References

References


[71] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 1995.
[74] GMP (GNU Multiple Precision Arithmetic Library). http://gmplib.org/.


[197] Unknown. *Der Handlungsreisende – wie er sein soll und was er zu thun hat, um Auftraege zu erhalten und eines gluecklichen Erfolgs in seinen Geschäften gewiss zu sein – Von einem alten Commis-Voyageur*. 1832.


Index

“folklore” (result), 79
15-puzzle, 248
Aarts, E. H. L., 255
(a, b)-tree, see under sorted sequence
Abello, J., 232
Ackermann, W., 224
Acknowledgement function (inverse), 224
Addition, 2
address, 24, 27
Adel'son-Vel'skii, G. M., 165
adacency array, see under graph
adacency list, see under graph
adacency matrix, see under graph
Adjacent, 49
Aggarwal, A., 120
Aho, A. V., 165
Ahuja, R. K., 201
al-Khwarizmi, Muhammad ibn Musa, 1, 6
ALD, see under shortest path
Algorithm, 1
algorithm analysis, 36, see also running time, 36
amortized, 60, 135, 158, 203
accounting method, 68
binary counter, 70
de amortization, 70
general definition, 71
operation sequence, 71
potential method, 68
token, 68
unbounded array, 66
universality of potential method, 73
approximation algorithm, 241
average case, 41, 84, 103, 107, 109, 115, 117, 124, 148, 199, 205, 245
global, 41
master theorem, 37, 104
randomized, 45, 107, 109, 115, 121
recursion, 9, 16, 37, 104
recursive, 9, 12
smoothed analysis, 262
sum, 4, 36
worst case, 109
Algorithm design, 1
“make the common case fast”, 66
algebraic, 9, 36, 87, 89, 101, 174, 174
black-box solvers, 234, 248, 261
certificate, 33, 36, 51, 187
deterministic, 46, 100
divide-and-conquer, 7, 34, 37, 103
building a heap, 131
mergesort, 103
MSD radix sort, 117
multiplication, 7
multiway mergesort, 119
quicksort, 108, 114
dynamic programming, 243, 261
Bellman–Ford algorithm, 260
changing money, 245
knapsack, 243, 245
matrix products, chained, 245
minimum edit distance, 245
principle of optimality, 243, 246
shortest paths, 193
Evolutionary algorithm, 259, 262
greedy, 101, 239, 257, 261
changing money, 245
cycle detection, 51
Dijkstra’s algorithm, 196
Jarník–Prim algorithm, 219
knapsack, 239, 240
Kruskal’s algorithm, 221
machine scheduling, 241
local search, 249, 262
hill climbing, 250
relaxing constraints, 256
restarts, 259
simplex algorithm, 250
simulated annealing, 252
Tabu search, 258
threshold acceptance, 258
lookup table, 203
preprocessing, 34, 100
random sampling, 120, 232
randomized, 45, 92, 125, 165, 226, 262
Las Vegas, 48, 85, 108, 114
Monte Carlo, 48, 101
recursion, 7, 9, 53, 104, 108, 113, 114, 117, 131, 178, 246
result checking, 6, 33, 101, 198
systematic search, 246, 248, 261
constraint programming, 248, 262
ILP solving, 248
iterative deepening, 248
knapsack, 246
use of sorting, 34, 99–101, 125, 172, 239
algorithm engineering, 1, 5, 10, 11, 92, 95, 111, 120, 123, 125, 163, 199, 209, 257, 261
alignment, 8, 163
all-pairs shortest path, see under shortest path allocate, 27
Alon, N., 97
amortized, see under algorithm analysis analysis, see also algorithm analysis ancestor, 52
AND, 24
Andersson, A, 125
antisymmetric, 264
Applegate, D. L., 230
approximation algorithm, 217, 240
approximation ratio, 240
Aragon, C. R., 257
Aragon, S. R., 165
arbitrage, 207
Arge, L., 123
arithmetic, 26
arithmetics, 24
array, 26, 26, 59
access [], 66
associative, 81
find, 82
forall, 82
insert, 82
remove, 82
Circular, 75, 201
growing, 66
popBack, 66
pushBack, 66
reallocate, 66
shrinking, 66
size, 66
sorting, 111
unbounded, 170
assertion, 32
assignment, 28
asymptotic, 11, 20, 21, 25
Ausiello, G., 54
average case, see under running time
AVL tree, see under sorted sequence
AWK, 81
B (block size), 25
B-tree, 163
bandwidth, 25
base, 25
Bast, H., 213
Bayer, R., 163
Beier, R., 245
Bellman-Ford algorithm, see under shortest path
Bender, M. A., 165
