|  |  |  |  |
| --- | --- | --- | --- |
| **Maximum N** | **Maximum complexity can be applied** | **Algorithms** | **Possible data structures to use** |
| 1,000,000,000 and higher | log (n), sqrt (n) | binary search, ternary search, fast exponentiation, euclid algorithm |  |
| 10,000,000 | n, n (log (log (n))), n log(n) | set intersection, Eratosthenes sieve, radix sort, KMP, topological sort, Euler tour, strongly connected components, 2sat | disjoint sets, tries, hash\_map, [rolling hash](http://www.infoarena.ro/blog/rolling-hash)deque |
| 1,000,000 | n log n | sorting, divide and conquer, sweep line, Kruskal, Dijkstra | segment trees, range trees, heaps, treaps, binary indexed trees, suffix arrays |
| 100,000 | n log2 n | divide and conquer | 2d range trees |
| 50,000 | n1.585, n sqrt n | Karatsuba, square root trick | two level tree |
| 1000 – 10,000 | n2 | largest empty rectangle, Dijkstra, Prim (on dense graphs) |  |
| 300-500 | n3 | all pairs shortest paths, largest sum submatrix, naive matrix multiplication, matrix chain multiplication, gaussian elimination, network flow |  |
| 30-50 | n4, n5, n6 |  |  |
| 25 – 40 | 3n/2, 2n/2 | meet in the middle | hash tables (for set intersection) |
| 15 – 24 | 2n | subset enumeration, brute force, dynamic programming with exponential states |  |
| 15 – 20 | n2 2n | dynamic programming with exponential states | bitsets, hash\_map |
| 13-17 | 3n | dynamic programming with exponential states | hash\_map (to store the states) |
| 11 | n! | brute force, backtracking, next\_permutation |  |
| 8 | nn | brute force, cartesian product |  |