Index

A
Accelerator vector, 252
AChem. See Artificial chemistry (AChem)
ACO. See Ant colony optimization (ACO)
Adaptive resolution genetic algorithm (arGA)
arLS kernel, 94–95
benchmarking problems, 97
constraint handling, 88–89
generation/population effect, 98, 99
genetic search, 93–94
GPU implementation, 100–103
information entropy, 90–91
input parameters, 96–97
iterations effect, 99, 100
local search algorithm, 92
local search probability effect, 98, 99
micro GA, 89–90
MINLP problem
ACO, 86
advanced genetic operators, 86
batch plant design problem, 86
BBO, 84
continuous and integer decision variables, 84
deterministic techniques, 86
equality and inequality constraints, 84
GENO, 87
hybridization of GA, 86
IGA approach, 87
mathematical form, 84
queue size effect, 99, 100
recursive behavior, 90, 91
redundancy avoidance, 92–93
RNG kernel, 96
variables encoding and genetic operators, 87–88
Amdahl’s law, 10–11, 40, 45, 242
Ant colony optimization (ACO)
cAS
colony model, 183, 184
cunning ant (c-ant), 183
donor ant (d-ant), 183
sequential description, 184
compute unified device architecture
cAS computation time, QAP, 188
kernel function, 187, 188
MINLP problem, 86
multi-GPU models
experimental setup, 196–197
island models, 194–198
master/slave model, 196, 198–200
QAPs (see Quadratic assignment problems (QAPs))
traveling salesman problem, 182
TS (see Tabu Search (TS))
Archive-based stochastic ranking evolutionary algorithm (ASREA), 268–269
child_pop, 272
description, 270–271
flow chart, 271
function evaluation on GPU, 271–272
global memory, 272
parallel SIMD ranking on GPU, 273
selection strategy, 274
arGA. See Adaptive resolution genetic algorithm (arGA)
Artificial chemistry (AChem)
algorithms
FRM, 397
Gillespie SSA, 395–397
NRM, 397
spatial simulation, 399–402
r-leaping method, 397–399
collision, 391–392
definition, 391
diffusion coefficient, 393–394
GP-GPU, 402–403
matrix notation, 392
parallelization
computational efficiency, 410–411
deterministic algorithm, 404–405
e xtinct fraction, 409–410
first kernel, 407
global state, 411–412
load balancing strategy, 413
Mersenne Twister algorithm, 413–414
optimization, 414–415
predator snapshots, 408–409
prey population, 399, 407
pseudorandom number generators, 413–414
reaction–diffusion system, 407–408, 411
spatial methods, 413
stochastic algorithm, 405–407
synchronization, 412
time constraints, 413
τ-leaping, 412–413
total runtime, 411–412
predator–prey interaction, 392–393
reaction/diffusion effects, 393
stoichiometric matrix, 392
ASREA. See Archive-based stochastic ranking evolutionary algorithm (ASREA)

B
Base evolutionary model, 108–109
Bioinformatics and computational intelligence, 315–316
Bi-population GA scheme (bGA), 111
Black-box optimizers (BBO), 84
Breast cancer, 316–317

C
Cartesian genetic programming (CGP)
classification
average CPU speed, 264
fitness function, 259
fitness scores, 263
global and shared memory, 260–263
GPU interpreter, 254–256
GPU.NET, 253–254
instructions code, 256–257
interpreters, 257–258
program length, 258
sizing for interpreter parameters, 258
speed results, 259
TidePowerd, 257
vector, 259–260
genotype structure, 250, 251
on GPUs, 252–253
INPP functions, 251
node function, 250
OUTPUT functions, 251
SKIPINP functions, 251
Cellular automata, 252
Cg compiler, 232
CGMC method. See Coarse-Grained Monte Carlo (CGMC) method
CGP. See Cartesian genetic programming (CGP)
CHC method, 110–111
Coalescent memory accesses, 9
Coarse-Grained Monte Carlo (CGMC) method, 405–407
Connection genes, 250, 251
Core i7, 430, 434, 438, 439
C2050 Tesla
components of, 317–318
gene tic programming, 335–339
Cunning Ant system (cAS)
colony model, 183, 184
computation time, QAP, 188
cunning ant (c-ant), 183
donor ant (d-ant), 183
sequential description, 184

