Cheat Sheet for PFPL

August 10, 2016

I. Judgements and Rules

1 Abstract Syntax

N	P	Expression	Name	Say	Meaning
1	5	$\mathcal{P}$	Script P	Proposition	Something to be proved
2	5	$\mathcal{P}(a)$	Script P of a	Proposition about tree a	Something to be proved about AST a
3	5	$\mathcal{O}$	Script O	Operator	An operator that can be used in an AST
4	5	$\mathcal{O}(a)$	Script O of a	Operator of arity a	An operator of a given arity
5	5	$\mathcal{X}_s$	Script X sub s	Variables x of sort s	Variables x of sort s
6	5	$\mathcal{S}$	S	A set of sorts	A set of sorts
7	5	$\{X_s\}_{s\in\mathcal{S}}$	Family	Family X of s	A sort-indexed family of disjoint finite sets X_s of variables x of sort s
8	6	$[b/x]\,a$	Substitution	Substitute b for x in a	Substitute b for x in a
9	7	$x_1, ..., x_n.a$	Abstractor	Bind variables x_n to expression a	Bind variables x_n to expression a
10	8	$\overrightarrow{x}$	X arrow	List of xs	$x_1, ..., x_n$
11	8	$\rho : \overrightarrow{x} \leftrightarrow \overrightarrow{x}'$	Fresh renaming	Freshen x using renaming ρ	A bijection between $\overrightarrow{x}$ and $\overrightarrow{x}'$ where $\overrightarrow{x}'$ is fresh.
12	8	$\widehat{\rho}_i(a_i)$	Rho hat sub i	Rename result	The result of applying the renaming ρ_i to a_i
13	8	$x =_\alpha y$	Equal alpha	*α-equivalence*	Trees x and y equal up to renaming
14	9	$x \triangleq y$	Delta equals	Replacement	Replace expression x with expression y

2 Inductive Definitions

N	P	Expression	Name	Say	Meaning
15	13	τ type	Type	Type τ	Judgement that τ is a type
16	13	$e : \tau$	Colon	e is of type τ	Judgement that expression e is of type τ
17	13	$e \Downarrow v$	Down arrow	e has value v	Judgement that expression e has value v
18	14	$\dfrac{J_1 ... J_k}{J}$	Surfboard	Infers	Judgements $J_1 ... J_k$ infer judgement J

3 Hypothetical and General Judgements

N	P	Expression	Name	Say	Meaning
19	23	$J_1 ... J_k \vdash_{\mathcal{R}} \mathcal{K}$	Turnstile	Entails	Given $\mathcal{R}$ and J infer $\mathcal{K}$
20	23	Γ	Gamma	Judgements Gamma	A finite set of judgements
21	23	Δ	Delta	Judgements Delta	A finite set of type judgements
22	25	$\Gamma \models_R J$	Double turnstile	Admissible	$\vdash_R \Gamma$ implies $\vdash_R J$
23	28	$\bigtriangledown$	Down triangle	Generic derivation	Generic derivation

II. Statics and Dynamics

4 Statics

N	P	Expression	Name	Say	Meaning
24	36	$n ::= s$	Colon colon equals	The syntax of n is s	Specifies the syntax of n
25	36	;	Semicolon	And	Separates arguments to expressionsin abstact notation

5 Dynamics

N	P	Expression	Name	Say	Meaning
26	41	$s \longmapsto s'$	Bar arrow	Transistion	State s transitions to state s'
27	42	$s \longmapsto^* s'$	Bar arrow star	Iterated transistion	State s transitions to state s' over more than zero transitions
28	42	$s \longmapsto^n s'$	Bar arrow n	N times iterated transistion	State s transitions to state s' over n transitions
29	44	$\mathcal{E}$	Script E	Expression context	Expression context
30	45	$\circ$	Circle	Hole	Placeholder to put an instruction
31	46	$e \equiv e'$	Equivalent	Definitional equivalence	e is definitionally equivalent to e'

6 Type Safety

N	P	Expression	Name	Say	Meaning
32	58	e??	Wrong	E goes wrong	Expression e goes wrong

7 Evaluation Dynamics

N	P	Expression	Name	Say	Meaning
33	58	$e \Downarrow^k v$	Downarrow k	E evaluates in k steps	Expression e evaluates to v in k steps

III. Total Functions

8 Function Definitions and Values

N	P	Expression	Name	Say	Meaning
34	63	$\{f\}$	Brace brackets	Function	Surround function f in abstract notation
35	63	$f.e$	Dot	Dot	Introduces the scope e of a function f in abstract notation
36	64	$f(\tau_1) : \tau_2$	Function	Function definition	A function taking an argument of type τ_1 and returning a value of type τ_2
37	64	$[\![x.e/f]\!]e'$	Script bracket	Function substitution	Function substitution
38	65	$\tau_1 \to \tau_2$	Right arrow	Maps to	A total function that maps elements of type τ_1 to elements of type τ_2
39	65	λ	Lambda	Lambda	Abstraction

9 System T of Higher-Order Recursion

N	P	Expression	Name	Say	Meaning
40	71	$\hookrightarrow$	Hook arrow	Select	Selector (used in System T recursion, sum types, and product types)
41	71	$\mid$	Bar	Either	A choice
42	71	$\overline{n}$	Overline	Church numbering	The Church numbering
43	76	$\lceil n \rceil$	Divided hat	Gödel numbering	The Gödel numbering

