__global__, 196
__shared__, 202
__shfl_down_sync, 201
__syncthreads(), 200
__syncwarp, 200
__threadfence(), 201
__threadfence_block(), 201
__threadfence_system(), 201
__align__, 204
4-Cover Property Analysis, 245
accelerated cascading, 227, 228
accelerated cascading: find minima, 227
acquire lock, 90
actors, 38
address space, 75
affinity, 125
Amdahl’s law, 64
application of prefix-sum: radix-sort, 249
associative caches, 9
asymptotically cost-optimal, 58
asynchronous communication, 102
asynchronously, 36
atomic, 78, 182
atomic pragma, 146
atomicAdd, 200
atomicAdd_block, 200
atomicAdd_system, 200
atomicCAS, 200
atomicity, 34
average concurrency, 123
Bakery algorithm, 94
bank conflicts, 205
Banyan switch, 20
barrier, 48, 145
barrier and consensus, 97
basic merge-sort, 238
binary tree computation, 212
binomial trees, 260
bisection bandwidth, 16
bisection width, 16
block decomposition, 116
block of threads, 195
blockCyclic, 181
blockDim, 196
blockIdx, 196
blocking network, 18
Brent’s work-time scheduling principle, 63
BSP, 47
BSP computation time, 48
BSP example, 49
bucket, 255
bulk-synchronous parallel model, 47
busy-wait, 87
cache, 8
cache coherence, 10
cache replacement policy, 9
cache-lines, 9
cache-miss, 8
capture, 146
causal consistency, 82
causality, 82
centralized protocol, 86
centralized push scheduling strategy, 127
Chapel, 180
Chapel tasks, 181
Chapel variable scope, 183
Cloth network, 21
colorall, 182
collective communication, 102
collective communication primitives, 103
communication, 99
communication edges, 122
communicator, 156
compare and swap, 95
concurrent kernels, 198
consensus, 97
consensus number, 95
contention, 13
cooperative groups, 201
cost, 58
cost-optimal, 58
counting locks, 90
cover, 242
CPU architecture, 5
CPU–GPU memory transfer, 197
CPUs, 4
CRCW, 53
CREW, 53
critical path, 42, 123
critical section, 90, 145
cross-bar network, 19
CUDA, 194
CUDA parallel memory access, 203
CUDA programming model, 195
CUDA shared memory, 202
CUDA synchronization, 199
cudaCpuDeviceId, 197
cudaDeviceSynchronize, 195, 198
cudaMalloc, 195
cudaMallocManaged, 195
cudaMemcpy, 197
cudaMemcpyAsync, 197
cudaMemcpyHostToDevice, 199
cudaMemAdvise, 197
cudaMemAlloc, 197
cudaMemAllocDefault, 199
cudaMemAdviseHostToDevice, 199
cudaStreamSynchronize, 198
cyclic decomposition, 116
data and function on device, 191
data decomposition, 116
data domain, 181
data races, 96
data-parallel, 37
deadlock, 88
debugging and profiling, 132
decare simd pragma, 152
deep pipeline, 11
depend, 149, 155
dependency edges, 122
design steps, 112
dim3, 196
direct memory access, 5
direct-mapped caches, 9
dist_schedule, 194
distribute pragma, 193
distributed protocol, 86
distributed push scheduling, 129
distributed-memory architecture, 4
distributed-memory programming model, 32
divide and conquer: merge two sorted lists, 217
divide and conquer: prefix-sum, 212
DMA, 5, 100
domain decomposition, 116
domain map, 181
domains, 181
dynamic task generation, 121
efficiency, 59
EREW, 52
Euler tour, 230
evicted, 8
exclusive scan, 217
execution homogeneity, 123
exploiting parallelism: minimum spanning tree, 257
exploiting parallelism: quick-sort, 250
exploratory decomposition, 121
fair, 89
false-sharing, 11, 206
FIFO and Processor Consistency, 82
FIFO eviction policy, 9
firstprivate, 142, 148
fixing processor count: sample-sort, 254
flush, 84, 142, 173
for, 148, 150, 154
for pragma, 147
for simd pragma, 152
forall, 182
fork, 76
fork-join model, 140
fragment, 32
full-duplex links, 13
functional decomposition, 116, 120
get, 169
GPU architecture, 11
GPU programming, 188
GPUs, 5
granularity, 36, 112
graph cut, 118
Gustafson’s Law, 65
Hadoop, 186
half-duplex, 13
HDFS, 186
hypercube network, 18
input/output, 130
instruction dependency, 7
instruction-level parallelism, 7
interconnect architecture, 13
iso-efficiency, 60
Karp–Flatt Metric, 66
kernel, 195
lastprivate, 148
latency, 57
latency and throughput, 57
linear, 148
linearizability, 85
link, 13
load balancing, 115, 125
locales, 180
lock, 90
lock-free, 87
LRU eviction policy, 9
makespan, 124
map, 184
map-reduce, 184
maximum speed-up, 58
memory and cache, 8
memory atom, 203
memory fence, 84
merge operation per tick, 248
message-passing, 32
message-passing synchronization, 158
MIMD: Multiple Instruction, Multiple Data, 3
minimal cut, 119, 126
model parallelism, 38
MPI, 155