Bentley, J. L., 124
Bertsekas, D. P., 262
best case, see under running time
best-first branch-and-bound, 128
bin packing, 146, 242
binary heap, see under priority queue
binary operation, 24
binary search, see under searching
Index 287

binary search tree, see under sorted sequence
binomial coefficient, 270
binomial heap, see under priority queue
binary tree, 137
bisection method, 35
bit operation, 24
Bixby, E. E., 230
Blelloch, G. E., 125
block, see memory block
Blum, N., 124, 165
Boolean formula, 242
Boolean value, 26
Boost, 57
Bellman–Ford algorithm, 214
Dijkstra’s algorithm, 214
graph, 173
graph traversal, 189
union–find, 231
Boruvka, O., 231
Botelho, F., 97
bottleneck shortest path, 217
bottom-up heap operation, 131
bounded array, 59
branch, 24
branch prediction, 125, 162
branch-and-bound, 128, 246
branch-and-cut, 249
Bro Miltersen, P., 97
Brodal, G., 141, 143
Brown, M. R., 79
Brown, R., 143
Buchbbaum, A., 232
bucket, 121
bucket sort, see under sorting
C, 26
C++, 17, 26, 31, 57, 78, 96, 123, 142, 164, 173, 214, 231
cache, 24
limited associativity, 123
cache-oblivious, 142, 165
calendar queue, see under priority queue
call by reference, 29
call by value, 29
carry, 1, 2
Carter, J., 97
cascading cut, 138
casting out nines, 6
Cayley, A., 174
census, 99
certificate, see algorithm design
certifying algorithm, 33
changing money, 245
characteristic function, 54
Chase, S., 123
Chazelle, B., 166, 232
checksum, 6
Cheriyan, J., 189
Cherkassky, B., 214
Chernoff bound, 122, 269
close, 81
child, 52
Chvátal, V., 230
class, 26, 27, 31
clique, see under graph
clock cycle, 25
clustering, 217
Coffman, E. G., 146
Cohen-Or, D., 174
collision, 82
combinatorial search, 81
comparison, 24
tree, 34, 108, 109
two-way, 35
comparison-based algorithm, 34, 106
competitive ratio, 242
compiler, 3, 26, 58, 81, 123
symbol table, 81
complex number, 31, 100
complexity, 24, see also running time
complexity theory, 54
composite data structure, 27
composition, 26
computation, model of, 24
concave function, 265
conditional branch instruction, 25
conditional statement, 28
cone, 251
congruent, 264
constant, 24
constant factor, 21, 25
constraint, 235
constraint programming, see under algorithm design, systematic search
contract, 32
corner, 265
convex, 265
convex polytope, 251
Cook, W. J., 18, 230
cooling schedule, 254
coprocessor, 25
core, 25
correctness, 31
cost vector, 235
crossover operation, 260
C#, 26
cuneiform script, 59
cycle, 50
  Hamiltonian, 50, 54
  simple, 50
  testing for, 51
DAG, see graph, directed, acyclic
Dantzig, G. B., 235
data dependency, 24
data struct. inv., see under invariant
data structure, VII
data type, see type
database, 147, 163
database join, 81
decision problem, 54
declaration, 26, 29
  implicit, 29
decrement (--), 28
degree, 49
delaunay triangulation, 232
demaine, E. D., 165
dementiev, R., 124, 125, 166, 225
deque, 75, 79
  first, 75
  last, 75
  popBack, 75
  pushFront, 75
  pushBack, 75
  pushFront, 75
dereference, 27
descendant, 52
design by contract, 32
deterministic algorithm, see under
  algorithm design
devroye, L., 148
dictionary, 81, 99
diet problem, 235
dietzfelbinger, M., 97
digit, 1
digraph, see graph, directed
Dijkstra’s algorithm, see under
  shortest path
Dijkstra, E., 196, 219
discrete-event simulation, 128
disk, see hard disk
dispose, 27
distributed system, 25
div, 24
division (integer), 6
driscoll, J., 166
dynamic programming, see under
  algorithm design
dynamic tree, 222
degree, 49
  associated information, 167
  backward, 175, 179
  contraction, 189
  cost, 50
  cross, 175, 179, 181
  crossing, 51
  forward, 175, 179
  parallel, 167, 173
  reduced cost, 207, see also
    node potential
tree, 175, 179
  weight, 50, 167
edge contraction, 226
dynamic programming, see under
edge query, 168, 171
