# Who is Interested in Algorithms and Why? Lessons from the Stony Brook Algorithms Repository*

Steven S. Skiena†
Department of Computer Science
SUNY Stony Brook, NY 11794-4400
skiena@cs.sunysb.edu

April 30, 1999

**Abstract**

We present "market research" for the field of combinatorial algorithms and algorithm engineering, attempting to determine which algorithmic problems are most in demand in applications. We analyze 1,503,135 WWW hits recorded on the Stony Brook Algorithms Repository (http://www.cs.sunysb.edu/~algorith), to determine the relative level of interest among 75 algorithmic problems and the extent to which publicly available algorithm implementations satisfy this demand.

## 1 Introduction

A primary goal of algorithm engineering is to provide practitioners with well-engineered solutions to important algorithmic problems. Our beliefs as to which problems *are* important to practitioners have been based primarily on anecdotal evidence. To provide more objective information, it seems useful to conduct "market research" for the field of combinatorial algorithms, by determining which algorithmic problems are most in demand in applications, and how well currently available implementations satisfy this demand.

This paper is an attempt to answer these questions. We present an analysis of 1,503,135 WWW hits recorded on the Stony Brook Algorithms Repository over a one year period, from February 22, 1998 to February 27, 1999. The Repository (http://www.cs.sunysb.edu/~algorith) provide a resource where programmers, engineers, and scientists can go to find implementations of algorithms for fundamental problems. User feedback and WWW traffic statistics suggest that it has proven valuable to people with widely varying degrees of algorithmic sophistication.

The structure of the Algorithms Repository makes it well suited to measure the interest in different algorithmic problems. For each of 75 fundamental algorithm problems, we have collected the best publicly available implementations that we could find. These problems have been indexed in major web search engines, so anyone conducting a search for information on a combinatorial algorithm problem is likely to stumble across our site. Further, special indexes and hyperlinks aboard our site help guide users to other relevant information.

This paper is organized as follows. In Section 2, we discuss the structure of the Algorithms Repository in more depth, to provide better insight into the nature of the data we present below.

*A preliminary version of this paper appeared in the Second Workshop on Algorithm Engineering, Saarbrucken, Germany, August 20-22, 1998.

†Supported in part by ONR Award 431-0857A and NSF Grant CCR-9625669.

In Section 3, we analyze WWW traffic to determine the most popular and least popular algorithmic problems. In Section 4, we report on what our users are finding. Each implementation available on the Repository has been rated as to its usefulness for the corresponding problem. By studying these ratings, we can assess the current state of the art of combinatorial computing, and see how well it matches user demand. Finally, in Section 5, we attempt to get a handle on where the interest in algorithms is located, both geographically and professionally.

The only other attempt at polling interest in algorithmic problems which I am aware of is by Crescenzi and Kann [1], who use WWW hits on their compendium of NP optimization problems to measure interest in different NP-hard problems. Any polling-based research is subject to a variety of bias and ambiguities. I make no grand claims as to how accurately this data measures the relative importance of different algorithmic research to society. I found many of the results quite surprising, and hope they will be of interest to the algorithmic community.

## 2 The Stony Brook Algorithms Repository

The Stony Brook Algorithms Repository was developed in parallel with my book [7], *The Algorithm Design Manual*, and the structure of the repository mirrors the organization of my book. The world has been divided into a total of 75 fundamental algorithmic problems, partitioned among data structures, numerical algorithms, combinatorial algorithms, graph algorithms, hard problems, and computational geometry. See Table 6 or http://www.cs.sunysb.edu/~algorith for the list of 75 problems.

For each problem, the book poses questions to try to reveal the issues inherent in a proper formulation, and then tries to suggest the most pragmatic algorithm solution available. Where appropriate, relevant implementations are noted in the book, and collected on the Algorithms Repository, most of which has been mirrored on a CD-ROM included with the book. In total, we have identified a collection of 56 relevant algorithm implementations. Finding these codes required a substantial effort. Since many of these implementations proved applicable to more than one problem, the repository contains an average of three relevant implementations per problem.

