Inf Technol Manag (2009) 10:55–65
DOI 10.1007/s10799-008-0046-x

Efficient heuristics for wireless network tower placement

Jason K. Deane · Terry R. Rakes · Loren Paul Rees

Published online: 8 January 2009

Abstract Over the last decade, telecommunications companies have invested nearly 100 billion dollars in the development of an impressive fiber optic backbone which is capable of transmitting data at incredible speeds. However, much of this backbone remains unused because of the data capacity bottleneck which exists at the user level. While various technologies have emerged to provide greater end-point capacity, many of these are of limited availability due to cost or technical considerations. One promising technology has been high-speed wireless service. Wireless service has the potential to provide widespread coverage, but the cost of developing the infrastructure such as antenna towers can be formidable. While models have been developed to assist in the planning for minimum cost tower placement, these models can be quite large and complex to solve. This paper explores alternate solution methods based on heuristic approaches which may allow for the solution of much larger tower placement problems.

Keywords Heuristics · Genetic algorithm · Wireless telecommunications · Location analysis

1 Introduction

Over the last decade, telecommunications companies have invested nearly 100 billion dollars in the development of an impressive fiber optic backbone capable of transmitting data at incredible speeds. However, the existence of this backbone, generally referred to as *broadband* because it employs a broad segment of the electromagnetic spectrum, has failed to change the way typical households and small businesses access network (e.g., Internet) services. The problem is that the connections from the backbone to the subscriber, generally referred to as “last mile” connections because of the relatively short distance involved, are often of significantly lower speed. Much of the current last mile infrastructure is owned by the telephone industry and cable TV industry and consists of copper cabling. While technologies like digital subscriber line (DSL) have maximized the potential of this copper cabling, this approach only provides about 1.5 megabits per second (mbps) and is not universally available (the subscriber must be within approximately 3.4 miles of a telephone switch). Cable service can provide up to about 5 mbps, still relatively slow in relation to industry need projections.

The terminology “last mile” is a generic term that can be slightly misleading. When the end-service domain was dominated by the telephone industry, it was called the “local loop.” As dedicated data providers came on line, the term last mile emerged because, in urban areas, most customers are located within a mile of a broadband source (switch). In rural or developing areas, however, this is commonly not the case as customers can, in many cases, be located many miles from the nearest broadband source. In these settings, as a result of the population sparcity, providing DSL or cable modem service is not a cost effective alternative (and even if one of these happens to be available, they suffer from the same speed limitations discussed above). One of the more promising broadband solutions that could potentially provide “last mile” service in these scenarios is wireless networking. Several technologies have been developed which can provide high-speed data

J. K. Deane (✉) · T. R. Rakes · L. P. Rees
Department of Business Information Technology, Pamplin College of Business, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, USA
e-mail: jason.deane@vt.edu

service over the last mile using radio waves. However, many issues, such as viable algorithms for determining the location of these radio towers, remain to be studied in order to determine the economic feasibility of wireless broadband for last mile connectivity.

While rural or developing areas, as described above, seem to be situations that could benefit greatly from the use of wireless transmission for last mile connectivity, they do not represent the only potential use for the technology/delivery strategy. Even in cases where customers are located within a mile of a broadband source, wireless tower transmission is still an important option. This is because replacing existing copper wire with high-speed fiber from a broadband switch to all homes within a large radius can be prohibitively expensive. Some studies in urban areas have shown the cost to run fiber to be in excess of $350 per meter (due to trenching, permits, landscaping, etc.). By erecting towers which can receive land-line signals from a backbone switch and then transmit them to customers, we could limit the expense and provide acceptable high-speed service at a fraction of the cost of hard media. In this case, we are limiting the fiber runs to those necessary to connect the towers to the backbone, and these costs are included in the tower cost. Thus, the last mile problem actually involves getting service from the backbone to the customer, whatever the distance may be.

