1-1-1981

Solution strategies and algorithm behavior in large-scale network codes

Richard S. Barr

Southern Methodist University

Edwin L. Cox School of Business

SOLUTION STRATEGIES AND ALGORITHM BEHAVIOR IN LARGE-SCALE NETWORK CODES

Working Paper 81-200*

by

Richard S. Barr

Southern Methodist University
Dallas, Texas 75275

SOLUTION STRATEGIES AND ALGORITHM BEHAVIOR IN LARGE-SCALE NETWORK CODES

Working Paper 81-200*

by

Richard S. Barr

Associate Professor of Management Science
Edwin L. Cox School of Business
Southern Methodist University
Dallas, Texas 75275

This paper was prepared for the Conference on Mathematical Programming, Testing and Validating Algorithms and Software, National Bureau of Standards, Boulder, Colorado, January 5-6, 1981.

1. INTRODUCTION

As detailed in recent reports [1,2,3], the problem of optimally merging microdata files results in the need to solve extremely large uncapacitated transportation problems. Problems with dimensions of over 20,000 constraints and tens of millions of variables are not uncommon and must be optimized on a regular basis. In the mid-1970's, an in/out-of-core, primal-simplex-based solution sytem was devised to met this need for the U.S. Department of the Treasury [1].

This study reports the results of recent experimentation with this system to test various solution strategy parameters and implementation schemes. Numerous runs were made to explore the effect on solvability of such considerations as pricing and pivoting rules, data storage technique, compiler, data page size, and problem density. In addition, computer graphics were used to study algorithm behavior during the solution process.

2. THE TESTING ENVIRONMENT

Problem overview. A microdata file is a collection of sample observation records, often based on a national survey. Two such files may be combined, or merged, by mating records from one file with similar records in the other file. Hence, the resultant composite file will consist of enhanced data records with each record containing items from both of the original surveys. The merged file provides an "enriched" data source for use in microanalytic models and policy research.

The optimal set of record pairings may be found by minimizing a linear interrecord dissimilarity function subject to transportation constraints [2]. In their fullest form, these problems are enormous. The merge model includes one structural variable, or network arc, for each pair of records which may be matched plus a network node constraint for each record in each file to avoid

over-matching or excluding records. So, recently, when a 75,000-record IRS file was to be merged with a 61,000-record Census file, the full problem involved 136,000 constraints and over 4.5 billion variables.

Computing environment. All runs reported herein were made on the U.S. Treasury's Univac 1100/81 mainframe with 400K 36-bit words of primary storage, running under the EXEC 8 operating system. Both the FORTRAN V and ASCII FORTRAN compilers were employed, each permitting buffered input/output from the system disks.

Matrix generator. Because of its size, direct solution of a full merge problem is typically well beyond mathematical programming's state-of-the-art, even with the advantageous network structure. For this reason, the matrix generator employs two techniques to reduce problem dimensions. First, the files are partitioned into several sub-files which are merged separately and combined, thus limiting the number of constraints to be considered simultaneously. Secondly, within each sub-problem, nondense problems are generated by including arcs for the "p-best" matches for each record, thus reducing the number of variables but making "infeasible" problems possible.

In designing for problems with up to 50,000 constraints, machine wordsize limits the magnitude of the dual variables and, hence, the problem costs. For this reason, the arc costs are scaled to range from 0 to 63. Finally, in order to accommodate percentage of optimality calculations for intermediate solutions (not used, see [2]), the arcs are sorted in ascending cost order.

Optimization software. The solution system was designed with minimal primary storage needs to accommodate large problems. A primal simplex algorithm was deemed best because of the low memory requirements and greater efficiency relative to other methodologies. Furthermore, only a spanning-tree basis need be maintained in-core with the arc cost data paged in piecewise

from secondary storage. The system was coded entirely in FORTRAN to simplify maintenance and enhance portability.

The heart of the system is an efficient primal-simplex transportation code which uses an independently-derived variant of the ATI algorithm [4] and maintains the basis in four node-length arrays. By packing, the basis data can be represented in only two node-length arrays. An initial basis is constructed from a single pass of the arc data and any artificials driven out of solution by either the Phase I-II or the Big-M (if the data is not packed) method.

