National Center for Geographic Information and Analysis

A Comparison of Strategies for Data Storage Reduction in Location-Allocation Problems

by

Paul A. Sorensen
Richard L. Church

University of California, Santa Barbara

Technical Report 95-4

May 1995

A Comparison of Strategies for Data Storage Reduction in Location-Allocation Problems

Paul A. Sorensen and Richard L. Church
NCGIA/Department of Geography
University of California
Santa Barbara, CA 93117

Abstract

Many solution techniques for discrete location-allocation problems make use of a sorted distance strings data structure in order to speed processing time. Such data structures are commonly used, for example, within interchange heuristics and lagrangian relaxation codes. The primary drawback to distance strings is that in a standard computer architecture they require approximately 50% additional memory storage in comparison to a standard distance matrix. This is due to the fact that distance strings must store not only distances, but also the order of the nodes sorted by increasing distance. To reduce the additional memory requirements of distance strings, researchers such as Hillsman [1980] and Densham and Rushton [1992a] have proposed a strategy for cutting the distance strings to include only sets of relatively close neighbor nodes. This has become an important implementation issue for solving relatively large location-allocation problems. In fact, the new Location-Allocation module of the ARC/Info GIS system uses a distance string structure and a string cutoff option to save storage and processing time. The danger in employing only partial distance strings is that if too few distance entries are stored in the distance strings, heuristic or algorithmic performance may be compromised in that the quality of the solutions generated may be less than desirable. This paper tests the effects on solution quality of imposing different distance string definitions and sizes. The goal is to establish guidelines concerning the degree to which the distance strings data structure may be reduced for memory savings without adversely affecting solution quality. The paper proposes and compares two alternative methods to that of Hillsman and of Densham and Rushton for the selection of nodes to be included within the distance strings data structure. We show that the two new alternative methods for defining distance strings appear to outperform the earlier approach. The results of this paper can be used to aid users in determining the extent that distance string cutoffs should be employed in application as well as aid in the development of location modeling software.

Introduction

Location-allocation problems may be characterized generally as follows: given a set of demand nodes and a set of potential facility sites (the sets may be partially or entirely overlapping), select some number of facilities and allocate service from the chosen set of facilities to the demand nodes in a manner that optimizes the given objective function. Usually, the objective function will include some measure of the distance or transportation cost (referred to hereafter simply as distance) between the demand nodes and the facilities to which their demand has been allocated. For example, in the p-Median location-allocation problem (refer to Hakimi [19641, Teitz and Bart [1968], and ReVelle and Swain [1970]), the goal is to select *p* facilities and assign each demand node to its nearest facility in order to minimize the total weighted service distance (demand multiplied by distance to nearest facility) of all demand nodes.

Because of the integral role of internodal distance information, solution techniques for location-allocation problems require the storage of internodal distance information for all potential facilities and demand nodes. A simple and memory efficient data structure for storing such information is a standard distance matrix. Each entry in the matrix corresponds to the distance between a demand node and a potential facility. If there are M potential facilities and N demand nodes, the size of the distance matrix is given as M*N.

While a standard distance matrix is efficient in terms of memory storage requirements, it is not always the most efficient data structure to support heuristic and optimal location algorithms. The processing speed of many solution methods for location-allocation problems can be greatly increased if internodal distance information can be retrieved in sorted order. The sorted distance strings data structure is commonly used to provide this type of access to distance information. A distance string is essentially comprised of a base node with a list of other nodes and corresponding distances to the base node stored in order of increasing distance (Hillsman [1980]). In practice, distance strings are often constructed as two separate matrices, one listing distances and one listing node indices. For example, consider a hypothetical distance string for node 1: distance 0.0 to node 1, distance 5.2 to node 2, distance 5.4 to node 6, distance 5.8 to node 3, distance 6.1 to node 5, etc. In this example, the distance matrix for row I would read: 0.0, 5.2, 5.4, 5.8, 6.1, etc.; the node index matrix for row I would read: 1, 2, 6, 3, 5, etc. In the distance strings data structure, one string exists for each demand node and/or each potential facility (depending on the implementation strategy). The sorted distance strings data structure has been used in algorithms proposed by numerous location-allocation researchers; examples may be found in Erlenkotter [1976], Weaver and Church [1983], Hillsman [1980] and Goodchild and Noronha [1983].

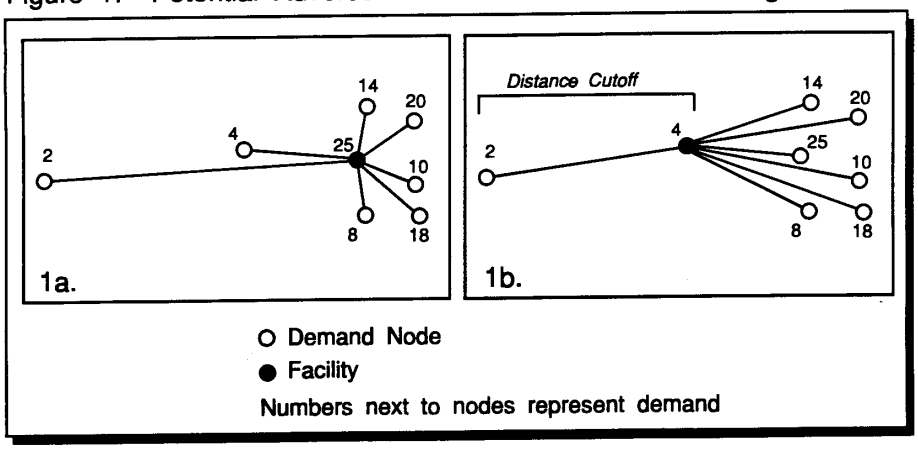
While distance strings may be used to increase the processing speed of some solution methods, they regrettably do so only at the cost of increased memory requirements. Whereas a distance matrix requires only distance information to be stored (the node indices are implicit in the position of the entry in the matrix), distance strings require both a distance measure and a node index for each distance entry in the distance string. In a typical computer architecture, this will result in a 50% increase in storage requirements. (This calculation assumes that distances are stored as floating point numbers and node indices are stored as integers. In the Intel 486 processor, for example, floating points require 4 bytes and integers require 2 bytes. Therefore, each entry in a distance string requires a total of 6 bytes, whereas each distance matrix entry requires only 4.)