Each problem page has a link to each relevant implementation page, as well as to pages associated with closely related problems. Each implementation page contains a link to the page associated with each problem to which it is applicable. Further, indexes contain links to implementations by programming language, subject area, and pictorial representation. Together these links enable the user to move easily through the site.

## 3 What are People Looking For?

Out of the 1.5 million hits recorded on this site over the one year interval, 393,467 of them were to primary html and shtml files. This latter count more accurately represents the number of mouse-clicks performed by users than the total hits, since most of the remaining hits are on image files associated with these pages. Therefore, we will limit further analysis to hits on these files.

Because user ID information is not logged on our WWW server, it is difficult to judge exactly how many different people accounted for these hits. Based on the roster of machines which accessed the site, I estimate that roughly 40,000 different people paid a visit during this 10 week study. Some fraction of hits came from webcrawler robots instead of human users, however I believe they had only a minor effect on our statistics. Observe that the least frequently clicked shtml file (containing the copyright notice for the site) was hit only 298 times versus 17,733 hits for the most frequently accessed page (the front page). The page advertising my book was hit 7,014 times, although I have no way of knowing whether any of them actually ordered it.

Table 1 reports the number of hits distributed among our highest level of classification – the seven major subfields of algorithms. Two different hit measures are reported for each

| Problem Category | Index Hits | Subsection Hits | Problems |
|---|---|---|---|
| Data Structures | 6698 | 14492 | 6 |
| Numerical Problems | 4499 | 12812 | 11 |
| Combinatorial Problems | 3419 | 10114 | 10 |
| Graph Problems: Polynomial | 4496 | 17492 | 12 |
| Graph Problems: Hard | 3849 | 13447 | 11 |
| Computational Geometry | 7031 | 25129 | 16 |
| Set and String Problems | 3003 | 10776 | 9 |
| Totals | 32995 | 104262 | 75 |

Table 1: Hits by Major Section Index

| Programming Language | Index Hits | Implementations |
|---|---|---|
| C language | 4776 | 37 |
| C++ | 5397 | 11 |
| Fortran | 868 | 6 |
| Lisp | 698 | 1 |
| Mathematica | 652 | 3 |
| Pascal | 1556 | 5 |
| Totals | 13947 | 63 |

Table 2: Hits by Programming Language Index

subfield, first the number of hits to the menu of problems within the subfield, and second the total number of hits to individual problem pages within this subfield. Computational geometry proved to be the most popular subfield by both measures, although outweighed by the interest in graph problems split across two subtopics. Data structures recorded the highest "per-problem" interest, but I was surprised by the relative lack of enthusiasm for set and string algorithms.

Table 2 reports the number of hits distributed among the various programming language submenus. C++ seems to have supplanted C as the most popular programming language among developers, although there is clearly a lag in the size of the body of software written in C++. C remains the source language for over half the implementations available on the Algorithm Repository. User interest in Mathematica rivals that of Fortran, perhaps suggesting that computer algebra systems are becoming the language of choice for scientific computation. There was no submenu associated with Java, reflecting what was available when I built the repository. The total number of implementations in Table 2 is greater than 56 because seven codes are written in more than one language.

Table 3 reports the 15 most popular and least popular algorithmic problems, as measured by the number of hits the associated pages received. Hit counts for all of the 75 problems appears in Table 6. Several observations can be drawn from this data:

- Shortest path (with 3660 hits) was the most popular of the algorithmic problems over the course of the study. Much of this interest was no doubt from students in algorithms courses seeking an edge. However its course-partner minimum spanning tree proved much less popular, finishing in seventh place with 2922 hits.
- Of the six data structure problems, only set union-find (number 31) failed to make the top 15 cut.
- There is more interest in kd-trees (3198 hits) than nearest-neighbor searching (2936 hits), even though the former is most often used as a means to solve the latter.
- People seem much more interested in generating permutations than subsets (1474 hits to