Figure 1 depicts the general system logic and indicates the parameters used to control arc pricing. Note the "double buffering" technique which

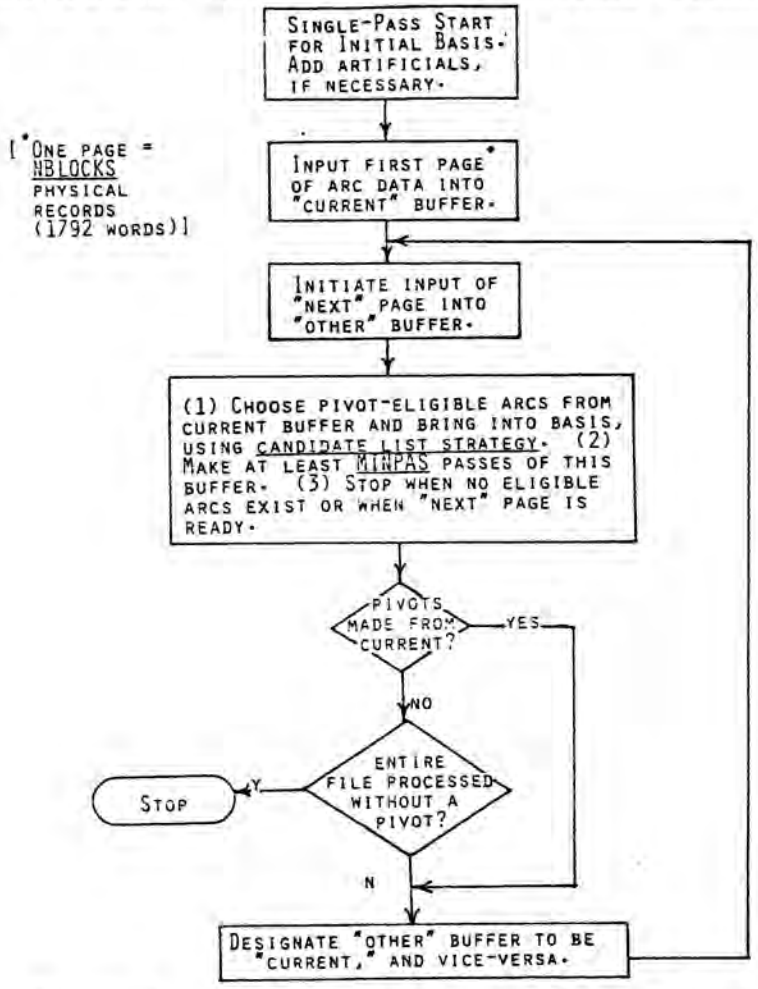

Figure 1. Solution System Flow Diagram

allows a page of arc data to be read in parallel with the pricing and pivoting operations, thus reducing the real-time, if not CPU-time, requiremnts.

In the pricing process, arc data is brought into core one page at a time; NBLOCK defines the page size in multiples of 1792 words. Arcs from the "current" page are selected for pivoting based on the "candidate list" technique of Mulvey which uses two parameters, K and L. With a K/L strategy, (1) L pivot-eligible arcs are selected and placed on the list, (2) arcs on the list are priced and the one with the most negative marginal value is chosen for pivoting into the solution, and (3) step two is repeated K times before a new list is made (see [5] for details). Candidate lists are built and processed until either MINPAS complete passes were made of the page, or until no pivot-eligible arcs remain. This process is applied anew to each page of arc data, and the values of NBLOCK, K, L, and MINPAS define a pricing strategy for the algorithm.

3. EXPERIMENTATION

The base problem. For test purposes, a relatively small test problem was used, consisting of 3115 origin nodes, 1463 destination nodes (4578 constraints), and 623,000 arcs. The initial solution contained 291 artificial arcs, the problem was feasible, and solution times ranged from five to ten CPU minutes.

Pricing and pivoting tests. To evaluate the effects of the pricing parameters, several strategies were used to solve the base problem, as described in Table 1. The FORTRAN V compiler was used, the basis data stored in packed form, and the Phase I-II method employed. Elapsed central processor (CP) time is shown for problem optimization only but may include a portion of the data input time.

