtt{K}\{\eta''/\mathtt{mt}\} \vdash e_1\{\eta''/\mathtt{mt}\} : \tau'\{\eta''/\mathtt{mt}\}.$$

Case T-Object Easy.

Case T-Check Similar to T-Snapshot.

Case T-Closure Easy.

■

Lemma 8 (Term Substitution Perserves Typing). *If* $\Gamma, \mathtt{y} : \tau_0; \mathtt{K} \vdash e : \tau$ *and* $\Gamma; \mathtt{K} \vdash \mathtt{s} : \tau_0$ *then* $\Gamma\{\mathtt{s}/\mathtt{y}\}; \mathtt{K} \vdash e\{\mathtt{s}/\mathtt{y}\} : \tau$.

Proof Easy induction on the derivation of $\Gamma, \mathtt{y} : \tau_0; \mathtt{K} \vdash e : \tau$.

■

Lemma 9. *If* $\mathtt{K} \vdash_{\mathtt{wft}} \mathtt{c}\langle\iota\rangle$, $\mathtt{omode}(\mathtt{c}\langle\iota\rangle) \neq ?$, $\mathtt{mtype}(\mathtt{md}, \mathtt{c}\langle\iota\rangle) = \overline{T} \to T$ *and* $\mathtt{mbody}(\mathtt{md}, \mathtt{c}\langle\iota\rangle) = \overline{\mathtt{x}}.e$ *then* $\overline{\mathtt{x}} : \overline{T}; \textbf{this} : T; \mathtt{K} \vdash e : T$.

Proof Induction on the derivation of $\mathtt{mbody}(\mathtt{md}, \mathtt{c}\langle\iota\rangle) = \overline{\mathtt{x}}.e$ using Lemmas 4 and 7.

Case MB-Class
$\overline{\mathtt{x}}.e = \overline{\mathtt{y}}.e_0\{\iota/\iota'\}$ $\quad \textbf{class}\; \mathtt{c}\;\Delta\; \textbf{extends}\; \mathtt{c}'\{\overline{F}\;\overline{M}\;A\}$
$\mathtt{param}(\Delta) = \iota'$ $\quad T_0\;\mathtt{md}(\overline{T_0}\;\overline{\mathtt{y}})\{\;e_0\;\} \in \overline{M}$

From T-Class and T-Method we have $\overline{\mathtt{y}} : \overline{T_0}; \textbf{this} : \mathtt{c}\langle\iota'\rangle; \mathtt{K}' \vdash e_0 : T_0$. Since $\mathtt{K} \vdash_{\mathtt{wft}} \mathtt{c}\langle\iota\rangle$ we have $\mathtt{K} \hookrightarrow \mathtt{K}'\{\iota/\iota'\}$ and $\mathtt{K}' = \mathtt{cons}(\Delta)$ from WF-Class. Using Lemmas 1 and 7 we have $\overline{\mathtt{y}} : \overline{T_0}\{\iota/\iota'\}; \textbf{this} : \mathtt{c}\langle\iota\rangle; \mathtt{K} \vdash e_0\{\iota/\iota'\} : T_0\{\iota/\iota'\}$.

Then, by MT-Class we have $\overline{T_0}\{\iota/\iota'\} = \overline{T}$ and $T_0\{\iota/\iota'\} = T$, from which $\overline{\mathtt{x}} : \overline{T}, \textbf{this} : \mathtt{c}\langle\iota\rangle; \mathtt{K} \vdash e : T$ is immediate.

Case MB-Super
$\overline{\mathtt{x}}.e = \mathtt{mbody}(\mathtt{md}, \mathtt{c}'\langle\iota\rangle)$ $\quad \textbf{class}\; \mathtt{c}\;\Delta\; \textbf{extends}\; \mathtt{c}'\{\overline{F}\;\overline{M}\;A\}$
$\mathtt{md} \notin \overline{M}$

Immediate from the inductive hypothesis.

■

Lemma 10. *If* $\mathtt{K} \vdash_{\mathtt{wft}} \mathtt{c}\langle\iota\rangle$, $\mathtt{fields}(\mathtt{c}\langle\iota\rangle) = \overline{T}\;\overline{\mathtt{fd}}$, *and* $\mathtt{init}(P, \mathtt{c}\langle\iota\rangle) = \overline{e}$ *then* $\varnothing; \mathtt{K} \vdash \overline{e} : \overline{T}$.