| Most Popular Problems | Hits | Least Popular Problems | Hits |
|---|---|---|---|
| shortest-path | 3660 | generating-subsets | 854 |
| kd-trees | 3198 | edge-coloring | 817 |
| dictionaries | 3022 | satisfiability | 792 |
| traveling-salesman | 2963 | independent-set | 789 |
| convex-hull | 2963 | cryptography | 786 |
| nearest-neighbor | 2936 | text-compression | 767 |
| minimum-spanning-tree | 2922 | maintaining-arrangements | 727 |
| voronoi-diagrams | 2815 | set-packing | 703 |
| triangulations | 2786 | planar-drawing | 678 |
| sorting | 2734 | median | 671 |
| graph-data-structures | 2596 | bandwidth | 629 |
| string-matching | 2304 | factoring-integers | 628 |
| suffix-trees | 2213 | shortest-common-superstring | 520 |
| priority-queues | 2208 | determinants | 520 |
| geometric-primitives | 2162 | feedback-set | 483 |

Table 3: Most and least popular algorithmic problems, by repository hits.

854 hits), presumably reflecting the perceived difficulty of the task.

- Surprisingly popular problems include traveling salesman (number 4), suffix trees (number 13), the knapsack problem (number 18). These might reflect educational interest, although Table 5 shows that almost twice as many total .com hits were recorded than total .edu hits.
- Surprisingly unpopular problems include independent set (number 64), planar drawing (number 69), and satisfiability (number 63). Such obviously commercial problems as cryptography (number 65) and text compression (number 66) proved unpopular presumably because better WWW resources exist elsewhere for these problems.
- My users (people seeking programs) rated NP-complete problems substantially differently than users of the compendium of optimization problems [1] (people seeking references). Our five most popular hard problems, in order were traveling salesman (2963 hits), knapsack (1926 hits), graph coloring (1847 hits), Hamiltonian cycle (1681 hits), and bin packing (1496 hits). Their most popular hard problem was vertex cover, which ranked 58th of all problems on our list.

It is interesting to note that only 17,733 hits occured to the front-page of the site, which suggests that most visitors never saw the main index of the site. This implies that most users initially entered the site through a keyword-oriented search engine, and gives credence to the notation that these hits measure problem interest more than just directionless wandering through the site.

# 4 What are They Finding?

The majority of visitors to the Algorithms Repository come seeking implementations of algorithms which solve the problem they are interested in. To help guide the user among the relevant implementations for each problem, I have rated each implementation from 1 (lowest) to 10 (highest), with my rating reflecting my opinion of the chances that an applied user will find that this implementation solves their problem.

My ratings are completely subjective, and in many cases were based on a perusal of the documentation instead of first-hand experience with the codes. Therefore, I cannot defend the correctness of my ratings on any strong objective basis. Still, I believe that they have proven useful in pointing people to the most relevant implementation. Of the two linear programming packages, lpsolve and linprog, the higher rated code received over five times as many hits (1859 to 340). [1]

Table 7 records the number of hits received for each implementation, along with the problem for which it received the highest rating, as well its average rating across all problems. LEDA [3] received almost as many hits (8806) as the two following implementations, both associated with popular books [5] (5339) and [2] (4360). The fourth most popular implementation was (surprisingly) Ranger [4] (3514), an implementation of kd-trees. This reflects the enormous popularity of nearest-neighbor searching in higher dimensions, as well as the fact that I have not updated the list of implementations since the publication of the book in November 1997. Arya and Mount's recently released ANN (http://www.cs.umd.edu/∼mount/ANN/) would be a better choice. Note that these counts record the number of people who looked at the information page associated with each implementation. The actual number of ftps is unknown but presumably much lower.

Despite their shortcomings, I believe that these ratings provide a useful insight into the state of the art of combinatorial computing today. Hits per problem page measures the level of interest in a particular algorithmic problem. *Program mass*, the sum of the rankings of all implementations for a given problem, provides a measure of how much effort has been expended by the algorithm engineering community on the given problem. By comparing the ranks of each problem by program mass and the popularity, we can assess which problems are most (and least) in need of additional implementations.