Table 1. Pricing and Pivoting Tests Using the Base Problem

Run No.	MINPAS (Maximum Passes/Page)	NBLOCK (No. of Data Blocks Per Buffer)	K/L (Candidate List Strategy)	Number of Pivots	CP Time (Seconds)
1.	99	4	20/40	32,071	497
2.	99	1	20/40	33,051	453
3.	99	4	1/1	45,010	601
4.	3	4	20/40	28,848	403
5.	3	1	20/40	30,625	433

This data indicates that solution times tend to improve when making fewer page passes (MINPAS=3) instead of processing a page of arc data until no eligible arcs remain (MINPAS=99). This is perhaps due to the decreasing rate of solution improvement with continued pricing of a page. Figure 2 illustrates this effect for a set of pivots chosen from early in Phase II of run 1. For

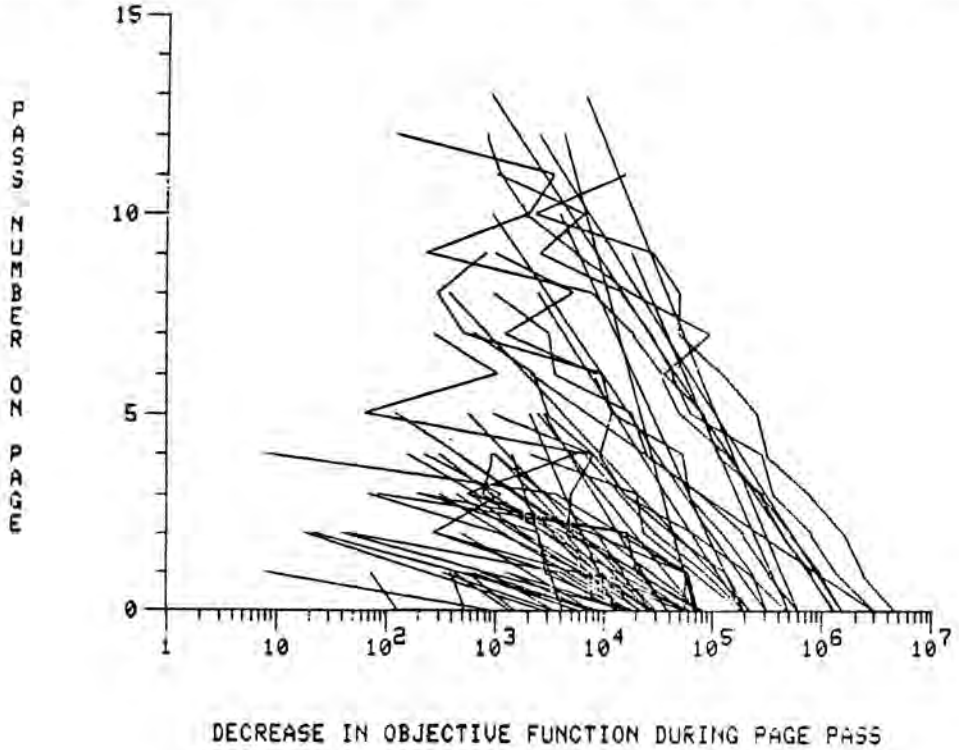

Figure 2. The Effect of Multiple Page Passes on Objective Function Value (MINPAS=99).

each new "current page" inspected, a "pass number" is initially set to zero and incremented by one each time the pricing routine reaches the end of the page. In Figure 2, these pass numbers are plotted against the objective function improvement from pivots made during the page pass. As might be expected, the improvement tends to diminish as the page is priced repeatedly. Hence, a strategy using a smaller MINPAS accepts a good improvement rather than seeking the maximum improvement per page.

Figure 3 illustrates the objective function behavior versus pivot number during the solution process with different MINPAS values (runs 1 and 4). The objective function increases are due to the search for a feasible solution in Phase I. The smaller MINPAS strategy achieves feasibility sooner, with the Phase II solution improvement rates approximately the same in both cases.