D
Delta coding method, 110
Differential evolution (DE)
 basic operations, 123
crossover operator, 129
CUDA-C language, 129
CUDA-C program structure and memory hierarchy, 127, 128
differential mutation, 123–125
function optimization, 131–132
implementation, 129–131
independent task scheduling
average fitness value, 141, 142
consistency, 141
CPU and GPU implementations, 140, 141
ETC matrix, 139
FCFS method, 139
fitness function, 140
flowtime, 139
inconsistency, 141
machine heterogeneity, 141
meta-task execution time, 139
task heterogeneity, 141
ThM*C* matrices, 143
TIM*C* matrices, 143, 144
initialization strategy, 124
LOP
applications, 133
cost function, 132
data sets, 133–134
graph problem, 132
heuristic algorithms, 134–135
LOLIB solutions, 135–137
normalized LOLIB library, 135, 137, 138
permutation, 132, 133
MCMC method, 129
mutation operator, 129
N real-valued vectors, 122
recombination/crossover operator, 125
scaling factor, 123
selection strategy, 124–125
target vectors, 123
trial vectors, 122–123
variants and applications, 125–127
Direct method (DA). See Gillespie stochastic simulation algorithm
Docking accuracy, 362–364
Dominance-checking algorithm, 295, 299

E
Easy specification of evolutionary algorithm (EASEA)
EASEA-CLOUD platform, 37
evolutionary trends, 36
four-connected three-dimensional net
AFX framework, 375–376
constants definition, 372
description, 370
genetic algorithm (GA), 371
genome speedup, 374–375
GPU card, 372
high-throughput (HT) method, 371
inputs, 372, 374
MCBH, 371
Rietveld refinement technique, 370–371
simple fitness function, 372–373
simulated annealing (SA), 371
solution view, 375, 377–379
subset of, 377, 380–381
unit cell configuration, 375, 377
GPU island model, 53–57

GPU parallelized evolutionary algorithm, 68–69
MFI zeolite
fitness evolution, 384–385
machines, 382–383
speedup for, 385
parallelization, 378, 380–382
-cuda command line option, 42–44
ezGPU, 45–47
graphic display, 43, 44
intrinsically parallel sector, 41
island parallelization (see Island parallelization)
nvcc NVIDIA C compiler, 43
Rosenbrock function, 45, 46
sequential and parallel algorithms, 41
sequential CPU algorithm, 43
360x speedup, 43, 45
weierstrass.ez source file, 41–43
versions, 37
Weierstrass benchmark function
crossover function, 38
evaluation function, 38
GenomeClass, 37
initialization function, 37–38
mutation function, 39
set of parameters, 39–40
Easy specification of evolutionary algorithm (EASEA)-CLOUD platform, 37
Estimated time to complete (ETC) matrix, 139–143, 163, 165, 172
Evolutionary strategy plus (ES+), 41

F
Fine-grained parallelization, 440–442
First-come first-served (FCFS) method, 139, 157–158
First reaction method (FRM), 397
Flexible molecular docking, 350–352
Flynn’s taxonomy, 18
Four-connected three-dimensional net
AFX framework, 375–376
constants definition, 372
description, 370
genetic algorithms (GA), 371
genome speedup, 374–375
GPU card, 372
high-throughput (HT) method, 371
inputs, 372, 374
MCBH, 371
Rietveld refinement technique, 370–371
simple fitness function, 372–373
simulated annealing (SA), 371
solution view, 375, 377–379
subset of, 377, 380–381
unit cell configuration, 375, 377