MPI barrier, 167
MPI collective communication, 164
MPI data types, 161
MPI dynamic parallelism, 177
MPI file IO, 173
MPI groups and communicators, 176
MPI process topology, 178
MPI reduction, 167
MPI send and receive, 156
MPI_Accumulate, 173
MPI_Allgather, 165
MPI_Allreduce, 169
MPI_Alltoall, 166
MPI_Alltoallv, 166
MPI_Alltoallw, 166
MPI_ANY_SOURCE, 157
MPI_Barrier, 167
MPI_Bsend, 158
MPI_Cart_create, 179
MPI_Cart_shfood, 179
MPI_Comm_create, 176
MPI_Comm_create_group, 177
MPI_Comm_get_parent, 178
MPI_Comm_rank, 156
MPI_Comm_spawn, 177
MPI_Comm_split, 177
MPI_COMM_WORLD, 156
MPI_Compare_and_swap, 173
MPI_Escan, 167
MPI_File_iread, 176
MPI_File_iwrite, 176
MPI_File_open, 175
MPI_File_read_all, 175
MPI_File_set_atomicity, 175
MPI_File_set_view, 175
MPI_File_sync, 175
MPI_File_write_all, 175
MPI_Group, 176
MPI_Group_union, 176
MPI_Ibcast, 165
MPI_Ibsend, 159
MPI_IN_PLACE, 165
MPI_Info, 175
MPI_INFO_NULL, 175
MPI_Init_thread, 156
MPI_INIT, 157
MPI_Init, 160
MPI_Info, 175
MPI_INIT_THREAD, 156
MPI_MODE_NOPUT, 171
MPI_MProbe, 160
MPI_Irecv, 159
MPI_Isend, 159
MPI_Issend, 159
MPI_Reduce, 167
MPI_Reduce_scatter, 169
MPI_Request_free, 160
MPI_Send, 159
MPI_Ssend, 159
MPI_Ssend, 159
MPI_SYSFS, 186
MPI_ANY_SOURCE, 157
MPI_Barrier, 167
MPI_Bsend, 158
MPI_Cart_create, 179
MPI_Cart_shfood, 179
MPI_Comm_create, 176
MPI_Comm_create_group, 177
MPI_Comm_get_parent, 178
MPI_Comm_rank, 156
MPI_Comm_spawn, 177
MPI_Comm_split, 177
MPI_COMM_WORLD, 156
MPI_Compare_and_swap, 173
MPI_Escan, 167
MPI_File_iread, 176
MPI_File_iwrite, 176
MPI_File_open, 175
MPI_File_read_all, 175
MPI_File_set_atomicity, 175
MPI_File_set_view, 175
MPI_File_sync, 175
MPI_File_write_all, 175
MPI_Group, 176
MPI_Group_union, 176
MPI_Ibcast, 165
MPI_Ibsend, 159
MPI_IN_PLACE, 165
MPI_Info, 175
MPI_INFO_NULL, 175
MPI_Init_thread, 156
MPI_INIT, 157
MPI_Init, 160
MPI_Init, 160
MPI_Init, 160
MPI_Init, 160
MPI_Init, 160
MPI_Init, 160
MPI_Init, 160
MPI_Init, 160
MPI_Init, 160
| MPI_Rget, 171 |
| MPI_Rput, 171 |
| MPI_Rsend, 158 |
| MPI_Scan, 167 |
| MPI_Scatter, 166 |
| MPI_Ssend, 158 |
| MPI_Status, 158 |
| MPI_STATUS_IGNORE, 160 |
| MPI_SUM, 168 |
| MPI_Test, 160 |
| MPI_Type_Ccommit, 163 |
| MPI_Type_create_darray, 164 |
| MPI_Type_create_struct, 163 |
| MPI_Type_create_subarray, 164 |
| MPI_Wait_all, 160 |
| MPI_Win_allocate, 170 |
| MPI_Win_attach, 170 |
| MPI_Win Completeme, 172 |
| MPI_Win_create_dynamic, 170 |
| MPI_Win_create, 170 |
| MPI_Win_flush, 173 |
| MPI_Win_lock_all, 172 |
| MPI_Win_post, 172 |
| MPI_Win_start, 172 |
| MPI_Win_wait, 172 |
| mpiexec, 156 |
| MST, 263 |
| mutual exclusion, 90 |
| name-space., 4 |
| network bandwidth, 15 |
| network comparison, 24 |
| network degree, 15 |
| network path length, 16 |
| network throughput, 15 |
| NIC, 100 |
| nonblocking, 87 |
| nonblocking network, 14 |
| nondeterministic., 77 |
| nowait, 148, 150 |
| num_teams(count), 193 |
| num_threads, 140 |
| NUMA, 3 |
| omega network, 20 |
| omp_get_num_devices, 190 |
| omp_get_team_num, 194 |
| omp_get_thread_num, 194 |
| one-sided communication, 169 |
| OpenMP, 139 |
| OpenMP GPU Off-Load, 188 |
| OpenMP Memory Model, 141 |
| OpenMP Reduction, 143 |
| OpenMP Synchronization, 144 |
| OpenMP Thread Creation, 140 |
| ordered, 148 |
| packets, 13 |
| pages, 75 |
| parallel p-ary search, 223 |
| parallel file system, 131 |
| parallel for, 150 |
| parallel for simd, 152 |
| Parallel Implementation, 185 |
| parallel merge, 218, 219, 222, 226 |
| parallel performance evaluation, 57 |
| parallel pragma, 140 |
| parallel prefix-sum, 214, 215 |
| parallel priority queue, 260 |
| parallel sections pragma, 150 |
| parallel work, 62 |
| partitioned global address space, 180 |
| partitioning, 222 |
| Peterson’s algorithm, 91 |
| PGAS, 180 |
| pinned memory, 101 |
| pipeline, 7 |
| pipeline decomposition, 122 |
| Pipeline parallelism, 38 |
| Pipelined Merges, 240 |
| pipelining, 238 |
| Pipelining: Merge-Sort, 238 |
| point-to-point communication, 99 |
| pointer jumping, 230 |
| ports, 13 |
| PRAM, 52 |
| PRAM computation time, 55 |
| PRAM consistency, 83 |
| PRAM example, 55 |
| PRAM model, 52 |
| preliminary task mapping, 125 |
| private, 141, 148 |
| private variables, 141 |
| process, 75 |
| processor consistency, 83 |
| producer-consumer, 81 |
profiling, 133
program order, 78
progress, 86
protocol control, 86
pull scheduling, 129
put, 169