Proof

Case FD-Class
$\mathtt{fields}(\mathtt{c}'\langle\iota\rangle) = \overline{T_1}\;\overline{\mathtt{fd}_1}$
$\textbf{class}\; \mathtt{c}\;\Delta\; \textbf{extends}\; \mathtt{c}'\{\overline{T_0}\;\overline{\mathtt{fd}} = \overline{e_0}\;\ldots\}$ $\quad \mathtt{param}(\Delta) = \iota'$

We have two subcases: either $\iota = ?, \overline{\eta}$ or $\iota = \eta', \overline{\eta}$. Without loss of generality, we argue the latter. From T-Class we have $\varnothing; \mathtt{K}' \vdash \overline{e_0} : \overline{T_0}$. Since $\mathtt{K} \vdash_{\mathtt{wft}} \mathtt{c}\langle\iota\rangle$ we have $\mathtt{K} \hookrightarrow \mathtt{K}'\{\iota/\iota'\}$ and $\mathtt{K}' = \mathtt{cons}(\Delta)$ from WF-Class. Using Lemmas 1 and 7 we have $\varnothing; \mathtt{K} \vdash \overline{e_0}\{\iota/\iota'\} : \overline{T_0}\{\iota/\iota'\}$

By the induction hypothesis, we have $\mathtt{init}(P, \mathtt{c}'\langle\iota\rangle) = \overline{e_1}$ with $\varnothing; \mathtt{K} \vdash \overline{e_1} : \overline{T_1}$. Now, we have $\varnothing; \mathtt{K} \vdash \overline{e_1}, \overline{e_0}\{\iota/\iota'\} : \overline{T_1}, \overline{T_0}\{\iota/\iota'\}$, but $\mathtt{init}(P, \mathtt{c}\langle\iota\rangle) = \mathtt{init}(P, \mathtt{c}'\langle\iota\rangle), \overline{e_0}\{\iota/\iota'\}$ and $\overline{T} = \overline{T_1}, \overline{T_0}\{\iota/\iota'\}$.

Case FD-Object Trivial.

■

Lemma 11. *If* $\Gamma; \mathtt{K} \vdash \mathtt{obj}(\alpha, \mathtt{c}\langle\iota\rangle, \overline{e})$, $\mathtt{K} \vdash_{\mathtt{wft}} \mathtt{c}\langle\iota\rangle$, *and* $\mathtt{fields}(\mathtt{c}\langle\iota\rangle) = \overline{T}\;\overline{\mathtt{fd}}$ *then* $\Gamma; \mathtt{K} \vdash \overline{e} : \overline{T}$.

Proof By Lemma 10 we have $\varnothing; \mathtt{K} \vdash \overline{e} : \overline{T}$. Lemma 1 gives us $\Gamma; \mathtt{K} \vdash \overline{e} : \overline{T}$. ■

Lemma 12. *If* $\Gamma; \mathtt{K} \vdash_{\mathtt{wft}} \mathtt{c}\langle ?, \iota\rangle$ *and* $\mathtt{abody}(\mathtt{c}\langle ?, \iota\rangle) = e_a$, *then* $\textbf{this} : \mathtt{c}\langle ?, \iota\rangle; \mathtt{K} \vdash e_a : \mathtt{modev}$.

Proof From T-Class and T-Attributor we have $\textbf{this} : \mathtt{c}\langle ?, \iota'\rangle; \mathtt{K}' \vdash e : \mathtt{modev}$ with $\mathtt{K}' = \mathtt{cons}(\Omega)$, $\iota' = \mathtt{param}(\Omega)$, and $\mathtt{K}' \vdash_{\mathtt{wft}} \mathtt{c}\langle ?, \iota'\rangle$. Since $\mathtt{K} \vdash_{\mathtt{wft}} \mathtt{c}\langle ?, \iota\rangle$ we have $\mathtt{K} \hookrightarrow \mathtt{K}'\{\iota/\iota'\}$. Using Lemmas 1 and 7 we have $\textbf{this} : \mathtt{c}\langle ?, \iota\rangle; \mathtt{K} \vdash e\{\iota/\iota'\} : \mathtt{modev}$; however, $e\{\iota/\iota'\}$ is e_a. ■

Lemma 13. *If* $\mathtt{K} \vdash \mathtt{c}\langle\mu', \iota\rangle <: \mathtt{c}\langle\mu, \iota\rangle$ *and* $\mathtt{fields}(\mathtt{c}\langle\mu, \iota\rangle) = \overline{T}\;\overline{\mathtt{fd}}$ *then* $\mathtt{fields}(\mathtt{c}\langle\mu', \iota\rangle) = \overline{T}\;\overline{\mathtt{fd}}$.

