

Index

- Acid properties of transactions, 301–303
 - atomicity, 302
 - consistency, 302–303
 - durability, 302–303
 - isolation, 302–303
- Adaptive Plan Correction (APC), 279–280
- Amdahl law, 10
- Analytical models, 33–46
 - cost models, 33–34
 - cost notations, 34–39
 - communication costs, 38–39
 - data parameters, 34–35
 - query parameters, 37
 - systems parameters, 36
 - time unit costs, 37–38
 - parallel database, operations in, *See* Databases, parallel
 - skew model, 39–43
- Architectures, grid database, 26–28
 - data-intensive applications working in, 26
 - grid middleware, 27
- Architectures, parallel database, 19–26
 - interconnection networks, 24–26
 - shared-disk architectures, 20–21
 - shared-memory architectures, 20–21
 - shared-nothing architecture, 22
- Association rules/Association rule data mining, 432, 440–450
 - association rules, 444–448
 - association rules generation, 445–448
 - frequent itemset generation, 444–445
 - concepts, 441–444
 - count distribution-based parallelism for, 448–449
 - data distribution-based parallelism for, 450
 - generation, 445–448
 - itemset, 441
 - literals, 441
- Asynchronous protocols, GRAP, 381
- Atomic commit protocols, 310–314
 - heterogeneous DBMSs, 313–314
 - Homogeneous DBMSs, 310–313
- Atomicity property, 302, *See also* Grid transaction atomicity and durability
 - for centralized and homogeneous DBMSs, 304
 - for heterogeneous distributed DBMSs, 306
- Autonomy, 294
- Basic data partitioning, 55–60
 - hash, 57–58
 - range, 58–59
 - round-robin, 56
- BERD (Bubba’s Extended Range Declustering), 67–69
- Binary merge sort, parallel, 85–86
 - cost model, 100–101
- Binary search, 71–72
- Bus interconnection network, 24
- Bushy-tree parallelization, 258
- Centralized DBMSs
 - transactions management in, 303–305
 - Atomicity, 304
 - Consistency, 304
 - solation, 304–305
- Classification, parallel, 477–495
 - data parallelism for a decision tree, 489–492
 - data set structure, 479–480
 - decision tree algorithm, 480–481
 - decision tree classification, 477–480
 - processes, 480–488
 - structure, 478–479
 - result parallelism for the decision tree, 492–495

- Classification, parallel (*Continued*)
 - splitting attributes or feature selection, 481–484
- Cluster/Clustering, parallel, 464–499
 - architectures, 23
 - cluster customers, 465
 - cluster students, 465
 - concepts, 467–468
 - hierarchical clustering, 468
 - in parallel data mining, 433
 - parallel *k*-means clustering, 471–477
 - partitional clustering, 468
 - query processing model, 270–275
 - architecture, 272–273
 - dynamic query processing, 271–272
 - load information exchange, 273–275
 - result parallelism parallel *k*-means, 475–477
 - similarity measures, 467–468
- Collection join queries, 219–255
 - algorithms for, 225
 - disjoint data partitioning, 226–227
 - parallel collection-equi join, 225–233
 - parallel double sort-merge collection-equi join algorithm, 227–228
 - parallel hash collection-equi join algorithm, 232–233
 - parallel sort-hash collection-equi join algorithm, 228–231
 - collection-intersect join algorithms, 233–246
 - non-disjoint data partitioning, 234–244
 - hash collection-intersect join algorithm, 246
 - relational division, 220
 - repeated relational division, 220
 - sort-hash collection-intersect join algorithm, 245–246
 - sort-merge nested-loop collection-intersect join algorithm, 244–245
 - subcollection join algorithms, 246–252
 - types, 222–225
 - array, 222
 - bag, 222
 - collection-equi join queries, 222–223
 - collection–intersect join queries, 223–224
 - list, 222
 - set, 222
 - subcollection join queries, 224–225
 - universal quantification and collection join, 220–221
- Communication, 11–12
 - cost, 38–39
 - parallel merge-all sort, 98–99
 - parallel partitioned sort, 104
 - parallel redistribution merge-all sort, 103
- Comparative analysis, 207–215
 - parallel index join, 213–215
 - parallel search index, 207–213
 - continuous-range search queries, 212
 - discrete-range search queries, 212
 - exact-match search queries, 212
 - intersection method, 209–210
 - multi-index search query processing, 209–212
 - one-index access method, 210–213
 - one-index search query processing, 207–209
- Comparison cost, 70, 72
- Compensate approach, 314
- Complex data partitioning, 60–69
 - BERD, 67–69
 - hybrid-range partitioning strategy, 60–65
 - MAGIC, 65–67
- Compute destination cost, 101
- Concurrency control protocols, 309–310
 - locking-based algorithms, 309
 - optimistic algorithms, 309
 - pessimistic algorithms, 309
 - timestamp ordering algorithms, 310
- Conjunctive predicates, 54
- Conjunctive prenex normal form (CPNF), 54
- Consistency property, 302–303
 - for centralized and homogeneous DBMSs, 304
 - for heterogeneous distributed DBMSs, 306–307
- Consolidation costs, 10–12
- Contingency GRAP, 378–381
 - correctness of, 383–384
 - read transaction operation for, 379
 - write transaction operation for, 379–381
- Continuous range search query, 53
- Correcting, 276
- Correction, dynamic cluster query optimization, 276–280
 - Adaptive Plan Correction (APC), 279–280
 - correcting, 276
 - deferring, 276
 - discarding, 276
 - Optimistic Plan Correction (OPC), 278
 - Pessimistic Plan Correction (PPC), 279
 - triggering, 276
- Correctness of GCC protocol, 336–338
- Cost models, 33–34
 - disjoint partitioning, 129–130
 - divide and broadcast, 128–129
 - for the early GroupBy with partitioning scheme, 156–158
 - for phase one (grouping phase), 156