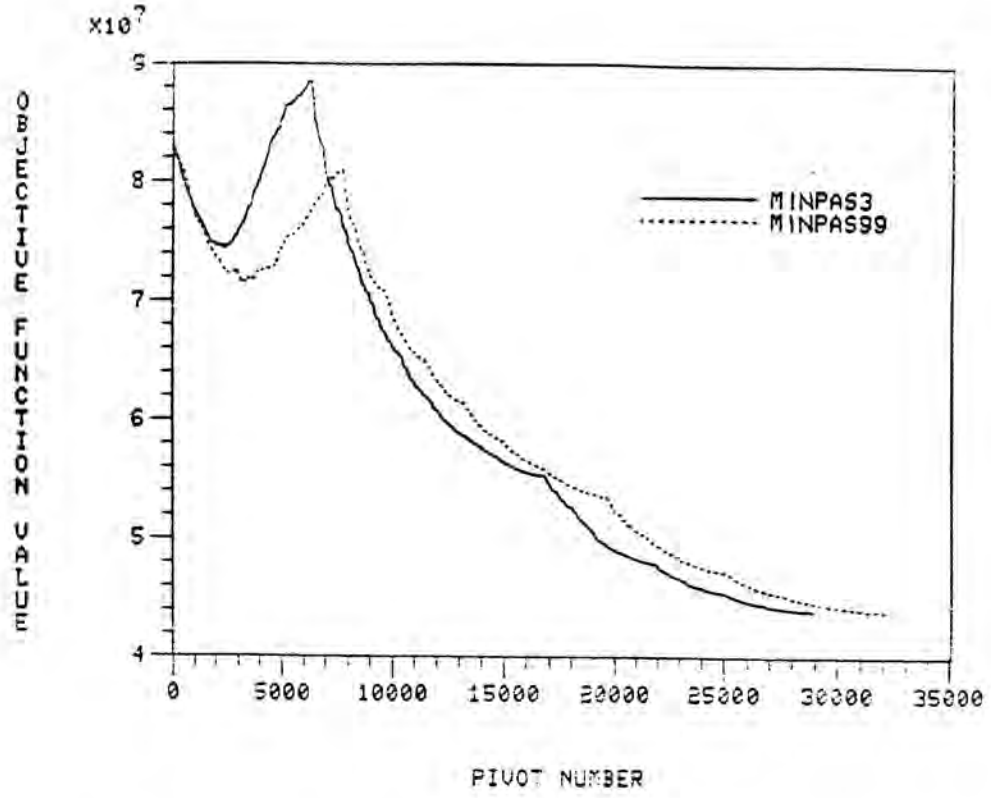

Figure 3. Objective Function Value versus Pivot Number Using Different MINPAS Values.

Varying the size of the data buffer (NBLOCK) yields mixed results (see runs 1,2,4, and 5 in Table 1). Generally, if MINPAS is exhaustive, the smaller page size is better; otherwise the larger page size is preferable. The best overall time, from run 4, used the larger NBLOCK value.

In comparing candidate list strategies, the larger 20/40 list tended to yield better solution times than the "modified row minimum" 1/1 strategy. Comparing runs 1 and 3 in Table 1, the 1/1 strategy was 20% slower and required 40% more pivots.

Complier, data storage, and methodology tests. Two compilers are available on the Univac 1100: FORTRAN V, a mature product, and a newer ASCII FORTRAN complier. Both packages generate well-optimized code. Table 2 shows the results of a series of runs using both compilers, with runs 1 and 2 indicating that the ASCII-generated code is slightly faster for the base problem.

The effect of packing the basis data can be seen by contrasting runs 2 and 3a from Table 2. By eliminating the unpacking operations and storing the basis in normal TYPE INTEGER format, solution time is reduced by 20 percent.

Table 2. Tests of Compiler, Data Storage Technique and Solution Method on Base Problem*

Run	Compiler	Basis Data	Solution Method	Pivots	CP Seconds
1.	FORTRAN V	Packed	Phase I-II	28,848	403
2.	ASCII	Packed	Phase I-II	28,848	371
3a.	ASCII	Unpacked	Phase I-II	29,401	297
3b.	ASCII**	Unpacked	Phase I-II	29,401	451
4a.	ASCII	Unpacked	Big-M	34,185	306
4b.	ASCII**	Unpacked	Big-M	34,185	397

*MINPAS=3, NBLOCKS=4, 20/40 candidate list.