Table 4 presents the results of such an analysis, showing the 20 most under (and over) implemented algorithmic problems. Kd-trees (rank 1) and suffix trees (rank 2) are the most needed data structure implementations, while the closely related problems of bin packing (rank 3) and knapsack (rank 4) are in the most need of algorithm implementations. There seems to be greater interest than activity in routing problems like Eulerian cycle/chinese postman (rank 9) and traveling salesman (rank 14). On the other hand, traditional algorithm engineering topics like matching (rank 59) and network flow (rank 56) have resulted in a relative abundance of codes for these problems.

# 5 Who is Looking?

By analyzing the domain names associated with each hit on the Algorithm Repository, we can see who is interested in algorithms. Table 5 records the number of hits by top-level domain. I believe that more hits were recorded by industrial users than educational ones, since the .com (82387) and .net (49172) domains together account for almost three times the number of .edu (46234) hits, although of course many students also have accounts with internet providers.

It is interesting and amusing to see the distribution of hits by country code. No less than 100 nations visited the Algorithm Repository during this one year interval, suggesting a much broader interest in algorithms than I would have thought. Hit count per nation is summarized in Table 8.

The most algorithmically inclined nation after the United States (presumably the source of most .com and .edu hits) was, not surprisingly, Germany (6099). The United Kingdom (3795), France (2811), and Spain (2501) each accounted for significantly more hits than Israel (1265), Japan (1668), and the Netherlands (1223) – suggesting that the interest does not completely correlate with my perception of the amount of algorithmic research activity in these nations. Ireland, which finished ninth in the survey of [1] ranked 40th among nations in ours. Two of the largest producers of graduate students in computer science, China (66) and India (230), ranked surprisingly low in the number of hits despite the presence of substantial software industries. Presumably this reflects limited WWW access within these countries.

[1] Any software developer who is dissatisfied with their ratings will perhaps be gratified to learn that my own *Combinatorica* [6] received the fourth lowest average score among the 56 rated implementations.

| Most Needed Implementations | Rank by Mass | Hits | Δ | Least Needed Implementations | Rank by Mass | Hits | Δ |
|---|---|---|---|---|---|---|---|
| kd-trees | 52 | 2 | -50 | network-flow | 5 | 19 | 14 |
| suffix-trees | 63 | 13 | -50 | random-numbers | 14 | 28 | 14 |
| bin-packing | 75 | 27 | -48 | dfs-bfs | 17 | 32 | 15 |
| knapsack | 58 | 18 | -40 | matching | 1 | 17 | 16 |
| polygon-partitioning | 66 | 35 | -31 | unconstrained-optimization | 36 | 52 | 16 |
| simplifying-polygons | 69 | 41 | -28 | vertex-coloring | 4 | 20 | 16 |
| nearest-neighbor | 32 | 6 | -26 | fourier-transform | 23 | 40 | 17 |
| minkowski-sum | 72 | 47 | -25 | cryptography | 44 | 65 | 21 |
| eulerian-cycle | 67 | 43 | -24 | satisfiability | 42 | 63 | 21 |
| dictionaries | 24 | 3 | -21 | high-precision-arithmetic | 15 | 37 | 22 |
| set-cover | 71 | 50 | -21 | bandwidth | 48 | 71 | 23 |
| set-data-structures | 51 | 31 | -20 | matrix-multiplication | 22 | 45 | 23 |
| motion-planning | 74 | 57 | -17 | drawing-trees | 30 | 54 | 24 |
| traveling-salesman | 21 | 4 | -17 | edge-coloring | 38 | 62 | 24 |
| scheduling | 65 | 49 | -16 | maintaining-arrangements | 43 | 67 | 24 |
| string-matching | 26 | 12 | -14 | planar-drawing | 40 | 69 | 29 |
| calendar-calculations | 59 | 46 | -13 | generating-graphs | 11 | 44 | 33 |
| clique | 45 | 33 | -12 | determinants | 39 | 74 | 35 |
| graph-partition | 54 | 42 | -12 | generating-subsets | 20 | 61 | 41 |
| graph-isomorphism | 50 | 39 | -11 | generating-partitions | 12 | 60 | 48 |

Table 4: Most needed and least needed implementations, based on program mass and hit ranks

| domain | .com | .edu | .gov | .mil | .net | .org | $[0-9]*$ | countries | totals |
|---|---|---|---|---|---|---|---|---|---|
| hits | 82387 | 46234 | 2022 | 1682 | 49172 | 1273 | 70002 | 47518 | 300290 |

Table 5: Hits by top level domain

# 6 Conclusions

Analysis of hits to the Stony Brook Algorithm Repository provides interesting insights to the demand for algorithms technology, and the state of the art of available implementations. It would be interesting to repeat this analysis at regular intervals to see how the demand changes over time.