While research has been done on the cell tower location problem, which is pertinent to data transmission for 3G mobile devices, this is a very different problem from the last mile problem addressed in this research. The cell tower location problem attempts to blanket a coverage area with a signal so that customers can move within the area. However, there are often “dead-zones” where signals cannot be obtained. Dedicated last mile solutions attempt to locate towers to provide point-to-multipoint, reliable, high-capacity signals from a fixed broadband backbone to fixed customer locations while constrained by line-of-sight (LOS) restrictions. 3G mobile connections will not suffice in this arena, even if signals could be reached by all customers, because 3G solutions are currently too slow. The goal of 3G is to provide 2 mbps when the receiver is stationary. While this is in the ballpark with DSL and cable, it is far below what the industry projects households and businesses will need to use evolving applications such as streaming audio and video. Fixed tower radio signals, with their wide spectrum, are capable of providing the higher bandwidth, but as we move to higher bandwidth the signals are unable to penetrate most objects, thus the LOS restriction. Finding solutions which allow economic tower placement given LOS restrictions is therefore very different from the cell tower location problem.

In a recent paper, Scheibe et al. [9] developed a spatial decision support system (SDSS) that allows the analysis of potential antennae locations for a last mile wireless service area. A grid of rectangular cells was used to overlay the service area. The center of each grid represents a potential antenna site, and the area within each grid represents a potential customer base for service. Because the available technologies are usually line-of-sight or near line-of-sight, information was necessary on ground topology along with antenna range in order to determine which cells (customers) could be reached by each antenna. The SDSS utilized information from a geographic information system (GIS) called GETWEBS and antenna specifications to identify the coverage provided by each antenna. A mathematical model was then used to determine the optimal location(s) for antennae subject to criteria such as mandated coverage or maximal profit.

The primary problem with the math programming based approach is model complexity. While the goal of the model is very similar to the set covering or facility location models (to make sure that we locate towers where all of the profitable customers will be covered by a signal), this problem is basically a fixed charge problem closely related to the fixed charge transportation problem (FCTP), which has been shown to be NP-hard in the literature. The sources of combinatorial complexity for this problem are easily identified. When a grid of reasonable size is placed over a large service area, the number of resulting cells can be extremely large. The United States Geological Survey produces Digital Elevation Models (DEMs) by digitizing cartographic map contour overlays or by scanning aerial photographs, and can achieve resolution of 30×30 m. Census data is available to determine numbers of households at this level of resolution, and greater accuracy of representation can be achieved with this level of detail. However, this fine granularity also increases the complexity of the underlying mathematical model. Exacerbating the complexity is the structure of the model. At the core of the model is a “coverage matrix” which treats each potential antenna location as a row and each potential customer as a column. Using the antenna range to determine reach and GETWEBS to determine line-of-sight visibility, a one in the matrix represents coverage and a zero represents no coverage. This can result in a very large matrix for which the necessary manipulation would tax the limits of most existing software, leaving intelligent problem reduction, or perhaps massively parallel solution algorithms, as the only viable alternatives. While GETWEBS has the capability to assist in intelligent aggregation of service areas, reduction of viable antenna locations is a significant problem.

The NP-hard nature of the proposed tower location problem makes it highly unlikely that it will ever be solved by an efficient optimal algorithm [4]; therefore, efficient and effective approximation algorithms are necessary. This paper explores three such alternative approximation solution techniques. All three approaches involve using a different representation scheme for the coverage matrix, which avoids storing all the "empty" (non-covered) cells of the sparse matrix resulting in tremendous data reduction. However, this alternate representation scheme is not amenable to solution by math programming techniques. The first proposed technique is a greedy heuristic approach which will provide near optimal solutions under certain circumstances and satisficing solutions when these conditions are violated. The second technique is a ratio heuristic which expands on the greedy heuristic by incorporating antenna cost data. The third approach utilizes an evolutionary or genetic algorithm (GA) approach to solve the modified problem.