- for phase three (GroupBy-JoinPhase), 157–158
 - for phase two (distribution phase), 157
 - scan cost, 156
- for the early GroupBy with replication scheme, 158–159
 - for phase one (grouping phase), 158
 - for phase three (grouping/joining phase), 159
 - for phase two (replication phase), 158–159
- for GroupBy-After-Join query processing, 159–163
 - for join partitioning scheme, 159–161
 - GroupBy partitioning scheme, 161–163
 - phase four (global aggregation phase), 161
 - phase one (data partitioning and broadcasting phase), 162
 - phase one (data partitioning phase), 159–160
 - phase three (redistribution phase), 161
 - phase two (join and aggregation phase), 162–163
 - phase two (join and local aggregation phase), 160
- for GroupBy-Before-Join query processing, 153–159
 - for the early distribution scheme, 153–156
- local join, 130
- for phase one (distribution phase), 153–154
 - data transfer cost, 154
 - destination cost, 154
 - scan cost, 153
 - select cost, 153
- for phase two (GroupBy-Join Phase), 154–156
 - aggregation and join costs, 154
 - disk cost of storing final result, 155
 - generating result records cost, 155
 - reading/writing of overflow buckets cost, 155
 - receiving records cost, 154
- notations, parallel GroupBy-Join, 151–153
 - join selectivity, 153
 - projectivity, 152
 - selectivity, 152
- parallel binary-merge sort, 100–101
- parallel groupby, 104–108
- parallel merge-all sort, 98–100
- parallel partitioned sort, 103–104
- parallel redistribution binary-merge sort, 101–102
- parallel redistribution merge-all sort, 102–103
- serial external merge-sort, 96–97
- Count distribution-based parallelism
 - for association rule mining, 448–449
- Cube queries, parallelization of, 412–417
 - basic CUBE queries, analysis, 413–416
 - partial CUBE queries, analysis of, 416–417
 - without using CUBE, 417
- Cumulative distribution function (CUME_DIST) queries, parallelization, 419–420
- Data computation cost, 46
- Data distribution-based parallelism
 - for association rule mining, 450
- Data mining, parallel data mining
 - association rules, 427–463
 - class description, 432
 - components, 430
 - data mining tasks, 431–433
 - descriptive data mining, 431
 - predictive data mining, 431
 - data parallelism, 437–438
 - data warehouse, 429
 - data-intensive applications, 428
 - definition, 430
 - from databases to data warehousing to data mining, 428–431
 - parallel association rules, 440–450
 - parallel sequential patterns, 450–461
 - parallelism, 436–440
 - querying vs. mining, 433–436
 - read-only queries, 429
 - result parallelism, 438–440
 - sequential patterns, 427–463
 - write queries, 429
- Data parallelism, 437–438
 - for a decision tree, 489–492
 - parallel *k*-means, 472–475
- Data parameters, 34–35
- Data partitioning method, 226
- Data scale up, 8, 9–10
- Data skew, 39
- Data transfer cost
 - disjoint partitioning, 129
 - divide and broadcast, 128
 - parallel binary-merge sort, 100
 - parallel redistribution binary-merge sort, 102
- Data virtualization approach
 - in grid environment, 28
- Databases, parallel, 4–5, 43–46
 - data computation, 45–46
 - data distribution, 45–46
 - disk operations, 44
 - main memory operations, 45
- Decision tree, 466
 - classification, 477–480