This most important conclusion of this work is that there is a demand for high quality implementations of algorithms for several important and interesting problems. I urge members of the algorithm engineering community to consider projects for problems on the left side of Table 4, for these represent the real open problems in the field. Indeed, I would be happy to add any results of this work to the Algorithm Repository for others to benefit from.


# 7 Acknowledgements

I would like to thank Ricky Bradley and Dario Vlah, who helped to build the software infrastructure which lies behind the Stony Brook Algorithm Repository. I also thank Gaurav Sehgal for his help collecting these statistics.


# References


[1] P. Crescenzi and V. Kann. How to find the best approximation results – a followup to Garey and Johnson. *ACM SIGACT News*, 29-4:90–97, December 1998.

[2] G. Gonnet and R. Baeza-Yates. *Handbook of Algorithms and Data Structures.* Addison-Wesley, Wokingham, England, second edition, 1991.

[3] K. Mehlhorn and S. Näher. LEDA, a platform for combinatorial and geometric computing. *Communications of the ACM*, 38:96–102, 1995.

[4] M. Murphy and S. Skiena. Ranger: A tool for nearest neighbor search in high dimensions. In *Proc. Ninth ACM Symposium on Computational Geometry*, pages 403–404, 1993.

[5] R. Sedgewick. *Algorithms in C++.* Addison-Wesley, Reading MA, 1992.

[6] S. Skiena. *Implementing Discrete Mathematics.* Addison-Wesley, Redwood City, CA, 1990.

[7] S. Skiena. *The Algorithm Design Manual.* Springer-Verlag, New York, 1997.

