Simulated Annealing for Profile and Fill Reduction of Sparse Matrices

Robert R. Lewis

University of British Columbia

Department of Computer Science

6356 Agricultural Road

Vancouver, BC V6T 1W5

CANADA

Internet: bobl@cs.ubc.ca

24 March, 1993

Revised: 22 July, 1993

Summary

Simulated annealing can minimize both profile and fill of sparse matrices. We applied these techniques to a number of sparse matrices from the Harwell-Boeing Sparse Matrix Collection. We were able to reduce profile typically to about 80% of that attained by conventional profile minimization techniques (and sometimes much lower), but fill reduction was less successful (85% at best). We present a new algorithm that significantly speeds up profile computation during the annealing process. Simulated annealing is, however, still much more time-consuming than conventional techniques and is therefore likely to be useful only in situations where the same sparse matrix is being used repeatedly.

1 Introduction

Problems that require the solution of systems of linear equations of the form

$$\mathbf{Ax} = \mathbf{b} \tag{1}$$

where $\mathbf{A}$ is $N \times N$, $|\mathbf{x}| = |\mathbf{b}| = N$ occur frequently in many areas, often with a large value of N. Directly solving linear systems like (1) is, in general, an $O(N^3)$ process. Iterative methods can usually improve on this and, when they converge, are often preferred.

If $\mathbf{A}$ is sparse, however, iterative methods have less of an edge: direct solutions can often do much better than $O(N^3)$. The principal drawback when solving sparse systems directly is the "fill" created during the decomposition procedure: elements of $\mathbf{A}$ which were initially zero must become non-zero and therefore enter into the later stages of the decomposition.

The success of direct methods hinges on techniques to reduce fill by solving the equivalent problem

$$\mathbf{A}'\mathbf{x}' = \mathbf{b}'$$

where $\mathbf{x}' = \mathbf{Px}$, $\mathbf{A}' = \mathbf{PAP}^T$, $\mathbf{b}' = \mathbf{Pb}$, and $\mathbf{P}$ is a permutation of the rows and columns of the identity matrix $\mathbf{I}$ such that $\mathbf{PP}^T = \mathbf{I}$. These techniques select a $\mathbf{P}$ that makes $\mathbf{A}'$ better (according to some criterion) than $\mathbf{A}$. This amounts to a reordering of $\mathbf{A}$ into $\mathbf{A}'$.

Finding an optimal $\mathbf{A}'$ is an NP-hard problem: there are $N!$ possible reorderings of $\mathbf{A}$ and no way to find the optimal one short of trying them all. The usual approach is to follow some heuristic procedure in reordering $\mathbf{A}$ such as: Reverse Cuthill-McKee (RCM), Minimum Degree Algorithm (MDA), Gibbs-King (GK), Nested Dissection (ND), etc.

All of these methods are capable of producing generally satisfactory results, but they all have their shortcomings. They might be dependent on the correct choice of a starting node (RCM) or an initial grouping

(ND). In any case, none of them makes any definite claims about minimizing fill.

What we'll explore in this paper is the application of a multivariate optimization technique called "simulated annealing" that can in principle be applied to any minimizant or maximizant in NP-hard problems. In particular, we'll investigate whether it can be practically applied to the minimizants of profile and fill to produce better orderings of $\mathbf{A}$. Previous work in [1] looked at the minimizants of profile, wavefront, and bandwidth. We will compare our results with this work where they overlap.

2 Simulated Annealing

(Most of this section follows the presentation in [11], although [8] presented the original idea.)

The statistical behaviour of physical systems with large numbers of degrees of freedom inspired the simulated annealing technique.

To "anneal" is "to heat (glass, metals, etc.) and then cool slowly to prevent brittleness". This is just one instance of a fundamental observation about nature: given sufficient time and a mechanism to do so, a system will always tend to adjust itself to a minimal energy state. For instance

- The surface of a lake is flat.
- (Slowly) cooled liquids form highly-regular crystals.
- Air molecules spread evenly in a room.

All of these represent systems with large numbers of degrees of freedom achieving global minima.

Most iterative techniques that attempt to solve global optimization problems are, in a sense, "greedy": as soon as they find a better solution, they adopt it. For this reason, these techniques don't always behave well in the presence of local minima.

How, then, are molecules able to "solve" the minimal energy problem globally? Because nature gives them sufficient time and energy to rearrange themselves in a way that ultimately satisfies the global minimum.

The Boltzmann distribution permits a system to exist in a state that is energy E *above* its minimum energy state with probability

$$P(E) \sim e^{\frac{-E}{k_B T}} \tag{2}$$

where T is the temperature and k_B is Boltzmann's constant. This means that for all $T > 0$ there's always a chance for a system stuck in a local minimum to acquire enough energy to move out of that minimum, but that as $T \rightarrow 0$, that probability becomes vanishingly small.

"Simulated" annealing (which we'll henceforth refer to as "SA") is then a computer simulation of physical annealing. Because it *is* a simulation, the minimizant can be any quantity we choose, not just energy.

The idea of sampling a simulation of a physical system which obeys a Boltzmann distribution originated in [10]. The development of SA as an optimization technique is due to [8].

In order to perform SA, we need (at least):

- an initial configuration for the system
- a set of "moves" (also known as "options"): distinct ways to randomly change the configuration. Each move should be reversible and it should be possible to reach any part of the configuration space in a finite number of moves.
- a minimizant ("E"): what we're trying to minimize. It (or, more commonly, its increment) needs to be evaluated after the application of any move.

- a control parameter ("T"): an indication of how willing we are at any given point to tolerate *increases* in the minimizant.
- an annealing schedule: analogous to the "cooling rate", this says how T decreases as the simulation proceeds. The schedule also specifies how many attempts we perform at each temperature.

Given these things, the SA algorithm (in only slightly simplified form) proceeds as follows:

$C \leftarrow$ initial configuration

$T \leftarrow$ initial temperature

repeat

    repeat a specified number of times (according to annealing schedule)

        $C' \leftarrow$ apply randomly-chosen move to C

        $\delta E \leftarrow E(C') - E(C)$

        if $\delta E < 0$ or $e^{-\frac{\delta E}{T}} > \text{random}(0,1)$

            $C \leftarrow C'$

    $T \leftarrow$ lower T according to annealing schedule

until we can't find a lower $E(C')$

The idea is that we always change configuration if we move to a lower energy state, but that we *sometimes* change configuration even if we move to a higher state, although as the temperature drops this is less and less likely to happen.

The main drawbacks of SA are:

- Proofs that it converges to an optimal solution are hard to come by.

- There are only very rough guidelines in the appropriate choice of the initial T and the annealing schedule. Often, they are chosen by trial-and-error.
- The algorithm needs to assess $E(C')$, the "energy" of the changed configuration, efficiently, since it is frequently needed.

3 Simulated Annealing and Sparse Matrices

How then do we use SA in sparse matrix problems? We can apply the algorithm given above if we can provide its required inputs.

In all further discussion, we will restrict ourselves to the case of symmetric matrices. This is purely for our own convenience: there is no reason why SA shouldn't work as well with non-symmetric matrices as it does with symmetric ones.

We have implemented a system in C on a Sun SPARCStation-2 that is capable of performing SA profile or fill reduction on an arbitrary symmetric matrix. Hereafter, we'll refer to the two procedures as SAPR (Simulated Annealing Profile Reduction) and SAFR (Simulated Annealing Fill Reduction).

Here follow the design choices affecting SA computation.

3.1 Initial Configuration

This is just a representation of the sparse matrix itself. In our case, allocating an array of nodes, each with a dynamically-sized array of adjacencies is advantageous. This takes up slightly more space than the usual double-array storage scheme (described in [6]), but its dynamic nature allows us to add adjacencies as they occur during fill.

3.2 Moves

One move that immediately suggests itself is to exchange two randomly-chosen rows (and the corresponding columns, since we're dealing with a symmetric matrix). We'll call this a "swap".

We'll discuss the impact of swaps on evaluations of the minimizant when we discuss the individual minimizants later.

3.3 Minimizant

We can choose this as needed. For this investigation, we'll take it to be either profile or fill. We'll talk about the practical limitations of this below. We could also have chosen bandwidth, as was done by [1], but bandwidth is a less reliable indicator of fill than profile.

3.4 Control Parameter and Annealing Schedule

The easiest way to think of "temperature" in this context is to recall that in analogy with (2), a configuration change that increases the minimizant by an amount equal to T will be accepted 63% $(= 1 - e^{-1})$ of the time. (We take $k_B = 1$ in this analogy.)

We want the system to be initially "hot": increases in the minimizant are *almost* as likely to be accepted as decreases. This is a tradeoff we make. If we start out with a T that is too low, we may miss global minima. On the other hand, if we start out with a T that is too high, we waste computing time randomly changing the system configuration. In most initial SA investigations, it's prudent to overestimate T until the researcher acquires some familiarity with the problem. This will be our approach:

$$T_{initial}(\mathbf{A}) = 10 \frac{\{\text{initial profile of } \mathbf{A}\}}{N}$$

This heuristic formula makes T large when we deal with a matrix that has a large profile, but the N^{-1} factor prevents it from getting large as the size of the matrix itself increases.