544 INDEX

- Deferring, 276
- Descriptive data mining, 431
- Destination cost, 46
- Direct Attached Storage (DAS), 27
- Discarding, 276
- Discrete range search query, 53
- Disjoint data partitioning, 226–227
- Disjoint partitioning join, 124–127
 - cost model, 129–130
- Disk cost
 - disjoint partitioning, 130
 - divide and broadcast, 129
 - local join, 131
- Disk writing cost, 71–72
- Distributed databases, 293–297
 - architectural model, 294
 - autonomy, 294
 - distribution, 294
 - eterogeneity, 294
 - distributed DBMS in grids, 296–297
 - partitioning, 296
 - replication, 296
 - transactions, 291–320, *See also* Transactions
 - working model, 294–296
- Divide and broadcast join, 121–124
 - cost model, 128–129
- Divide and broadcast, and, 234–236
- Divide and partial broadcast, 236–244
 - one-way, 242–243
 - two-way, 238–244
- Double sort-merge collection-equi join
 - algorithm, 227–228
- Duplicate removal, 78
- Durability property, 302–303, *See also* Grid transaction atomicity and durability
 - for centralized and homogeneous DBMSs, 304–305
 - for heterogeneous distributed DBMSs, 306–307
- Dynamic cluster query optimization, 275–284
 - correction, 276–280, *See also* Correction
 - load information exchange, 275
 - migration, 280–281
 - partition, 281–284
 - query plan correction, 275
 - semijoin-based query optimization, 284
 - static query plan formulation, 275
 - subquery migration, 275
 - subquery partition, 275
- Dynamic Query Processing, 271–272
- Early distribution scheme, GroupBy-Before-Join query processing, 143–144
 - distribution phase, 143
 - GroupBy-Join phase, 143–144
- Early GroupBy with partitioning scheme, 145–147
 - distribution phase, 145
 - final grouping and join phase, 145
 - local grouping phase, 145, 147
- Early-abort Grid-ACP, 346–348
- Equi-join query, 112
- Euclidean distance, 468
- Euler’s constant, 40
- Exact match search, 52
- Execution Among Subqueries, 261–263
- Exhaustive search, 69
- External sorting
 - cost models for, 96–104
 - parallel, 83–91
 - binary-merge sort, 85–86
 - merge-all sort, 83–84
 - partitioned sort, 90–91
 - redistribution binary-merge sort, 86–88
 - redistribution merge-all sort, 88–89
 - serial, 80–83
- Failure recovery algorithm for Grid-ACP, 353–359
 - originator recovery procedure, 357–359
 - participant recovery procedure, 354–357
- File sorting, 77
- Final merging costs, 98
- Find_node algorithm, 186–187
- Finding destination cost
 - disjoint partitioning, 129
- Flat-tree parallelization, 258
- Frequent itemset generation, 444–445
- Fully replicated indexing (FRI) structure, 168, 178–180
 - FRI-1, 178–179
 - FRI-3, 180–181
 - maintaining, 188
- Gain criterion, 482
- Generating result cost
 - local join, 131
 - parallel binary-merge sort, 100
 - parallel merge-all sort, 98–99
 - parallel partitioned sort, 104
 - parallel redistribution binary-merge sort, 102
 - parallel redistribution merge-all sort, 103
 - serial external merge-sort, 97
- Global subtransaction ready log, 352
- Global transaction active log, 352
- Global transaction monitor (GTM), 294
- Global transaction termination log, 353