Proof Trivial from the fact that $\mathtt{param}(\mathtt{c}\langle\mu, \iota\rangle) = \mathtt{param}(\mathtt{c}\langle\mu', \iota\rangle)$. ■

Lemma 14 (Replacement). *If* $\mathbf{D}$ *a deduction concluding* $\Gamma; \mathtt{K} \vdash \mathbf{E}[\,e_1\,] : \tau$, $\mathbf{D}_1$ *is a subdeduction of* $\mathbf{D}$ *concluding* $\Gamma'; \mathtt{K}' \vdash e_1 : \tau'$, $\mathbf{D}_1$ *occurs in* $\mathbf{D}$ *in the position corresponding to the hole* $\odot$ *in* $\mathbf{E}$, *and* $\Gamma'; \mathtt{K}'; \vdash e_2 : \tau'$, *then* $\Gamma; \mathtt{K} \vdash \mathbf{E}[\,e_2\,] : \tau$.

Proof Similar argument to [5] Lemma 4.2 and [4] pg. 181: Let us view the deduction $\mathbf{D}_1$ of $\Gamma'; \mathtt{K}' \vdash e_1 : \tau'$ as a tree. If there exists a deduction $\mathbf{D}_2$ of $\Gamma'; \mathtt{K}' \vdash e_2 : \tau'$ then we may replace $\mathbf{D}_1$ and e_1 with $\mathbf{D}_2$ and e_2 in $\mathbf{D}$, thus reaching $\Gamma; \mathtt{K} \vdash \mathbf{E}_{:}[\,e_2\,]\tau$. ■

Lemma 15 (Preservation). *If* $\Gamma; \mathtt{K} \vdash e : \tau$, $e \xRightarrow{\mathtt{m}} e'$, *then* $\Gamma; \mathtt{K} \vdash e : \tau$.

Proof By induction on the derivation of $e \xRightarrow{\mathtt{m}} e'$, with a case analysis on the last rule used. We consider the interesting cases.

Case R-New, R-Cast Easy.

Case R-Msg
$e = o.\mathtt{md}(\overline{v'})$ $\quad e' = \mathtt{cl}(\mu, e\{\overline{v'}/\overline{\mathtt{x}}\}\{o/\mathbf{this}\})$
$\tau = T$ $\quad o = \mathtt{obj}(\alpha, \mathtt{c}\langle\mu,\iota\rangle, \overline{v})$
$\emptyset \looparrowright \mu \leq \mathtt{m}$

From T-Msg and T-Object we have

$$\begin{array}{l}\Gamma; \mathtt{K} \vdash \mathtt{obj}(\alpha, \mathtt{c}\langle\mu,\iota\rangle, \overline{v}) : \mathtt{c}\langle\mu,\iota\rangle \\ \mathtt{omode}(\mathtt{c}\langle\mu,\iota\rangle) \neq ? \\ \mathtt{mtype}(\mathtt{md}, \mathtt{c}\langle\mu,\iota\rangle) = \overline{T} \to T \\ \Gamma; \mathtt{K} \vdash \overline{v'} : \overline{T'} \\ \Gamma; \mathtt{K} \vdash \mathbf{this} : T_0 \\ \Gamma; \mathtt{K} \vdash o.\mathtt{md}(\overline{v}) : T\end{array}$$

From Lemma 9 we have $\overline{\mathtt{x}} : \overline{T}, \mathbf{this} : \mathtt{c}\langle\mu,\iota\rangle; \mathtt{K} \vdash \overline{\mathtt{x}}.e_b : T$. Using Lemma 8 twice gives us $\varnothing; \mathtt{K} \vdash e_b\{\overline{v'}/\overline{\mathtt{x}}\}\{o/\mathbf{this}\} : T$. Now, we may weaken $\varnothing$ to Γ by Lemma 1 which gives us $\Gamma; \mathtt{K} \vdash e_b\{\overline{v'}/\overline{\mathtt{x}}\}\{o/\mathbf{this}\} : T$.

Then, by T-Closure we have

$$\Gamma; \mathtt{K} \vdash \mathtt{cl}(\mathtt{m}', e_b\{\overline{v'}/\overline{\mathtt{x}}\}\{o/\mathbf{this}\}) : T.$$

Case R-Field
$e = o.\mathtt{fd}_i$ $\quad e' = v_i$
$\tau = T_i$ $\quad o = \mathtt{obj}(\alpha, \mathtt{c}\langle\mu,\iota\rangle, \overline{v})$

From T-Field and T-Object we have

$$\begin{array}{l}\Gamma; \mathtt{K} \vdash \mathtt{obj}(\alpha, \mathtt{c}\langle\mu,\iota\rangle, \overline{v}) : \mathtt{c}\langle\mu,\iota\rangle \\ \mathtt{fields}(\mathtt{c}\langle\mu,\iota\rangle) = (\overline{T}\ \overline{\mathtt{fd}}) \\ \Gamma; \mathtt{K} \vdash o.\mathtt{fd}_i : T_i\end{array}$$

By Lemma 11 we have $\Gamma; \mathtt{K} \vdash \overline{v} : \overline{T_i}$. Choosing v_i finishes the case.

