Proceedings WAE'98, Saarbrücken, Germany, August 20–22, 1998
Ed. Kurt Mehlhorn, pp. 204–212

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

Steven S. Skiena[1]
Department of Computer Science
SUNY Stony Brook, NY 11794-4400
skiena@cs.sunysb.edu

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 249,656 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 249,656 WWW hits recorded on the Stony Brook Algorithms Repository over a ten-week period, from February 10 to April 26, 1998. 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. 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.

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.

Problem Category	Index Hits	Subsection Hits	Problems
Data Structures	1067	2651	6
Numerical Problems	735	2271	11
Combinatorial Problems	539	2108	10
Graph Problems: Polynomial	842	3955	12
Graph Problems: Hard	729	2846	11
Computational Geometry	1247	5398	16
Set and String Problems	519	1898	9
Totals	5678	21127	75

Table 1: Hits by Major Section Index

2. The Stony Brook Algorithms Repository

The Stony Brook Algorithms Repository was developed in parallel with my book [6], *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, 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 almost quarter-million hits recorded on this site over the ten-week interval, 58289 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 10,000 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 41 times versus 2752 hits for the most frequently accessed page (the front page). [2]

[2] By contrast, the page advertising my book was hit 1364 times, although I have no way of knowing whether any of them actually ordered it.

Programming Language	Index Hits	Implementations
C language	805	37
C++	929	11
Fortran	125	6
Lisp	99	1
Mathematica	104	3
Pascal	272	5
Totals	2334	63

Table 2: Hits by Programming Language Index

Most Popular Problems	Hits	Least Popular Problems	Hits
shortest-path	681	shape-similarity	156
traveling-salesman	665	factoring-integers	141
minimum-spanning-tree	652	independent-set	137
kd-trees	611	cryptography	136
nearest-neighbor	609	maintaining-arrangements	134
triangulations	600	text-compression	133
voronoi-diagrams	578	generating-subsets	133
convex-hull	538	set-packing	126
graph-data-structures	519	planar-drawing	120
sorting	485	median	118
string-matching	467	satisfiability	116
dictionaries	459	bandwidth	107
geometric-primitives	452	shortest-common-superstring	105
topological-sorting	424	feedback-set	83
suffix-trees	423	determinants	78

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

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 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:

- Although shortest path was the most popular of the algorithmic problems over the ten week time period, a preliminary study done two weeks earler showed traveling-salesman comfortably in the top spot (560 hits to 513). I attribute the late surge of interest in shortest path to coincide with the end of the academic year in the United States, and students from algorithms courses seeking an edge. Minimum spanning tree showed a similar surge in this time interval.
- Of the six data structure problems, only priority queues (number 17) and set union-find (number 37) failed to make the top 15 cut.
- People seem twice as interested in generating permutations than subsets (258 hits to 133 hits), presumably reflecting the perceived difficulty of the task.
- Surprisingly popular problems include topological sorting (number 14), suffix trees (number 15), the knapsack problem (number 19). 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 set cover (number 59), planar drawing (number 69), and satisfiability (number 71). Such obviously commercial problems as cryptography (number 64) and text compression (number 66) proved unpopular presumably because better WWW resources exist for these problems.

It is interesting to note that only 2752 hits occured to the front-page of the site, which means that most users 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. [3]

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 [2] received almost as many hits (2084) as the two following implementations, both associated with popular books [4] (1258) and [1] (994). The fourth most popular implementation was (surprisingly) Ranger [3] (846), 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. 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 significantly lower.

[3] Any software developer who is dissatisfied with their ratings will perhaps be gratified to learn that my own *Combinatorica* [5] received the fourth lowest average score among the 56 rated implementations. Anybody who cannot better appreciate the merits of *Combinatorica* is clearly unfit to judge other people's software.