**Recompiled for core references beyond 65K.

The 1100-series architecture affords greater efficiences if all program data can be stored in 65K words or less. To reference array locations beyond this limit, a compiler option directs the generation of code which does not use the index registers for indirect addressing. A comparison of runs 3a, 3b, 4a, and 4b indicates that invoking this option escalates solution times by from 30 to 50 percent.

The method by which artificial arcs are handled can have an effect on both the number of pivots and time required to reach optimality. Contrasting runs 3a and 4a of Table 2, there seems to be little difference between the Big-M and Phase I-II methods, but runs 4a and 4b indicate a distinct advantage for the Big-M method. An extremely large value of Big M was used in these runs which, as discovered in the testing described below, was not the best choice since smaller artificial costs yielded a 15 percent improvement in this solution time.

Table 3. Comparative Solution Statistics for the Base Problem and a Totally Dense Problem

	13.6% Density	100% Density
Origin nodes	3115	3115
Destination nodes	1463	1463
Node constraints	4578	4578
Arcs	623,000	4,361,000
Cost range	0-63	0-63
MINPAS	3	3
Candidate list strategy	20/40	20/40
NBLOCK	3	7
Pivots made	29,401	25,534
Degenerate Pivots	255	273
CP Times (seconds)		
Start	14	98
Phase I	71	55
Phase II	212	782
Total (minutes)	297 (4.9)	935 (15.5)

Problem density. To investigate the effect of problem density on solvability, the base problem was regenerated with all arcs included -- not just those for the 200 best matches per records. Table 3 indicates that while the totally dense problem has seven times as many arcs, it requires only three times the solution time. Moreover, the denser problem required fewer pivots for optimization, perhaps due to the higher dimensionality, but both runs had few degenerate pivots as is characteristic of transportation problems.

Big-M values. In the folklore of mathematical programming, smaller Big-M values are preferable to larger ones. To test this hypothesis, the runs described in Table 4 were made. In each case, the cost assigned to basic artificials was set to an initial value; when the pricing routine completed a pass of all pages of the arc data, the cost of any remaining artificials was increased by a multiple of the previous cost. In the second run shown, for example, the first four values of Big M were 62, 80, 104, and 135. Generally the folklore seems to be true. The longest solution time was with the largest

Table 4. Solving the Base Problem with Various Big-M Strategies*

Big-M Value**			
Initial Value	Multiplier at End-of-File	Pivots Made	CP Time (Seconds)
3	0	33,377	257
1	1.3	30,454	255
1.1	1.3	29,741	251
1.5	1.5	31,876	259
2	0	32,222	257
10	0	34,168	266

*MINPAS = 3, NBLOCK = 4, 20/40 candidate list strategy.

**Expressed in multiples of 62, the largest cost.

Big-M cost and the best strategy began with a small cost that increased gradually, yielding a six percent improvement in solution time over the worst case.

Summary. Large-scale testing is made difficult by the high cost of a single observation and the resultant limit on the number of strategies to be considered. However, even this modest amount of testing led to the identification of solution strategies which solved a 19,406-constraint, 4.5-million-variable problem from a recent merge by making 298,893 pivots in 185 minutes. This is a considerable improvement over previous experience where, for example, 382 minutes were required to solve a comparable problem on the slightly slower Univac 1110 [1].

REFERENCES

[1] Barr, Richard S., "Design and Evolution of a Mathematical Programming System for Ultra-Large-Scale Transportation Problems," research report, Edwin L. Cox School of Business, Southern Methodist University, Dallas, TX (1981).

[2] Barr, Richard S. and J. Scott Turner, "A New, Linear Programming Approach to Microdata File Merging," in 1978 Compendium of Tax Research, U.S. Government Printing Office, Washington, D.C. (1978), 131-155.

[3] Barr, Richard S. and J. Scott Turner, "Optimal Microdata File Merging through Large-Scale Network Technology," to appear in Mathematical Programmming Studies (1981).

[4] Glover, Fred, Darwin Klingman, and Joel Stutz, "Augmented Threaded Index Method for Network Optimization," INFOR 12,3 (1974) 293-298.