Function gene, 250

G

G80 chip, 19
GDE. See Guided differential evolution (GDE)
GeneChip data mining
breast cancer, 316–317
genetic programming (see Genetic programming (GP))

General evolutionary numerical optimizer (GENO), 87
Generic local search algorithm. See Memetic algorithms (MAs)

Genetic algorithm (GA)
accuracy
average number of generation, 427–429
comparison of, 426–428
experimental parameters, 426
mutation, 426–427
optimum solution, 427–428
crossover operation, 424–425

easy specification of evolutionary algorithm (EASEA), 370
parallel GA model (see Parallel genetic algorithm model)

Genetic programming (GP)
Affymetrix probe, 333–335
aircraft model, principles of, 243–244
aircraft nonlinear state space, 244–245
Cg compiler, 232
considered trajectory, functions, 245–247
constants, 325–326
C2050 Tesla, 335–339
CUDA code, 322, 324
evaluation function
evaluation time vs. fitness cases, 236–237
function set influences, 238–239
population size, 241
tree construction methods, 239–240
tree depth influence, speed up, 237–238
experimental process, 235
Fermi L1 caches, 328, 330
fitness calculation, 326, 328–330
GA/ES, 228
GP-tree population and problem-related learning set, 229, 230
8800GTX card, 232
implementation
algorithm, 233–234
evaluation, 234–235
input classifier, 334, 337
large-scale data mining, 331–333
master-slave model implementation, 228
MIMD parallelization, 230–231
.Net and MS Accelerator, 232
NVIDIA hardware, 233
parameters, 333
population, 330–331
RapidMind, 232
single-instruction multiple data interpreter, 321–325
stack interpreter, 323–324
theoretical speedups, 242–243
thread divergence, 324, 327–328
thread layout, 321, 325–327
time repartition, 228
training data, 327, 334, 336

Genetic programming operations per second (GPOps/s), 259
GENITOR algorithm, 109
GF110 chips, 20
Gillespie multiparticle method (GMP), 401, 406
Gillespie stochastic simulation algorithm, 395–397
GK110 chips, 9
Global and shared memory interpreter type, 263
interpreting on GPU, 261
program length, 262
Regs array, 260
Global memory coalescing, 155
GMP. See Gillespie multiparticle method (GMP)