Most Needed Implementations	Rank by Mass	Hits	Δ	Least Needed Implementations	Rank by Mass	Hits	Δ
suffix-trees	57	15	-42	high-precision-arithmetic	16	29	13
bin-packing	74	34	-40	priority-queues	4	17	13
knapsack	59	19	-40	edge-coloring	43	57	14
kd-trees	42	4	-38	drawing-trees	37	53	16
eulerian-cycle	66	31	-35	maintaining-arrangements	47	64	17
polygon-partitioning	65	32	-33	matching	1	18	17
nearest-neighbor	36	5	-31	unconstrained-optimization	40	58	18
minkowski-sum	71	42	-29	satisfiability	50	70	20
simplifying-polygons	68	44	-24	dfs-bfs	9	30	21
motion-planning	73	55	-18	network-flow	2	23	21
traveling-salesman	19	2	-17	random-numbers	18	40	22
scheduling	64	48	-16	bandwidth	48	71	23
set-data-structures	53	37	-16	matrix-multiplication	25	49	24
edge-vertex-connectivity	60	45	-15	planar-drawing	41	68	27
thinning	62	47	-15	cryptography	33	63	30
graph-partition	52	39	-13	generating-graphs	12	46	34
string-matching	24	11	-13	fourier-transform	13	50	37
hamiltonian-cycle	32	21	-11	generating-subsets	28	66	38
set-cover	70	59	-11	generating-partitions	21	60	39
approximate-pattern-matching	34	24	-10	determinants	30	74	44

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	15310	8421	266	341	5193	183	9149	19426	58289

Table 5: Hits by top level domain

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. Suffix trees (rank 1) and kd-trees (rank 4) are the most needed data structure implementations, while the closely related problems of bin packing (rank 2) and knapsack (rank 3) 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 5), traveling salesman (rank 11), and Hamiltonian cycle (rank 18). On the other hand, traditional algorithm engineering topics like matching (rank 61) and network flow (rank 65) 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 (15310) and .net (5193) domains together account for more than twice the number of .edu (8421) hits. Roughly one third of all hits came from users outside the United States, presumably similarly split between educational and industrial users.

It is interesting and amusing to see the distribution of hits by country code. No less than 84

nations visited the Algorithm Repository during this ten week 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 (2340). The United Kingdom (1677), France (1445), and Spain (1054) each accounted for significantly more hits than Israel (315), Japan (709), and the Netherlands (590) – suggesting that the interest does not completely correlate with my perception of the amount of algorithmic research activity in these nations. Two of the largest producers of graduate students in computer science, China (39) and India (89), ranked surprisingly low in the number of hits despite the presence of substantial software industries. Presumably this reflects limited WWW access within these countries.

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.

References

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

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

[3] 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.