race condition, 77
race condition and consistency of state, 77
rank, 156, 218
Ranklist, 220
RDMA, 100
re-entrant lock, 90
ready send, 158
receive, 157
recursive decomposition, 120
recursive doubling, 230
recursive doubling: connected components, 233
recursive doubling: Euler tour, 231
recursive doubling: list ranking, 230
reduce, 184
reduction, 51, 103, 142, 148
registers, 5
release lock, 90
remote memory access, 169
remote procedure call, 102
request, 159
retires, 7
routing, 13
RPC, 102, 188

safe, 88
sampling, 255
scalability, 57
scan, 217
schedule, 148, 152
scheduled, 125
section, 150
send, 157
separators, 255
seq_cst, 146, 147
sequential consistency, 78
sequentially consistent, 79
shared, 141
shared variables, 141
shared-memory architecture, 3
shared-memory programming model, 52

sharing a loop’s work, 147
short-circuiting, 230
shuffle-exchange network, 20
signal-wait, 87
SIMD, 2
SIMD lanes, 152
SIMD Pragma, 151
simple parallel model, 46
simplex links, 13
single pragma, 151
SISD, 2
speculative decomposition, 122
speed-up, 58
SPMD, 155
starvation, 88
status, 158
storage servers, 131
storage targets, 131
stream processing, 38
streams, 198
strong scaling, 60
summary, 39, 67, 104, 133, 207, 264
super-steps, 47
symmetry breaking, 235
sync, 182
synchronization, 114
synchronization condition, 86
synchronization hazards, 88
synchronization variables, 84
synchronizing, 182
synchronous communication, 102
system architecture, 4
tag, 157
target data, 191
target pragma, 189
task, 36, 153
task cost variance, 123
task decomposition, 115
task execution, 124
task graph degree, 123
task graph metrics, 123
task graph model, 35
task pragma, 153
task scheduling framework, 126
taskwait pragma, 153
taskyield pragma, 153
teams pragma, 193
282 Index

thrashing., 113
thread blocks in OpenMP, 193
thread_limit(count), 193
threadprivate pragma, 142
threads, 76
threads and processes, 75
throughput, 57
torus network, 16
transactional memory, 96
transactions, 85
tree network, 22
types and quality of networks, 14
variants of task parallelism, 37
vectorized, 152
virtual memory, 75

volatile, 201
voting, 200
wait-free, 87
warp, 195
weak consistency, 84
weak scaling, 60
window, 169
words, 3
work stealing, 130
work-optimal, 63
work-sharing pragmas, 150
work-time optimal, 63
work-time scheduling principle, 63
write-back, 9
write-through, 9