- Grace hash join, 117
- Grid atomic commit protocol (Grid-ACP), 343–351, 387–398, *See also* Modified Grid-ACP
 - algorithm, 344–346
 - originator's, 345, 347
 - participant's, 345–346
 - correctness of recovery algorithm, 361–365
 - transaction submission procedure, 362–363
 - correctness of, 350–351
 - early-abort grid-ACP, 346–348
 - failure recovery algorithm for, 353–359
 - handling failure of sites with, 351–365
 - logs required at originator sites, 352–353
 - logs required at participant site, 353
 - storing log files at originator and participating sites, 351–352
 - in replicated data, 387–398
 - message complexity analysis, 349–350
 - recovery protocols, comparison, 359–361
 - state diagram of, 343–344
 - compensate states, 343
 - pre-abort state, 343
 - sleep state, 343
 - time complexity analysis, 349
- Grid concurrency control (GCC) protocol, 321–340
 - basic functions required, 324–325
 - active trans(DB), 324
 - append TS(ST_i j), 325
 - cardinality(Any set), 325
 - DB accessed(T_i), 324
 - split trans(T_i), 324
 - correctness of, 336–338
 - features of, 338–339
 - serializability theory, 325–329
 - submission phase, 329–330
 - termination phase, 331–333
 - traditional versus, 334–336
- Grid Data Distribution (GDD), 27
- Grid databases, 4–5
 - challenges, 292–293
 - definition, 3
 - transactions, 291–320, *See also* Transactions
- Grid replica access protocol (GRAP), 371–378
 - correctness of, 377–378
 - read transaction operation for, 371–372
 - write transaction operation for, 372–375
 - if the participant decides to commit, 373
 - if the participant decides to abort, 373
- Grid transaction atomicity and durability, 341–366
 - motivation, 342–343
- Grid-ACP, *See* Grid atomic commit protocol
- GroupBy-Join queries, 141–166
 - cost model notations, 151–153, *See also* Cost model
 - cost models for parallel, 104–108
 - early GroupBy with partitioning scheme, 145–146
 - early GroupBy with partitioning scheme, 146–147
 - GroupBy After Join query, 142–143
 - GroupBy Before Join query, 142
 - GroupBy partitioning scheme, 150–151
 - aggregate operations, 151
 - consolidation, 151
 - data partitioning, 150–151
 - join operations, 151
 - GroupBy-After-Join query processing
 - parallel algorithms for, 148–151
 - GroupBy-Before-Join query processing, 143
 - early distribution scheme, 143
 - parallel algorithms for, 143–147
 - parallel algorithms for, 92–96
 - redistribution method, 94–96
 - traditional methods, 92–93
 - two-phase method, 93–94
- Hashing collections/multivalued, 232–233
 - hash collection-equi join algorithm, 232–233
 - hash collection-intersect join algorithm, 246
 - hash subcollection join algorithm, 251–252
 - hash table, 36
 - hash-based join, 117–120
 - partitioning, 57–58, 126–127
- Heterogeneity, 294
- Heterogeneous distributed DBMSs
 - atomic commit protocols, 313–314
 - compensate, 314
 - redo, 314
 - retry, 314
 - transactions management in, 305–307
 - atomicity, 306
 - consistency, 306–307
 - durability, 307
 - isolation, 307
- Hierarchical clustering, 468
- Hierarchical merging method, 93
- High-level replica management architecture, 368–369
- Histogram queries, parallelization, 420–422
- Homogeneous DBMSs
 - atomic commit protocols, 310–313
 - Three-phase commit (3PC), 312–313
 - Two-Phase Commit (2PC), 311–312
 - transactions management in, 303–305
 - atomicity, 304