In computer platforms with large memory resources available, such additional memory requirements may be of little concern, unless the problem data set consists of thousands of nodes. On computers with less available memory (such as many personal computers), though, additional memory usage can be problematic, particularly for large problems. In order to reduce the memory requirements of the distance strings data structure, several researchers (see, for example, Densham and Rushton [I 992a] and Hillsman [1980]) have proposed a strategy for cutting the distance strings in a manner such that for each node, distance information is only stored for other nodes in relatively close proximity. The rationale for this type of cut is that in many location-allocation models (such as the p- - Median), demands are unlikely to be served by facilities located at a great distance; therefore, greater internodal distances need not be represented in the data structure.

The attraction in using distance string cutoffs is readily apparent. As fewer nodes are stored in each string, the memory savings for the data structure continues to increase. The danger that must be considered, though, is that if too few distance entries are stored, the quality of location-allocation solutions generated by various solution methods may be degraded due to insufficient distance data. The reason for this is as follows. When distance strings are reduced or cut in length, all internodal distances not falling within the cut are generally assumed to be arbitrarily large numbers (Densham and Rushton [1992a]). This effectively prevents the assignment of any demand to a facility that is not included within its reduced distance string; the service distance for any such assignment would be calculated as an arbitrarily large number, and this would cause the resulting solution to be inferior to any in which all assignments fell within the partially cut distance strings. As a result, the optimal solution for a given problem may not be attainable simply because the optimal configuration may involve an assignment which lies beyond the distance cutoff for the demand and potential facility.

An example of this potential problem is shown in Figure 1. This figure represents a small p- Median problem in which only one facility is to be sited. The goal is to minimize the average weighted service distance. In Figure I a, no distance cutoff is imposed. The optimal single facility locates close to the cluster of high demand nodes, thereby minimizing the average weighted distance. In Figure 1b, a distance cutoff is imposed. The distance cutoff forces the facility to move to the candidate in the middle, in order that the relatively low demand node on the left can be served within the cutoff distance. This move, however, results in an increase in average weighted distance.

Figure 1: Potential Adverse Effects of Cut Distance Strings



While the problem illustrated in Figure 1 is for a very small data set, the same problem can occur in much larger data sets as well. Given the potential for such a problem to occur, a natural question to investigate concerns the appropriate level at which to cut distance strings for a given problem in order to maximize memory savings without adversely affecting solution quality. Without such guidelines, the use of distance string cutoffs in such location packages as LADSS (Densham [1992]) and ARC/Info (ESRI [1994]) may be ill- advised. In this paper a series of tests are presented which are designed to help suggest guidelines to address this issue. We also propose two alternative methods for determining distance string lengths, both of which appear to be superior to that used by others in the literature.

Determining the Length of Individual Distance Strings

When the distance strings data structure is cut in order to conserve memory, it is possible to cut the individual distance string for each candidate and/or facility to be the same length. It is also possible, however, to cut the individual strings at varying lengths. As an example, the method for cutting distance strings within LADSS and ARC/Info involves a maximal distance cutoff and results in strings of varying lengths. Considering such alternatives, a natural question to ask is as follows: if a given memory savings level is required, what is the best manner of determining the length of each of the individual distance strings in order to achieve the overall memory savings?

As an example, consider a 1000 node problem in which each node is both a demand and a potential facility. Suppose that sorted distance strings are to be used and that one wishes to cut the strings in a manner that requires only 25% of the memory for full distance strings. Given that each node is both a demand and a potential facility, there would be 1,000,000 entries in the full distance strings data structure: 1000 entries in each string for 1000 nodes. In order to achieve the desired level of memory savings, the cut-off distance strings can only include 250,000 of the 1,000,000 potential entries. In other words, each of the 1000 individual distance strings must hold on average only 250 entries.

Once the average desired size of the distance strings has been determined, the next question to address is the method of selecting the entries to be included in each individual distance string in order that the average distance string length goal is achieved. Barring additional constraints (constraints prohibiting, for example, the service of a given demand by a

given potential facility), it is assumed that if a particular distance string is to be of length *q*, then that distance string will hold the *q* closest entries to the base node. With this in mind, the problem to be solved becomes essentially this: choose *q* for each distance string such that the sum of all *q*'s divided by the total number of distance strings equals the average desired distance string length.

There are, of course, several approaches for solving this problem; three intuitively appealing alternatives are considered here. The first approach is to establish a fixed service distance. Any entry falling within the fixed service distance should be included in a distance string. Any entry greater than the service distance is to be discarded. This method was originally defined by Hillsman [1980] and subsequently used by Densham and Rushton [1992a]. With such an approach, the distance string length for different nodes will of course vary; isolated nodes will have relatively small distance strings, since fewer nodes will fall within the service distance, whereas nodes located in highly clustered areas will have longer distance strings. Because the lengths of different distance strings will vary, the challenge in this approach is to select the service distance in such a manner that the goal for the average string length is preserved. In practice, this may be accomplished through a trial and error selection of service distances or through a procedure that exhaustively checks all distances and continues to increase the service distance until the correct number of entries is included. It should be noted that for the p-Median model, the imposition of distance strings cut by a fixed service distance is equivalent to the p- Median with maximum service distance constraints as formulated by Khumawala [1975]. This approach has been implemented within the LADSS location-allocation package (Densham [1992]) as well as in the new location- allocation module of the widely- used ArcInfo Geographic Information System (ESRI [1994]). In subsequent testing, this approach will be referred to as the fixed service distance (FSD) approach.