[5] Mulvey, John M., "Pivot Strategies for Primal-Simplex Network Codes," JACM 25, 2 (1978) 266-270.

The following papers are currently available in the Edwin L. Cox School of Business Working Paper Series.

79-100 "Microdata File Merging Through Large-Scale Network Technology," by Richard S. Barr and J. Scott Turner

79-101 "Perceived Environmental Uncertainty: An Individual or Environmental Attribute," by Peter Lorenzi, Henry P. Sims, Jr., and John W. Slocum, Jr.

79-103 "A Typology for Integrating Technology, Organization and Job Design," by John W. Slocum, Jr., and Henry P. Sims, Jr.

80-100 "Implementing the Portfolio (SBU) Concept," by Richard A. Bettis and William K. Hall

80-101 "Assessing Organizational Change Approaches: Towards a Comparative Typology," by Don Hellriegel and John W. Slocum, Jr.

80-102 "Constructing a Theory of Accounting--An Axiomatic Approach," by Marvin L. Carlson and James W. Lamb

80-103 "Mentors & Managers," by Michael E. McGill

80-104 "Budgeting Capital for R&D: An Application of Option Pricing," by John Kensinger

80-200 "Financial Terms of Sale and Control of Marketing Channel Conflict," by Michael Levy and Dwight Grant

80-300 "Toward An Optimal Customer Service Package" by Michael Levy

80-301 "Controlling the Performance of People in Organizations," by Steven Kerr and John W. Slocum, Jr.

80-400 "The Effects of Racial Composition on Neighborhood Succession," by Kerry D. Vandell

80-500 "Strategies of Growth: Forms, Characteristics and Returns," by Richard D. Miller

80-600 "Organization Roles, Cognitive Roles, and Problem-Solving Styles," by Richard Lee Steckroth, John W. Slocum, Jr., and Henry P. Sims, Jr.

80-601 "New Efficient Equations to Compute the Present Value of Mortgage Interest Payments and Accelerated Depreciation Tax Benefits," by Elbert B. Greynolds, Jr.

80-800 "Mortgage Quality and the Two-Earner Family: Issues and Estimates," by Kerry D. Vandell

80-801 "Comparison of the EEOCC Four-Fifths Rule and A One, Two or Three σ Binomial Criterion," by Marion Gross Sobol and Paul Ellard

80-900 "Bank Portfolio Management: The Role of Financial Futures," by Dwight M. Grant and George Hempel

80-902 "Hedging Uncertain Foreign Exchange Positions," by Mark R. Eaker and Dwight M. Grant

80-110 "Strategic Portfolio Management in the Multibusi ess Firm: An Implementation Status Report," by Richard A. Bet is and William K. Hall

80-111 "Sources of Performance Differences in Related and Unrelated Diversified Firms," by Richard A. Bettis

80-112 "The Information Needs of Business with Special Application to Managerial Decision Making," by Paul Gray

80-113 "Diversification Strategy, Accounting Determined Risk, and Accounting Determined Return," by Richard A. Bettis and William K. Hall

80-114 "Toward Analytically Precise Definitions of Market Value and Highest and Best Use," by Kerry D. Vandell

80-115 "Person-Situation Interaction" An Exploration of Competing Models of Fit," by William F. Joyce, John W. Slocum, Jr., and Mary Ann Von Glinow

80-116 "Correlates of Climate Discrepancy," by William F. Joyce and John Slocum

80-117 "Alternative Perspectives on Neighborhood Decline," by Arthur P. Solomon and Kerry D. Vandell

80-121 "Project Abandonment as a Put Option: Dealing with the Capital Investment Decision and Operating Risk Using Option Pricing Theory," by John W. Kensinger

80-122 "The Interrelationships Between Banking Returns and Risks," by George H. Hempel

80-123 "The Environment For Funds Management Decisions In Coming Years," by George H. Hempel

81-100 "A Test of Gouldner's Norm of Reciprocity In A Commercial Marketing Research Setting," by Roger Kerin, Thomas Barry, and Alan Dubinsky

81-200 "Solution Strategies and Algorithm Behavior in Large-Sclae Network Codes," by Richard S. Barr