Case R-Snapshot
$e = \mathbf{snapshot}\ o\ [\mathtt{m}_1, \mathtt{m}_2]$ $\quad e' = \mathbf{check}(e_a\{o/\mathbf{this}\}, \mathtt{m}_1, \mathtt{m}_2, o)$
$\tau = \exists \mathtt{m}_1 \leq \mathtt{mt} \leq \mathtt{m}_2.\mathtt{c}\langle\mathtt{mt},\iota\rangle$ $\quad o = \mathtt{obj}(\alpha, \mathtt{c}\langle ?,\iota\rangle, \overline{v})$
$\mathbf{class}\ \mathtt{c} \cdots \{ \cdots A \} \in P$ $\quad \mathtt{abody}(\mathtt{c}\langle ?,\iota\rangle) = e_a$

From T-Snapshot and T-Object we have

$$\begin{array}{l}\Gamma; \mathtt{K} \vdash \mathtt{obj}(\alpha, \mathtt{c}\langle ?,\iota\rangle, \overline{v}) : \mathtt{c}\langle ?,\iota\rangle \\ \Gamma; \mathtt{K} \vdash \mathbf{snapshot}\ o\ [\mathtt{m}_1, \mathtt{m}_2] : \exists\omega.\mathtt{c}\langle\mathtt{mt},\iota\rangle \\ \omega = \mathtt{m}_1 \leq \mathtt{mt} \leq \mathtt{m}_2\end{array}$$

From Lemma 12 we have $\mathbf{this} : \mathtt{c}\langle ?,\iota\rangle; \mathtt{K} \vdash e_a : \mathtt{modev}$. Then, by Lemma 8 we have $\varnothing; \mathtt{K} \vdash e_a\{o/\mathbf{this}\} : \mathtt{modev}$. Using Lemma 1 gives us $\Gamma; \mathtt{K} \vdash e_a\{o/\mathbf{this}\} : \mathtt{modev}$.

Then, by T-Check we have $\Gamma; \mathtt{K} \vdash \mathbf{check}(e_a\{o/\mathbf{this}\}, \mathtt{m}_1, \mathtt{m}_2, o) : \exists \mathtt{m}_1 \leq \mathtt{mt} \leq \mathtt{m}_2.\mathtt{c}\langle\mathtt{mt}',\iota\rangle$. We may alpha rename $\mathtt{mt}'$ to $\mathtt{mt}$, giving us exactly what is needed.

Case R-Check
$e = \mathbf{check}(\mathtt{m}', \mathtt{m}_1, \mathtt{m}_2, o)$ $\quad e' = \mathtt{obj}(\alpha', \mathtt{c}\langle\mathtt{m}',\iota\rangle, \overline{v})$
$\tau = \exists \mathtt{m}_1 \leq \mathtt{mt} \leq \mathtt{m}_2.\mathtt{c}\langle\mathtt{mt},\iota\rangle$ $\quad o = \mathtt{obj}(\alpha, \mathtt{c}\langle\mu,\iota\rangle, \overline{v})$
$\emptyset \looparrowright \{\mathtt{m}_1 \leq \mathtt{m}', \mathtt{m}' \leq \mathtt{m}_2\}$ $\quad \alpha'$ fresh

From T-Check and T-Object we have

$$\begin{array}{l}\Gamma; \mathtt{K} \vdash \overline{v} : \overline{T} \\ \mathtt{fields}(\mathtt{c}\langle\mu,\iota\rangle) = \overline{T}\ \overline{\mathtt{fd}} \\ \Gamma; \mathtt{K} \vdash \mathtt{obj}(\alpha, \mathtt{c}\langle\mu,\iota\rangle, \overline{v}) : \mathtt{c}\langle\mu,\iota\rangle \\ \omega = \mathtt{m}_1 \leq \mathtt{mt} \leq \mathtt{m}_2 \\ \Gamma; \mathtt{K} \vdash \mathbf{check}(\mathtt{m}', \mathtt{m}_1, \mathtt{m}_2, o) : \exists\omega.\mathtt{c}\langle\mathtt{mt},\iota\rangle\end{array}$$

Lemma 13 gives $\mathtt{fields}(\mathtt{c}\langle ?,\iota\rangle) = \mathtt{fields}(\mathtt{c}\langle\mathtt{m}',\iota\rangle)$, from which we have $\Gamma; \mathtt{K} \vdash \mathtt{obj}(\alpha', \mathtt{c}\langle\mathtt{m}',\iota\rangle, \overline{v})$ by T-Object.