IV. Finite Data Types

10 Product Types

N	P	Expression	Name	Say	Meaning
44	81	$<>$	Angle brackets	Null tuple	Null tuple
45	81	$< e_1, e_2 >$	Angle brackets	Ordered pair	Ordered pair
46	81	e.l	Left	Left projection	Select left member of the ordered pair
47	81	e.r	Right	Right projection	Select right member of the ordered pair

11 Sum Types

N	P	Expression	Name	Say	Meaning
48	87	l.e	Left	Left injection	Create sum type element using left type
49	87	r.e	Right	Right injection	Create sum type element using right type
50	93	$\triangleq$	Delta equals	Delta equals	Replacement

VI. Infinite Data Types

14 Generic Programming

N	P	Expression	Name	Say	Meaning
51	121	$t.\tau$	Dot	Type operator	Bind t to type τ

15 Inductive and Coinductive Types

N	P	Expression	Name	Say	Meaning
52	133	$\cong$	Tilde equal	Isomorphism	Isomorphism

VII. Variable Types

17 Abstract Types

N	P	Expression	Name	Say	Meaning
53	149	$\exists(t.\tau)$	Existential quantifier	Exists	Defines an interface
54	153	$\forall(t.\tau \longrightarrow \tau_2)$	Universal quantifier	For all	Defines universal type

18 Higher Kinds

N	P	Expression	Name	Say	Meaning
55	157	::	Colon colon	Kind type constructor	Maps types to types

VIII. Partiality and Recursive Types

19 System PCF of Recursive Functions

N	P	Expression	Name	Say	Meaning
56	166	$\mapsto$	Short bar arrow	Maps to	Function definition
57	166	$\bot$	Bottom	Bottom	Totally undefined partial function
58	167	$\tau_1 \rightharpoonup \tau_2$	Harpoon	Partial function	Partial function

20 System FPC of Recursive Types

N	P	Expression	Name	Say	Meaning
59	177	_	Underscore	Underscore	Unfree variable

IX. Dynamic Types

21 The Untyped λ-Calculus

N	P	Expression	Name	Say	Meaning
60	185	Λ	Lambda	Lambda calculus	The lambda calculus
61	188	Y	Y	Y Combinator	The Y combinator
62	190	$x^\dagger$	Superscript cross	Superscript cross	Language isomorphism

X. Subtyping

24 Structural Subtyping

N	P	Expression	Name	Say	Meaning
63	213	$\tau' <: \tau$	Subtype	τ' is a subtype of τ	τ' is a subtype of τ

XI. Dynamic Dispatch

27 Inheritance

N	P	Expression	Name	Say	Meaning
64	252	$()^\ddagger$	Isomorphism	Isomorphism	$()^\ddagger$ is a method isomorphism

XII. Control Flow

28 Control Stacks

N	P	Expression	Name	Say	Meaning
65	257	$k \rhd e$	Right triangle	Evaluation state	Evaluate e on k
66	257	$k \lhd e$	Left triangle	Return state	Evaluate k on e
67	258	ϵ	Epsilon	Empty frame	Empty frame
68	258	(—)	Frame hole	Frame hole	A place to put an evaluated expression into a frame
69	258	$k; f$	Stack with frame	Stack k with frame f	Stack k has frame f at the bottom
70	259	$k \lhd: \tau$	Triangle colon	Stack k expects value of type τ	Stack k expects value of type τ
71	259	$f : \tau \rightsquigarrow \tau'$	Squiggle arrow	Transform	Frame f transforms expression of type τ into expression of type τ'
72	261	$s \looparrowright e$	Loop arrow	Unravel	State s goes to expression e
73	261	$k \bowtie e = e'$	Bowtie	Goes to	Stack k and expression e goes to expression e'

29 Exceptions

N	P	Expression	Name	Say	Meaning
74	266	$k\blacklozenge$	Dark left triangle	Failed	Stack k is in a failed state

XIII. Symbolic Data

31 Symbols

N	P	Expression	Name	Say	Meaning
75	282	$a \sim \tau$	Tilde	Define a	Define symbol a as type τ
76	282	Σ	Sigma	Symbol context	Set of symbol definitions
77	284	$'a$	Quote	Symbol reference	Symbol reference

32 Fluid Binding

N	P	Expression	Name	Say	Meaning
78	290	$\mu' \otimes a \hookrightarrow e$	Tensor product	Tensor product	Map symbol a to expression e
79	290	$\mu' \otimes a \hookrightarrow \bullet$	Tensor product	Tensor product	Symbol a is undefined
80	290	$\mu' \otimes a \hookrightarrow \_$	Tensor product	Tensor product	Symbol a is indeterminate

XIV. Mutable State

34 Modernized Algol

N	P	Expression	Name	Say	Meaning
81	308	$a := e$	Colon equals	Assign	Assign e to a
82	308	$*a$	Star	Reference	Get the value of a
82	309	$m \| \mu$	State	State	Command m with memory map μ
83	314	$m \sim \tau$	Dotted tilde	Dotted tilde	Command m returns a value of type τ