We'll also follow a common (naive) approach to an annealing schedule. We are given two dimensionless parameters: F_T and A_T. At any given temperature, we try a maximum of NA_T swaps. If no successful reductions of the minimizant have occurred after this, we assume we can't find a lower $E(C')$ and stop. Otherwise, we lower the temperature by a factor of F_T and iterate. In addition, if at any temperature we ever have $N(1 + 0.1A_T)$ successes, we take this to indicate that the temperature is too high and lower the temperature early. Typically, $F_T = 0.9$ and $A_T = 100$, but we'll consider others.

4 SAPR: Simulated Annealing Profile Reduction

It would be possible to compute the profile of the matrix *a priori* after each swap, but this is inefficient, being $O(N - \min(i, j)) = O(N)$ for the storage scheme being used. Instead, we can note that swapping row i with row $j : j > i$ affects only the areas of the matrix shown in gray in Figure 1. Arrows indicate which elements are being effectively exchanged by the swap.

Only individual row profiles of rows whose indices are greater than or equal to i can be affected. Also, the gray areas are themselves sparse (in general) so we can efficiently compute changes to the individual row profiles by traversing the adjacency arrays of the affected nodes of the matrix.

The overall change in profile as a result of a swap can therefore be computed in $O(B_h)$ (integer operations) where B_h is the (mean) half-bandwidth of the matrix.

ordering	profile
Original	32
Reverse Cuthill-McKee	18
SAPR	17

Table 1: Results of Profile Minimization on Figure 2

4.1 A Simple Example

We first present a small example: a sparse matrix whose graph is shown in Figure 2. (This was originally used as an example in [6].)

Figures 3 and 4 show the sparsity patterns before and after RCM-ordering of this matrix. (All examples of RCM in this paper start from a pseudoperipheral node using the algorithm given in [6].) In this and all of the other sparsity diagrams we show, the original elements of the matrix are in black and those that would be created by fill are in gray.

Figure 5 and Table 1 show the results of SAPR on this matrix. The difference between SAPR and RCM is quite small (= 1) in this case, but at least it demonstrates that SAPR can find a better solution.

4.2 A 2-D Grid

Figures 6 and 7 show a more realistic example: a 100×100 array corresponding to a 2-dimensional, 5-point PDE problem on a 10×10 grid.

Figure 8 and Table 2 show the results of SAPR on this matrix. It's important to note the times involved: on a Sun SPARCstation-2 RCM required less than a second to run, while SAPR took about 15 minutes.

ordering	profile
Original	909
Reverse Cuthill-McKee	829
SA Profile Reduction	812

Table 2: Results of Profile Minimization of 100×100 2-D 5-point PDE solution matrix

This suggests that any practical application of SAPR seeking to minimize the amount of CPU time used should take into account both profile reduction and solution time. Of course, if the same matrix is being used many times, the SAPR cost can be amortized and the (presumed) reduction in solution time for each use may more easily make up the difference.

f_{SAPR} is the improvement ratio of SAPR compared to the best non-SA-obtained profile reported in our sources. If we examine f_{SAPR} for a wide range of N, we get the results shown in Figure 9: as N increases, the solution gets (almost) monotonically worse, but not dramatically so.

4.3 The Harwell-Boeing Sparse Matrix Collection

To examine how SAPR does on a wider range of matrices, we will apply it to a subset of matrices from the Harwell-Boeing Sparse Matrix collection (see [4]).

Tables 6–10 show the results of SAPR on a subset of the matrices in the Harwell-Boeing test set (description in [4]). Table 11 is the key to these tables.

Taken overall, there's a wide variation of f_{SAPR}. There's an improvement in the majority of cases (71 out of 95), but in some cases (e.g. HB#13/NOS2) there's no improvement at all! How can we account for this?

Figure 10 plots f_{SAPR} as a function of N for all the matrices in the subset. There is no obvious correlation between the two: SAPR appears to be as likely to be successful with a small matrix as with a large one. The near-monotonic increase in f_{SAPR} we observed for a single class of problem in Figure 9 is not evident here.

Perhaps some other attribute correlates better with f_{SAPR}. Figure 11 plots f_{SAPR} against the original source of the matrix, as reported in [4]. The abscissa is arranged in increasing order of mean f_{SAPR} for all matrices within the given source. Figure 11 also shows the discipline the matrices in each source came from. Performance for structural engineering and finite element matrices appears widespread. The single 9-point 30×30 PDE appears close to $f_{SAPR} = 1$, as we might have expected from the 5-point PDE results shown above in Figure 9. All the electric power matrices have low f_{SAPR}'s.

Nevertheless, even within a given discipline, there can be a wide spread of f_{SAPR}. Could this be the result of the annealing schedule?

4.4 Altering the Annealing Schedule

There's no guarantee that the annealing schedule we've used for all our work so far is optimal. Let's choose a subset of the matrices we've been using and see what happens when we change the schedule.

The subset we'll choose correspond to Harwell-Boeing Tape File #7 (hereafter "HBTF#7"). We have several reasons for selecting these:

- They arise from a common problem domain: finite element modelling.
- They show a wide range in f_{SAPR}, from 0.576 to 1.541.
- They correspond to the same matrices used in [1], so we can compare our results and timings.

We'll perform SAPR on all 30 matrices in HBTF#7 with the five different annealing schedules S1-5 described in Table 3. S1 is identical to the schedule used for Tables 6–10. The other schedules vary F_T and A_T.

Schedule Name	F_T	A_T	RCMP used?
S1	0.90	100	no
S2	0.95	50	no
S3	0.95	100	no
S4	0.90	50	no
S5	0.90	100	yes

Table 3: SAPR Schedule Specifications

S5 is special. Before starting SA, the matrix is preordered with the RCM algorithm. We refer to this as "RCM Preconditioning" (hereafter, "RCMP").

We do not propose this as an improvement to SAPR: since we're supposed to start with a "hot" matrix, its initial configuration should not matter. What we *can* do, though, is use it as an indicator of how sensitive SAPR is to the annealing schedule. If two values of f_{SAPR} with and without RCMP differ greatly, we take it to be an indication that our schedule is sub-optimal.

Tables 12–13 show the results[1]. Table 14 is the key to these tables. It is interesting to note that while no single schedule outperforms the others, the most rapidly-cooled, least iterated one, S4, is with one exception always worse than the rest.

4.5 Comparison With Previous Work

As mentioned above, we can compare our results for HBTF#7 not only with those of the minimal non-SA profiles, but also with the previous SAPR work done by Armstrong in [1]. Tables 15–16 show this. Table 17

[1]The timings for Tables 6–10 and those for S1 in Tables 12–13 differ as the former were taken from an earlier, less efficient version of the software with a different compiler and operating system on a different SPARCstation-2.

is the key to these tables.

Varying annealing schedules and using minimal profiles has improved our results. Now we have uniformly lower profiles than non-SA methods, except in the one small case (DWT 66) for which we suggest that both GK and SAPR find the minimal profile.

In all but one case (DWT 2680 – the largest matrix in HBTF#7), we also find lower profiles than Armstrong. This is something of a surprise, as he chose a somewhat more elaborate annealing schedule with a parameterization that varied matrix-by-matrix.

Nevertheless, it is safe to conclude that we have found no clear heuristic on choosing an annealing schedule for profile redution. One must simply try a number of points in the schedule parameter space. (This conclusion is often reached in SA problems.)

Armstrong also cites timing results: The total CPU time required for SAPR profiles of all 30 matrices was 600 hours on a DEC 20/60 mainframe. The total CPU time for our SAPR results for the same 30 matrices in Tables 15–16 was 10.1 hours on a SPARCStation 4/690 server. Certainly, a large part of the difference is attributable to the difference in hardware technology, but it is likely that our profile increment computation speedup also contributed.

5 SAFR: Simulated Annealing Fill Reduction

It is conceptually easy in SA to switch minimizants from profile to fill. Problems arise, however, in computing the change in fill as a result of a swap, which SA requires frequently.

Since a single swap of rows i and j can introduce a large amount of fill that cascades into all rows $k > \min(i, j)$, fill increment computation is much more expensive than profile increment computation. It would

ordering	fill
Original	15
Reverse Cuthill-McKee	3
SAFR	2

Table 4: Results of Fill Minimization on Figure 2

be extremely useful to try to find a cheaper way of computing the fill increment.

We'll investigate fill vs. profile increment computation in greater detail below.

5.1 A Simple Example

We can perform SAFR on the same simple example shown in Figure 2. Figure 12 and Table 4 show the results.

Again, an insignificant improvement (= 1) in fill over RCM, but, again, this is just an illustration of the feasibility of the technique.

5.2 A 2-D Grid

We again use the more realistic example of Figure 6. Figure 8 and Table 5 show the fill minimization results. The table also includes the profile minimization results by way of illustrating something suggested by a comparison of Figures 8 and 13: orderings which minimize fill can be quite distinct from those which minimize profile.

ordering	profile	fill
Original	909	720
Reverse Cuthill-McKee	829	640
SAPR	812	623
SAFR	3123	400

Table 5: Results of Minimization of 100×100 2-D 5-point PDE solution matrix

5.3 Time Requirements of Profile vs. Fill Computation

The 2-D, 5-point grid is a scaleable problem that allows us to see how fill and profile increment computations scale with the size of the problem. Figure 14 shows this for a wide range of N. Note that while both curves are steeper than $O(N)$, the fill curve is much steeper than the profile curve. For this reason, we can only do fill minimization on small matrices.