edgeArrow, 168
efficiency, see running time
eight-queens problem, 248, 256
element, 26, 99
equality, see under
empty sequence ⟨⟩, 27
equals (==), 24
equivalence relation, 265
Eratosthenes, 31
event, 266
exchange argument, 219, 239
exclusive OR (⊕), 24
execution time, see running time
existence problem, 233
expected value, 41, 266
exponential search, 35
external memory, see also machine model
  building heap, 132
lower bound, 120
merging, 119
MST, 225
parallel disks, 120, 125
priority queue, 139
queue, 76
scanning, 119
semiexternal algorithm, 226
sorting, 118, 120, 124
stack, 76
Fakcharoenphol, J., 215
false, 24
Farach-Colton, M., 165
fast memory, 25
ferry connections, 217
Fibonacci, L., 135
Fibonacci heap, see under priority queue
field (algebraic), 86, 265
field (of variable), 27
FIFO queue, 74, 177
external-memory, 76
first, 74
popFront, 74
pushBack, 74
using circular array, 75
using two stacks, 75
file, 27
filing card, 145
Flajolet, P., 40
Fleischer, R., 142
floating-point, 24, 56, 203
flow, 237
Floyd, R. W., 58, 124
for, 28
Ford, L. R., Jr., 206
forest, 51
Fredkin, E., 166
Fredman, M. L., 97, 135, 143
frequency allocation, 258
Frigo, M., 142
function object, 96
function pointer, 123
Funke, S., 212
Gabow, H., 189
Gärtnner, B., 262
garbage collection, 57
Garey, M. R., 54, 146
generic methods, 233
generic programming, 31, 173
genome, 259
geometric series, see under sum
graph, 49
generation, 2, 7
geometric, 252
GMP, 17
Goldberg, A., 205, 212, 214
Goodrich, M. T., 174
Graefe, G., 163
Graham, R. L., 40, 58, 241
2-edge-connected components, 187
adjacency array, 168
adjacency list, 170
adjacency matrix, 171
undirected, 171
average degree, 228
BFS, 176, 192
implementation, 188
biconnected components, 188, 189
bidirected, 49, 167, 170
bipartite, 34, 174
breadth-first search, see BFS
Cayley, 174
citation network, 167
clique, 34, 55
coloring, 34, 54, 55, 255, 257
fixed-K annealing, 258
Kempe chain annealing, 255
penalty function annealing, 256
XRLF greedy algorithm, 257
communication network, 175
complete, 54
component, 50
connected components, 50, 177
construction, 160
conversion, 168, 169
counting paths, 171
cut, 172, 218
cycle detection, 170
DAG, see graph, directed, acyclic (DAG)
dense, 171
depth-first search, see DFS
DFS, 175, 178, 206
backtrack, 178
init, 178
root, 178
traverseNonTreeEdge, 178
traverseTreeEdge, 178
diameter, 209
directed, 49
  acyclic (DAG), 50, 51, 52, 180
even, 180, 170
core-decomposition, 199
dia, see under edge
diameter, 209
directed, 49
  subgraph (induced), 50
topological sorting, 180, 195
transitive closure, 177
traversal, 175
triconnected components, 189
undirected, 49
vertex, see node
visitor, 189
DAG, 50, 51, 52, 180
dynamic, 168, 170
more algorithms, 189
open, 183
undirected, 49

Edges

interval, 100
interval graph, 172
layer, 176
linked edge objects, 170
minimum spanning tree, see MST
MST, see MST
multigraph, 167, 173
network design, 217
node, see node
output, 168
planar, 51, 174
4-coloring, 255
5-coloring, 256
embedding, 189
testing planarity, 189
random, 208, 257
random geometric graph, 257
representation, 167
reversal information, 168
SCC, see graph, strongly connected component
shortest path, see shortest path
shrunken graph, 182
sparse, 170
static, 168
Steiner tree, 228
2-approximation, 228
street network, 51
strongly connected component
  certificate, 187
  open, 182
strongly connected components, 50, 175, 181
  closed, 183
  implementation, 188
  invariant, 182
  more algorithms, 189
  open, 183
subgraph (induced), 50
topological sorting, 180, 195
transitive closure, 177
traversal, 175
triconnected components, 189
undirected, 49
vertex, see node
visitor, 189
graphics processor, 25
greedy algorithm, see under algorithm
design
Grossi, R., 166
Pham, N. K., 143
Han, Y., 125, 143
