
Index

Symbols

`:=` operator, 65
`<-` operator, 65

A

acknowledgments, xi
additional resources, ix
Amdahl, Gene, 2
Amdahl's law, 2
anonymous functions, 37
atomicity, 6-7

B

batch processing, 103
bridge-channels, 122
Broadcast method, 55
bugs, 149
(see also error handling)

C

cancellations (see timeouts and cancellations)
channels
best use of, 64
bidirectional, 65
blocking channels, 67, 126
bridge-channels, 122
buffered, 70-74
closing, 68
creating, 65
data types constraints, 66
history of, 26
multiple goroutines, 69, 114
mutexes and, 29

nil, 74
operations reference chart, 75
or-channel, 94-97
or-done-channels, 119
ownership assignment, 76
ranging over, 69
reading from, 68
reading from closed, 68
tee-channels, 120
unbuffered, 71
unidirectional, 65
writing to, 67
checkStatus, 97-100
chunking, 127
circular wait, 12
cloud computing, 3
code examples, obtaining and using, x
Coffman Conditions, 12
Coffman, Edgar, 12
comments and questions, xi
community-oriented resources, ix
concurrency
common issues
atomicity, 6-7
deadlocks, 10-13
determining concurrency safety, 18-20, 85
livelocks, 13-15
memory access synchronization, 8-10, 17, 29, 47
race conditions, 4-6, 214
starvation, 16-18
coroutines and, 38
definition of term, 1

fork-join model, 39-47
Go's approach to, 29-31
Go's philosophy on, 31-35
history of, 2
vs. parallelism, 23-26, 30

concurrency patterns
bridge-channels, 122
confinement, 85-89
context package, 131-145
empty interfaces and, 85
error handling, 97-100
fan-out, fan-in, 114-119
for-select loops, 89
or-channel, 94-97
or-done-channels, 119
pipelines, 100-114
preventing goroutine leaks, 90-94
queuing, 124-131
tee-channels, 120

concurrency primitives
channels, 64-78
Cond type, 52-57
GOMAXPROCS function, 83
goroutines, 37-47
select statement, 31
select statements, 78-82
sync package, 47-64
Cond type, 52
Mutex and RWMutex, 49
once variable, 57
Pool, 59
WaitGroup, 47

Cond type, 52-57
confinement
ad-hoc, 86
benefits and drawbacks of, 86
immutable data and, 85
lexical, 40, 59, 87-89

contact information, xi

context, 6-7, 24

context package
benefits of, 131
cancelling call-graph branches with, 132
Context type, 131
Deadline method, 139-141
vs. done channel pattern, 136-139
Done method, 132
example, 134
functions in, 133

guidelines for use, 144
purposes served by, 132
transporting request-scoped data with, 141-143

context switching, 45
coroutines, 38
critical sections, 8-10

CSP (Communicating Sequential Processes)
concept of, 26-29
concurrency vs. parallelism, 23-26
Go's approach to concurrency, 29-31
Go's philosophy on concurrency, 31-35

D

data race, 4-6
databases, goroutine cancellations and, 159
DDoS (distributed denial of service) attacks, 175
deadlocks, 10-13, 156
death-spirals, 128
debugging
goroutine errors, 213
healing unhealthy goroutines, 188-194
pprof profiler, 216
race detection, 214

decision-tree, 33

Dijkstra, Edgar, 28

double-ended queues (deques), 199

duplicate messages, 159-161

E

embarrassingly parallel, 2

empty interfaces (`interface{}`)
benefits of, 85
chan `interface{}` variable, 65
pipeline interfaces, 111

errata, xi

error handling
complete example, 150-155
components of, 148
displaying errors to users, 150
error anatomy, 213
error categories, 149
error correctness, 150
error packages, 155
importance of, 147
multiple module example, 149
responsibility for, 97-100

F

fair scheduling strategy, 197
fan-out, fan-in technique, 114-119
files, goroutine cancellations and, 159
for-select loops, 89
fork-join model
 benefits of, 43
 child branches, 39
 closures and, 40
 context switching and, 45
 graphical representation of, 39
 interdependency of tasks in, 198
 join points, 39
 memory management in, 42
 parent cancellations, 157
 synchronization in, 43
tasks vs. continuations, 204-210