| | | All Implementations | | Best Implementation | |
|---|---|---|---|---|---|
| Problem | Hits | Impl. Count | Avg Score | Program Name | Rating |
| approximate-pattern-matching | 1403 | 3 | 6.3 | agrep | 10 |
| bandwidth | 629 | 2 | 7.5 | toms | 9 |
| bin-packing | 1496 | 1 | 3.0 | xtango | 3 |
| calendar-calculations | 1109 | 1 | 10.0 | reingold | 10 |
| clique | 1344 | 3 | 5.3 | dimacs | 9 |
| convex-hull | 2963 | 7 | 5.4 | qhull | 10 |
| cryptography | 786 | 3 | 5.3 | pgp | 10 |
| determinants | 520 | 4 | 4.5 | linpack | 8 |
| dfs-bfs | 1356 | 7 | 3.9 | LEDA | 8 |
| dictionaries | 3022 | 4 | 6.0 | LEDA | 10 |
| drawing-graphs | 1339 | 3 | 7.0 | graphed | 9 |
| drawing-trees | 932 | 3 | 7.0 | graphed | 9 |
| edge-coloring | 817 | 4 | 4.5 | stony | 6 |
| edge-vertex-connectivity | 950 | 3 | 4.0 | combinatorica | 4 |
| eulerian-cycle | 1154 | 2 | 3.0 | combinatorica | 3 |
| feedback-set | 483 | 1 | 4.0 | graphbase | 4 |
| finite-state-minimization | 1610 | 4 | 5.7 | grail | 9 |
| fourier-transform | 1192 | 5 | 5.0 | fftpack | 10 |
| generating-graphs | 1146 | 5 | 6.8 | graphbase | 10 |
| generating-partitions | 859 | 5 | 6.6 | wilf | 8 |
| generating-permutations | 1474 | 4 | 7.0 | ruskey | 8 |
| generating-subsets | 854 | 4 | 6.3 | wilf | 8 |
| geometric-primitives | 2162 | 5 | 5.4 | LEDA | 8 |
| graph-data-structures | 2596 | 6 | 6.7 | LEDA | 10 |
| graph-isomorphism | 1217 | 2 | 6.5 | nauty | 10 |
| graph-partition | 1154 | 2 | 6.0 | link | 8 |
| hamiltonian-cycle | 1681 | 5 | 4.2 | toms | 6 |
| high-precision-arithmetic | 1266 | 5 | 5.6 | pari | 9 |
| independent-set | 789 | 2 | 6.0 | dimacs | 7 |
| intersection-detection | 2128 | 5 | 5.2 | LEDA | 7 |
| kd-trees | 3198 | 3 | 4.0 | ranger | 8 |
| knapsack | 1926 | 2 | 5.0 | toms | 6 |
| linear-equations | 1221 | 3 | 6.6 | lapack | 10 |
| linear-programming | 1841 | 5 | 4.4 | lpsolve | 9 |
| longest-common-substring | 927 | 2 | 5.0 | cap | 8 |
| maintaining-arrangements | 727 | 2 | 8.0 | arrange | 9 |
| matching | 1973 | 10 | 5.2 | goldberg | 9 |
| matrix-multiplication | 1128 | 5 | 5.0 | linpack | 7 |
| median | 671 | 2 | 5.0 | handbook | 6 |
| minimum-spanning-tree | 2922 | 9 | 4.0 | LEDA | 6 |
| minkowski-sum | 1102 | 1 | 4.0 | eppstein | 4 |
| motion-planning | 899 | 1 | 3.0 | orourke | 3 |
| nearest-neighbor | 2936 | 4 | 5.2 | ranger | 7 |
| network-flow | 1860 | 8 | 5.0 | goldberg | 10 |
| planar-drawing | 678 | 3 | 5.7 | graphed | 8 |
| point-location | 1584 | 4 | 4.8 | LEDA | 7 |
| polygon-partitioning | 1333 | 1 | 8.0 | geompack | 8 |
| priority-queues | 2208 | 8 | 4.5 | LEDA | 9 |
| random-numbers | 1489 | 6 | 4.8 | simpack | 7 |
| range-search | 1511 | 4 | 4.8 | LEDA | 8 |
| satisfiability | 792 | 2 | 8.0 | posit | 8 |
| scheduling | 1074 | 2 | 4.0 | syslo | 4 |
| searching | 1267 | 3 | 5.6 | handbook | 7 |
| set-cover | 1063 | 1 | 5.0 | syslo | 5 |
| set-data-structures | 1375 | 3 | 4.3 | LEDA | 5 |
| set-packing | 703 | 1 | 5.0 | syslo | 5 |
| shape-similarity | 861 | 2 | 6.5 | snns | 7 |
| shortest-common-superstring | 520 | 1 | 8.0 | cap | 8 |
| shortest-path | 3660 | 7 | 5.0 | goldberg | 9 |
| simplifying-polygons | 1159 | 1 | 5.0 | skeleton | 5 |
| sorting | 2734 | 7 | 4.9 | moret | 7 |
| steiner-tree | 1093 | 2 | 7.5 | salowe | 8 |
| string-matching | 2304 | 5 | 4.4 | watson | 7 |
| suffix-trees | 2213 | 2 | 4.0 | stony | 6 |
| text-compression | 767 | 1 | 5.0 | toms | 5 |
| thinning | 1039 | 1 | 9.0 | skeleton | 9 |
| topological-sorting | 1772 | 6 | 3.5 | LEDA | 7 |
| transitive-closure | 930 | 2 | 4.0 | LEDA | 6 |
| traveling-salesman | 2963 | 5 | 5.0 | tsp | 8 |
| triangulations | 2786 | 8 | 5.9 | triangle | 9 |
| unconstrained-optimization | 972 | 3 | 6.3 | toms | 8 |
| vertex-coloring | 1847 | 8 | 5.0 | dimacs | 7 |
| vertex-cover | 876 | 3 | 5.0 | clique | 6 |
| voronoi-diagrams | 2815 | 6 | 5.3 | fortune | 9 |

