Index

OCFA, 101

★ “quick check” exercise, xii
★★ easy exercise, xii
★★★ moderate exercise, xii
★★★★ challenging exercise, xii
→ exercise without solution, xii

A-normal form, 253
abbreviations, see type definitions
abstract types, 245–289, see also modules, 454
access control with linear types, 3
adequacy, 277
admissible property, 259
affine types, see substructural types
Agda, 66
ALF, 68, 69
Alfa, 66
algebraic data types and type inference,
453–458
algorithmic type checking, see also undecidability
for the calculus of constructions, 66
for LF, 56–60, 62–63
for linear lambda-calculus, 11–14
aliasing, see also syntactic control of interference
and typed assembly language, 156
alias types, see typed assembly language applicative bisimilarity, 288
arrays
and typed assembly language, 170–171
in linear type systems, 24–28
AUTOMATH, 86, 384
avoidance problem, see signatures
Barendregt cube, see lambda cube
behavioral type systems, 105
bisimilarity, 288
bunched logic, see substructural logics
bunched types, see substructural types
C, 43, 90, 106, 133, 343
C#, 142
Calculus of Capabilities, 134
Calculus of Constructions, 64–71, 86
with dependent sum types, 69–71
Calculus of Inductive Constructions,
66–69
Caml, 389
capability types and typed assembly language, 175
Cayenne, 74, 305
CC, see Calculus of Constructions
CIC, see Calculus of Inductive Constructions
ciu-equivalence, 264, 288
Clean, 43, 389
CLI, see Common Language Infrastructure
closing substitution, 263
closure analysis, 100
closures in TAL, 168–170
CLU, 343
coherence, see modules
Common Language Infrastructure, 139, 142, 178
compilation, separate, see modules
compilation, type-preserving, 141
compiler optimizations
  enabled by affine and relevant types, 39–40
  enabled by linear types, 19
computational λ-calculus, 105
concatenation of records, see type inference
constraints, see also type inference
  for type inference, 407–422
  generation, 429–434
  solving, 438–450
containment rules, see substructural types
context
  evaluation, 256
context splitting, see substructural types
contextual equivalence, 249, 261–266
  vs. bisimilarity, 288
continuation-passing style and regions, 132
contraction, see structural properties
control flow analysis, 100
control flow safety, see typed assembly language
Coq, 66, 67, 86, 175, 384
cryptographic authentication infrastructure, see proof-carrying code
Curry-Howard correspondence
  and dependent types, 48–49
  and linear logic, 41
cut-off compilation, 303
Cyclone, 43, 90, 132–134, 174

Damas–Milner type system, 399–406
  relation with HM(X), 428–429
data types and type inference, 453–458
decidability, see undecidability
definitional equality, see equivalence checking
definitions, see type definitions
delta-reduction, 395
delta-reduction of type definitions, 354
Dependent ML, 74–82
dependent types, 45–86, see also LF,
  calculus of constructions, calculus of inductive constructions
  and typed assembly language, 171–172
decidable type checking for restricted systems, 75
higher-order abstract syntax and,
  implementation, 83–85
indexed types, 75
products, 46
semantics, 86
sums, 61–63, 69–71
sums vs. existentials (weak sums), 70
type inference, 82
undecidability of type checking, 74–75
with substructural types, 43
dependently vs. statically typed languages, 305
determinate module, 363
DML, see Dependent ML
dot notation for existential types, 308