The second and perhaps most obvious alternative is to fix the number of distance entries in each string to equal the average string length required by the total number of permissible entries. In the above 1000 node example, this would entail including within each distance string the 250 closest nodes and excluding 0 others. This approach is straightforward and easy to compute. One notable characteristic of this approach is that the maximum distance entry stored within each string will be different. Relatively isolated nodes will by nature include some entries that are located a great distance away, whereas nodes located in areas of high clustering may only include nearby entries. In subsequent testing, this approach will be referred to as the fixed string length (FSL) approach.

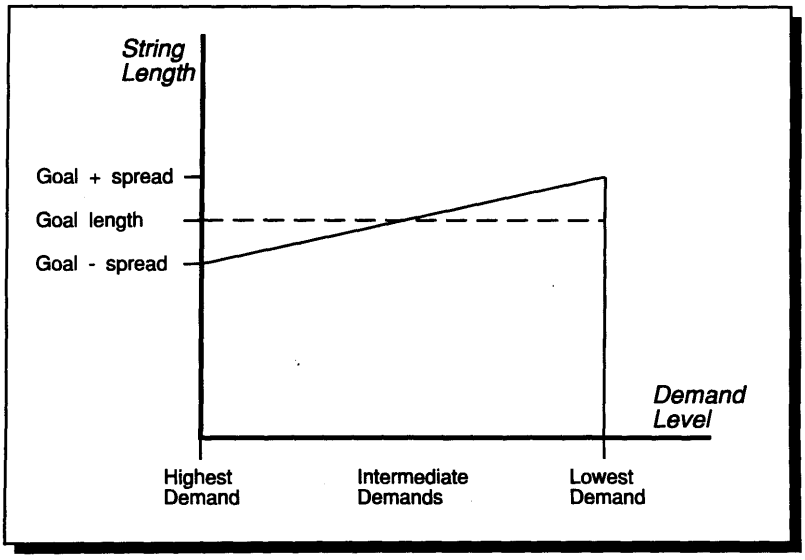
The third approach considered in this paper is termed the demand-varied string length (DVSL) approach. It is based on the following observation. In the p-Median model, due to the fact that the objective function is based on minimizing the average weighted service distance, facilities are likely to locate at or near nodes of high demand (thus reducing weighted service distance). For this reason, it would seem likely that the distance strings for nodes of high demand need not be as long as the distance strings for nodes of low demand. Under this approach, then, distance strings are set up so that high demand nodes have fewer entries than the goal average string length, and low demand nodes have more entries than the goal average string length.

Specifically, the demand-varied string length technique works as follows. Establish a goal length and a percent spread (for example, goal length is 25% of all possible entries and goal spread is 5%). Next, sort the nodes by decreasing demand. Then, assign the length of the distance string for each node, based on its position in the sorted demand list, so that the node with the

highest demand has not fewer than goal length minus goal spread percent of the potential entries (in the above example this would be 25% - 5% 20%) and the node with the lowest demand has not more than goal length plus goal spread percent of the potential entries (25% + 5% = 30%). The string length of all intermediate demand levels will vary accordingly between the longest string and the shortest string based on the relative demand. In implementing this approach, some care is required to ensure that the average distance string length works out to be the goal length. In all tests performed in this study, the spread has been set at 5%, a level which, from preliminary testing, appeared to provide the best results.

Figure 2 gives a graphical depiction of the DVSL approach. In Figure 2, it can be seen how the string length increases as the demand level decreases, while the average string length remains at the target level.

Figure 2: Demand-Varied String Length Approach



It should be noted that of the three alternatives to distance string cutoff methods discussed, only the fixed service distance (FSD) approach has been discussed or proposed in the literature (Hillsman [1980] and Densham and Rushton [1992a, 1992b]). The FSL and the DVSL methods were both developed as a part of the research reported here.

Using the P-Median Model for Testing the Viability of Reduced Distance Strings

The p-Median model serves as a representative location-allocation model for use in the testing performed in this research. The primary reason for the choice of the p-Median is based on the strong theoretical links that have been shown to exist between it and a host of other location-allocation models (Church [1974]; Hillsman [1984]). Essentially, through straightforward mathematical transformations, the p-Median can be converted into a large number of other specialized location-allocation problems.

As a method for generating solutions for the p-Median, the recently introduced interchange heuristic referred to as GRIA (the global-regional interchange algorithm, Densham and Rushton [1992b]) is employed. GRIA begins with a randomly selected set of facilities and seeks to improve the solution by iteratively swapping out marginal existing facilities and swapping in promising candidate facilities. Specifically, the GRIA process searches for two

distinct types of swaps: global swaps, in which the best global drop facility (that which would have the least adverse effect on the objective function if dropped from the current solution) can be replaced by any candidate facility; and regional swaps, in which any current facility can be replaced by a candidate facility within its service region. Essentially, at each stage GRIA evaluates a set of potential swaps, either global or regional, and performs the most beneficial swap identified (i.e., the swap within a service region or the swap involving the most marginal site that results in the greatest improvement in the objective function). Thus, the quality of the solution improves with each iteration until no further beneficial swaps can be identified, at which point the algorithm terminates.

GRIA was selected for this study because in test runs it has proven to be capable of finding consistently good p-Median solutions in rapid time. In numerous test comparisons with the Teitz and Bart [1968] heuristic, well known for its robustness in solving p-Median problems (see, for example, Rosing, Hillsman, and Rosing [19791), GRIA proved capable of generating solutions of comparable quality in substantially reduced time (Sorensen [1994]). However, as with all heuristics, including that of Teitz and Bart, GRIA does not always find the optimal solution. Furthermore, it may generate different solutions (i.e., different local optima) based on different starting configurations. For this reason, in each test performed in this study, GRIA is run with 10 different initial starting configurations in order to provide data on the average performance of the heuristic for a given cutoff level of the distance strings.