G

garbage collection, 20
generators, 106-114
Get method, 59
Go
 benefits of, 20-21
 channels, 28-29, 64-78
 concurrency patterns in, 85-145
 concurrency primitives, 37-84
 memory management in, 42
 online resources, ix
 origins of, 197
 vs. other languages, 29-31
 philosophy on concurrency, 31-35
 decision-tree, 33
 prerequisites to learning, viii
 select statement, 31
 Wiki, 32
go keyword, 212
GOMAXPROCS function, 83
goroutines
 benefits of, 42, 212
 creating, 37
 error handling, 213
 healing unhealthy, 188-194
 main goroutine, 37
 multiple, 69, 114
 operation of, 38-47
 preventing leaks, 90-94
 restarting, 188-194
 scheduling, 197-212

termination of, 90
guarded commands, 28

H

healing unhealthy goroutines, 188-194
heartbeats
 before each unit of work, 166-168
 demonstration of, 162-166
 deterministic tests using, 169
 interval-based, 168, 170-172
 roles of, 161, 166, 168
 types of, 161
higher order functions, 102
Hoare, Tony, 26-29

I

immutable data, 85
indivisible, 7

L

leaks, preventing, 90-94
Little's law, 129
livelocks, 13-15

M

M:N scheduler, 39
make function, 65
memory access synchronization, 8-10, 17, 29, 47
memory management, 20, 42
monads, 102
Moore, Gordon, 2
Moore's law, 2
multiplexing, 90, 117
Mutex and RWMutex, 49-52
mutexes, 29
mutual exclusion, 12

N

negative feedback loops, 128
no preemption, 12

O

object pool pattern, 59
once variable
online resources, ix
or-channel, 94-97

or-done-channels, 119

order-independence, 115

OS threads, 25

P

parallelism, 23-26, 30

parent cancellations, 157

patterns (see concurrency patterns)

pipelines

- batch processing, 103

- benefits of, 100

- best practices for constructing, 104-109

- definition of term, 101

- empty interfaces and, 111

- handy generators, 109-114

- properties of, 102

- stages, 101

- stream processing, 103

pool pattern, 59-64

pprof profiler, 216

preemptability, 157-159

primitives (see concurrency primitives)

process calculi, 27

Q

questions and comments, xi

queueing

- benefits and drawbacks of, 124

- best use of, 129

- blocking and, 126

- chunking and, 127

- decoupling ability of, 126

- Little's law, 129

- negative feedback loops, 128

- performance concerns and, 124

- using, 126

R

race conditions, 4-6, 214

rate limiting

- benefits of, 175

- definition of term, 174

- implementing, 176-183

- purpose of, 174

- single vs. aggregate, 183-188

replicated requests, 157, 172-174

resources, ix

runtime/pprof package, 216

RWMutex and Mutex, 49-52

S

scaling concurrent operations

- dynamic scaling, 31

- error propagation, 147-155

- healing unhealthy goroutines, 188-194

- heartbeats, 161-172

- horizontal scaling, 3

- rate limiting, 174-188

- replicated requests, 172-174

- timeouts and cancellation, 155-161

select statement, 31, 78-82

shared state, 159

Signal method, 55

spigot algorithms, 2

stale data, 156

stalling joins, 205

starvation, 16-18

stream processing, 103

Sutter, Herb, 3

sync package

- Cond, 52-57

- memory access synchronization, 8-10, 17,

- 29, 47

- Mutex and RWMutex, 49-52

- once variable, 57-59

- Pool, 59-64

- WaitGroup, 47

system saturation, 155

T

tee-channels, 120

thread pools, 21, 26, 29

timeouts and cancellations

- duplicated messages and, 159-161

- preemptability and, 157-159

- role of cancellations, 157

- role of timeouts, 155

- shared state modifications, 159

tools and commands

- goroutine errors, 213

- pprof profiler, 216

- race detection, 214

typographical conventions, x

U

uninterruptible, 7

W

wait for condition, [12](#)
WaitGroup, [47](#)
web scale, [3](#)

work cancellation, [90](#), [155](#)

(see also timeouts and cancellations)
work stealing strategy, [197-212](#)