ECC, see Extended Calculus of Constructions
Edinburgh Logical Framework, see LF
effect type systems, 89–90, 102–123,
  see also regions
  applications, 87
  and interference analysis, 105
  polymorphism, 114
  and protocol verification, 105
  region inference, 89–90
and soundness of value flow analysis, 104
with substructural types, 43
Tofte-Talpin type system, 89, 101, 114–123
value restriction and polymorphism, 123
effects, 390
equirecursive types, 454, 459
equivalence, see contextual equivalence, cuiequivalence
equivalence checking, 223–244
definitional equality, 54
for LF, 53–54
erasure
in region-based analysis, 111–114
in value flow analysis, 93–97
evaluation context, 256
evaluation frame, 257
exchange, see structural properties
exercises, difficulty ratings, xii
existential types, see also abstract types
in typed assembly language, 168
vs. Sigma types, 70
vs. signatures, 307, 308
in typed assembly language, 167
Extended Calculus of Constructions, 70
extensionality principle, 225, 249, 250, 252, 279
external name, see modules
external references between modules, 294
families of modules, see modules
families of signatures, see signatures
fibered signatures, see signatures
Finitary PCF, 90
first-class modules, see modules
foundational proof-carrying code (FPCC), see proof-carrying code
frame
evaluation, 257
functors, see modules
fundamental property, see logical relations
Galois connection, 267
garbage collection, see memory management
generalization of a type scheme, 402–404
generic programming, 345
Glasgow Haskell Compiler, 39, 43
Haskell, 43, 74, 334, 342, 344, 389
Herbrand universe, 411
hiding, see abstract types, modules
higher-order abstract syntax in dependent type systems, 49, 206
higher-order modules, see modules
HM(X), see type inference
Hope, 343
Howe’s method, 288
implicit syntax, see type inference and dependent types
incremental compilation, see modules
indexed types, see dependent types
inference, see type inference
information hiding, see abstract types, modules
instantiation of a type scheme, 402–404, 407, 408
interfaces, see signatures
interference, see aliasing
interference analysis via effect type systems, 105
internal name, see modules
isorecursive types, 289, 454, 458, 459
Java, 90, 141, 142, 187, 300, 303, 305, 343
Java Virtual Machine, 139, 142, 178, 189
judgments-as-types, see LF
Kripke logical relation, 237
lambda cube, 71–73, 86
language-based security, see proof-carrying code, typed assembly language
LCF, 389
LEGO, 66, 70, 85, 86
LF, 49–63, 86, 175
algorithmic type checking, 56–60, 62–63
with dependent sum types, 61–63
implicit, in proof-carrying code systems, 211–214
Linear, 42
in proof-carrying code systems, 205–214
linear lambda-calculus, 6–30
algorithmic type checking, 11–14
and arrays, 24–28
polymorphic, 20–24
with reference counting, 28–30
Linear LF, 42
linear logic, see substructural logics
linear types, see substructural types
linking, see modules
Lisp, 343
logical equivalence, 234
Logical Frameworks, see LF
logical relations, 223–289
fundamental property, 239–243, 274
history, 243–244
Kripke, 237
monotonicity, 235–237
operationally based, 266
and “recursive language features”, 289
manifest types, see type definitions
memory management, see also regions
with linear types, 7, 14
with linear types and regions, 42, 132
reference counting, 28–30, 41
reuse, 111
stack discipline, 30, 89, 99, 157
with substructural types, 4
and typed assembly language, 174
memory safety, see typed assembly language
Microsoft Common Language Infrastructure, see Common Language Infrastructure
mixin modules, 343
ML, 141, 142, 389–489
meanings of the term, 389
ML Kit, 90, 123, 128–130, 133
ML module system, see modules
ML type inference, see type inference
mobile code, see proof-carrying code, typed assembly language
Modula-2 and Modula-3, 343
modules, see also signatures
abstract type components, 307–317
applicative vs. generative functors, 336–338
coherence, 327–333
determinacy, 312–315
in existing programming languages, 341–343
external references between, 294
families of, 324–338
first-class, 312, 338–339
functors, 324–338
functors and determinacy, 336–338
hierarchies, 317–320
higher-order, 339–340
internal vs. external names, 296, 317–320
linking, 303–304
mixin modules, 343
ML module system, 341–342
phase distinction, 305–307
pragmatics of functors, 333–336
recursive, 341
separate and incremental compilation, 302–303
static vs. dynamic equivalence, 340
units, 343
monad, 105
monotonicity property of logical relations, 271
nominal vs. structural signature matching, 299
nonstructural subtyping, 412
normalize-and-compare algorithm for equivalence checking, 225
NuPrl, 54

object encodings in TAL, 168–170
Objective Caml, 342, 343
objects, type inference for, 459
occurs check, 439
opaque interface, 358
operational extensionality, see extensionality principle
operational reasoning using types, 245–289
ordered lambda-calculus, 30–36, 42
ordered logic, see substructural logics
ordered types, see substructural types