The next section provides some background including a brief discussion of the referenced math programming model. The following sections present the heuristic approaches and the metaheuristic genetic algorithm approach. Finally, we describe the results of a comparative study of the heuristics and provide conclusions as to their relative performance.

2 Background

While several models for wireless tower placement have been suggested in the literature, we will use the model of Scheibe et al. [9] as it incorporates the geo-spatial and economic information which we believe is necessary to ascertain the economic viability of a wireless installation.

Rather than use an overlay grid, which may have numerous empty cells that do not contain either a potential customer or a potential antenna location, we will form a list of customers and antenna locations. Let $i = \{1,2,\ldots n\}$ represent the potential customers which require a signal, and $j = \{1,2,\ldots m\}$ be the potential tower locations. The wireless broadband fixed, point-to-multipoint (PMP) profit maximization model may be written as:

$$\text{Max } Z = \sum_i (R_i - CP_i)S_i - \sum_j CT_jX_j \tag{1a}$$

s.t.

$$\sum_j V_{ij}X_j \geq S_i \qquad \text{for each } \; i = 1, 2, \ldots n \tag{1b}$$

$$\sum_j CT_jX_j + \sum_i CP_iS_i \leq B \tag{1c}$$

where $X_j = 1$ if a tower is placed at location j, 0 otherwise; $S_i = 1$ if customer i receives at least one signal, 0 otherwise; $V_{ij} = 1$ if a tower at j is capable of providing a signal to customer i, 0 otherwise; R_i = monthly revenue from customer i; CP_i = amortized cost per month of customer i's premise equipment; CT_j = amortized cost per month of placing a tower at location j; B = monthly budget limit.

This model is basically a fixed charge model where S_i (the variable representing the signal to customer i) is the profit-making decision variable. But, in order to get a signal to customer i, we must incur the "fixed charge" of building at least one tower which has line-of-sight to the customer as indicated by V_{ij}. The fixed charge is triggered by X_j being forced to assume a value of 1. Of course, one tower can "see" many customers, so the model will examine combinations until it finds the set of tower locations which, in spite of their fixed cost, generate enough customer revenue to be most profitable.

While cost(s), potential tower locations, and customer availability are all major factors in the determination of where towers should be placed, two of the prime determinants are the coverage matrix V_{ij} and the budget, B. V_{ij} is called the "view-shed" matrix, and captures the signal coverage between towers and customers. Values for V_{ij} would be determined using the antenna range and terrain to determine line-of-sight feasibility. In Scheibe et al. [9], the visibility matrix was generated using the modified GET-WEBS software embedded within a spatial DSS. The budget may be specified by a for-profit firm as the maximum desirable expenditure, or by a governmental agency as the maximum subsidy. It is worth noting that in constraint 1c, because revenue exceeds CP_i, additional added customers will always bring in enough revenue to cover the cost of their equipment. However, what is included in a budget limit may not be based just on having enough funds to construct a tower or add a customer. Companies often have budget ceilings designed to reflect the amount they are willing to have invested in infrastructure, the total amount subject to maintenance, etc. We believe that a case could be made either way for including or not including CP_i in the budget limit, so we have included it for generality. In any particular implementation, leaving it out would not significantly change the model.

The basic model above assumes that degree of customer coverage is allowed to be whatever is necessary to maximize profit. In some cases, especially those where wireless service provision is part of a government funded project, coverage may be mandated. If we wanted to stipulate that everyone must be covered, then we would relax the budget restriction in Eq. 1c and replace Eq. 1b with:

$$\sum_j V_{ij}X_j \geq 1 \qquad \text{for each } i = 1, 2, \ldots n \tag{1d}$$

If we desired a certain percentage of coverage, then in lieu of Eq. 1d we could retain Eq. 1b in the model and add:

$$\sum_i S_i \geq SP \times n \tag{1e}$$

where SP is the percentage of customers that must be covered.