Since $\emptyset \looparrowright \{\mathtt{m}_1 \leq \mathtt{m}', \mathtt{m}' \leq \mathtt{m}_2\}$ we have $\mathtt{K} \looparrowright \{\mathtt{m}_1 \leq \mathtt{m}', \mathtt{m}' \leq \mathtt{m}_2\}$. $\mathtt{omode}(\mathtt{c}\langle\mathtt{m}',\iota\rangle) = \mathtt{m}'$, from which $\mathtt{K} \vdash \mathtt{c}\langle\mathtt{m}',\iota\rangle <: \mathtt{c}\langle\mathtt{mt},\iota\rangle\{\mathtt{mt}/\mathtt{m}'\}$ is immediate.

Then, by T-Sub and S-ExistAbstract we have

$$\Gamma; \mathtt{K} \vdash \mathtt{obj}(\alpha', \mathtt{c}\langle\mathtt{m}',\iota\rangle, \overline{v}) : \exists\omega.\mathtt{c}\langle\mathtt{mt},\iota\rangle.$$

Case R-McaseProj, R-Closure1, R-Closure2 Easy.

Case R-Context
$e = \mathbf{E}[\,e_1\,]$ $\quad e' = \mathbf{E}[\,e_1'\,]$
$e_1 \stackrel{\mathtt{m}}{\Longrightarrow} e_1'$

Let us assume $\Gamma; \mathtt{K} \vdash \mathbf{E}[\,e_1\,] : \tau$ with $\Gamma; \mathtt{K} \vdash e_1 : \tau'$. By the induction hypothesis, $\Gamma; \mathtt{K} \vdash e_1' : \tau'$. Then, by Lemma 14, $\Gamma; \mathtt{K} \vdash \mathbf{E}[\,e_1'\,] : \tau$.

∎

Lemma 16.
(1) If $\Gamma; \mathtt{K} \vdash v : \tau$ and $\mathtt{K} \vdash \tau <: \mathtt{c}\langle\iota\rangle$, then $\tau = \mathtt{c}'\langle\iota\rangle$ with $\mathtt{K} \vdash \mathtt{c}'\langle\iota\rangle <: \mathtt{c}\langle\iota\rangle$.
(2) If $\Gamma; \mathtt{K} \vdash v : \tau$ and $\mathtt{K} \vdash \tau <: \mathbf{mcase}\langle T\rangle$, then $\tau = \mathbf{mcase}\langle T'\rangle$ with $\mathtt{K} \vdash T' <: T$.

Proof Easy case analysis on the induction of the derivations of $\mathtt{K} \vdash \tau <: \mathtt{c}\langle\iota\rangle$ and $\mathtt{K} \vdash \tau <: \mathbf{mcase}\langle T\rangle$. ∎

Lemma 17 (Canonical Forms)**.** *Given $\Gamma; \mathtt{K} \vdash v : \tau$,*

(1) If $\tau = \mathtt{c}\langle\iota\rangle$ then v has the shape $\mathtt{obj}(\alpha, \tau', \overline{v})$ with $\mathtt{K} \vdash \tau' <: \mathtt{c}\langle\iota\rangle$.
(2) If $\tau = \mathbf{mcase}\langle T\rangle$ then v has the shape $\mathbf{mcase}\langle T'\rangle\{\overline{\mathtt{m} : v}\}$ with $\mathtt{K} \vdash T' <: T$.
(3) If $\tau = \mathtt{modev}$ then v has the shape $\mathtt{m}$ with $\mathtt{m} \in \mathtt{modes}(P)$.

Proof

(1) Induction on the derivation $\Gamma; \mathtt{K} \vdash v : \mathtt{c}\langle\iota\rangle$. Two rules may apply: T-Obj and T-Sub.

Case T-Obj $\quad v = \mathtt{obj}(\alpha, \mathtt{c}\langle\iota\rangle, \overline{v})$
Letting τ' be $\mathtt{c}\langle\iota\rangle$ finishes the case.

Case T-Sub
$v = v_1$
$\Gamma; \mathtt{K} \vdash v_1 : \tau_1 \quad \mathtt{K} \vdash \tau_1 <: \mathtt{c}\langle\iota\rangle$
By Lemma 16 $\tau_1 = \mathtt{c}'\langle\iota\rangle$. Then, by the induction hypothesis, $v_1 = \mathtt{obj}(\alpha, \tau_1', \overline{v})$ with $\mathtt{K} \vdash \tau_1' <: \tau_1$. By S-Trans, $\mathtt{K} \vdash \tau_1' <: \mathtt{c}\langle\iota\rangle$. Then, by T-Sub, $\Gamma; \mathtt{K} \vdash \mathtt{obj}(\alpha, \tau_1', \overline{v}) : \mathtt{c}\langle\iota\rangle$.