546 INDEX

- Homogeneous DBMSs (*Continued*)
 - consistency, 304
 - isolation, 304–305
- Horizontal data partitioning, 55
- Hybrid-range partitioning strategy (HRPS), 60–65
 - advantages, 63–65
- Hypercube interconnection network, 25–26
- I/O bottleneck, 4
- Independent parallelism, 15, 18
- Indexing, parallel, 167–218
 - comparative analysis, 207–215, *See also* Comparative analysis
 - index join algorithms, 200–207
 - one-index join query, 200–203
 - two-index join query, 200, 203–207
 - maintenance, 180–188
 - algorithms, 185–188
 - complexity degree of, 188
 - fully replicated index, 188
 - nonreplicated index, 182
 - partially replicated index, 182–188
 - restructuring algorithms, 187
 - restructuring step, 183
 - steps for, 180–188
 - one-index method, 199–200
 - initialization module, 200
 - one-index access module, 200
 - search queries parallel processing using, 192–200
 - storage analysis, 188–192
 - structures, 168–180
 - fully replicated index (FRI), 168, 178–180
 - nonreplicated index (NRI), 168, 169–171
 - partially replicated index (PRI), 168, 171–178
- Interconnection networks, 24–26
 - bus, 24
 - hypercube, 25–26
 - mesh, 24–25
- Interference, 11–12
- Interoperation parallelism, 12, 15–18
- independent parallelism, 15, 18
- pipeline parallelism, 15–18
- Interquery parallelism, 12, 13–14
- Intertree node parallelism, 492
- Intraoperation parallelism, 12, 15, 16
- Intraquery parallelism, 12, 14–15
- Isolation property, 302–303
 - for centralized and homogeneous DBMSs, 304–305
 - for heterogeneous distributed DBMSs, 306–307
- Itemset, 441
 - anti-monotonicity, 442
 - association rules, 441–442
 - candidate itemset, 441
 - frequent itemset, 441
 - itemset mining, 441
- Join algorithms for the collection-intersect join, 244–245
- Join costs
 - local join, 131
- Join partitioning scheme
 - for GroupBy-After-Join query processing, 148–150
 - consolidation, 150
 - data partitioning, 148
 - global aggregation, 149
 - join operation, 149
 - local aggregation, 149
 - redistribution, 149
- Join selectivity notation, parallel GroupBy-Join, 153
- Join, parallel, 112–134
 - cost models, 128–131
 - join algorithms, 120–127
 - divide and broadcast-based, 121–124
 - disjoint partitioning join, 124–127
 - join operations, 103
 - optimization, 132–134
 - load balancing, 133–134
 - main memory, 132–133
- k -Means clustering, parallel, 81–82, 471–477
 - algorithm, 468–471
 - data parallelism parallel k -means, 472–475
- Leaf nodes, 189–190
- Left-deep tree parallelization, 258
- Linear scale up, 8
- Linear search, 69
- Linear speed up objective, parallel query processing, 7
- Literals, 441
- Load cost
 - parallel binary-merge sort, 100
 - parallel merge-all sort, 99
 - parallel partitioned sort, 104
 - parallel redistribution binary-merge sort, 102
 - parallel redistribution merge-all sort, 103
 - serial external merge-sort, 97
- Load imbalance, 133–134
- Load information exchange, 273–275
 - high load processing node, 273
 - low load processing node, 273
 - medium load processing node, 273

- Load skew in single-table queries, 260
- Local database management system (LDBMS), 294
- Local join, 131
- Local merge-sort costs, 98
- Local searching method, 73
- Locking-based algorithms, 309

- MAGIC (Multiattribute Grid Declustering), 65–67
- Massively Parallel Processing (MPP) machines, 22
- Merge-all sort, 83–84
 - cost model, 98–100
- Merging cost
 - parallel binary-merge sort, 100
 - parallel merge-all sort, 98–99
 - parallel partitioned sort, 104
 - parallel redistribution binary-merge sort, 102
 - parallel redistribution merge-all sort, 103
 - serial external merge-sort, 97
- Mesh interconnection network, 24–25
- Message complexity analysis, Grid-ACP, 349–350
- Migration, dynamic cluster query optimization, 280–281
 - subquery migration, 280
- Mixed parallelism, 18–19
- Modeling skew, 40
- Modified Grid-ACP, 390–395
 - algorithm, 390–393
 - correctness of, 393–395
 - ACP properties, 393–394
 - for originator site, 392
 - using replication at multiple levels, 391
- Moving average queries, parallelization, 422–424
- Multiattribute search query, 54
- Multidatabase systems, 297–299
 - architecture, 297
 - communication autonomy, 297
 - design autonomy, 297
 - execution autonomy, 297
 - in grids, 297–299
- Multi-index search query processing, 195–200
 - intersection method, 195
 - algorithm, 198
 - Case 1 (one index is based on NRI-1, PRI-1, or FRI-1), 196
 - Case 2 (one index is based on NRI-3, PRI-3, or FRI-3), 197
 - Case 3 (one index is based on NRI-2 or PRI-2), 197
 - individual index access module, 198
 - initialization module, 198
 - intersection module, 198
 - record loading module, 198
 - Multiple ROLLUP queries, 409–411
- Nested-loop join, 114–115
- Network partitioning, 315–316
- Node architectures, 23
- Non-disjoint data partitioning, 234–244
 - divide and broadcast, and, 234–236
 - divide and partial broadcast, 236–244
 - simple replication, 234
- Nonleaf nodes, 189–190
- Nonreplicated Indexing (NRI) Structures, 168, 169–171
 - maintaining, 182
 - NRI-1, 170
 - NRI-2, 171–172
 - NRI-3, 171, 173
- Nonskewed Subqueries, 264–265
- NTILE queries, parallelization, 420–422