5.4 The Harwell-Boeing Sparse Matrix Collection

Table 18 shows the results of SAFR on a small subset of the matrices in the Harwell-Boeing test set. Most of them were chosen from those members of the subset we used for SAPR which had $N < 100$. Two additional matrices, 685 BUS and 1138 BUS, were chosen to include results from the tie-breaking minimum degree results presented in by Cavers in [2]. Table 19 is the key to these tables.

All of the matrices underwent SAFR with default schedule S1 except for the two large matrices, which used RCMP combined with the initial assumption of $T = 1$ (a very cool system) to get these numbers in a reasonable length of time[2].

[2] If we consider over 5 CPU-hours to be "reasonable"!

SAFR was able to improve over the minimum degree algorithm (or Cavers' enhancement of it) in 8 out of 15 cases, and only did significantly worse in the one largest case.

The times to produce conventional fill results were never more than a few seconds.

6 Conclusions and Further Work

SA has potentially wide application in direct sparse matrix solution. Although SAPR takes considerably longer than RCM or GK (and probably all the others), it can produce a significant reduction in profile for a wide spectrum of matrix types. SAFR, on the other hand, takes a considerably longer time than either ND or MDA, and reduces fill less dramatically than SAPR reduces profile.

SA is particularly applicable in two areas:

- *situations where the same matrix* **A** *is used repeatedly*

  Then, the additional cost of SA would be amortized over a large number of solutions.

- *when validating other, faster algorithms for profile and fill reduction*

  (This was pointed out in [1].) Given the optimal annealing schedule (which we do not claim to have used here), it should be possible to determine what the actual minimum value of the minimizant is.

Possibilities for additional work include:

- More work needs to be done on heuristics for choosing an appropriate annealing schedule.
- There may be some way to improve fill increment computation so that SAFR will be practical on large matrices.

- SA should lend itself well to nested dissection, as it has been used in analogous applications for reducing interconnectivity in VLSI layout (see [8] and [12]).

7 Acknowledgements

The author would like to thank Dr. Joseph Liu for his helpful comments on an earlier version of this paper and Dr. James Varah for his help and encouragement. Special thanks go to Ian Cavers for his advice and help in accessing the Harwell-Boeing Sparse Matrix Collection.

References

[1] B. Armstrong, 'Near-Minimal Matrix Profiles and Wavfronts for Testing Nodal Resequencing Algorithms', *Int. j. numer. methods eng.*, **21**, 1785-1790 (1985).

[2] I. A. Cavers, *Using Deficiency Measure for Tiebreaking the Minimum Degree Algorithm*, Techical Report 89–2, Department of Computer Science, University of British Columbia, 1989.

[3] I. A. Cavers, *work in progress.*

[4] I. S. Duff, R. G. Grimes, J. G. Lewis, and B. Poole, 'User's Guide for the Harwell-Boeing Sparse Matrix Collection', *ACM SIGNUM Newsletter*, **17** (1982), p. 22.

[5] G. C. Everstine, 'A Comparison of Three Resequencing Algorithms for the Reduction of Matrix Profile and Wavefront', *Int. j. numer. methods eng.*, **14**, 837-853 (1979).

[6] J. A. George and Joseph W-H. Liu, *Computer Solution of Large Sparse Positive Definite Systems*, Prentice-Hall, New Jersey, 1981.

[7] J. A. George and Joseph W-H. Liu, *User Guide for SPARSPAK: Waterloo Sparse Linear Equations Package*, Research Report CS-78-30, Department of Computer Science, University of Waterloo, 1978.

[8] S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, 'Optimization by Simulated Annealing', *Science*, **220** (1983), pps. 671-680.

[9] J. G. Lewis, 'Implementation of Gibbs-Poole-Stockmeyer and Gibbs-King Algorithms', *ACM Trans. on Math. Softw.*, **8** (2) (1982), pps. 180-189.

[10] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, 'Equation of State Calculations by Fast Computing Machines', *J. Chem. Phys.*, **21** (1953), pps. 1087-1092.

[11] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes*, Cambridge University Press, New York, 1988.

[12] M. P. Vecchi and S. Kirkpatrick, 'Global Wiring by Simulated Annealing', *IEEE Trans. on CAD*, **CAD-2** (1983), pps. 215-222.

H-B #	Matrix Name	N	# of Matrix Nonzeros	Conventional Profiles RCM	Conventional Profiles GK	Simulated Annealing Profile	Simulated Annealing Time	f_{SAPR}
3	ASH292	292	1250	3738	3479	2694	457	0.774
	ASH85	85	304	617	564	491	91	0.871
4	BCSPWR01	39	85	99	100	83	19	0.838
	BCSPWR02	49	108	214	180	113	24	0.628
	BCSPWR03	118	297	774	644	478	87	0.742
	BCSPWR04	274	943	4331	3609	2224	316	0.616
	BCSPWR05	443	1033	10784	7305	3015	420	0.413
	BCSPWR06	1454	3377	63182	48265	17570	1940	0.364
	BCSPWR07	1612	3718	74344	56119	21147	2400*	0.377
	BCSPWR08	1624	3837	78187	57242	24741	2490*	0.432
	BCSPWR09	1723	4117	79260	64788	26793	2300*	0.414
5	BCSSTK01	48	224	667	544	468	44	0.860
	BCSSTK02	66	2211	2145	2145	2145	3	1.000
	BCSSTK03	112	376	272	272	306	101	1.125
	BCSSTK04	132	1890	3965	3228	3162	1060	0.980
	BCSSTK05	153	1288	2254	2226	2194	464	0.986
	BCSSTK06	420	4140	13226	13723	12975	2980	0.981
	BCSSTK07	420	4140	13226	13723	12975	2980	0.981
	BCSSTK09	1083	9760	75581	66281	57194	9250	0.863
	BCSSTK10	1086	11578	19943	19685	28488	13300	1.447
	BCSSTK11	1473	17857	73117	66919	62103	20400	0.928

Table 6: Results of Profile Reduction of Harwell-Boeing Matrices

H-B #	Matrix Name	N	# of Matrix Nonzeros	Conventional Profiles RCM	GK	Simulated Annealing Profile	Time	f_{SAPR}
5	BCSSTK12	1473	17857	73117	66919	62103	20400	0.928
	BCSSTM10	1086	11589	19790	19685	28424	12300	1.444
6	CAN 24	24	92	97	98	95	16	0.979
	CAN 61	61	309	408	367	340	85	0.926
	CAN 62	62	140	269	254	178	32	0.701
	CAN 73	73	225	774	699	524	52	0.750
	CAN 96	96	432	1234	1114	1078	123	0.968
	CAN 144	144	720	1074	979	972	227	0.993
	CAN 161	161	769	2610	2610	2473	238	0.948
	CAN 187	187	839	2374	2231	2163	294	0.970
	CAN 229	229	1003	4475	4423	3890	326	0.879
	CAN 256	256	1586	7977	6845	4166	650	0.609
	CAN 268	268	1675	7732	9665	4637	749	0.600
	CAN 292	292	1416	9275	8014	4057	545	0.506
	CAN 445	445	2127	20234	18786	14448	948	0.769
	CAN 634	634	3931	35131	32155	26111	2470	0.812
	CAN 715	715	3690	38449	35451	20722	2390	0.585
7	DWT 59	59	163	255	255	214	35	0.839
	DWT 66	66	193	151	127	127	47	1.000
	DWT 72	72	147	297	255	147	33	0.576
	DWT 87	87	314	609	595	428	84	0.719

Table 7: Results of Profile Reduction of Harwell-Boeing Matrices (continued)

H-B #	Matrix Name	N	# of Matrix Nonzeros	Conventional Profiles RCM	GK	Simulated Annealing Profile	Time	f_{SAPR}
7	DWT 162	162	672	1479	1417	1117	216	0.788
	DWT 193	193	1843	5312	4416	4225	911	0.957
	DWT 198	198	795	1219	1115	1096	281	0.983
	DWT 209	209	976	3610	3823	2494	325	0.691
	DWT 221	221	925	2004	2002	2204	316	1.101
	DWT 234	234	534	1305	1115	814	173	0.730
	DWT 245	245	853	5196	3568	1939	271	0.543
	DWT 307	307	1415	8336	8540	6290	526	0.755
	DWT 310	310	1379	2836	2697	2639	541	0.878
	DWT 346	346	1786	7688	7335	5925	740	0.808
	DWT 361	361	1657	4714	4699	4646	647	0.989
	DWT 419	419	1991	8230	7654	6173	838	0.807
	DWT 492	492	1824	6575	5021	3156	716	0.629
	DWT 503	503	3265	14816	14539	11817	1930	0.813
	DWT 512	512	2007	4807	4456	4031	864	0.905
	DWT 592	592	2848	10848	10333	8940	1510	0.865
	DWT 607	607	2869	17261	14283	13013	1400	0.911
	DWT 758	758	3376	7822	7417	11433	1830	1.541
	DWT 869	869	4077	18424	14589	15820	2600	1.084
	DWT 878	878	4163	21513	18818	17499	2860	0.930
	DWT 918	918	4151	22187	19580	15286	2630	0.781

Table 8: Results of Profile Reduction of Harwell-Boeing Matrices (continued)