Graphics processing units (GPUs)
ACO (see Ant colony optimization (ACO)), bioinformatics and computational intelligence, 315–316
breast cancer, 316–317
components of
C2050 Tesla, 317–318
Fermi design, 319
NVIDIA multiprocessor, 318–319
stream processor (SP), 318–319
computation
CUDA, 181–182
single instruction, multiple threads, 181
CPU clock speed, 312
development, 6
experimental platform, 191–192
experimental results, 193–194
Genetic Programming (GP)
  Affymetrix probe, 333–335
  constants, 325–326
  C2050 Tesla, 335–339
  CUDA code, 322, 324
  Fermi L1 caches, 328, 330
  fitness calculation, 326, 328–330
  input classifier, 334, 337
  large-scale data mining, 331–333
  parameters, 333
  population, 330–331
  single-instruction multiple data interpreter, 321–325
  stack interpreter, 323–324
  thread divergence, 324, 327–328
  thread layout, 321, 325–327
  training data, 327, 334, 336
history and origins, 18
market, 17
memory latency, 318–321
Moore’s law, 312–313
on-chip transistor, 337
OpenCL application, 340
SIMD, 313–315
systolic genetic search
  compute capability, 211, 212
  crossoverAndMutation kernel, 213
  elitism, exchange, and evaluate kernels, 214
  memory multiprocessors, 211
  SIMT, 211
  structure of, 212, 213
  threads organization, 212, 213
TS (see Tabu Search (TS))
GT200 chips, 17
GTX 460/CUDA programming, 429
GTX295 nVidia card vs. 3.6 GHz Pentium IV
genome transfer time, 77–78
gettimeofday() POSIX function, 74
Rosenbrock function, 74–75
speedup, 75–77
GTX275 NVIDIA card vs. Intel XEON 550 CPU, 53
Guided differential evolution (GDE)
  CUDA vs. MolDock parallelisation, 358–359
  MolDock, 352–353
H
High-throughput (HT) method, 371
Hill-climbing algorithm, 92
Hyper-threading, 21, 28–29
I
Intel processors, 3, 4
IP+ port number, 52
Island parallelization
  barycentric-type crossover, 49
  complex systems, 47–48
  EASEA island model, 51–52
  local optima, 50
  multi-island algorithm, 50
  population-based algorithms, 48
  premature convergence, 47
  Rastrigin function, 48, 49
  search algorithms, 50
  4WD vehicles, 48
K
K20X card, 10
L
Linear ordering problem (LOP)
  applications, 133
  cost function, 132
  data sets, 133–134
  graph problem, 132
  heuristic algorithms, 134–135
  LOLIB solutions, 135–137
  normalized LOLIB library, 135, 137, 138
  permutation, 132, 133
  Local search algorithm, 69–71, 73, 92
  Logarithmic direct method (LDM), 405
  Logical thread organization, 29–30
M
Markov chain Monte Carlo (MCMC) method, 129
MATA. See Move-cost adjusted thread assignment (MATA)
Max-Sat problem, 67
Memetic algorithms (MAs)
  crystal structure prediction, zeolites, 79–80
  GTX295 nVidia card vs. 3.6 GHz Pentium IV
genome transfer time, 77–78
gettimeofday() POSIX function, 74
Rosenbrock function, 74–75
speedup, 75–77
origin, 64–65
parallel memetic algorithm (see Parallel memetic algorithm)
  SEA, 65–67
  Mersenne Twister algorithm, 413–414
MGA. See Multi-compartmental Gillespie’s algorithm (MGA)
Min–max algorithm, 158
Min–min algorithm, 158
MOEA. See Multi-objective evolutionary algorithm (MOEA)
MolDock
   flexible docking, 354
guided differential evolution, 352–353
parallelisation of, 355–356