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

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

[6] 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	298	3	6.3	agrep	10
bandwidth	107	2	7.5	toms	9
bin-packing	258	1	3.0	xtango	3
calendar-calculations	188	1	10.0	reingold	10
clique	248	3	5.3	dimacs	9
convex-hull	538	7	5.4	qhull	10
cryptography	136	3	5.3	pgp	10
determinants	78	4	4.5	linpack	8
dfs-bfs	268	7	3.9	LEDA	8
dictionaries	459	4	6.0	LEDA	10
drawing-graphs	275	3	7.0	graphed	9
drawing-trees	189	3	7.0	graphed	9
edge-coloring	169	4	4.5	stony	6
edge-vertex-connectivity	214	3	4.0	combinatorica	4
eulerian-cycle	262	2	3.0	combinatorica	3
feedback-set	83	1	4.0	graphbase	4
finite-state-minimization	290	4	5.7	grail	9
fourier-transform	197	5	5.0	fftpack	10
generating-graphs	208	5	6.8	graphbase	10
generating-partitions	163	5	6.6	wilf	8
generating-permutations	258	4	7.0	ruskey	8
generating-subsets	133	4	6.3	wilf	8
geometric-primitives	452	5	5.4	LEDA	8
graph-data-structures	519	6	6.7	LEDA	10
graph-isomorphism	251	2	6.5	nauty	10
graph-partition	232	2	6.0	link	8
hamiltonian-cycle	320	5	4.2	toms	6
high-precision-arithmetic	274	5	5.6	pari	9
independent-set	137	2	6.0	dimacs	7
intersection-detection	410	5	5.2	LEDA	7
kd-trees	611	3	4.0	ranger	8
knapsack	341	2	5.0	toms	6
linear-equations	218	3	6.6	lapack	10
linear-programming	317	5	4.4	lpsolve	9
longest-common-substring	175	2	5.0	cap	8
maintaining-arrangements	134	2	8.0	arrange	9
matching	363	10	5.2	goldberg	9
matrix-multiplication	200	5	5.0	linpack	7
median	118	2	5.0	handbook	6
minimum-spanning-tree	652	9	4.0	LEDA	6
minkowski-sum	219	1	4.0	eppstein	4
motion-planning	181	1	3.0	orourke	3
nearest-neighbor	609	4	5.2	ranger	7
network-flow	312	8	5.0	goldberg	10
planar-drawing	120	3	5.7	graphed	8
point-location	293	4	4.8	LEDA	7
polygon-partitioning	259	1	8.0	geompack	8
priority-queues	398	8	4.5	LEDA	9
random-numbers	230	6	4.8	simpack	7
range-search	289	4	4.8	LEDA	8
satisfiability	116	2	8.0	posit	8
scheduling	204	2	4.0	syslo	4
searching	235	3	5.6	handbook	7
set-cover	168	1	5.0	syslo	5
set-data-structures	241	3	4.3	LEDA	5
set-packing	126	1	5.0	syslo	5
shape-similarity	156	2	6.5	snns	7
shortest-common-superstring	105	1	8.0	cap	8
shortest-path	681	7	5.0	goldberg	9
simplifying-polygons	216	1	5.0	skeleton	5
sorting	485	7	4.9	moret	7
steiner-tree	190	2	7.5	salowe	8
string-matching	467	5	4.4	watson	7
suffix-trees	423	2	4.0	stony	6
text-compression	133	1	5.0	toms	5
thinning	206	1	9.0	skeleton	9
topological-sorting	424	6	3.5	LEDA	7
transitive-closure	195	2	4.0	LEDA	6
traveling-salesman	665	5	5.0	tsp	8
triangulations	600	8	5.9	triangle	9
unconstrained-optimization	168	3	6.3	toms	8
vertex-coloring	328	8	5.0	dimacs	7
vertex-cover	223	3	5.0	clique	6
voronoi-diagrams	578	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	132	unconstrained-optimization	6	1	6.0
LEDA	2084	graph-data-structures	10	30	6.2
agrep	223	approximate-pattern-matching	10	1	10.0
arrange	87	maintaining-arrangements	9	3	7.3
bipm	127	matching	8	1	8.0
cap	119	shortest-common-superstring	8	2	8.0
clarkson	123	convex-hull	6	1	6.0
clique	154	clique	6	6	5.5
combinatorica	603	generating-graphs	8	28	4.0
culberson	133	vertex-coloring	6	2	5.0
dimacs	288	matching	9	10	5.6
eppstein	373	minkowski-sum	4	2	4.0
fftpack	107	fourier-transform	10	1	10.0
fortune	368	voronoi-diagrams	9	2	8.0
genocop	113	unconstrained-optimization	5	1	5.0
geolab	146	geometric-primitives	5	1	5.0
geompack	187	polygon-partitioning	8	2	8.0
goldberg	446	network-flow	10	3	9.3
grail	282	finite-state-minimization	9	1	9.0
graphbase	569	generating-graphs	10	17	4.4
graphed	398	drawing-graphs	9	7	6.4
handbook	994	dictionaries	8	12	4.9
htdig	107	text-compression	7	1	7.0
lapack	120	linear-equations	10	1	10.0
link	144	graph-partition	8	4	4.5
linpack	68	determinants	8	2	7.5
linprog	76	linear-programming	4	1	4.0
lpsolve	354	linear-programming	9	1	9.0
math	105	matrix-multiplication	6	1	6.0
moret	466	sorting	7	17	3.8
nauty	243	graph-isomorphism	10	1	10.0
north	209	drawing-graphs	7	2	7.0
orourke	598	geometric-primitives	6	8	4.4
pari	281	high-precision-arithmetic	9	2	9.0
pgp	44	cryptography	10	1	10.0
phylip	89	steiner-tree	7	1	7.0
posit	55	satisfiability	8	1	8.0
qhull	268	convex-hull	10	4	7.0
ranger	846	kd-trees	8	3	7.0
reingold	225	calendar-calculations	10	1	10.0
ruskey	206	generating-permutations	8	4	7.2
salowe	98	steiner-tree	8	1	8.0
sedgewick	1258	sorting	5	11	3.2
simpack	286	priority-queues	7	2	7.0
skeleton	187	thinning	9	2	7.0
snns	189	shape-similarity	7	1	7.0
stony	272	suffix-trees	6	3	6.0
syslo	506	set-cover	5	11	4.1
toms	546	bandwidth	9	24	5.0
triangle	232	triangulations	9	1	9.0
trick	193	vertex-coloring	7	2	5.5
tsp	331	traveling-salesman	8	1	8.0
turn	55	shape-similarity	6	1	6.0
watson	322	finite-state-minimization	8	2	7.5
wilf	259	generating-partitions	8	12	4.5
xtango	644	sorting	6	19	3.2