Table 6: Hits by algorithmic problem, with implementation ratings

| | | Major Problem | | All Problems | |
|---|---|---|---|---|---|
| Software | Hits | Problem Name | Rating | Count | Average |
| ASA | 631 | unconstrained-optimization | 6 | 1 | 6.0 |
| LEDA | 8806 | graph-data-structures | 10 | 30 | 6.2 |
| agrep | 895 | approximate-pattern-matching | 10 | 1 | 10.0 |
| arrange | 430 | maintaining-arrangements | 9 | 3 | 7.3 |
| bipm | 586 | matching | 8 | 1 | 8.0 |
| cap | 524 | shortest-common-superstring | 8 | 2 | 8.0 |
| clarkson | 524 | convex-hull | 6 | 1 | 6.0 |
| clique | 667 | clique | 6 | 6 | 5.5 |
| combinatorica | 2447 | generating-graphs | 8 | 28 | 4.0 |
| culberson | 587 | vertex-coloring | 6 | 2 | 5.0 |
| dimacs | 1308 | matching | 9 | 10 | 5.6 |
| eppstein | 1244 | minkowski-sum | 4 | 2 | 4.0 |
| fftpack | 651 | fourier-transform | 10 | 1 | 10.0 |
| fortune | 1675 | voronoi-diagrams | 9 | 2 | 8.0 |
| genocop | 422 | unconstrained-optimization | 5 | 1 | 5.0 |
| geolab | 693 | geometric-primitives | 5 | 1 | 5.0 |
| geompack | 792 | polygon-partitioning | 8 | 2 | 8.0 |
| goldberg | 2215 | network-flow | 10 | 3 | 9.3 |
| grail | 1123 | finite-state-minimization | 9 | 1 | 9.0 |
| graphbase | 2658 | generating-graphs | 10 | 17 | 4.4 |
| graphed | 1736 | drawing-graphs | 9 | 7 | 6.4 |
| handbook | 4360 | dictionaries | 8 | 12 | 4.9 |
| htdig | 371 | text-compression | 7 | 1 | 7.0 |
| lapack | 446 | linear-equations | 10 | 1 | 10.0 |
| link | 655 | graph-partition | 8 | 4 | 4.5 |
| linpack | 68 | determinants | 8 | 2 | 7.5 |
| linprog | 340 | linear-programming | 4 | 1 | 4.0 |
| lpsolve | 1859 | linear-programming | 9 | 1 | 9.0 |
| math | 460 | matrix-multiplication | 6 | 1 | 6.0 |
| moret | 1831 | sorting | 7 | 17 | 3.8 |
| nauty | 970 | graph-isomorphism | 10 | 1 | 10.0 |
| north | 805 | drawing-graphs | 7 | 2 | 7.0 |
| orourke | 2579 | geometric-primitives | 6 | 8 | 4.4 |
| pari | 1080 | high-precision-arithmetic | 9 | 2 | 9.0 |
| pgp | 44 | cryptography | 10 | 1 | 10.0 |
| phylip | 442 | steiner-tree | 7 | 1 | 7.0 |
| posit | 349 | satisfiability | 8 | 1 | 8.0 |
| qhull | 1315 | convex-hull | 10 | 4 | 7.0 |
| ranger | 3514 | kd-trees | 8 | 3 | 7.0 |
| reingold | 1305 | calendar-calculations | 10 | 1 | 10.0 |
| ruskey | 998 | generating-permutations | 8 | 4 | 7.2 |
| salowe | 466 | steiner-tree | 8 | 1 | 8.0 |
| sedgewick | 5339 | sorting | 5 | 11 | 3.2 |
| simpack | 1290 | priority-queues | 7 | 2 | 7.0 |
| skeleton | 833 | thinning | 9 | 2 | 7.0 |
| snns | 527 | shape-similarity | 7 | 1 | 7.0 |
| stony | 1312 | suffix-trees | 6 | 3 | 6.0 |
| syslo | 2550 | set-cover | 5 | 11 | 4.1 |
| toms | 2538 | bandwidth | 9 | 24 | 5.0 |
| triangle | 938 | triangulations | 9 | 1 | 9.0 |
| trick | 926 | vertex-coloring | 7 | 2 | 5.5 |
| tsp | 1386 | traveling-salesman | 8 | 1 | 8.0 |
| turn | 282 | shape-similarity | 6 | 1 | 6.0 |
| watson | 1355 | finite-state-minimization | 8 | 2 | 7.5 |
| wilf | 1126 | generating-partitions | 8 | 12 | 4.5 |
| xtango | 2945 | sorting | 6 | 19 | 3.2 |