Han, Y., 152
half-space, 25
Halperin, S., 232
Haralson, C., 212
Harrelson, C., 212
hash function, 82
hashing, 81, 100
hash table, see hashing
hashing, 81, 100
  closed, 90
  large elements, 96
  large keys, 96
  linear probing, 83, 90
cyclic, 91
find, 90
insert, 90
remove, 90
unbounded, 91
open, 90
perfect, 92
perfect (dynamic), 95
realistic analysis, 86
universal, 85
bit strings, 86
by integer multiplication, 89
by shifting, 89
by table lookup, 89
simple linear, 89
using bit matrix multiplication, 88
using scalar products, 87
universal family, 86
unrealistic analysis, 84
use of, 100, 101, 108, 168
with chaining, 82, 83
average case, 85
fat, 95
find, 83
implementation, 95
insert, 83
remove, 83
slim, 95
unbounded, 85
heap property, 130, see also
priority queue
heapsort, see under sorting
Held, M., 230
Held–Karp lower bound, see under
MST
heuristic, 44
high-performance computing, 25
hill climbing, see under algorithm design,
local search
Hn, see sum, harmonic
Hoare, C. A. R., 58
Hollerith, H., 99
Hopcroft, J., 165
Huddleston, S., 79, 165
hyperplane, 250
Hoyer, P., 143
I/O step, 25
Iacono, J., 143
IBM, 99
IEEE floating-point, 56
if, 28
iff, 265
ILP, see linear program, integer
imperative programming, 26
implementation note, 25
incident, 49
increment (++), 28
incumbent, 246
indentation, 28
independent random variable, 268
index, 26, 59
indicator random variable, 41, 110
inequality
Chernoff, 269
Jensen’s, 270
Markov’s, 48, 268
infinity (∞), 26, 56
initialization, 26
inlining, 29
input, 24
input size, 20, 23
inserting into a sequence, 60
insertion sort, see under sorting
instance, 20
instruction, 24, 24
integer, 26
integer arithmetics, 1
internal memory, 25
invariant, 32, 182
data structure invariant, 32, 33, 60, 129,
133, 149, 159, 165, 202, 222
loop invariant, 32, 34, 90, 102
inverse, 265
Itai, A., 165
Italiano, G., 165
item, 165
iteration, 28
iterative deepening search, 248
iterator, see under STL
Jarník, V., 219
Jarník–Prim algorithm, see under
MST
Java, 18, 26, 31, 57, 79, 90, 124, 144, 214,
231
deque, 79
hashCode, 96
hashMap, 96
linked list, 79
memory management, 79
PriorityQueue, 142
SortedMap, 164
SortedSet, 164
sorting, 124
stack, 79
TreeMap, 164
TreeSet, 164
vector, 79
JDSL, 57
Dijkstra’s algorithm, 214
graph, 174
graph traversal, 189
MST, 231
PriorityQueue, 142
Jiang, T., 125
Johnson, D. S., 54, 146, 257
jump, 24
Kaligosi, K., 125
Kaplan, H., 143
Karatsuba, A., 9
Karger, D., 232
Karlin, A., 97
Karmakar, N., 237
Karp, R., 230
Katajainen, J., 79, 141
Katriel, I., 232
Kellerer, H., 233
Kempe, A. B., 256
Kempe chain, see under graph
Kettner, L., 124, 166
key, 82, 99, 127
Khachian, L., 237
King, V., 232
Klin, P., 232
knapSack, 54, 191
knapSack Problem, 233
2-approximation (round), 240
as an ILP, 238
average case, 245
branch-and-bound algorithm, 246
dynamic programming, 243
by profit, 245
evolutionary algorithm, 260
fractional, 238, 239, 247
fractional solver, 239
greedy algorithm, 240
local search, 250
simulated annealing, 255
use of, 233
knot, 59
Knuth, D., 40, 58, 97, 125
Komlos, J., 97
Konheim, A. G., 165
Korf, R. E., 248
Korst, J., 255
Korte, B., 232
Kosaraju, S. R., 189
Kothari, S., 123
Kruskal, J., 221
Landis, E. M., 165
Larsen, P.-A., 163
Las Vegas algorithm, see under algorithm design, randomized
latency, 25
Lawler, E. L., 230
leading term, 22
left, 52
LED, 17, 57
Bellman–Ford algorithm, 214
bounded stack, 78
Dijkstra’s algorithm, 214
graph, 173
graph traversal, 188
h_array, 214
list, 78
map, 78
MST, 231
node_pq, 214
priority queue, 142
queue, 78
sortseq, 164
stack, 78
static graph, 173
union–find, 231
Lee, L. W., 173
left-to-right maximum, 42, 48, 110, 200
Leiserson, C. E., 125, 142
Lenstra, J. K., 230
less than (<), 43
Levenshtein distance, 243
Levin, D., 174
lexicographic order, 100, 265