parameterized modules, see modules
parameterized signatures, see signatures
parametricity, see relational parametricity
parametric polymorphism, see polymorphism
Pascal, 343
PCC, see proof-carrying code
Pebble, 74, 305
phantom types, 455
phase distinction, see also modules
and dependent types, 75
phase splitting, see type definitions
Pi types, see dependent types
pointers, shared vs. unique, 157
polymorphic record update, 460
polymorphic recursion, 154, 452
polymorphic variants, 483–486
polymorphism, see also type inference
and regions, 110
in effect type systems, 114
in linear type systems, 20–24
and regions, 108
in typed assembly language, 146
in value flow analysis, 101
pre- and postconditions, in proof-carrying code, 184
principal signature, see signatures
principal type schemes, 405, 430
principal typings, 430
privacy, guaranteeing with PCC, 216–218
program analysis, type-based, 87–135
program equivalence, see typed operational reasoning
programming languages
C, 43, 90, 106, 133, 343
C#, 142
Caml, 389
Cayenne, 74, 305
Clean, 43, 389
CLU, 343
Cyclone, 43, 90, 132–134, 174
Dependent ML, 75–82
Haskell, 43, 74, 334, 342, 344, 389
Hope, 343
Java, 90, 141, 142, 187, 300, 303, 305, 343
LF, 86
Lisp, 343
ML, 141, 142, 389–489
ML Kit, 90, 123, 128–130, 133
Modula-2 and Modula-3, 343
Objective Caml, 342, 343
Pascal, 343
Pebble, 74, 305
Prolog, 90, 127, 134
Quest, 74
Russell, 305
Scheme, 305
Standard ML, 255, 341, 343, 345, 389
Vault, 43, 90
Prolog, 90, 127, 134
proof-carrying code, 139–140, 177–220
architecture, 178–180
beyond types, 216–218
costs, 211, 220
for cryptographic authentication, 219
efficient proof representation in implicit LF, 211–214
foundational, 155, 175, 178