Table 7: Hits by implementation, with associated ratings

| Count | Country | Code | Count | Country | Code |
|---|---|---|---|---|---|
| 6099 | Germany | de | 115 | Soviet Union | su |
| 3795 | United Kingdom | uk | 115 | Estonia | ee |
| 3062 | Canada | ca | 101 | Yugoslavia | yu |
| 2811 | France | fr | 83 | Latvia | lv |
| 2501 | Spain | es | 75 | Cyprus | cy |
| 2425 | Australia | au | 66 | China | cn |
| 1946 | Italy | it | 62 | United Arab Emirates | ae |
| 1668 | Japan | jp | 47 | Great Britain | gb |
| 1420 | Poland | pl | 46 | Kazakhstan | kz |
| 1267 | Korea | kr | 38 | Luxembourg | lu |
| 1265 | Israel | il | 32 | Philipines | ph |
| 1223 | Netherlands | nl | 32 | Honduras | hn |
| 1175 | Sweden | se | 24 | Malta | mt |
| 1134 | Finland | fi | 23 | Trinidad | tt |
| 1001 | Brazil | br | 23 | Costa | cr |
| 789 | Portugal | pt | 22 | Saudi Arabia | sa |
| 721 | Russia | ru | 20 | Macau | mo |
| 708 | Switzerland | ch | 19 | Vietnam | vn |
| 695 | Hong Kong | hk | 19 | Bolivia | bo |
| 678 | Belgium | be | 18 | Jamaica | jm |
| 652 | Singapore | sg | 15 | Egypt | eg |
| 615 | Solvenia | si | 14 | Peru | pe |
| 597 | Austria | at | 14 | Iceland | is |
| 583 | Norway | no | 13 | Jordan | jo |
| 557 | Ukraine | ua | 12 | Dominican Republic | do |
| 546 | Romania | ro | 12 | Bahrain | bh |
| 406 | Czech Republic | cz | 12 | Armenia | am |
| 403 | Taiwan | tw | 11 | Panama | pa |
| 396 | Greece | gr | 11 | Nicaragua | ni |
| 388 | Denmark | dk | 6 | Namibia | na |
| 388 | Chile | cl | 5 | Pakistan | pk |
| 350 | Malaysia | my | 5 | Georgia | ge |
| 339 | Uruguay | uy | 5 | Botswana | bw |
| 337 | Mexico | mx | 4 | Kuwait | kw |
| 334 | United States | us | 4 | Iran | ir |
| 329 | Colombia | co | 4 | Belize | bz |
| 311 | Hungary | hu | 3 | Mauritius | mu |
| 273 | Argentina | ar | 3 | Ethiopia | et |
| 259 | Venezuela | ve | 2 | Zimbabwe | zw |
| 256 | Ireland | ie | 2 | Oman | om |
| 230 | India | in | 2 | New Calidonia | nc |
| 201 | Thailand | th | 2 | Moldova | md |
| 178 | Indonesia | id | 2 | Ecuador | ec |
| 163 | Lithuania | lt | 2 | Dominica | dm |
| 163 | Croatia | hr | 1 | St. Helena | sh |
| 161 | New Zealand | nz | 1 | Qatar | qa |
| 132 | Slovakia | sk | 1 | Guatemala | gt |
| 127 | Bulgaria | bg | 1 | Greenland | gl |
| 122 | Turkey | tr | 1 | Byelorussian | by |
| 116 | South Africa | za | 1 | Barbados | bb |

Table 8: Hits by Nation