scoring function, 353–354
Molecular docking method
benchmark, 364–366
CUDA vs. MolDock parallelisation
   guided differential evolution, 358–359
   pose computation, 359–360
   preprocessing phase, 356–358
   program flow, 356–357
   scoring function, 359–360
dataset, 362–363
docking accuracy, 362–364
flexible molecular docking, 351–352
MolDock
   flexible docking, 354
guided differential evolution, 352–353
parallelisation of, 355–356
scoring function, 353–354
virtual screening approach, 350
Monte Carlo basin hopping (MCBH), 371
MOO. See Multi-objective optimization (MOO)
Moore’s law, 312–313
Move-cost adjusted thread assignment (MATA), 180, 190–194
Multi-compartmental Gillespie’s algorithm (MGA), 401
Multi-objective evolutionary algorithm (MOEA)
   CPU and GPU implementations, 304–305
   DMAX model, 289–290
   3D rendering, 291
duality principle, 290
   I1 value, 279
   NSGA-II, 274
   objective vector, 290
   parallel MOEA algorithm, 293, 294
crossover and mutation, 298–299
cross-validation approach, 301–304
data organization, 296–297
dominance-checking algorithm, 295, 299
fitness value evaluation, 297–298
multi-objective function, 293
   non-dominated-selection algorithm, 295–296, 299–300
   parent selection, 298
   prediction models, 300–301
   scoring function, 292, 293
Pareto front, 291
Pareto optimal solution, 291
performance assessment
   EVAL stages, 278
   function bounds, 276
   I_R2 values, 277
   ZDT functions, 280
shaders, 291
shading languages, 292
SPEA2, 274
Multi-objective optimization (MOO)
   ASREA, 268–269
   child_pop, 272
crowding distance, 270
   flow chart, 271
   function evaluation, 271–272
global memory, 272
   parallel SIMD ranking on GPU, 273
   selection strategy, 274
dominance and pareto optimality, 269–270
dominance rank method, 268
EVAL stages, 278
GPU simulations
   common parameter, 280
   function evaluation time, 283
   population sizes, 281
   ranking comparison time, 282
   ZDT functions, 278
   hypervolume indicator, 279
   MOEA (see Multi-objective evolutionary algorithm (MOEA))
   MOOP, 269
   NSGA-II, 268
   objective function, 269
   parameters, 275–276
   performance assessment tools, 275
   R-indicator values, 275
   salient features, 284
Multi-objective optimization problem (MOOP), 269
   Multiple instruction multiple data (MIMD), 151, 230–231
   N
Next reaction method (NRM), 397, 400
Next subvolume method (NSM), 400
Niching, 83, 85, 88
Non-dominated-selection algorithm, 295–296, 299–300
NVIDIA card, 7
NVIDIA GeForce GTX285 GPU, 109–110
NVIDIA GP-GPU architecture
    compute capability, 19
    core organization
        divergence, 23–24
        multi-processor, 21–22
        streaming processors, 20–21
        warp schedulers, 22–23
CUDA software framework
    logical thread organization, 29–30
    scheduling evolution, 32–33
    scheduling process, 31–32
    stream, 33
3D rendering, 19
Flynn’s taxonomy, 18
G80 chip, 19
GF110 chip, 20
memory hierarchy
    cache memory, 24–25
    device memory, 27–28
    host memory, 28
    registers, 26–27
    shared memory, 25–26
memory space, 19
SMT, 28–29
warp notion, 18
NVIDIA multiprocessor, 318–319