Li, M., 125
linear algebra, 171, 250
linear order, 99, 215, 262
linear program (LP), 234
fractional solution, 238
integer (ILP), 236, 238
0–1 ILP, 238, 248
branch-and-cut, 249
Index

knapsack, 238
pigeonhole principle, 242
set covering, 239
maximum flow, 237
minimum-cost flow, 237
mixed integer (MILP), 238
relaxation of ILP, 238
rounding, 238
shortest path, 238
simplex algorithm, 250
smoothed analysis, 262
solver, 262
strict inequality, 251
tight inequality, 251
linearity of expectations, 41, 85, 86, 110, 228, 267
list, 27, 59, 83, 170
blocked, 76, 106, 118
bulk insert, 105
circular, 136, 170
concat, 64, 65
c晰enate, 60, 65
doubly linked, 60, 145
dummy item, 61, 170
empty, 61
find, 63, 65
findNext, 64, 65
first, 64, 65
head, 64, 65
insert, 62, 64, 65
interference between ops., 64
invariant, 60
isEmpty, 64, 65
last, 64, 65
linked, 60
makeEmpty, 64, 65
memory management, 61, 64
move item, 61
popBack, 64
popFront, 64, 65
pushBack, 64, 65
pushFront, 64, 65
remove, 61, 64, 65
rotation, 64
singly linked, 65, 95
size, 64
sorting, 105
splice, 61, 65
swapping sublists, 64
load instruction, 24
local search, see under algorithm design
locate, see under sorted sequence
logarithm, 264
logical operations, 24
loop, 28, 36
loop fusion, 3
loop invariant, see under invariant
lower bound, 241
“breaking”, 116
element uniqueness, 108
external sorting, 120
minimum, 107
pairing heap priority queue, 143
sorting, 106
lower-order term, 22
LP, see linear program
Lucas, E., 75
Lumsdaine, A., 173
Lustig, I. J., 262
M (size of fast memory), 25
machine instruction, see instruction
machine model, 21, 23
accurate, 25
complex, 25
external memory, 25
parallel, 24, 25
RAM, 23, 26
real, 21
sequential, 23
simple, 25
von Neumann, 23
word, 125
machine program, 24, 26
machine scheduling, 241
decreasing-size algorithm, 242
online algorithm, 241
shortest-queue algorithm, 241
machine word, 23, 24
Maggs, B. M., 125
makespan, 241
map coloring, 255
Markov, A., 48
Markov’s inequality, see under
inequality
Martello, S., 233
Martinez, C., 124
master theorem, see under algorithm
analysis
matem, 202
Matrix, 171
Matrix products, chained, 245
Mauer, D., 58
Maximization problem, 233
Maximum flow, 237
McCready, E. M., 163
McGeoch, L. A., 257
McIlroy, M. D., 124
median, 114, see also selection, 265
Mehlhorn, K., 79, 97, 165, 166, 189, 201, 209, 215, 229
Mehnert, J., 166
member variable, 31
memcpy, 78
memory access, 24
memory block, 25
memory cell, 23, see also machine word
memory management, 27
memory size, 24
MergeSort, see under sorting
merging, 103, 244
external, 119
multiway, 119
Meyer auf der Heide, F., 97
Meyer, B., 56
Meyer, U., 189, 205, 214
Michel, L., 262
Minimization problem, 233
Minimum edit distance, 245
Minimum spanning forest, see MST
minimum spanning tree, see MST
mobile device, 25
mod, 24
modulo, 7, 264
Monte Carlo algorithm, see under algorithm design, randomized
Moret, B., 231
Morris, R., 97
most significant distinguishing index, 202
move-to-front, 44
msd, see most significant distinguishing index
MST, 217
2-approximation of TSP, 230
Boruvka’s algorithm, 231
clustering, 217, 232
cut property, 218, 221
cycle property, 219, 221, 232
Euclidean, 232
external memory, 225
Held–Karp lower bound, 230
Jarník–Prim algorithm, 219
maximum-cost spanning tree, 218
parallel, 232
semiexternal Kruskal algorithm, 226
streaming algorithm, 222
uniqueness conditions, 219
use of, 217, 228, 232
multicore processor, 25
multikey quicksort, 113
multiplication (integer)
Karatsuba, 9
refined, 12
recursive, 7
school method, 1, 3
use of, 1
multithreading, 25
mutation, 25
Näher, S., 166, 171
Nemhauser, G., 231, 248
network, 25, see also graph
communication network, 49
design, 217
Neubert, K. S., 125
Nilsson, N. J., 268
node, 49
active, 118
associated info., 167
deep, 52, 176