Testing Design

At this juncture, the basic issues to be addressed in this study have been outlined. First, to what degree can distance strings be cut in order to preserve memory while not adversely affecting solution quality? Second, if a given memory savings level is required, what is the most promising strategy for selecting the length of each individual distance string within the data structure in a manner designed to preserve the integrity of the remaining distance data: FSL, FSD, or DVSL? For the purpose of this study, a test has been devised which is capable of addressing both issues simultaneously. To facilitate discussion of the testing strategy, however, it is first necessary to introduce a new term: the matrix representation percentage (MRP). When distance strings are cut, some subset of the total distance information (that which is contained within the strings rather than that which is discarded) remains available for analysis. The term MRP is used to refer to the amount of available information within the reduced distance strings as a percentage of the total distance information which would appear in a full distance matrix. Thus, an MRP of 25% would designate that 25% of all distance information is contained within the distance strings and 75% has been cut.

Having introduced the MRP concept, it is now possible to discuss the details of the testing strategy. For a given problem (a specific data set with a specific number of facilities to be sited), begin by generating 10 random starting configurations (recall that GRIA may generate different solutions based on different starting configurations; thus it is beneficial to perform multiple trials for any given problem). Solve each of these starting configurations with full distance strings (in other words, solve the problems with all distance information available). Track the best, average, and worst solutions found. In addition, track a measure of the amount of work performed (in this case, the number of swaps or interchanges evaluated and performed) by GRIA for each of the trials. These initial results represent a benchmark measure of the best possible results.

Next, test the same problem with varying MRPs for each of the cutoff methods, FSD, FSL, and DVSL. Begin with FSD. Set the cutoff in such a manner that the MRP equals 10%. Solve each of the 10 random configurations, again tracking the best, average, and worst case solutions as well as the work measure. Increase the MRP to 15% and solve the 10 trials again. Continue to increase the MRP by 5% at a time, in each case solving all 10 trials and tracking all results. At each stage, compare the best, average, and worst case solution values along with the work value to the benchmark measures. For low MRP values, it may not be possible to find any feasible solutions (solutions in which all demands are allocated to facilities included in their distance strings). For slightly higher MRP values, the solutions may be feasible but relatively poor in comparison to the benchmark measures. At some point, however, the MRP will reach a level at which the solutions and work measures are equivalent to those of the benchmark with full distance strings. This MRP level represents the critical cutoff level at which the GRIA solution process is unaffected by the distance string cutoffs. This process is repeated for the FSL and the DVSL methods.

Critical Cutoff Levels

Before proceeding with the test results, it is necessary to comment further on the critical distance cutoff levels. For any of the cutoff strategies, as the MRP is increased, at some point it will reach a level at which the results of the tests with the cut distance strings will be exactly identical to those of the benchmark tests (i.e., those with full distance strings). When this point is reached, the best, average, and worst case objective values AND the average work value will be identical to the benchmark measures. At this MRP level, it is evident that the reduction in the distance strings does not at all alter the performance of GRIA. At each step of the heuristic, the exact same swaps are evaluated and performed; hence, the objective values as well as the average work measure are identical to those of the benchmark.

In many of the tests, however, the MRP level yielding results that mirror the benchmark measures is not the only cutoff level of concern. Often, in fact, the following situation occurs. As the MRP is increased, at some point a level is reached at which the best, average, and worst case results are as good, but not identical, to those of the benchmark. Alternatively, the objective results may be identical to the benchmark measures but the work value will differ.

When either of these two circumstances occur, it may be inferred that the current distance cutoffs have caused GRIA to alter the series of swaps performed but have still provided enough distance information to allow GRIA to find good or optimal solutions. In practice, this type of situation would occur if one of the swaps performed in the trials with full distance strings was prevented in the cutoff distance strings tests because it would have led, at least temporarily, to a solution in which one or more demand nodes was assigned to a facility not located within the cutoff. As a result, GRIA would opt for a different swap at that particular iteration. Even with making the alternate swap, however, GRIA could eventually find the same local optima or perhaps an alternate local optima that was as good as or better than the original.

As a result, there are two important MRP levels to track for each test. The first is that in which the solution quality of the 10 trial runs is at least as good as that of the benchmark runs. In other words:

Best objective with cutoff ≥ best benchmark objective, and
Average objective with cutoff ≥ average benchmark objective, and
Worst objective with cutoff ≥ worst benchmark objective.

This MRP level represents the optimal point at which memory savings is maximized without adversely affecting the quality of the solutions. The second critical MRP level occurs when the best, average, and worst objective values and the average work measure are identical to the benchmark measures. In other words:

Best objective with cutoff = best benchmark objective, and
Average objective with cutoff = average benchmark objective, and
Worst objective with cutoff = worst benchmark objective, and
Average work measure with cutoff = average benchmark work objective.

The second MRP level represents the point at which the actual performance of the algorithm (the step-by-step series of swap evaluations and performances) remains unaltered by the distance string cutoffs. In practice, both MRP level 1 and level 2 may occur at the same cutoff percentage. Many times, however, MRP level 1 will occur at a lower percentage than MRP level 2.

In addition to MRP 1 and 2, a third important MRP value will also be tracked throughout the tests. The third important MRP, designated MRP 0, represents the MRP level at which GRIA is first able to generate all feasible solutions. MRP 0 does not, of course, represent a good MRP level for generating solutions of high quality (in general, at MRP 0 all of the solutions, though feasible, will be of rather poor quality). MRP 0 is useful, however, in subsequent calculations aimed at estimating the appropriate MRP for various problems.

Before proceeding further, it should be noted that although the desired MRP levels to be tested are 10%, 15%, 20%, 25%, etc., in practice it is not always possible to establish cutoffs with each of the methods that exactly match these desired MRP goals. With the Swain data set with 55 nodes, for example, it is not possible to establish a fixed string length (FSL) cutoff that will store exactly 25% of the distance information. A fixed string length of 13 will yield a percentage of 23.64% (13/55 = .2364) while a fixed string length of 14 will yield a percentage of 25.45%. In cases such as this, the actual feasible MRP closest to the goal percent will be used (in the example, the FSL would be set at 14). In the results of the testing given later, the actual percents rather than the goal percents are listed.