- Obstacles objective, parallel query processing, 10–12
 - consolidation costs, 10–12
 - start up costs, 10–12
- One-index join query, 192–195, 200–203
 - Case 1 (NRI-1 and NRI-3), 201
 - Case 2 (NRI-2), 201
 - Case 3 (PRI), 201
 - Case 4 (FRI), 201–203
- Online analytic processing (OLAP) and business intelligence, 9, 401–426
 - cube queries, parallelization of, 412–417
 - cume_dist queries, parallelization, 419–420
 - histogram queries, parallelization, 420–422
 - moving average queries, parallelization, 422–424
 - NTILE queries, parallelization, 420–422
 - parallel multidimensional analysis, 402–405
 - parallelization without using ROLLUP, 412
 - ranking queries, parallelization of, 418–419
 - rollup queries, parallelization, 405–412
 - top-N queries, parallelization of, 418–419
 - windowing queries, parallelization of, 422–424
- Open Grid Service Architecture (OGSA), 27
- Optimistic algorithms, 309
- Optimistic Plan Correction (OPC), 278
- Originator's algorithm for Grid-ACP, 345

- Page, 34
- Parallel association rules, 440–450, *See also* Association rule mining

- Parallel universal qualification, *See* Collection join queries
- Parallelism
 - forms of, 12–19
 - independent parallelism, 15
 - interoperation parallelism, 12, 15–18
 - interquery parallelism, 12, 13–14
 - intraoperation parallelism, 12, 15, 16
 - intraquery parallelism, 12, 14–15
 - mixed parallelism, 18–19
 - pipeline parallelism, 15–18
- Partial CUBE queries, analysis of, 416–417
- Partial ROLLUP queries, 411–412
- Partially Replicated Indexing (PRI) Structures, 168, 171–178
 - index entry deletion, 185
 - index entry insertion in, 184
 - multiple node pointers model for, 174
 - PRI-1, 172, 174
 - PRI-2, 176–177
 - maintaining, 182–188
 - PRI-3, 177–178
 - replication in, 177
- Participant's algorithm for Grid-ACP, 346
- Partition/Partitioning, 296
 - dynamic cluster query optimization, 281–284
 - hash join, 283
 - simple join, 283
 - partitional clustering, 468
 - partitioned tree construction, 493
 - tuning, 263
- Pessimistic algorithms, 309
- Pessimistic Plan Correction (PPC), 279
- Pipeline merging costs, 102
- Pipeline parallelism, 15–18
 - drawbacks, 17–18
- Predictive data mining, 431–432
- Probing, 119
- Processing skew, 40
- Projectivity notation, parallel GroupBy-Join, 152
- Projectivity ratio, 37

- Query processing, parallel, 5–6
 - motivations, 5–6
 - objectives, 7–12
 - communication, 11–12
 - interference, 11–12
 - parallel obstacles, 10–12
 - scale up, 8–10
 - skew, 12
 - speed up, 7–8
 - parameters, 37
 - results generation cost, 45
- Query scheduling and optimization, 256–287
 - cluster query processing model, 270–275
 - degree of parallelization, 258
 - bushy-tree parallelization, 258
 - flat-tree parallelization, 258
 - left-deep tree parallelization, 258
 - right-deep tree parallelization, 258
 - dynamic cluster query optimization, 275–284, *See also individual entry*
 - query execution plan, 257–259
 - scheduling rules, 269–270
 - serial vs. parallel execution scheduling, 264–269
 - subqueries execution scheduling strategies, 259–263
- Querying vs. Mining, 433–436
 - supervised learning, 436
 - unsupervised learning, 433–435
- Quorum-based protocols, 317–318