(2) Induction on the derivation $\Gamma; \mathtt{K} \vdash v : \mathbf{mcase}\langle T\rangle$. Two rules may apply: T-Mcase and T-Sub.

Case T-Mcase $\quad v = \mathbf{mcase}\langle T\rangle\{\overline{\mathtt{m} : v}\}$
Letting T' be T finishes the case.

Case T-Sub
$v = v_1$
$\Gamma; \mathtt{K} \vdash v_1 : \tau_1 \quad \mathtt{K} \vdash \tau_1 <: \mathbf{mcase}\langle T\rangle$
By Lemma 16 $\tau_1 = \mathbf{mcase}\langle T_1\rangle$ with $\mathtt{K} \vdash T_1 <: T$. Then, by the induction hypothesis, $v_1 = \mathbf{mcase}\langle T_1'\rangle\{\overline{\mathtt{m} : v}\}$ with $\mathtt{K} \vdash T_1' <: T_1$. By S-Trans, $\mathtt{K} \vdash T_1' <: T$. Then, by (T-Sub), $\Gamma; \mathtt{K} \vdash \mathbf{mcase}\langle T_1\rangle\{\overline{\mathtt{m} : v}\} : \mathbf{mcase}\langle T\rangle$.

(3) Only (T-ModeValue) may apply from which $\mathtt{m} \in \mathtt{modes}(P)$ is immediate.

∎

Definition 1 (Bad Cast)**.** *Expression $(T')\mathtt{obj}(\alpha, T, \overline{v})$ is a bad cast iff $\emptyset \vdash T <: T'$ does not hold.*

Definition 2 (Bad Check)**.** *Expression $\mathbf{check}(\mathtt{m}, \mathtt{m}', \mathtt{m}'', o)$ is a bad check iff $\emptyset \looparrowright \{\mathtt{m}' \leq \mathtt{m}, \mathtt{m} \leq \mathtt{m}''\}$ does not hold.*

Lemma 18. *If $e = \mathbf{E}[\,e'\,]$, and $\Gamma; \mathtt{K} \vdash \mathbf{E}[\,e'\,] : \tau$ whose subderivation is rooted at $\Gamma'; \mathtt{K}' \vdash e' : \tau'$ then $\mathtt{omode}(\Gamma(\mathbf{this})) = \mathtt{omode}(\Gamma'(\mathbf{this}))$.*

Proof Case analysis on the structure of e. Consider the case, $e = \odot[e']$. We have $e = e'$ with $\Gamma = \Gamma'$ from which $\mathtt{omode}(\Gamma(\mathbf{this})) = \mathtt{omode}(\Gamma'(\mathbf{this}))$ is immediate. The rest follows from easy induction.

∎

Lemma 19 (Progress)**.** *Suppose* **this** $: T; \emptyset \vdash e : \tau$ *and* $\mathtt{omode}(\Gamma(\mathbf{this})) = \mathtt{m}$*, then either*

(1) $e \overset{\mathtt{m}}{\Longrightarrow} e'$ *for some* e'.

(2) e *is a value.*

(3) $e = \mathbf{E}[\,\mathbf{check}(\mathtt{m}', \mathtt{m}_1, \mathtt{m}_2, o)\,]$ *where* $\emptyset \looparrowright \{\mathtt{m}_1 \le \mathtt{m}', \mathtt{m}' \le \mathtt{m}_2\}$ *does not hold.*

(4) $e = \mathbf{E}[\,(T')\mathtt{obj}(\alpha, T, \overline{v})\,]$ *where* $\emptyset \vdash T <: T'$ *does not hold.*

(5) $e = \mathtt{cl}(\mathtt{m}, \mathbf{E}[\,\mathbf{check}(\mathtt{m}', \mathtt{m}_1, \mathtt{m}_2, o)\,])$ *where* $\emptyset \looparrowright \{\mathtt{m}_1 \le \mathtt{m}', \mathtt{m}' \le \mathtt{m}_2\}$ *does not hold.*

(6) $e = \mathtt{cl}(\mathtt{m}, \mathbf{E}[\,(T')\mathtt{obj}(\alpha, T, \overline{v})\,])$ *where* $\emptyset \vdash T <: T'$ *does not hold.*

Proof By induction on the derivation of **this** $: \mathtt{c}'\langle\iota\rangle; \emptyset \vdash e : \tau$. We use o to stand for $\mathtt{obj}(\alpha, \mathtt{c}\langle ?, \iota\rangle, \overline{v})$.

Case T-Var, T-New Easy.