H-B #	Matrix Name	N	# of Matrix Nonzeros	Conventional Profiles RCM	Conventional Profiles GK	Simulated Annealing Profile	Simulated Annealing Time	f_{SAPR}
11	LSHP 265	265	1009	3479	3355	3168	169	0.944
	LSHP 406	406	1561	6541	6346	5970	329	0.941
	LSHP 577	577	2233	11012	10730	10113	570	0.942
	LSHP 778	778	3025	17158	16773	15893	854*	0.948
	LSHP1009	1009	3937	25245	24741	23501	1300*	0.950
	LSHP1270	1270	4969	35539	34900	33419	1750*	0.958
	LSHP1561	1561	6121	48306	47516	46133	2170*	0.971
	LSHP1882	1882	7393	63812	62855	61523	2810*	0.979
	LSHP2233	2233	8785	82323	81183	90032	3290*	1.109
	LSHP2614	2614	10297	104105	102766	102100	3890*	0.994
	LSHP3025	3025	11929	129424	127870	124753	4610*	0.976
13	NOS1	237	627	467	468	872	82	1.867
	NOS2	957	2547	1907	1908	3937	506	2.064
	NOS3	960	8402	47536	41273	36487	3970	0.884
	NOS4	100	347	744	750	652	40	0.876
	NOS5	468	2820	25228	22471	19997	717	0.890
	NOS6	675	1965	9305	9095	12501	443	1.374
	NOS7	729	2673	34110	34110	34201	732	1.003
19	GR.30.30	900	4322	33872	28251	27170	1690*	0.962
20	PLAT362	362	3074	9261	13388	8238	1050	0.890
23	BLCKHOLE	2132	8502	171437	169219	107710	3210	0.637

Table 9: Results of Profile Reduction of Harwell-Boeing Matrices (continued)

H-B #	Matrix Name	N	# of Matrix Nonzeros	Conventional Profiles RCM	Conventional Profiles GK	Simulated Annealing Profile	Simulated Annealing Time	f_{SAPR}
23	SSTMODEL	3345	13047	105421	104562	82805	6030	0.792
24	BCSSTK19	817	3835	9594	8152	9505	1120	1.166
	BCSSTK20	485	1810	5069	4408	3859	336	0.875
	BCSSTK21	3600	15100	174478	172754	272798	6820	1.579
	BCSSTK22	138	417	851	842	729	63	0.866
28	494 BUS	494	1080	13272	8412	2921	446	0.347
	662 BUS	662	1568	28727	16601	6847	720	0.412
	685 BUS	685	1967	25606	16314	6857	897	0.420
	1138 BUS	1138	2596	43680	36707	12278	1450	0.334
29	ZENIOS	2873	15302	12981	12723	23172	12100*	1.821
30	BCSSTK26	1922	16129	194842	188032	102187	17900	0.543

Table 10: Results of Profile Reduction of Harwell-Boeing Matrices (continued)

Column	Description
H-B #	Harwell-Boeing "Tape File Number"
Matrix Name	Harwell-Boeing designation of the matrix
N	dimension of the matrix is $N \times N$
# of Matrix Nonzeros	number of non-zero elements in the matrix
RCM	Reverse Cuthill-McKee profile derived from SPARSPAK library (source: [3])
GK	Gibbs-King profile (source: [9])
Profile	the SAPR result
Time	total (user + system) CPU time (sec) required on a Sun SPARCstation-2 to perform SAPR
f_{SAPR}	ratio of Profile (with no RCMP) to the minimum of RCM and GK
*	flags certain SA profile computations that did not converge after 100 or more temperature decrements (all profile changes were $< 0.1\%$ at that point, so this should not noticeably affect overall results)

Table 11: Key to Tables 6–10

Matrix Name	Simulated Annealing Schedule S1 Profile	S1 Time	S2 Profile	S2 Time	S3 Profile	S3 Time	S4 Profile	S4 Time	S5 Profile	S5 Time
DWT 59	$\boxed{214}$	19.6	215	21.1	215	40.7	215	9.8	215	19.2
DWT 66	$\boxed{127}$	26.6	$\boxed{127}$	24.4	$\boxed{127}$	50.6	204	12.8	$\boxed{127}$	26.0
DWT 72	$\boxed{147}$	17.8	155	17.4	153	35.0	163	8.7	151	18.5
DWT 87	$\boxed{428}$	50.4	$\boxed{428}$	47.8	429	90.1	462	22.5	431	49.2
DWT 162	1117	127.0	1183	131.0	1258	234.8	1262	60.1	$\boxed{1110}$	131.4
DWT 193	4225	536.5	4233	535.8	4230	1045.0	4228	254.5	$\boxed{4217}$	543.0
DWT 198	1096	165.7	1094	172.8	1091	323.8	1094	79.8	$\boxed{1090}$	158.1
DWT 209	2494	193.1	$\boxed{2493}$	199.9	3029	371.4	2503	96.6	2503	194.2
DWT 221	2204	185.4	1643	197.8	$\boxed{1634}$	364.9	1650	92.9	2062	178.2
DWT 234	$\boxed{814}$	95.3	818	94.1	815	172.8	838	45.7	818	88.6
DWT 245	$\boxed{1939}$	153.9	2022	144.7	2041	297.4	1986	76.8	2024	157.2
DWT 307	$\boxed{6290}$	304.5	6569	318.0	6315	627.2	6437	169.2	6391	310.1
DWT 310	2639	302.5	2636	311.3	$\boxed{2631}$	603.5	2642	171.3	2645	320.6
DWT 346	5925	457.8	5843	475.2	5882	863.5	5945	231.6	$\boxed{5842}$	446.2
DWT 361	4646	377.8	4647	377.8	$\boxed{4638}$	773.1	4681	194.9	4642	381.5
DWT 419	6173	509.7	6345	490.3	$\boxed{6147}$	1034.5	6309	255.8	6242	483.2
DWT 492	3156	427.7	3466	439.5	$\boxed{2855}$	902.4	4663	239.7	3083	451.0
DWT 503	11817	1180.6	$\boxed{11599}$	1079.6	11682	2167.8	11827	554.6	11611	1099.4
DWT 512	4031	495.0	$\boxed{3918}$	438.5	3937	908.0	4325	219.8	3991	532.6
DWT 592	8940	841.4	$\boxed{8914}$	805.6	10414	1535.2	9068	461.2	8933	884.0

Table 12: SAPR Results For HBTF#7 With Several Annealing Schedules

Matrix	Simulated Annealing Schedule									
	S1		S2		S3		S4		S5	
Name	Profile	Time	Profile	Time	Profile	Time	Profile	Time	Profile	Time
DWT 607	13013	772.2	12700	773.2	$\boxed{12613}$	1428.6	13108	396.8	13075	820.7
DWT 758	11433	953.2	9886	945.6	10020	1798.0	11342	541.0	$\boxed{7111}$	1409.6
DWT 869	15820	1363.6	18164	1124.7	12416	2275.1	$\boxed{11723}$	701.8	12274	1307.9
DWT 878	$\boxed{17499}$	1472.7	17733	1225.2	17638	2367.3	18262	702.2	17953	>1602.8
DWT 918	$\boxed{15286}$	1394.2	19452	1368.3	21280	2400.9	28358	694.0	15572	1392.7
DWT 992	32204	4048.5	32258	3919.7	$\boxed{31940}$	7637.3	32526	2134.0	32794	4099.3
DWT 1005	32213	>1504.4	33116	1502.9	$\boxed{32167}$	2816.1	33189	810.5	32584	1604.5
DWT 1007	19246	1811.2	19362	1455.2	$\boxed{16965}$	2805.7	20791	856.5	19288	1721.4
DWT 1242	32452	2046.6	40672	2185.1	$\boxed{32440}$	3695.6	33676	>1182.4	32549	2340.9
DWT 2680	86068	>5589.2	112866	>6928.0	109228	11967.7	143576	>3395.7	$\boxed{85048}$	>6474.1

Table 13: SAPR Results For HBTF#7 With Several Annealing Schedules (continued)

Column	Description
Matrix Name	Harwell-Boeing designation of the matrix
Profile	SAPR profile resulting from annealing schedule S1 - S5 described in Table 3 (best results boxed)
Time	total (user + system) CPU time (sec) required on a Sun SPARCstation-2 4/690 to perform the corresponding SAPR
$>$	flags certain SA profile computations that did not converge after the temperature fell below 0.01 (all profile changes were $< 0.1\%$ at that point, so this would not noticeably affect overall results)

Table 14: Key to Tables 12–13

Matrix Name	Conventional Profiles RCM Cav	Eve	Cur	GK	Current SAPR Profile	Time	f_{SAPR}	Previous SAPR Profile	f_{Arm}
DWT 59	$\boxed{255}$	313	$\boxed{255}$	$\boxed{255}$	214	19.6	0.839	273	0.784
DWT 66	151	211	151	$\boxed{127}$	127	24.4	1.000	193	0.658
DWT 72	297	$\boxed{245}$	297	255	147	17.8	0.600	219	0.671
DWT 87	609	687	$\boxed{575}$	595	428	50.4	0.744	515	0.831
DWT 162	1479	1604	1456	$\boxed{1417}$	1110	131.4	0.783	1272	0.873
DWT 193	5312	4844	4948	$\boxed{4416}$	4217	543.0	0.955	4409	0.956
DWT 198	1219	1366	1186	$\boxed{1115}$	1090	158.1	0.978	1287	0.847
DWT 209	3610	3950	$\boxed{3568}$	3823	2493	199.9	0.699	2693	0.926
DWT 221	2004	2166	2147	$\boxed{2002}$	1634	364.9	0.816	1848	0.884
DWT 234	1305	1544	1366	$\boxed{1115}$	814	95.3	0.730	1016	0.801
DWT 245	5196	4018	5155	$\boxed{3568}$	1939	153.9	0.543	2161	0.897
DWT 307	8336	$\boxed{8136}$	8730	8540	6290	304.5	0.773	6535	0.963
DWT 310	2836	3007	2789	$\boxed{2697}$	2631	603.5	0.976	2940	0.895
DWT 346	7688	7508	7714	$\boxed{7335}$	5842	446.2	0.796	6136	0.952
DWT 361	4714	5090	4714	$\boxed{4699}$	4638	194.9	0.987	4992	0.929
DWT 419	8230	9050	8246	$\boxed{7654}$	6147	1034.5	0.803	6512	0.944
DWT 492	6575	6691	6982	$\boxed{5021}$	2855	902.4	0.569	3304	0.864
DWT 503	14816	15945	14829	$\boxed{14539}$	11599	1079.6	0.798	11958	0.970
DWT 512	4807	5325	4740	$\boxed{4456}$	3918	438.5	0.879	4384	0.894
DWT 592	10848	14563	11207	$\boxed{10333}$	8914	805.6	0.863	9417	0.947