O
OpenCL application, 340
Oracle penalty method, 85, 87, 89

P
Parallel computing
    CUDA framework, 151–153
    GPU architecture
        CUDA cores, 153, 154
        global memory coalescing, 155
        off-chip memory, 153
        SIMT, 153, 154
        streaming multiprocessor, 153–155
        parallel reduction, 156–157
        parallel systems, 151
Parallel genetic algorithm model
    GPU computation
        definition, 431–432
        high-speed shared memory, 432–433
        initial value dependency problem, 428, 432
sub-chromosome, parallel processing, 433
Sudoku region blocks, 430–431
multi-core processor, 428, 434
Parallel independent run model
    CUDA programming model, 106, 107
    GPU vs. CPU computation, 116–119
QAP
    base evolutionary model, 108–109
    cost minimisation, 107–108
    parallel evolutionary model, 109–110
    restart strategy, 110–112
QAPLIB benchmark library, 113
run time distribution, 113, 114
run time gain, 115
speed, analytical estimation, 115–118
statistical estimation, 114
Parallelization
    Amdahl’s law, 10–11
    generic evolutionary loop, 11–12
    island model, 12–13
Parallel memetic algorithm
    EASEA platform
        GPU parallelized evolutionary algorithm, 68–69
        GPU parallelized MA algorithm, 69–71
        local search algorithm, 69–73
        Max-Sat problem, 67
Parallel multi-objective evolutionary algorithms
    algorithm, 293, 294
crossover and mutation, 298–299
cross-validation approach, 301–304
data organization, 296–297
dominance-checking algorithm, 295, 299
fitness value evaluation, 297–298
multi-objective function, 293
non-dominated-selection algorithm, 295–296, 299–300
parent selection, 298
prediction models, 300–301
scoring function, 292, 293
Particle swarm optimization (PSO) algorithm
    basic high-level form, 159
degrees of parallelism, 159
fitness function, 159
global memory fitness kernel, 174, 175
machine count, 172–173
multi-swarm PSO, 160–161
    continuous and discrete PSO, 164
    CPU control loop, 168
data organization, 164–165
distinct components, 163
    ETC matrix, 163
fitness updation, 166–167
particle best and global best value updation, 167–168
particle initialization, 166
position and velocity updation, 166
problem size, 163
swap particles, 168
repulsive factor, 160
shared memory fitness kernel, 174, 175
single-swarm PSO, 161–162
solution quality tests, 174–176
swarm counts
execution time, 169, 170
GTX 260 and 570 GPU, 170–172
GTX 570 GPU, 171, 172
sequential CPU and GPU algorithm, 169
task count, 173–174
task matching problem
evolutionary computing, 162–163
definition, 157, 158
FCFS algorithm, 157–158
min–max algorithm, 158
min–min algorithm, 158
Pipelining replaces cache memory, 7–8
Pixel-based algorithms, 6, 7
PSO algorithm. See Particle swarm optimization (PSO) algorithm
Quadratic assignment problems (QAPs)
base evolutionary model, 108–109
compute move costs, 186
cost minimisation, 107–108
definition, 185
parallel evolutionary model, 109–110
restart strategy, 110–112
Tabu search, 185–187
Queue size effect, 99, 100
Random number generator (RNG) kernel, 96
Rosenbrock function, 45, 46, 74–75
Scoring function
CUDA vs. MolDock parallelisation, 359–360
MolDock, 353–354
SGS. See Systolic Genetic Search (SGS)
Silico-alumino-phosphates (SAPO), 371
Simulated annealing (SA), 371
Simultaneous multi-threading (SMT), 21, 28–29
Single-instruction multiple-data (SIMD), 151, 313–315, 402–403
Single-instruction multiple-threads (SIMT), 153, 154, 211
Single-program multiple-data (SPMD) parallelism, 7, 8
Spatio-temporal parallelism, 7–8
SSA method. See Stochastic simulation algorithm (SSA) method
Standard evolutionary algorithm (SEA), 65–67
Stochastic simulation algorithm (SSA) method
FRM, 397
Gillespie SSA, 395–397
NRM, 397
r-leaping method, 397–399
Streaming multiprocessor (SM), 153–155
Stream processor (SP), 314, 318–319, 323
Sudoku puzzles
evaluation experiment
execution platform, 434–435
minimum numbers of generations, 437
number of individuals, 436–437
scalability, 434–436
fine-grained parallelization, 440–442
genetic algorithm
accuracy, 426–429
crossover operation, 424–425
parallel GA model (see Parallel genetic algorithm model)
high-speed evolutionary computation, 422–423
individual setting, 437–439
initial value dependence, 435–438
rules of, 423–424
system architecture
Core i7, 430
GTX 460 and CUDA programming, 429
Systolic arrays (SAs), 205
Systolic Genetic Search (SGS) algorithms, 215–216
crossover points, 208–210
exchange directions, 210
GPU implementation
compute capability, 211, 212
crossoverAndMutation kernel, 213
elitism, exchange, and evaluate kernels, 214
memory multiprocessors, 211
SIMT, 211
structure of, 212, 213
threads organization, 212, 213
grid sizes, 207
knapsack problem, 214–215
mutation, 207–208
numerical efficiency, 217–219
parallel performance
GeForce GTX 285, 220
GeForce 9800 GTX+, 218
GPU vs. CPU implementation,
221, 222
graphic card, 218
GTX 480, 221
improvement factors, 223
solutions, 223
Tesla C1060, 219–221
parameters setting and test environment,
216–217
pseudocode algorithm, 206
systolic arrays, 205
systolic cell, 205, 206
two-dimensional toroidal grid cells,
207
move-cost adjusted thread assignment,
190–191
move cost computations, 189–190
quadratic assignment problem, 185–186
TidePowerd, 257

V
Vertex-based algorithms, 6, 7
Very high memory transfer bandwidth, 8–9

W
Warp schedulers, 22–23
Weierstrass benchmark function
crossover function, 38
evaluation function, 38
GenomeClass, 37
initialization function, 37–38
mutation function, 39
set of parameters, 39–40

X
X-ray powder diffraction (XRD), 370, 372

Z
Zeolites. See Easy specification of evolutionary algorithm (EASEA)