dfsNum, 178
finishing time, 178
interior, 52
marked, 178
numbering, 167
ordering relation (≺), 179
potential, 207, 211, 230
reached, 176, 197
representative, 177, 182
scanned, 196
Index

NodeArray, 168, 173
Noshita, K., 200
NOT, 24
NP, 53
NP-complete, 54
NP-hard, 55, 233
numeric type, 2
O(·), 21
o(·), 21
object, 26
object-oriented, 31
objective function, 233
of (in type declaration), 26, 27, 71
Ofman, Y., 9
Ω(·), 21
ω(·), 21
online algorithm, 44, 241
optimization, 233
optimization problem, 56, 233
OR, 24
Orlin, J., 201
oversampling, 121
P, 53
Pagh, R., 97
pair, 27
pairing heap, see under priority queue
parallel assignment, 28
parallel processing, 24, 25, 121, 214, 232, 259, 262
parameter, 29
actual, 29
formal, 29
parameterized class, 31
parent, 52
Pareto, V., 244
Pareto-optimal, 244, 261
parser, 26
partition, 222
Pascal, 26
Patashnik, O., 40, 58
path, 50
simple, 50
Perl, 81
permutation, 42, 100, 101, 106
random, 42, 45
persistent data structure, 166
Peru, 59
Peterson, W. W., 90
Petrank, E., 97
Pettie, S., 143, 232
Pferschy, U., 233
pigeonhole principle, 242
pipelining, 4
Pisinger, D., 233
pivot, 108, 121
selection, 111, 124
Plaxton, C. G., 125
pointer, 26
polynomial, 22, 101, see also under running time
polytope, 251
population, 259
postcondition, 32
potential function, see node, potential powers (of numbers), 32
Pratt, V. R., 124
precedence relation, 49
precondition, 32
predecessor, 60, 60
Priebe, V., 209
Prim, R. C., 219
Prim's algorithm, see MST, Jarník–Prim algorithm
prime number, 31, 86, 101, 265
abundance, 88
primitive operation
full adder, 1
product, 2
principle of optimality, 243, 246
priority queue, 127
addressable, 129, 133, 198
binary heap, 129, 199
addressable, 129, 133
call on deleteMin, 142
building, 131
bulk insertion, 133
deleteMin, 131
insert, 130
invariant, 129
siftDown, 131
siftUp, 130
bounded, 129
bucket, 143
bucket queue, 201
invariant, 202
calendar queue, 143
decrease key, 128, 199
deleteMin, 127
double-ended, 156
external, 139
fat heap, 143
Fibonacci heap, 135, 199, see also
priority queue, heap-ordered forest
decreaseKey, 138
deleteMin, 136
item, 136
rank, 136
heap-ordered forest, 133
cut, 133
decreaseKey, 133
deleteMin, 133
insert, 133
invariant, 133
link, 133
merge, 135
new tree, 133
remove, 135
insert, 128
integer, 142, 143, 201
item, 133
memory management, 141
merge, 128
minimum, 127, 130, 133
monotone, 128, 143, 198, 201
naive, 129, 199
pairing heap, 135, see also
priority q., heap-ordered forest
complexity, 143
three-pointer items, 135
two-pointer items, 135
radix heap, 201
base b, 204
remove, 128
thin heap, 143
unbounded, 129
use of, 102, 120, 125, 128, 198, 226
probability, 266
probability space, 41, 266
problem instance, 20
procedure, 29
profit vector, see cost vector
program, 24
program analysis, see algorithm analysis
programming language, 26, 28, 58

functional, 105
logical, 105
programming model, see machine model
Prokop, H., 142
pseudo-polynomial algorithm, 245
pseudocode, 26, 56
Puget, J.-F., 262
Pugh, W., 165
quartile, 114, see also selection
queue, 27, 170, see also FIFO
quicksort, see under sorting
quiqui, 59
radix sort, see under sorting
Radzik, T., 214
RAM model, see under machine model
Ramachandran, S., 142
Ramachandran, V., 232
Raman, R., 125
Ranade, A., 123
random experiment, 266
random number, 46
random source, 57
random variable, 41, 266
independent, 268
indicator, 266
product, 268
randomized algorithm, see under
algorithm design; algorithm analysis
rank, 103, 265
Rao, S., 215
reallocation, 78
recombination, 259, 260
record, see composite type
recurrence relation, 9, 16, 35, 37, 58
recursion, 29, see also under
algorithm design; algorithm analysis
elimination, 113, 141
red–black tree, see under sorted sequence
reduction, 55
reflexive, 265
register, 24, 24, 25