Data Sets and Problems Tested

Before presenting the results of the testing, comments on the data sets used are necessary. To provide adequate testing for the various distance string cutoff strategies and levels, it is useful to use several data sets which vary in size as well as in spatial layout. (In this context, the term spatial layout refers to the relative distribution of demands within the problem - for example, are the nodes of highest demand clustered in a small area within the node set, or are they relatively spread out). To serve this goal, four data sets were selected: the Swain data set (see Swain [1971]), the reversed Swain data set, the London data set (see Goodchild and Noronha [1983]), and the Aztec data set (see Ruggles and Church [1994]). The Swain and reversed Swain data set contain 55 nodes. The London data set contains 150 nodes. The Aztec data set contains 366 nodes. The Swain data set was included as a representative small problem. The London data set was included as a representative medium problem. The Aztec data set was included as a

representative large problem. No very large problems (> 500 nodes) were included because of the excessive amounts of computer time required to test such problems for multiple runs. The reversed Swain data set was included to facilitate the goal of testing data sets with varying spatial layouts. In the Swain data set, most of the high demand nodes are clustered in the center of the region. In the reversed Swain data set, in contrast, most of the high demand nodes are scattered around the fringe of the region.

For each of these data sets, the following values of p (the number of facilities to be located) are tested:

Swain: p = 4, 5, 6, 7, 8, and 9
Reversed Swain: p = 4, 5, 6, 7, 8, and 9
London: p = 6, 9, 12, 15, 18, 21, and 24
Aztec: p = 10, 20, 30, and 40

Testing Results

The basic goal of the testing described above is to find, for each specific test (each value of p for each data set), the critical MRP levels for each of the three methods for cutting distance strings discussed earlier. Before summarizing the findings for all of the tests, it is useful to study in detail the results of a single test in order to see how the critical MRP levels are determined. Table 1 lists the results for p = 6 for the London data set.

In Table 1, the first column lists the method used for cutting the distance strings. The second column lists the actual MRP of the distance strings information stored within the cutoff strings. The third and fourth columns give the objective value and penalty value of the best solution out of the 10 trials. The penalty value represents the amount of demand that could not be served within the distance cutoff; when a penalty value is present, the objective value is not meaningful and therefore has not been listed. The fifth and sixth columns list the average objective and penalty value for the 10 trials. The seventh and eighth columns list the worst objective and penalty value found during the 10 trials. The ninth column gives the average work measure performed by GRIA during the course of the 10 trials. Recall that this measure is helpful in determining the second critical MRP level - that in which the GRIA algorithm is entirely unaffected by the restriction of the distance strings. The tenth column shows the points at which MRP 0, 1, and 2 are reached.

The base row in Table 1 shows the results of the 10 trials in the absence of distance cutoffs. It is against these values that the results of all subsequent tests in the table are compared. Following the base row, the next set of tests listed are for the fixed service distance (FSD) method of cutting distance strings. For the first four tests (10.00% - 25.07%), only infeasible solutions can be found. For 30.04% and 34.92%, some, but not all, of the solutions are infeasible; this can be noted by the average and worst case results. MRP 0 occurs at 39.96%, the first point at which all solutions are feasible. For 39.96% through 64.97%, progressively better feasible solutions are found, though they are still not as good as the base results. At 69.98%, the results are actually slightly better than the base results. Thus, 69.98% represents MRP level 1, the first MRP at which the results are at least as good as the base results'. Because some of the objective values along with the work measure are different from the base results, though, it is clear that MRP level 2 has still not been reached. MRP level 2, in fact, is not reached until 80.01 %, at which all of the objective values as well as the work value are identical to the base measures. From similar analysis of the fixed string length (FSL) method it can be seen that the critical MRP

levels are MRP 0 = 20.00%, MRP I = 35.33%, and MRP 2 = 45.33%. For the demand-varied string length (DVSL) method, MRP 0 = 25.00%, MRP I = 30.00% and MRP 2 = 45.00%.

Table 1: Sample Test Data for London Data Set, p = 6, Trials = 10

Cutoff Method	Actual MRP	Best		Average		Worst		Work Meas.	Cut Level
		Obj.	Pen.	Obj.	Pen.	Obj.	Pen.		
Base	100	293210	---	297001	---	303891	---	1328	
FSD	10.00	---	611	---	660	---	685	1275	
	14.98	---	306	---	353	---	405	1265	
	19.96	---	126	---	161	---	215	1293	
	25.07	---	27	---	55	---	88	1303	
	30.04	407788	---	---	9	---	23	1230	
	34.92	336135	---	---	1	---	4	1339	
	39.96	351169	---	372537	---	381046	---	1266	0
	45.08	301977	---	326649	---	384674	---	1356	
	50.11	293210	---	298574	---	316128	---	1439	
	55.03	293210	---	300271	---	307043	---	1426	
	60.02	293210	---	299820	---	306725	---	1255	
	64.97	293210	---	297222	---	309848	---	1304	
	69.98	293210	---	296565	---	303891	---	1305	1
	74.96	293210	---	296549	---	303891	---	1312	
	80.01	293210	---	297001	---	303891	---	1328	2
FSL	10.00	---	407	---	444	---	500	1401	
	15.33	---	12	---	49	---	114	1340	
	20.00	296085	---	306393	---	317804	---	1340	0
	24.67	293210	---	299351	---	306752	---	1297	
	30.00	293210	---	296926	---	306824	---	1373	
	35.33	293210	---	296565	---	303891	---	1350	1
	40.00	293210	---	297001	---	303891	---	1361	
	45.33	293210	---	297001	---	303891	---	1328	2
DVSL	10.00	---	669	---	673	---	682	1421	
	15.00	---	131	---	184	---	287	1246	
	20.00	299595	---	---	11	---	27	1260	
	25.00	293210	---	298429	---	311903	---	1342	0
	30.00	293210	---	296411	---	302116	---	1280	1
	35.00	293210	---	296565	---	303891	---	1350	
	40.00	293210	---	297001	---	303891	---	1361	
	45.00	293210	---	297001	---	303891	---	1328	2

Based on similar analyses of all the tests performed, a summary of the critical MRP levels has been compiled in Table 2. The first column in Table 2 lists the name of the data set tested. The second column lists the value of p for each specific test. The third, fourth, and fifth column list the MRP level 0 for each of the cutoff methods: fixed service distance (FSD), fixed string length (FSL), and demand-varied string length (DVSL). The sixth, seventh, and eighth

columns list the MRP level 1 for each of the methods. The ninth, tenth, and eleventh columns list the MRP level 2 for each of the methods.