Table 15: Comparison of SAPR With Conventional and Previous SA Profile Reduction Results

Matrix Name	Conventional Profiles RCM Cav	Eve	Cur	GK	Current SAPR Profile	Time	f_{SAPR}	Previous SAPR Profile	f_{Arm}
DWT 607	17261	15721	17151	$\boxed{14283}$	12613	1428.6	0.883	13065	0.965
DWT 758	7822	11370	7808	$\boxed{7417}$	7111	1409.6	0.959	7123	0.998
DWT 869	18424	16163	15660	$\boxed{14589}$	11723	701.8	0.804	13207	0.888
DWT 878	21513	20545	21485	$\boxed{18818}$	17499	1472.7	0.930	17835	0.981
DWT 918	22187	22399	22201	$\boxed{19580}$	15286	1394.2	0.781	15949	0.958
DWT 992		$\boxed{35018}$	37136		31940	7637.3	0.912	32528	0.982
DWT 1005		$\boxed{41104}$	42176		32167	2816.1	0.783	32513	0.989
DWT 1007		24168	$\boxed{23685}$		16965	2805.7	0.716	19913	0.852
DWT 1242		51419	$\boxed{48837}$		32440	3695.6	0.664	33098	0.980
DWT 2680		105324	$\boxed{102112}$		85048	6474.1	0.833	84900	1.002

Table 16: Comparison of SAPR With Conventional and Previous SA Profile Reduction Results (continued)

Column	Description
Matrix Name	Harwell-Boeing designation of the matrix
Conventional Profiles	non-SA profiles from various sources (best results boxed)
Cav	Reverse Cuthill-McKee profile derived from SPARSPAK library (source: [3])
Eve	RCM profile (source: [5])
Cur	RCM profile computed by author using pseudoperipheral node algorithm described in [6]
GK	Gibbs-King profile (source: [9])
Profile	best SAPR profile from Tables 12–13
Time	total (user + system) CPU time (sec) required on a Sun SPARCstation-2 4/690 to obtain the best SAPR profile
f_{SAPR}	ratio of the best Sn profile to the minimum of the RCM and GK profiles
Previous SAPR Profile	simulated annealing profile reported by [1]
f_{Arm}	ratio of [1] results to the best SAPR profile

Table 17: Key to Tables 15–16

H-B #	Matrix Name	N	# of Matrix Nonzeros	Conventional Fills ND	MDA	CTB	Simulated Annealing Fill	Time	f_{SAFR}
3	ASH85	85	304	390	$\boxed{211}$		195	6122.9	0.924
4	BCSPWR01	39	85	23	$\boxed{19}$		19	118.4	1.000
	BCSPWR02	49	108	49	$\boxed{21}$		20	206.2	0.952
5	BCSSTK01	48	224	336	$\boxed{269}$		231	5849.2	0.859
6	CAN 24	24	92	34	$\boxed{27}$		23	179.6	0.852
	CAN 61	61	309	112	$\boxed{56}$		52	2699.9	0.929
	CAN 62	62	140	97	$\boxed{44}$		44	430.2	1.000
	CAN 73	73	225	263	$\boxed{163}$		164	4901.6	1.006
	CAN 96	96	432	699	$\boxed{544}$		545	40062.6	1.002
7	DWT 59	59	163	121	$\boxed{81}$		79	771.8	0.975
	DWT 66	66	193	110	$\boxed{0}$		8	554.8	ERROR
	DWT 72	72	147	87	$\boxed{37}$		37	351.0	1.000
	DWT 87	87	314	319	$\boxed{105}$		99	4388.6	0.943
28	685 BUS	685	1967	4212	1692	$\boxed{1586}$	1523	20700.0	0.960
	1138 BUS	1138	2596	3838	673	$\boxed{641}$	691	22900.0	1.078

Table 18: SAFR Results

Column	Description
H-B #	Harwell-Boeing "Tape File Number"
Matrix Name	Harwell-Boeing designation of the matrix
N	dimension of the matrix is $N \times N$
# of Matrix Nonzeros	number of non-zero elements in the matrix
Conventional Fills	non-SA fills from various sources (best results boxed)
ND	best fill from conventional nested dissection ordering (source: [7])
MDA	best fill resulting from conventional minimum-degree algorithm ordering (source: [7])
CTB	best fill resulting from minimum-degree algorithm with the tie-breaking strategy in [2]
Fill	the SAFR result
Time	total (user + system) CPU time (sec) required on a Sun SPARCstation-2 to perform SAFR
f_{SAFR}	ratio of Fill to the minimum of ND, MDA, and CTB

Table 19: Key to Table 18

8 Figures

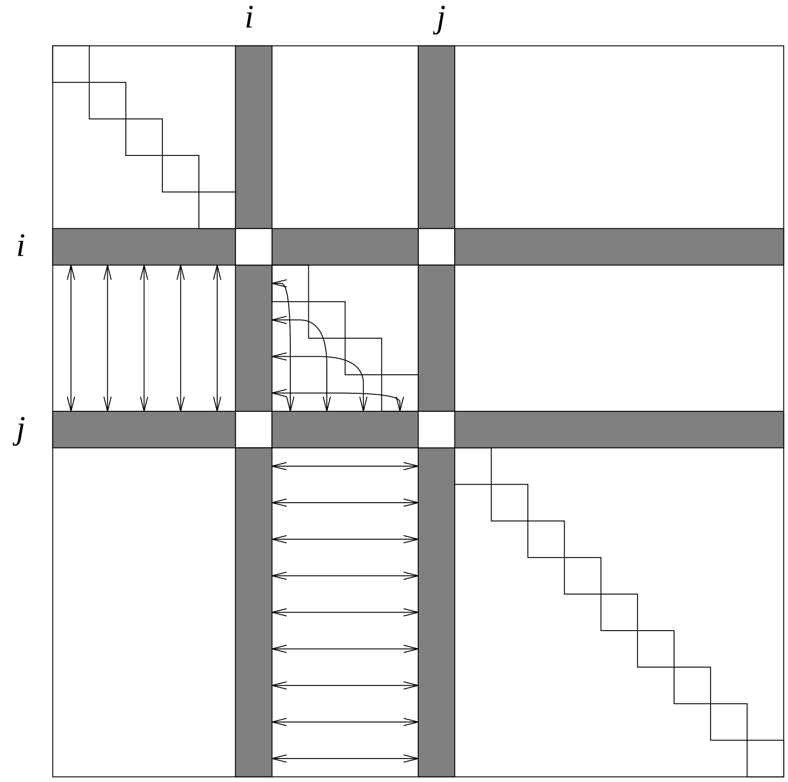

Figure 1:

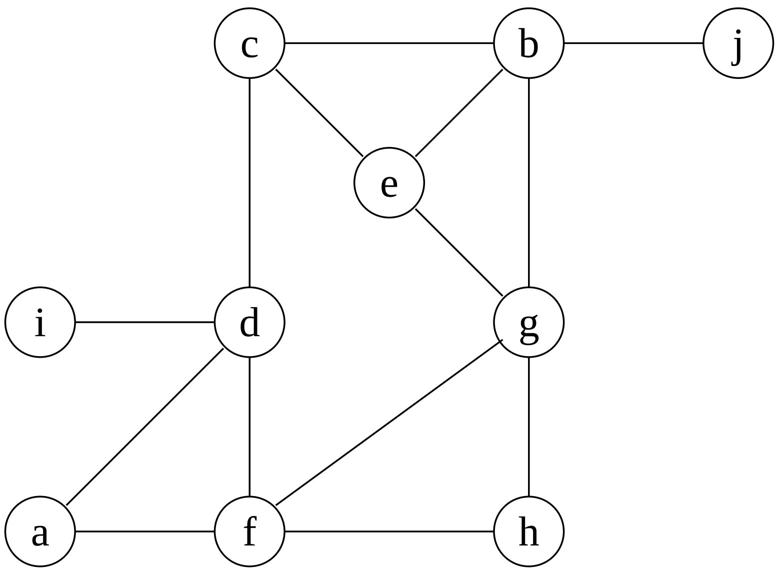

Figure 2:

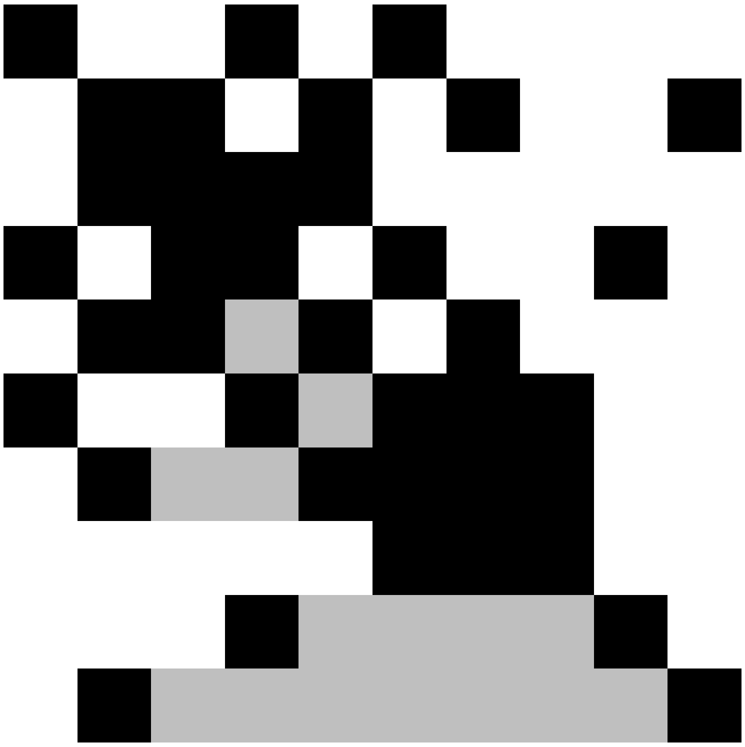

Figure 3:

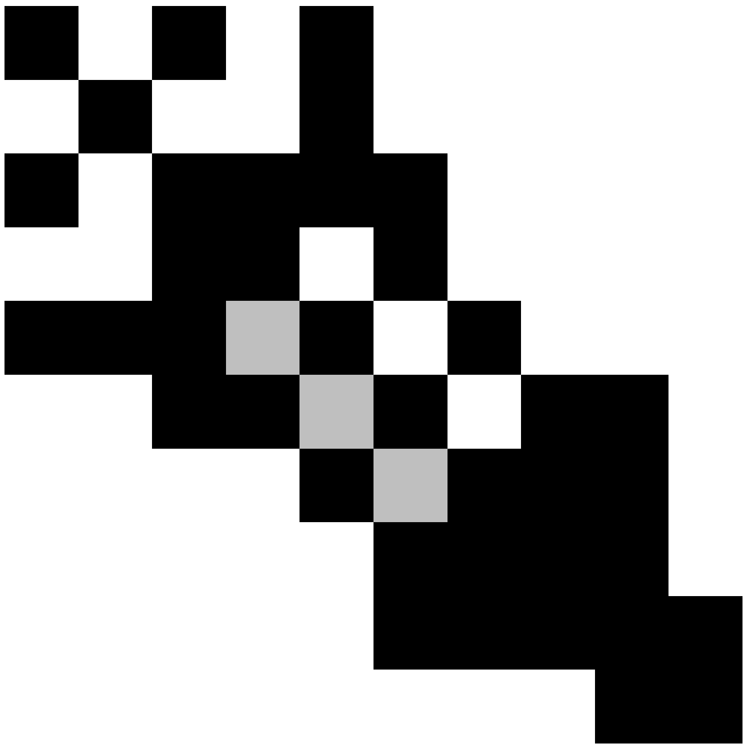

Figure 4:

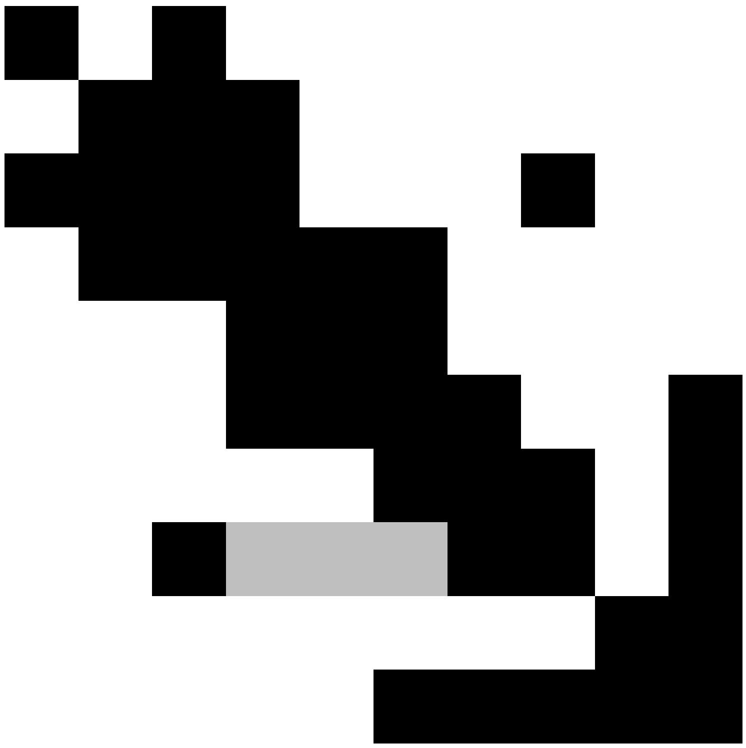

Figure 5:

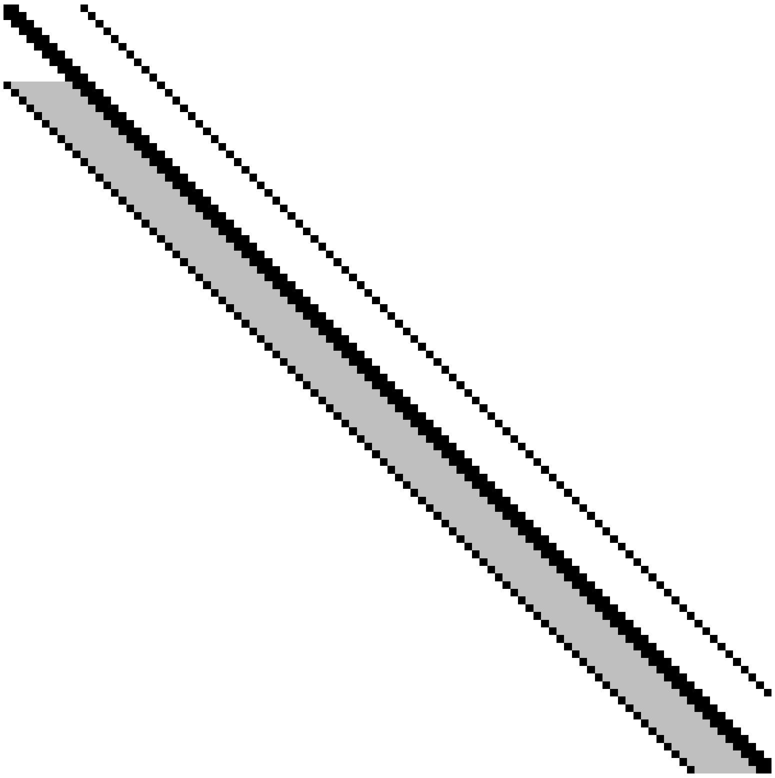

Figure 6:

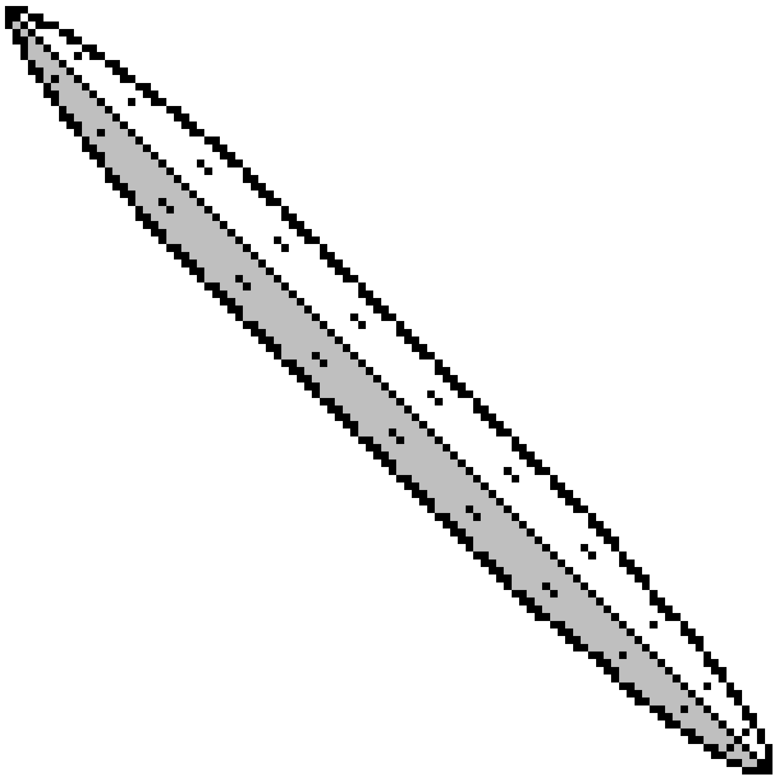

Figure 7:

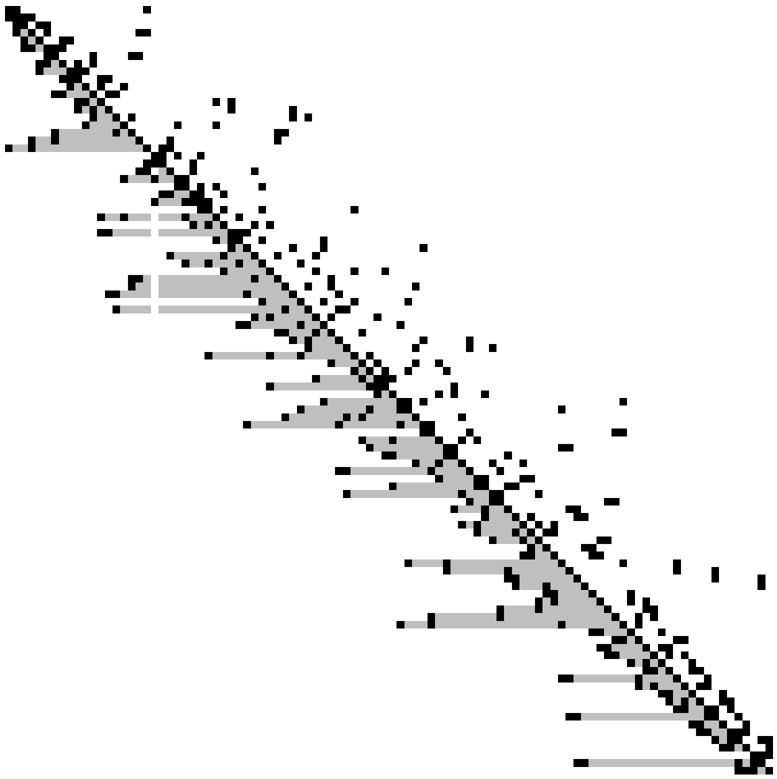

Figure 8:

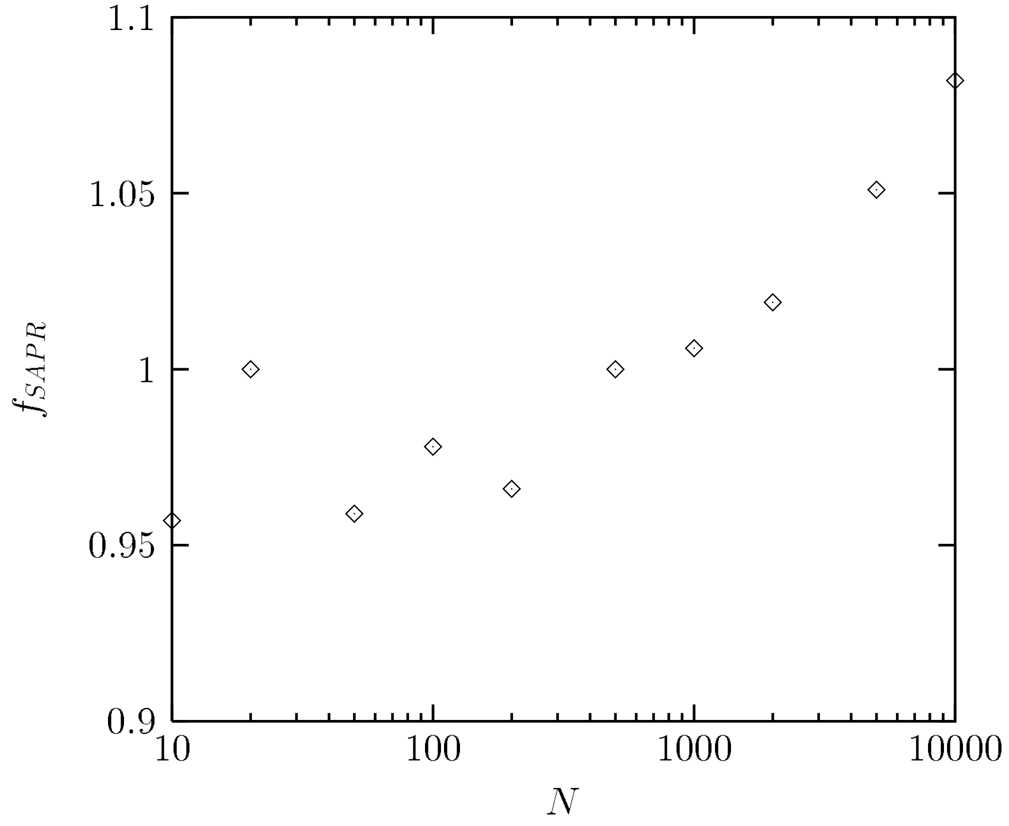

Figure 9:

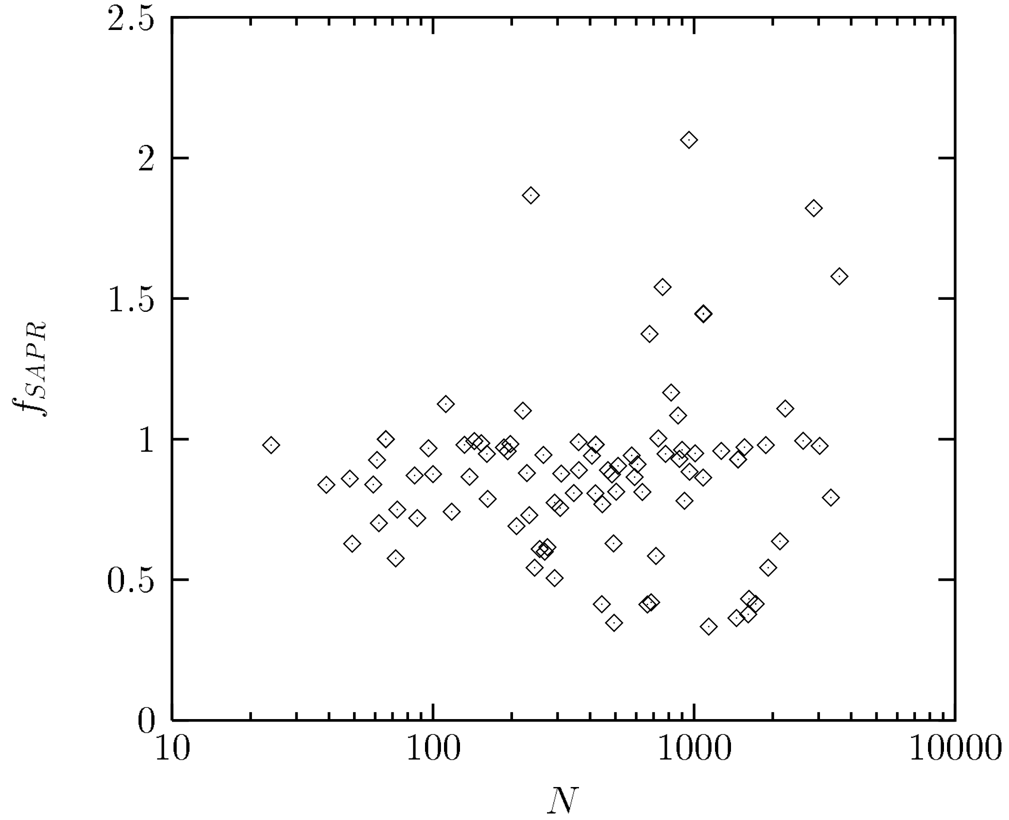

Figure 10:

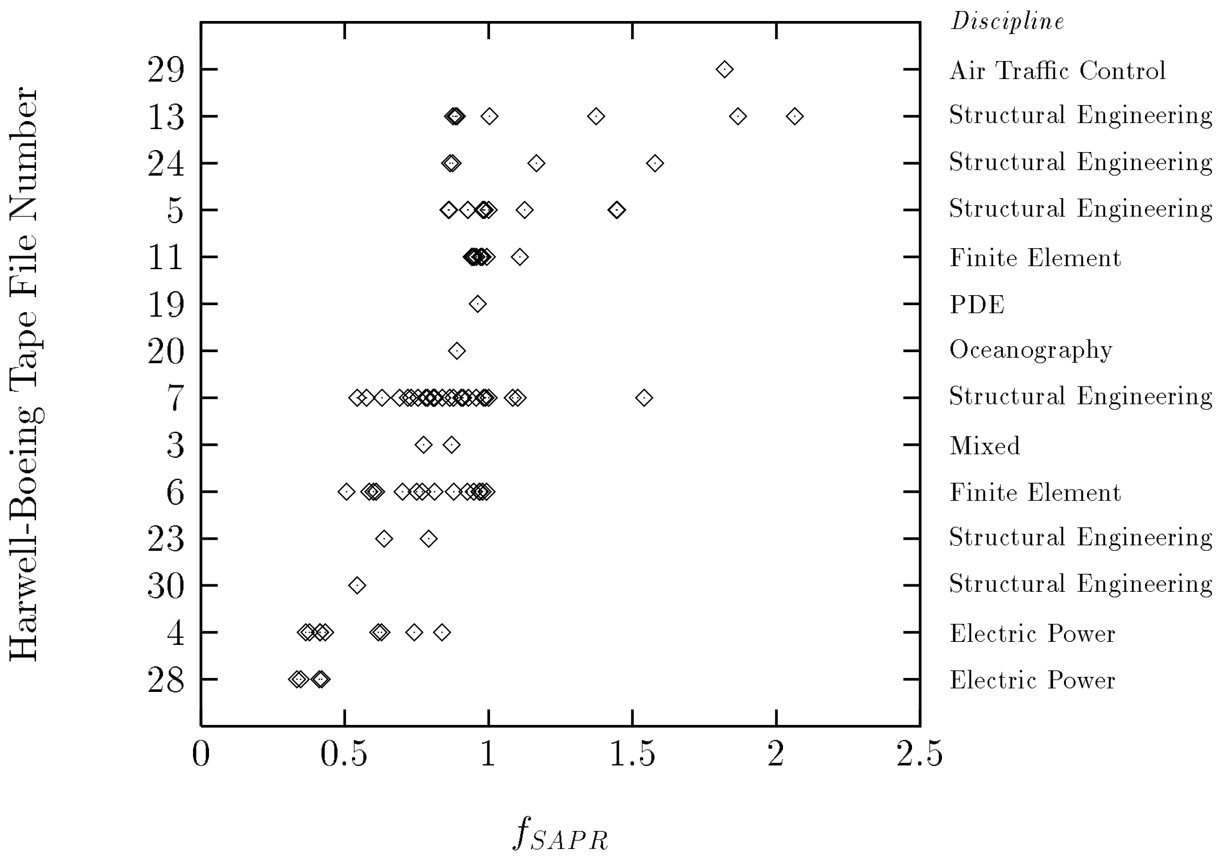
Discipline
29
13
24
5
11
19
20
7
3
6
23
30
4
28
Harvell-Boeing Tape File Number
Air Traffic Control
Structural Engineering
Structural Engineering
Structural Engineering
Finite Element
PDE
Oceanography
Structural Engineering
Mixed
Finite Element
Structural Engineering
Structural Engineering
Electric Power
Electric Power
0
0.5
1
1.5
2
2.5
f_{SAPR}