Reif, J., 189
relation, 265
antisymmetric, 264
equivalence, 265
reflective, 265
symmetric, 265
transitive, 265
weakly antisymmetric, 265
relaxation, 256, see also under
linear program
remainder, 24
Remez, O., 174
removing from a sequence, 60
repeat, 28
result checking, see under algorithm design
return, 29
Rivest, R. L., 124
road map, 49
Robertson, N., 255
Robins, G., 229
Rodeh, M., 165
root, see under tree
Roura, S., 124
run, see under sorting
running time, 20, 24, 28, 36, see also
algorithm analysis
average case, 20, 41
best case, 20, 24
polynomial, 53
worst case, 20
sample space, 266
Sanders, D. P., 255
Sanders, P., 124, 125, 141, 142, 166, 212,
214, 215, 225, 232
Santos, R., 174
Sarnak, N., 166
SAT solver, 242
satisfiability problem, 54
satisfiable, 242
Schäfer, G., 209
Schaffer, R., 142
scheduling, 128, 191, 241
Schevyn, C., 257
Schönhage, A., 18
Schrijver, A., 262
Schultes, D., 212, 225
search tree, see sorted sequence
searching, 145, see also sorted sequence
binary search, 34, 56, 100, 121, 151
dynamic, 43
exponential, 35, 56
linear, 43
range, 100
shortest path, see under shortest path
Sedgewick, R., 40, 125, 142, 143
Seidel, R., 165, 174, 224
selection, 114
deterministic, 124
quickselect, 114
streaming, 115
self-loop, 49
semicolon (in pseudocode), 28
sentinel, 63, 95, 102, 106, 141
sequence, 27, 27, 59, 100
overview of operations, 77
space efficiency, 77
series, see sum
server, 25
set, 27
set covering, 239
Seymour, P., 255
Shapiro, H. D., 231
shared memory, 25
Sharir, M., 189
Shell sort, see under sorting
Shepherdson, J., 23
tail, 24
Shmoys, D. B., 230
shortest path, 191
acyclic, 192
ALD (average linear Dijkstra), 205, 214
all-pairs, 207
arbitrary edge costs, 206
as a linear program, 236
A* search, 211
Bellman-Ford algorithm, 206
refined, 214
bidirectional search, 209
bottleneck, 217, 233
by table lookup, 212
constrained, 215, 246
correctness criterion, 194
DAG, 195
Dijkstra’s algorithm, 196
invariant, 201
directional search, 211
generally, 215
goal-directed search, 212
integer edge cost, 192
linear average time, 205
multicriteria, 215
negative cycle, 192
nonnegative edge cost, 192
parallel, 214
parent pointer, 193
public transportation, 196
query, 209
relaxing of edges, 194
single-source, 191
subpath, 193
tentative distance, 194
transit node routing, 212
tree, 193
uniqueness, 193
unit edge cost, 192
use of, 191, 207
shortest-queue algorithm, 231
shrunken graph, 182
Sibeyn, J., 225
sibling, 52
sibling pointer, 136
Siek, J. G., 173
sieve of Eratosthenes, 31
SIMD, 25, 95
simplex algorithm, see under linear programming
simulated annealing, see under algorithm design, local search
Singler, J., 124
Sipser, M., 54
Sivadasan, N., 209
Sleator, D., 79, 143, 165, 166, 222
slow memory, 25
Smith, S. J., 125
snow plow heuristic, 125
solution
feasible, 233
potential, 233
sorted sequence, 34, 145
(a,b)-tree, 149
split (node), 152
amortized update cost, 158
augmentation, 160
balance, 153
build/rebuild, 156
concatenation, 157
fusing, 153
height, 150
insert, 151
invariant, 149
item, 150
locate, 150
parent pointer, 161
reduction, 161
remove, 153
removing a range, 158
splitter, 149
splitting, 157
adaptable, 165
AVL tree, 165
binary search tree, 147
degenerate, 148
expected height, 148
implicit, 149
insert, 147
locate, 147
perfect balance, 147
rotation, 149
selection, 161
cache-oblivious, 165
finger search, 161
first, 146, 156
last, 146, 156
locate, 146, 146
merging, 161
navigation, 142
persistent, 166
pred, 146
randomized search tree, 165
range searching, 156
red–black tree, 155, 164
remove, 145
skip list, 165
sparse table, 165
splay tree, 165
strings, 166
succ, 146
trie, 166
use of, 146, 147
weight-balanced tree, 160, 165
sorting, 99
almost sorted inputs, 103
bottom-up heapsort, 142
bucket, 116
comparison-based, 116