- Random-unequal data partitioning, 59
- Range partitioning, 58–59, 124–126
- Range search query, 53
- Ranking queries, parallelization of, 418–419
- Read transaction operation for GRAP, 371–372
- Read-one-write-all (ROWA) approach, 316
- Real Application Cluster (RAC), 28
- Receiving cost
 - parallel binary-merge sort, 100
 - parallel redistribution binary-merge sort, 102
- Receiving records cost, 107
 - disjoint partitioning, 130
 - divide and broadcast, 129
- Record, 34
- Recovery algorithm for Grid-ACP, correctness of, 361–365
- Recovery protocols of Grid-ACP, comparison, 359–361
- Redistribution binary-merge sort, 86–88
 - cost model, 101–102
- Redistribution merge-all sort, 88–90
 - cost model, 102–103
- Redistribution method, 94–96
 - cost model, 107–108
- Redo approach, 314
- Replica management in grids, 367–386, *See also* Grid replica access protocol (GRAP)
 - comparison among protocols, 381–383
 - asynchronous, 381
 - synchronous, 381
 - handling multiple partitioning, 378–384
 - contingency GRAP, 378–381
 - motivation, 367–368
 - replica architecture, 368–370

- high-level replica management architecture, 368–369
- Replica synchronization protocols, 314–318
 - network partitioning, 315–316
 - primary copy, 317
 - quorum-based protocols, 317–318
 - read-one-write-all (ROWA) approach, 316
 - ROWA-Available (ROWA-A), 316–317
- Replicated data, grid atomic commitment in, 387–398
 - transaction properties, 395–397
- Replication, 296
- Result generation cost, 70, 72
- Result parallelism, 438–440
 - for the decision tree, 492–495
 - parallel k -means, 475–477
- Retry approach, 314
- Right-deep tree parallelization, 258
- Rollup queries, parallelization, 405–412
 - multiple ROLLUP queries, 409–411
 - parallelization without using ROLLUP, 412
 - partial ROLLUP queries, 411–412
 - single ROLLUP queries, 405–409
- Round-robin data partitioning, 56
- ROWA-Available (ROWA-A), 316–317

- Save cost
 - parallel binary-merge sort, 100
 - parallel merge-all sort, 98–99
 - parallel partitioned sort, 104
 - parallel redistribution binary-merge sort, 102
 - parallel redistribution merge-all sort, 103
 - serial external merge-sort, 97
- Scalar aggregate, 79
- Scale up objective, parallel query processing, 8–10
 - calculation, 8
 - data scale up, 8, 9–10
 - linear scale up, 8
 - transaction scale up, 8, 9
- Scanning cost, 44, 70, 72
 - disjoint partitioning, 129
 - divide and broadcast, 128
 - local join, 130
- Scheduling rules, 269–270
- Search, parallel, 51–74
 - algorithm, 69–74
 - comparison, 74
 - local searching method, 73–74
 - processor activation or involvement, 73
 - serial search algorithms, 69–72
 - data partitioning, 54–69
 - basic, 55–60
 - complex, 60–69
 - search queries, 51–54
 - exact match search, 52
 - multiattribute search query, 54
 - range search query, 53
- Search queries parallel processing using index, 192–200, *See also* One-index join query; Two-index join query
 - multi-index, 195–200
 - intersection method, 195
 - one-index, 192–195
 - algorithm for, 195
 - index tree traversal, 192–194
 - parallel exact-match search queries, 192–194
 - parallel range selection query, 194–195
 - processor involvement, 192–193
 - record loading, 192, 194
- Select cost, 45, 70, 72
 - disjoint partitioning, 129
 - divide and broadcast, 128
 - local join, 130
 - parallel binary-merge sort, 100
 - parallel merge-all sort, 98–99
 - parallel partitioned sort, 104
 - parallel redistribution binary-merge sort, 102
 - parallel redistribution merge-all sort, 103
 - serial external merge-sort, 97
- Selection, 51
- Selectivity notation, parallel GroupBy-Join, 152
- Selectivity ratio, 37
- Semantic atomicity, 343
- Sequential patterns
 - data mining, 427–463
 - concepts, 452–456
 - count distribution, 459
 - data distribution, 459–461
 - joining phase, 457
 - pruning phase, 458–459
- Serial execution among subqueries, 259–261
- Serial external sorting, 80–83
- Serial join algorithms, 114–120
 - algorithm comparison, 120
 - hash-based, 117–120
 - nested-loop, 114–115
 - sort-merge, 116–117
- Serial search algorithms, 69–72
 - binary search, 71–72
 - linear search, 69–71
- Serial subqueries execution scheduling, 490
- Serial vs. parallel execution scheduling, 264–269
 - nonskewed subqueries, 264–265, 267–269
 - skewed subqueries, 265–269