Table 2: Comparison of Critical MRP Values for Cutoff Methods

Data Set	P	Critical MRP 0			Critical MRP 1			Critical MRP 2		
		FSD	FSL	DVSL	FSD	FSL	DVSL	FSD	FSL	DVSL
Swain	4	50.02	30.91	**24.99**	69.98	40.00	**35.01**	80.03	**50.91**	**50.02**
	5	40.36	**25.45**	**24.99**	75.01	45.45	**35.01**	75.01	**45.45**	**44.99**
	6	40.36	**20.00**	**20.00**	60.00	**34.55**	44.99	69.98	**40.00**	44.99
	7	25.36	**20.00**	**20.00**	60.00	**34.55**	**35.01**	69.98	40.00	**35.01**
	8	25.36	**20.00**	**20.00**	60.00	**34.55**	**35.01**	65.16	**34.55**	**35.01**
	9	19.80	**14.55**	**15.01**	60.00	30.91	**24.99**	65.16	**34.55**	**35.01**
Rev. Swain	4	50.02	**30.91**	**30.02**	75.01	**45.45**	**44.99**	90.02	**85.45**	**84.99**
	5	40.36	**25.45**	**24.99**	54.84	**45.45**	**44.99**	84.99	80.00	**75.01**
	6	29.98	**25.45**	**24.99**	**44.79**	60.00	55.01	80.03	**65.45**	**64.99**
	7	29.98	**20.00**	**20.00**	**44.79**	**45.45**	**44.99**	80.03	**65.45**	**64.99**
	8	25.36	**20.00**	**20.00**	44.79	45.45	**40.00**	65.16	60.00	**55.01**
	9	25.36	**14.55**	**15.01**	44.79	**30.91**	**30.02**	54.84	**50.91**	**50.02**
London	6	39.96	**20.00**	25.00	69.98	35.33	**30.00**	80.01	**45.33**	**45.00**
	9	30.04	**15.33**	20.00	55.03	**24.67**	**25.00**	64.97	**35.33**	**35.00**
	12	19.96	**15.33**	**15.00**	50.11	**24.67**	**25.00**	64.97	**24.67**	**25.00**
	15	19.96	**10.00**	**10.00**	50.11	20.00	**15.00**	55.03	**24.67**	**25.00**
	18	14.98	**10.00**	**10.00**	39.96	20.00	**15.00**	50.11	**20.00**	**20.00**
	21	14.98	**10.00**	**10.00**	45.08	**15.33**	**15.00**	45.08	**24.67**	**25.00**
	24	**10.00**	**10.00**	**10.00**	45.08	**10.00**	15.00	50.11	**15.33**	20.00
Aztec	10	20.00	**15.03**	**15.00**	40.00	**30.05**	**30.00**	60.00	**50.00**	**50.00**
	20	**10.00**	**10.11**	**10.00**	35.00	25.14	**20.00**	40.00	30.05	**25.00**
	30	**10.00**	**10.11**	**10.00**	35.00	30.05	**25.00**	35.00	30.05	**25.00**
	40	**10.00**	**10.11**	**10.00**	**20.00**	25.14	**20.00**	**20.00**	25.14	**20.00**

Note: Each problem solved with 10 trial starting configurations.

For each test, the lowest MRP level 0, level 1, and level 2 for all of the methods has been shown in bold. In cases of ties, each of the best methods are shown in bold. It should be noted that in determining ties, the goal percentages (though not shown) rather than the actual percentages are considered. For example, for MRP level 1 in the Swain data set with p = 7, the critical MRPs are 60.00% for FSD, 34.55% for FSL, and 35.01 % for DVSL. Because 34.55% and 35.01 % both correspond to a goal of 35%, FSL and DVSL are considered to tie for the best cutoff level I for this test.

Conclusions Drawn from the Testing

From examination of the results in Table 2, the demand-varied string length cutoff approach appears to be the most successful. The fixed string length method is also fairly robust. In contrast, with the exception of a few tests, the fixed service distance method performs rather poorly for all problems. Table 3 quantifies these results. Note that Table 3 deals explicitly with MRP level 1 and MRP level 2, the two MRP levels which are important in determining the

appropriate amount of information to include within reduced distance strings in order to maximize memory savings without adversely affecting solution quality.

Table 3: Summary of Cutoff Method Results

Critical MRP	Approach	Best	Tied	Lost	Greatest Deviation
1	FSD	1	2	20	40%
	FSL	2	10	11	15%
	DVSL	9	11	3	10%
2	FSD	0	1	22	40%
	FSL	2	15	6	5%
	DVSL	5	16	2	5%

The first column in Table 3 designates the MRP level to which the summary applies. The second column lists the cutoff approach used. The third column gives the number of times that the cutoff method gave the single lowest critical MRP value. The fourth column lists the number of times that the method tied for the best critical MRP value. The fifth column shows the number of times that the cutoff method did not yield the best results. Note that in each row, the sum of the values in columns three, four, and five is equal to 23, the total number of test problems. The sixth column lists the greatest difference between the critical MRP found by the method for a particular test and the best critical cutoff found by either of the other methods for the same test. The results in table 3 clearly indicate the inferiority of FSD in comparison to FSL and DVSL. They also show the slight but none-the-less existing edge that DVSL enjoys over FSL.