Case T-Cast
$e = (T')e_1$ $\quad \tau = T'$
this $: \mathtt{c}'\langle\iota\rangle; \emptyset \vdash e_1 : \mathtt{c}\langle\iota\rangle$

We consider the case where e_1 is a value. By Lemma 17 we have $e_1 = \mathtt{obj}(\alpha, T, \overline{v})$. If $\emptyset \vdash T <: T'$ then R-Cast applies, giving $e' = \mathtt{obj}(\alpha, T, \overline{v})$. If $\emptyset \vdash T <: T'$ does not hold, then $e = \mathbf{E}[\,(T')\mathtt{obj}(\alpha, T, \overline{v})\,]$, a bad cast.

Case T-Msg
$e = e_1.(\overline{e_1})$ $\quad \tau = T'$

We consider the case where e_1 and all e_{1_i} are values. By Lemma 17, $e_1 = \mathtt{obj}(\alpha, \tau', \overline{v})$ with $\emptyset \vdash \tau' <: \mathtt{c}\langle\iota\rangle$. Then, by Lemma 18 we have $\emptyset \looparrowright \{\mathtt{omode}(T) \le \mathtt{m}\}$. R-Msg now applies, giving $e' = \mathtt{cl}(\mathtt{m}', e_b\{\overline{v}'/\overline{\mathtt{x}}\}\{o/\mathbf{this}\})$ with $\mathtt{mbody}(\mathtt{md}, T) = \overline{\mathtt{x}}.e_b$.

Case T-Field Similar.

Case T-Snapshot
$e = \mathbf{snapshot}\ e_1\ [\mathtt{m}_1, \mathtt{m}_2]$ $\quad \tau = \exists\omega.\mathtt{c}\langle \mathtt{mt}_1, \iota\rangle$
this $: T; \emptyset \vdash e_1 : \mathtt{c}\langle ?, \iota\rangle$ $\quad \omega = \mathtt{m}_1 \le \mathtt{mt}_1 \le \mathtt{m}_2$
We consider the case where e_1 is a value. Let o_1 stand for $\mathtt{obj}(\alpha, \tau, \overline{v})$. By Lemma 17, $e_1 = \mathtt{obj}(\alpha, \tau, \overline{v})$ with $\emptyset \vdash \tau <: \mathtt{c}\langle ?, \iota\rangle$. R-Snapshot applies, giving $e' = \mathbf{check}(e_a\{o_1/\mathbf{this}\}, \mathtt{m}_1, \mathtt{m}_2, o_1)$.

Case T-MCase Easy.

Case T-ElimCase
$e = e_1 \rhd \eta$ $\quad \tau = T$
this $: T; \emptyset \vdash e_1 : \mathbf{mcase}\langle T\rangle$ $\quad \eta \in \mathtt{modes}(P)$ or η appears in $\emptyset$

Similar, except for the case that e_1 is a value. By Lemma 17, e_1 has the shape $\mathbf{mcase}\langle T'\rangle\{\overline{\mathtt{m} : v}\}$ with $\emptyset \vdash T' <: T$, from which R-McaseProj applies, giving us $e' = v_j$.

Case T-ModeValue, T-Sub, T-Object Easy.

Case T-Check
$e = \mathbf{check}(e_1, \mathtt{m}_1, \mathtt{m}_2, e_2)$ $\quad \tau = \mathtt{modev}$
this $: T; \emptyset \vdash e_1 : \mathtt{modev}$
this $: T; \emptyset \vdash e_2 : \mathtt{c}\langle ?, \iota\rangle$

We consider the case where e_1 and e_2 are values. By Lemma 17, e_1 has the shape $\mathtt{m}'$ and e_2 has the shape $\mathtt{obj}(\alpha, \tau', \overline{v})$ with $\mathtt{K} \vdash \tau' <: \mathtt{c}\langle ?, \iota\rangle$. Now, we have two cases: If $\emptyset \looparrowright \{\mathtt{m}_1 \le \mathtt{m}, \mathtt{m} \le \mathtt{m}_2\}$ then T-Check applies, giving us $e' = \mathtt{obj}(\alpha', \mathtt{c}\langle \mathtt{m}', \iota\rangle, \overline{v})$. Otherwise we have a bad check, with $e = \mathbf{E}[\,\mathbf{check}(\mathtt{m}', \mathtt{m}_1, \mathtt{m}_2, \mathtt{obj}(\alpha, \mathtt{c}\langle ?, \iota\rangle, \overline{v}))\,]$.