Figure 11:

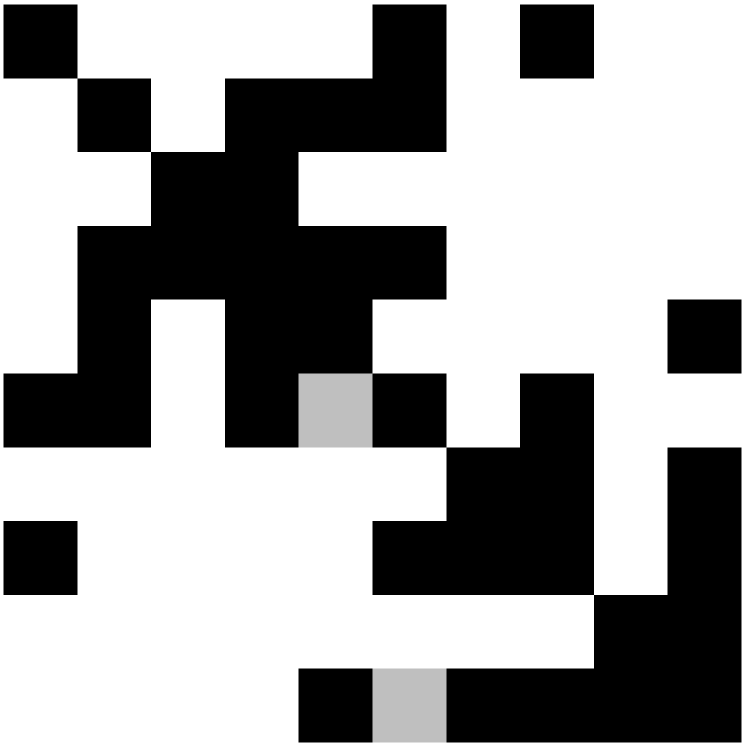

Figure 12:

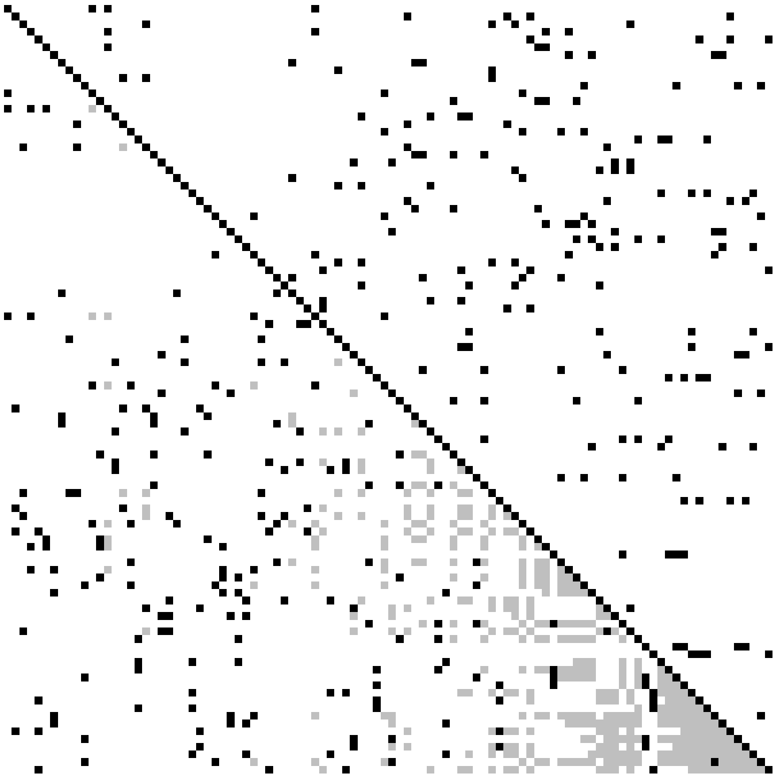

Figure 13:

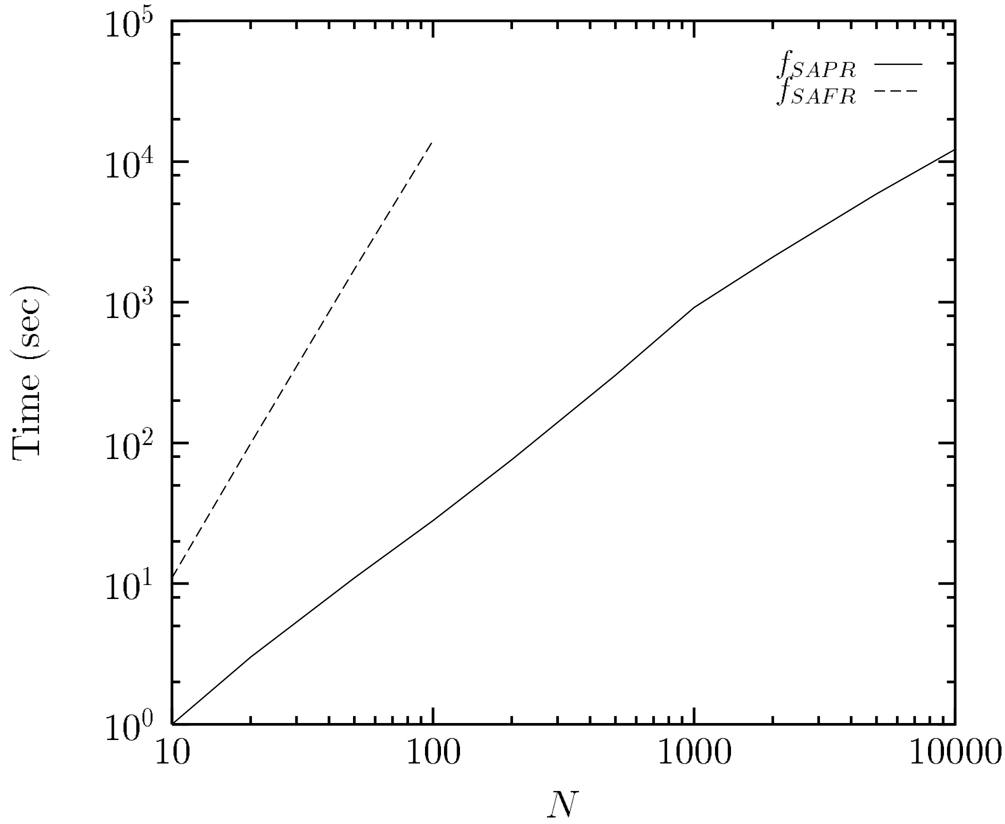
f_{SAPR}
f_{SAFR}
10^5
10^4
10^3
10^2
10^1
10^0
Time (sec)
10
100
1000
10000
N

Figure 14:

LEGENDS

Figure 1: Swapping Two Rows And Columns of a Symmetric Matrix

Figure 2: A Simple Example

Figure 3: Sparsity Pattern of Original Figure 2 Matrix

Figure 4: Sparsity Pattern of RCM-ordered Figure 2 Matrix

Figure 5: Sparsity Pattern of SA Profile Reduction of Figure 2 Matrix

Figure 6: Sparsity Pattern of Original 100×100 2D 5-point PDE Matrix

Figure 7: Sparsity Pattern of RCM-ordered 100×100 2D 5-point PDE Matrix

Figure 8: Sparsity Pattern of SA Profile Reduction of 100×100 2-D 5-point PDE Matrix

Figure 9: f_{SAPR} vs. N for a Range of 2-D 5-point PDE Matrices

Figure 10: Relative Performance of SAPR vs. N

Figure 11: Relative Performance of SAPR vs. Harwell-Boeing Tape File Number

Figure 12: Sparsity Pattern of SAFR of Figure 2 Matrix

Figure 13: Sparsity Pattern of SAFR of 100×100 2-D 5-point PDE Matrix

Figure 14: Times Required for SA Fill and Profile Minimization of 2-D 5-point PDE Matrix