dynamic, 102
external, 118
flash, 125
heapsort, 125, 132
in-place, 101, 111
insertion, 36, 102, 105
large elements, 123
list, 105
lower bound, 116
mechanical, 99
mergesort, 103, 125
multiway merge, 119
numbers, 116, 116, 122, 170
parallel, 121, 123
parallel disks, 125
quicksort, 108, 123, 124, 148
radix, 116
LSD, 116
MSD, 117, 123
random numbers, 117
run formation, 119, 125
sample, 120, 125
selection, 101, 128
Shell sort, 125
small inputs, 102, 108
small subproblems, 111
stable algorithm, 116
strings, 113, 116
use of, 34, 99–101, 125, 172, 226, 239
word model, 125
source node, 49
spellchecking, 125
Spielmann, D., 262
splitter, 121, 147
stack, 27, 29, 74, 75
bounded, 75
external-memory, 76
pop, 74
push, 74
top, 74
unbounded, 75
statement, 28
static array, 27, 59
statistics, 114
Stirling’s approximation, 107, 118, 270
STL, 13, 57, 164
deque, 78
hash_map, 96
hash_multiset, 96
hash_set, 96
iterator, 78, 123
list, 78
map, 164
multimap, 164
multiset, 164
priority_queue, 142
set, 164
sort, 123
stack, 78
store instruction, 24
Strassen, V., 18
streaming algorithm, 115, 222
string, 27, 59, 100
striping, 125
struct, see composite type
Sturgis, H., 23
STXXL, 124, 141, 142
subroutine, 29
successor, 60, 60
succinct data structure, 97
Sudoku, 255
sum, 58, see also under algorithm analysis
estimation by integral, 271
geometric, 38, 270
harmonic, 43, 45, 88, 110, 200, 228, 264, 270
Sumerian, 59
survival of the fittest, 259
swap, 28
sweep-line algorithm, 146
symmetric, 265
syntax, 26
Szemerédi, E., 97
table, 59
tablet, 59
tabu list, see tabu search
tabu search, see under algorithm design,
local search
tail bound, 269
tail recursion, see recursion, elimination
Tamassia, R., 174
Tardos, E., 97
target node, 49
Tarjan, R. E., 79, 97, 124, 135, 143, 165,
166, 189, 201, 214, 222, 224, 232
telephone book, 99
template programming, 31, 123
Teng, S. H., 262
termination, 33, 34
Θ(·), 21
Thomas, R., 255
Thompson, K., 246
Thorup, M., 95, 125, 143, 144
thread, 25
threshold acceptance, see under
algorithm design, local search
time, see running time
time step, 24
Toom, A., 18
total order, 99, 265, 265
Toth, P., 233
tournament tree, 125
tower of Hanoi, 75
Träff, J. L., 232
transitive, 265
translation, 27–30
traveling salesman problem, 54, 55, 56, 230
2-exchange, 250
3-exchange, 250
Held–Karp lower bound, 230
hill climbing, 250
tree, 51, 147
depth, 52
dynamic, 222
expression tree, 53
height, 52
implicitly defined, 129
interior node, 52
ordered, 53
representation, 136
root, 52
sorting tree, 106
traversal, 53
triangle inequality, 230, 250
trie, see under sorted sequence
triple, 27
true, 24
truth value, 24
Tsitsiklis, J. N., 262
TSP, see traveling salesman problem
tuple, 27, 100
type, 26
Udupa, R., 123
Ullman, J., 165
Ullmann, Z., 244
unary operation, 24
unbounded array, 60, 66
undefined value (⊥), 26
uniform memory, 23
union–find, 222
path compression, 223
union by rank, 223
universe (Ω), 233
upper bound, see worst case
Vöcking, B., 245
van Emde Boas layout, 165
van Emde Boas, P., 166
Van Hentenryck, P., 262
Vanderbei, R. J., 262
variable, 26, 235
Vazirani, V., 232
tensor (in C++), 78
verification, 32, 103
vertex, see node
Vishkin, U., 189
visitor, see under graph
Vitányi, P., 125
Vider, J., 120, 123
von Neumann, J., 23
von Neumann machine, see under
machine model
Vuillemin, J., 137
Vygen, J., 232
weakly antisymmetric, 265
Wegener, I., 54, 142
Wegmann, M., 97
Weidling, C., 97
Westbrook, J., 232
while, 28
Wickremasinghe, R., 123
Wilhelm, R., 58
Williams, J. W. J., 129
Winkel, S., 125, 142
witness, see algorithm design, certificate
Wolsey, L., 248
word, see machine word
worst case, see under running time
XOR (⊕), 24, 203
Zagha, M., 125
Zang, I., 215
Zelikowski, A., 229
Ziegelmann, M., 215
Ziviani, N., 97
Zlotowski, O., 171
Zwick, U., 143, 232