550 INDEX

- Serializability theory, grid, 325–329
- global-global conflict, 329
- global-local conflict, 329
- local-local conflict, 329
- Set/bag hashing, 229
- Shared-disk architectures, 20–21
- Shared-everything architecture, 54
- Shared-memory architectures, 20–21
- Shared-nothing architecture, 22, 54
- Similarity measures, 467–468
- Simple replication, 234
- Single ROLLUP queries, 405–409
- Skew/Skewness, 12, 39–40, 260
 - skewed subqueries, 265–267
- Sort, parallel, 77–91
 - binary-merge sort, 85–86
 - merge-all sort, 83–84
 - partitioned sort, 90–91
 - redistribution binary-merge sort, 86–88
 - redistribution merge-all sort, 88–89
 - sort-hash collection-equi join algorithm, 228–231
 - sort-hash collection-intersect join algorithm, 245–246
 - sort-hash sub-collection join algorithm, 249–251
- Sorting cost
 - parallel merge-all sort, 98
 - parallel partitioned sort, 104
 - serial external merge-sort, 97
- Sort-merge nested-loop subcollection join algorithm, 116–117, 248–249
- Speed up objective, parallel query processing, 7–8
 - linear speed up, 7
 - sublinear speed up, 7
 - superlinear speed up, 7
- Start up costs, 10–12
- State diagram of Grid-ACP, 343–344
 - compensate states, 343
 - pre-abort state, 343
 - sleep state, 343
- Storage analysis, index, 188–192
 - parallel processors, storage cost models for, 191–192
 - FRI Storage, 192
 - NRI Storage, 191
 - PRI Storage, 191
 - uniprocessors, storage cost models for, 189–191
 - index storage, 189–191
 - record storage, 189
- Subcollection join algorithms, 224–225, 246–252
 - data partitioning, 247–248
 - hash subcollection join algorithm, 251–252
 - sort-hash sub-collection join algorithm, 249–251
 - sort-merge nested-loop subcollection join algorithm, 248–249
- Sublinear speed up objective, parallel query processing, 7
- Submission phase of GCC protocol, 329–330
- Subqueries execution scheduling strategies, 259–263
 - parallel execution among subqueries, 261–263
 - dynamic resource division, 262
 - static resource division, 262–263
 - serial execution among subqueries, 259–261
- Superlinear speed up objective, parallel query processing, 7
- Symmetric multi processor (SMP) machines, 21
 - cluster of, 23
- Synchronous protocols, GRAP, 381
- Synchronous tree construction approach, 491
- Systems parameters, 36
- Table, 34–35
- Task stealing, 263
- Termination phase of GCC protocol, 331–333
- Testing data set, 466
- Three-phase commit (3PC), 312–313
- Time complexity analysis, Grid-ACP, 349
- Time equalization method, 263
- Time unit costs, 37–38
- Time-series analysis, parallel data mining, 433
- Timestamp ordering algorithms, 310
- Top-N queries, parallelization of, 418–419
- Training data set, 466
- Transactions in distributed and grid databases, 291–320
 - acid properties of, 301–303
 - atomic commit protocols, 310–314
 - basic definitions on transaction management, 299–301
 - concurrency control protocols, 309–310
 - management, 303–307
 - centralized DBMSs, 303–305
 - heterogeneous distributed DBMSs, 305–307
 - homogeneous DBMSs, 303–305
 - replica synchronization protocols, 314–318
- Transactions/Transaction properties in replicated environment, 395–397
 - atomicity, 395
 - consistency and isolation, 396
 - durability, 396

- scale up, 8, 9
 - submission procedure, 362–363
- Triggering, 276
- Two-index join query, 200, 203–207
 - Case 1, 203–205
 - Case 2, 205–207
- Two-Phase Commit (2PC), 93–94, 311–312
 - cost model, 104–105
- Uniprocessors, storage cost models for, 189–191
- Vertical data partitioning, 55
- Windowing queries, parallelization of, 422–424
- Write transaction operation for GRAP, 372–375
- Writing cost, 44
- Zipf distribution, 265