guaranteeing privacy, 216–218

pre- and postconditions, 184

program annotation, 193

proof checking as LF type checking, 209–211

proof generation, 214–215

proof representation in LF, 205–214

safety policy, 182–187

and substructural types, 40

symbolic evaluation, 190–192, 194–195

vs. typed assembly language, 141, 155, 178, 189

verification condition generation, 187–190

propositions-as-types, see Curry-Howard correspondence

protocol verification with effect type systems, 105

pure type systems (PTS), 71–73

PVS, 74

qualified types, 488

qualifiers, see type qualifiers

Quest, 74

record operations, 460–489

record update and extension, polymorphic, 460

recursive definitions, 398

recursive modules, see modules

recursive types, see also modules, recursive

in linear type systems, 17

and type inference, 453–460

reference counting, see also memory management

in linear type systems, 28–30, 41

references, 390, 398, 435, 452, see also effects

regions, 87–135, see also effect type systems

and continuation-passing style, 132

erasure, 111–114

imperative, 131–132

inference, 89–90, 101, 123–127

lexically scoped, 89, 99–100

and linear types, 42, 132

polymorphic, 108–110

practical memory-management systems, 133–135

reuse of deallocated memory, 111

safety properties, 87, 106

and stack-oriented memory management, 89, 99

and typed assembly language, 173, 175

register file type, 146

relational parametricity, 245, 271, 286, 287

relevant logic, see substructural logics

relevant types, see substructural types

resource management, see memory management, regions

row variables, see type inference

Russell, 305

safety policy, see proof-carrying code

Scheme, 305

scheme, see type scheme

Scott induction, 259

sealing, see signatures, 362

security, see proof-carrying code, typed assembly language

separate compilation, see modules

set-based analysis, 101

Sigma types, see dependent types

signatures, see also modules

avoidance problem, 315–317, 365

dot notation, 307

vs. existential types, 307, 308

families of, 320–324

fibered vs. parameterized, 322–324

matching, 299

nominal vs. structural matching, 299

opaque, 307

principal, 298, 301

role in separate compilation, 295

sealing, 310–312
sealing, static vs. dynamic, 338
subsumption principle for, 299
translucent, 307–310
transparent, 307
singleton kinds, see type definitions
singleton types, 385
software fault isolation, 140
sorts in pure type systems, 72
stack typing, see typed assembly language
Standard ML, 255, 341, 343, 345, 389
statically vs. dependently typed languages, 305
strictness analysis, 43
strict types, see relevant types
strong sum types, see dependent types
structural properties, 4–6
contraction, 4, 11, 41
exchange, 4, 11, 31, 32
weakening, 4, 11, 41
structural subtyping, 412
structural vs. nominal signature matching, 299
structures, see modules
submodules, see modules
substructural logics, 40–42
substructural types, 3–43
affine types, 5, 36–40, 43
bunched types, 42
containment rules, 9, 33
context splitting, 9, 42
context splitting, algorithmic, 11
with dependent types, 43
with effect type systems, 43
linear types, 5–30, 41, 43
ordered types, 5, 30–36, 42
relevant types, 5, 39
temporal resource management, 36
uniqueness types, 43
subtyping, see also constraints
and typed assembly language, 173
co- and contra-variance, 412, 415
structural vs. nonstructural, 412
sum types, see also algebraic data types,
variant types
in dependent type systems, 61–63
in linear type systems, 17
surjective pairing, 62
symbolic evaluation, see proof-carrying code
syntactic control of interference, 41
syntax-directedness, see algorithmic type checking
TAL, see typed assembly language
TAPL, ix
temporal resource management with substructural types, 36
theorem provers
Agda, 66
ALF, 68, 69
Alfa, 66
AUTOMATH, 86, 384
Coq, 66, 67, 86, 175, 384
LCF, 389
LEGO, 66, 70, 85, 86
NuPrl, 54
PVS, 74
TIL, see typed intermediate language
Tofte–Talpin type system, see effect type systems
Touchstone PCC architecture, see proof-carrying code
translucent sums, see type definitions
transparent interface, 358
type-based program analysis, 87–135
type checking, see algorithmic type checking
type definitions, 347–385
for algebraic data types, 454
delta-reduction, 354
in module interfaces, 358–367
manifest types, 358
phase splitting, 378–384
primitive, 351–358
singleton kinds, 367–384
translucent sums, 358–367
type inference, 389–489
and algebraic data types, 453–458
and dependent types, 82
HM(X), 389–489
objects, 459, 461, 486
polymorphic variants, 483–486
records, 460–489
and recursive types, 453–460
regions, 101, 123–127
regions and effect types, 89–90
row variables, 460–489
in typed assembly language, 154
and value flow analysis, 97–98
type-preserving compilation, 141
type qualifiers, 7
  linear qualifier, 7
  ordered qualifier, 32
  quantification over, 21
  reference counting qualifier, 28
  unrestricted qualifier, 7
type reconstruction, see type inference
type scheme, 402–404, 407
typed assembly language, 139–175
  and aliasing, 156
  and alias types, 157
  and arrays, 170–171
  and capability types, 175
closures, 168–170
compiling to, 164–172
control flow safety, 142–155
and dependent types, 171–172
encoding objects, 168–170
ensuring memory safety, 155–172
and existential types, 167–168
memory management, 174
and origins of Cyclone, 134
polymorphism, 146
vs. proof-carrying code, 141, 155, 178, 189
and regions, 173, 175
stack-allocated memory, 157
and substructural types, 40
and stack typing, 173
and subtyping, 173
TALT, 173–175
TALx86, 170, 171, 173, 174
and type inference, 154
typed intermediate language, 142, see also typed assembly language
typed operational reasoning, 245–289
typed operational semantics, 86
type and effect systems, see effect type systems
Types and Programming Languages, ix
undecidability
  of dependent type checking, 54, 74–75
  of module type systems, 339
  of type inference with polymorphic recursion, 452
unification, 439–442
  with record types, 476
uniqueness types, see substructural types
units, 343
untrusted code, see proof-carrying code, typed assembly language
unwinding property, 259–260
UTT, 86
value flow analysis, 88, 90–102
  constraint-based, 101
  erasure, 93–97
  polymorphic, 101
  soundness with effect types, 104
  type inference, 97–98
  unsoundness without effect types, 88, 99
value restriction, 255, 437
  in effect type systems, 123
variants, polymorphic, 483–486
Vault, 43, 90
verification conditions, see proof-carrying code
weak head normalization, 57, 230
weak sum types, see dependent types
weakening, see structural properties
web resources, xii