Case T-Closure
$e = \mathtt{cl}(\mathtt{m}', e_1)$ $\quad \tau = \tau_1$
$\Gamma, \mathbf{this} : T_1; \emptyset \vdash e_1 : \tau_1$ $\quad \mathtt{omode}(T_1) = \mathtt{m}'$

Let $\Gamma' = \Gamma, \mathbf{this} : T_1$. Since $\mathtt{omode}(\Gamma'(\mathbf{this})) = \mathtt{m}'$, by the inductive hypothesis, $e_1 \overset{\mathtt{m}'}{\Longrightarrow} e_1'$, e_1 is a value, $e_1 = \mathbf{E}_1[\,\mathbf{check}(\mathtt{m}'', \mathtt{m}_1, \mathtt{m}_2, o)\,]$ where $\emptyset \looparrowright \{\mathtt{m}_1 \le \mathtt{m}'', \mathtt{m}'' \le \mathtt{m}_2\}$ does not hold, or $e_1 = \mathbf{E}_1[\,(T')\mathtt{obj}(\alpha, T, \overline{v})\,]$ where $\emptyset \vdash T <: T'$ does not hold.

If e_1 is a value, R-Closure2 applies. If $e_1 \overset{\mathtt{m}'}{\Longrightarrow} e_1'$ then R-Closure1 applies. If $e_1 = \mathbf{E}_1[\,\mathbf{check}(\mathtt{m}'', \mathtt{m}_1, \mathtt{m}_2, o)\,]$ where $\emptyset \looparrowright \{\mathtt{m}_1 \le \mathtt{m}'', \mathtt{m}'' \le \mathtt{m}_2\}$ does not hold, or $e_1 = \mathbf{E}_1[\,(T')\mathtt{obj}(\alpha, T, \overline{v})\,]$ where $\emptyset \vdash T <: T'$ does not hold then we have $e = \mathtt{cl}(\mathtt{m}', \mathbf{E}_1[\,\mathbf{check}(\mathtt{m}'', \mathtt{m}_1, \mathtt{m}_2, o)\,])$ where $\emptyset \looparrowright \{\mathtt{m}_1 \le \mathtt{m}'', \mathtt{m}'' \le \mathtt{m}_2\}$ does not hold, or $e = \mathtt{cl}(\mathtt{m}', \mathbf{E}_1[\,(T')\mathtt{obj}(\alpha, T, \overline{v})\,])$ where $\emptyset \vdash T <: T'$ does not hold.

■

Theorem 1 (Type Soundness)**.** *If P is well-typed and* $\mathtt{boot}(P) = \mathtt{cl}(\top, e)$*, then either* $e \overset{\top}{\Longrightarrow}_* v$*,* $\mathtt{cl}(\top, e) \Uparrow$*, or* $e \overset{\top}{\Longrightarrow}_* e'$ *and e' is a bad cast or a bad check.*

Proof Immediate from Lemmas 15 and 19. ■

Let us say $\mathtt{cl}(\mathtt{m}_0, e_0)$ is a *sub-redex* of reduction $e \overset{\mathtt{m}}{\Longrightarrow} e'$ iff $e_0 \overset{\mathtt{m}_0}{\Longrightarrow} e_0'$ is a sub-derivation of $e \overset{\mathtt{m}}{\Longrightarrow} e'$.

Theorem 2 (Waterfall Invariant with Mixed Typing)**.** *If P is well-typed,* $\mathtt{boot}(P) = \mathtt{cl}(\top, e)$*,* $e \overset{\top}{\Longrightarrow} \ldots e_1 \overset{\top}{\Longrightarrow} e_2$*, and* $\mathtt{cl}(\mathtt{m}, \mathtt{obj}(\alpha, T, \overline{v}).\mathtt{md}(\overline{v'}))$ *is a sub-redex of* $e_1 \overset{\top}{\Longrightarrow} e_2$*, then* $\emptyset \looparrowright \mathtt{omode}(T) <: \mathtt{m}$*.*

Proof Let us consider the sub-redex $\mathtt{cl}(\mathtt{m}, \mathtt{obj}(\alpha, T, \overline{v}).\mathtt{md}(\overline{v'}))$. By Theorem 19 we can either take a step, or have a bad cast or bad check. However, since neither a bad cast or bad check may occur during (R-Msg) we may take a step, granting the condition $\emptyset \looparrowright \mathtt{omode}(T) <: \mathtt{m}$. Hence, runtime errors never occur at message invocation time.

■

References

[1] Jrapl home, `http://kliu20.github.io/jRAPL/`.

[2] Watts up? power meters, `https://www.wattsupmeters.com/secure/index.php`,.

[3] Watts up logger, `https://github.com/kjordahl/Watts-Up--logger`,.

[4] J. R. Hindley and J. P. Seldin. *Introduction to Combinators and Λ-calculus*. 1986.

[5] A. Wright and M. Felleisen. A syntactic approach to type soundness. *Inf. Comput.*, 115(1):38–94, Nov. 1994.