Table 7: Hits by implementation, with associated ratings

Count	Country	Code	Count	Country	Code
2340	Germany	de	50	Thailand	th
1692	Canada	ca	49	Venezuela	ve
1677	United Kingdom	uk	48	Soviet Union	su
1445	France	fr	48	Cyprus	cy
1054	Spain	es	47	Slovakia	sk
932	Australia	au	46	Yugoslavia	yu
842	Italy	it	46	Ukraine	ua
709	Japan	jp	44	Indonesia	id
591	Korea	kr	42	Kazakhstan	kz
590	Netherlands	nl	41	Turkey	tr
528	Sweden	se	39	China	cn
463	Finland	fi	32	Honduras	hn
423	Brazil	br	26	South Africa	za
417	Hong Kong	hk	23	Romania	ro
346	Norway	no	23	Philippines	ph
345	Belgium	be	21	Malta	mt
317	Switzerland	ch	20	Great Britain	gb
315	Israel	il	17	Lithuania	lt
302	Austria	at	17	Costa Rica	cr
294	Portugal	pt	12	Egypt	eg
282	Poland	pl	12	Dominican Republic	do
267	Slovenia	si	12	Armenia	am
238	Russia	ru	11	Trinidad and Tobago	tt
218	Singapore	sg	11	Jordan	jo
170	Denmark	dk	10	Iceland	is
162	Greece	gr	9	Uruguay	uy
159	Czech Republic	cz	9	Bahrain	bh
148	Chile	cl	7	Peru	pe
143	Mexico	mx	6	United Arab Emirates	ae
137	Taiwan	tw	5	Georgia	ge
125	Argentina	ar	4	Belize	bz
115	United States	us	3	Saudi Arabia	sa
115	Malaysia	my	3	Bolivia	bo
113	Hungary	hu	2	New Caledonia	nc
97	Columbia	co	2	Mauritius	mu
89	India	in	2	Moldova	md
85	Ireland	ie	2	Ecuador	ec
80	Croatia	hr	1	Pakistan	pk
77	Estonia	ee	1	Namibia	na
60	New Zealand	nz	1	Luxembourg	lu
58	Latvia	lv	1	Kuwait	kw
52	Bulgaria	bg	1	Barbados	bb

Table 8: Hits by Nation