Based on these results, it may be argued that when distance strings are to be reduced, the most promising strategy to employ is either FSL or DVSL. While DVSL is likely to be the most effective, FSL is also quite reasonable and is easier to implement (it does not require a complicated algorithm to sort the nodes by demand and vary the string lengths according to the sorted order in a method that will preserve the overall desired cutoff level).

The first goal stated in this study was to establish guidelines for appropriate cutoff levels for distance strings: cutoffs that would maximize memory savings without adversely affecting solution quality. Regrettably, deriving such guidelines is a rather difficult task. Ideally, a simple formula based on the number of facilities to be sited relative to the total number of nodes could be derived which would be capable of giving consistently good results. After all, as the proportion of facilities to demands increases, the average service distance will decrease; hence, the critical MRP values should decrease as well. A close examination of the results in Table 2, however, indicates that such a simple formula may not exist. As one would expect, MRP level 0 decreases consistently as p increases for any given data set. MRP level 1 and level 2 do not, however, show such consistent results. For example, in the Aztec data set under the DVSL approach, the critical MRP level 1 for $p = 20$ is 20.00% while the critical cutoff level 1 for $p = 30$ is 25.00%. Intuitively, one would expect the opposite to occur.

To add to this inconsistency, the appropriate cutoff levels appear to vary based on the spatial distribution of the demands within each data set. This can be seen by contrasting the results for the Swain and the Reversed Swain data sets. In these two data sets, though the number as well as the location of the nodes is the same (only the demand levels at each node are

different), the critical MRP levels for tests with the same number of facilities to be sited vary by as much as 10% for MRP level 1 and by as much as 35% for MRP level 2.

Based on the observation of these inconsistencies, it is difficult to develop a general - purpose set of guidelines for appropriate distance cutoff levels. This is not to say, however, that distance cutoffs should be avoided. Clearly they can be very effective in reducing the required memory. In fact, out of all the tests, the largest MRP level 1 for FSL was 60.00%, and for DVSL it was 55.01 %. These MRP levels ranged as low as 10.00% for FSL and as low as 15.00% for DVSL. These numbers correspond to highly desirable levels of memory savings. They also represent computational savings associated with the generation and processing of a reduced set of distance information. In order to take advantage of the potential memory and computational savings without adversely affecting solution quality, though, it appears that preliminary testing will be required for any given data set and number of facilities in order to determine the appropriate MRP levels.

One of the primary reasons for reducing distance strings is to eliminate the need, due to memory requirements, of generating a full set of distance strings. Obviously, however, the tests outlined and performed earlier require the initial generation of full distance strings for the purpose of establishing benchmark measures. In using reduced distance strings for an untested dataset and value of p, though, it would be beneficial to be able to estimate the appropriate MRP without having to generate a full set of distance strings, particularly if the problem to be solved is very large. Two alternative methods through which the estimation of an acceptable MRP level may be obtained are presented here.

The first method works as follows. For a given data set and value of p, generate some number of trials. With whatever cutoff method is to be used, begin with a low MRP value and solve for all of the trials. Increase the MRP value and solve for all trials again. Continue to increase the MRP value until all trials yield feasible results - solutions in which all demands are served within the partial distance strings. In other words, establish the MRP level 0. Then, estimate what the difference between MRP level 0 and MRP level 1 should be. Add this difference to MRP level 0, and use the result as approrpiate MRP value to use in the tests.

How may the difference between MRP 0 and MRP1I be estimated? To provide insight into this question, Table 4 shows the difference between MRP level 1 and MRP level 0 for each of the 23 problems tested in this study. In Table 4, the first column lists the data set. The second column gives the value of p tested. The third column gives the difference between MRP 1 and MRP 0 for the FSD method. The fourth column lists the difference for the FSL method. The fifth column lists the difference for the DVSL method.

Table 4: Difference Between Critical MRP 0 and Critical MRP 1

Data Set	P	FSD	FSL	DVSL
Swain	4	19.96	9.09	10.02
	5	34.65	20.00	10.02
	6	19.64	14.55	24.99
	7	34.64	14.55	15.01
	8	34.64	14.55	15.01
	9	40.20	16.36	9.98
Rev. Swain	4	24.99	14.54	14.97
	5	14.48	20.00	20.00
	6	14.81	34.55	30.02
	7	14.81	25.45	24.99
	8	19.43	25.45	20.00
	9	19.43	16.36	15.01
London	6	30.02	15.33	5.00
	9	24.99	9.34	5.00
	12	30.15	9.34	10.00
	15	30.15	10.00	5.00
	18	24.98	10.00	5.00
	21	30.10	5.33	5.00
	24	35.08	0.00	5.00
Aztec	10	20.00	15.02	15.00
	20	25.00	15.03	10.00
	30	25.00	19.94	15.00
	40	10.00	15.03	10.00

From the results of Table 4, it is evident that there does not exist a simple formula that can accurately predict the difference between MRP 0 and MRP 1. For FSD, the difference varies between 10% and 40%. For FSL, the difference varies between 0% and 35%. For DVSL, the difference varies between 5% and 30%. Unfortunately, however, the differences do not vary consistently based on the value of *p*. Therefore, the safest approach would be to use a fixed estimate of the difference and add this to the established MRP level 0. Based on the results of these tests, the safe estimates would be 40% for FSD, 35% for FSL, and 30% for DVSL. These estimates will lead to the establishment of a new MRP that is at least as large as MRP level 1 in all of the cases tested, though in many instances the new MRP will be far greater than the actual MRP level 1.

The second method for establishing an appropriate MRP level is as follows. Again, beginning with a data set and a number of facilities to site, generate some number of test trials. With whatever cutoff method is to be used, establish a relatively low MRP level and solve for all trials, tracking the best, average, and worst results. Increase the MRP level by 5% and repeat all tests. Continue to increase the MRP level 5% at a time until the best, average, and worst case solution values are not any better than those at the prior 2 MRP levels. When this point occurs, the first of the prior 2 MRP levels can be used as an estimate of MRP level 1. (Note that the prior 2 MRP levels are both checked because at times the results may stay the same with the first 5% increase and then improve again with the second 5% increase.) This method appears to be quite successful based on an analysis of the data, similar to that in Table 1, used in the compilation of Table 2. Unlike the method for estimating the difference between MRP 0 and MRP 1, this method will never lead to a gross overestimate of the appropriate MRP for use with the data set. It does, however, require more test runs to establish the MRP value. It continues to run tests all the way until the estimated MRP level 1 is encountered, whereas the estimated difference method only runs tests until MRP level 0 is determined and then adds a fixed increment.

Summary

In implementations of location-allocation algorithms, the sorted distance string data structure may be cut- off in order to reduce the required memory. In performing such a reduction, the goal is to maximize memory savings and minimize computational time associated with generating and processing shortest distance information without adversely affecting solution quality. This paper has addressed two issues surrounding the use of the cut-off distance strings data structure. First, what is the best strategy for cutting individual strings within the overall distance string data structure in order achieve a given level of memory savings. Second, to what extent can distance strings be reduced without degrading the quality of solutions generated by the algorithm.

In relation to the first question, three different strategies were tested: the fixed service distance method (FSD) of Hillsman [1980] and Densham and Rushton [1992a], the fixed string length method (FSL) proposed in this paper, and the demand-varied string length method (DVSL) proposed in this paper. Of these, FSD was by far the least successful. In contrast, both FSL and DVSL performed rather well. DVSL did have a slight edge in performance over FSL. FSL, on the other hand, is much easier to implement. Based on these findings, if a string cutoff method is to be implemented, it is easy to recommend FSL or DVSL and not consider FSD.

In relation to the second question, the results of the testing suggested that it would be difficult to derive a general formula for determining the appropriate cutoff level for a given data set and number of facilities to be sited. Unfortunately, the critical MRP levels appear to vary inconsistently with the data set as well as with the number of facilities to be sited. As such, some care is required in using distance string cutoffs; preliminary testing should be performed for any given data set and number of facilities in order to determine the appropriate cutoff levels at which memory savings may be maximized without adversely affecting solution quality. Two viable methods for estimating the appropriate MRP levels without generating full distance strings have been presented.

As a final comment, all testing performed in this paper involved the p-Median location-allocation problem (selected for its theoretical similarity to many other location-allocation problems) and the GRIA heuristic. Distance string cutoffs may, of course, be used with other

algorithms and other location-allocation problems, though the results of such usage may not correspond exactly to the results reported in this paper. It is considered likely, however, that the results would be similar.

References

Church, R., 1974. "Synthesis of a Class of Public Facilities Location Models", PhD Thesis, Department of Geography and Environmental Engineering, The Johns Hopkins University, Baltimore, MD.

Densham, P. 1992. "Location Allocation Decision Support System", Software release S-92- 3, National Center for Geographic Information and Analysis, University of California, Santa Barbara, CA.

Densham, P. and G. Rushton, 1992(a). "Strategies for Solving Large Location-Allocation Problems by Heuristic Methods", *Environment and Planning A*, 24, 289-304.

Densham, P. and G. Rushton, 1992(b). "A More Efficient Heuristic for Solving Large P-Median Problems", *Papers in Regional Science*, 17(3), 307-329.

Environmental Systems Research Institute (ESRI), 1994. "ARC/Info Revision 7.0", Location-Allocation Module Documentation, Redlands, CA.

Erlenkotter, D, 1978. "A Dual Based Procedure for Uncapacitated Facility Location", *Operations Research*, 25, 992-1009.

Goodchild, M. and V. Noronha, 1983. "Location-Allocation for Small Computers", Monograph No. 8, Department of Geography, The University of Iowa, Iowa City.

Hakimi, S., 1964. "Optimum Locations of Switching Centers and the Absolute Centers and Medians of a Graph", *Operations Research*, 12(3), 450-459.

Hillsman, E., 1980. "Heuristic Solutions to Location- Allocation Problems: A User's Guide to Alloc IV, V, and VI", Monograph No. 7, Department of Geography, The University of Iowa, Iowa City.

Hillsman, E., 1984. "The P-Median Structure as a Unified Linear Model for Location-Allocation Analysis", *Environment and Planning A*, 16, 305- 318.

Khumawala, B., 1973. "An Efficient Algorithm for the P-Median Problem with Maximum Distance Constraints", *Geographical Analysis*, 5(4), 309-321.

ReVelle, C. and R. Swain, 1970. "Central Facilities Location", *Geographical Analysis*, 2(l), 30-42.

Rosing, K., E. Hillsman, and H. Rosing-Vogelaar, 1979. "The Robustness of Two Common Heuristics for the P-Median Problem", *Environment and Planning A*, 11, 373-380.

Ruggles, A. and R. Church, 1994. "An Analysis of Late Horizon Settlement Patterns in the Teotihuacan-Temascalapa Basins: A Location-Allocation and GIS Approach", To appear in an edited volume on Archaeology and GIS.

Sorensen, P., 1994. "Analysis and Design of Heuristics for the P-Median Location-Allocation Problem", Masters Thesis, Department of Geography, The University of California, Santa Barbara, CA.

Swain, R., 197 1. "A Decomposition Algorithm for a Class of Facility Location Problems", Ph.D. Thesis, Cornell University, Ithaca, NY, 1971.

Swain, R., 1974. "A Parametric Decomposition Approach for the Solution of Uncapacitated Location Problems", *Management Science,* 21(2), 189-198.

Teitz, M. and P. Bart, 1968. "Heuristic Methods for Estimating the Generalized Vertex Median of a Weighted Graph", *Operations Research*, 16(5), 955-961.

Weaver, J. and R. Church, 1983. "Computational Procedures for Location Problems on Stochastic Networks", *Transportation Science*, 17, 168-180.