As an illustration, consider the hypothetical customer market shown in Fig. 1 representing 30 potential customers and six possible antenna locations.

The circles of coverage are governed by antenna range, and this range plus the areas of topographical relief (which may interfere with line-of-sight) combine to define the visibility matrix *V*. Note that antenna location B can provide a full radius of coverage because it would be located on top of a hill, while the coverage of potential antenna A would be blocked by that same high terrain. Figure 2 is the visibility matrix for this example.

Suppose we use a customer profit level of $80 per month for each customer and a tower cost of $500 per month for each tower (this would be the amortized cost of the tower over its life, plus any variable operating costs). Ignoring for now the budget constraint and assuming that customer coverage is not mandatory but governed by profitability, Solver would yield two alternate optimal solutions [3]. The first would be to place towers in locations a and c at a net profit of $360 per month. Customers four through 13, 19, 21, 22, and 25 through 28 would be covered by this solution. The second is to place towers in locations b and c at, again, a profit of $360. Customers 1 through 10, 19, 21, 22, and 25 through 28 would be covered by this solution.

There are two important issues to note from this example. First, the major source of problem complexity comes from the data storage requirement for the visibility matrix. As noted earlier, for a realistic problem, this matrix could necessitate storage of hundreds of thousands, or even millions, of cells. And yet in this example the matrix is relatively sparse with only 25% of the cells containing a non-zero value. As the problem size increases, the sparseness of the matrix will also increase.

Second, the factor that necessitates using a linear programming approach for this problem is the matching of unique combinations of profits/cost with the decision variables. In the next section, we will look at a heuristic which exploits both of these facts.

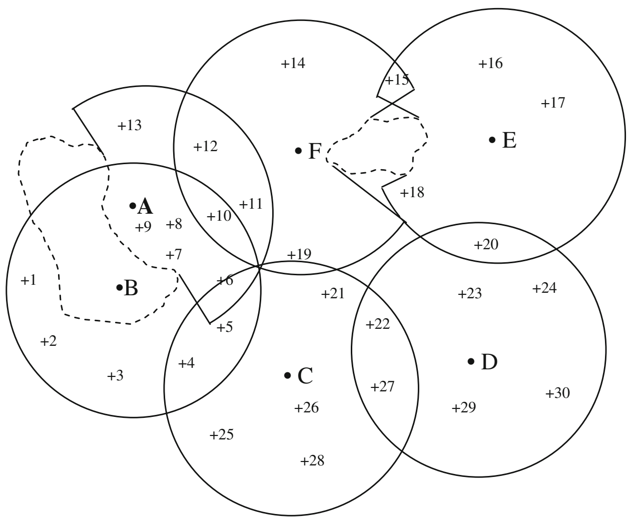

Fig. 1 Hypothetical customer market

	Towers					
Customers	a	b	c	d	e	f
1	0	1	0	0	0	0
2	0	1	0	0	0	0
3	0	1	0	0	0	0
4	0	1	1	0	0	0
5	0	1	1	0	0	0
6	1	1	1	0	0	0
7	1	1	0	0	0	0
8	1	1	0	0	0	0
9	1	1	0	0	0	0
10	1	1	0	0	0	1
11	1	0	0	0	0	1
12	1	0	0	0	0	1
13	1	0	0	0	0	0
14	0	0	0	0	0	1
15	0	0	0	0	1	1
16	0	0	0	0	1	0
17	0	0	0	0	1	0
18	0	0	0	0	1	0
19	0	0	1	0	0	1
20	0	0	0	1	1	0
21	0	0	1	0	0	0
22	0	0	1	1	0	0
23	0	0	0	1	0	0
24	0	0	0	1	0	0
25	0	0	1	0	0	0
26	0	0	1	0	0	0
27	0	0	1	1	0	0
28	0	0	1	0	0	0
29	0	0	0	1	0	0
30	0	0	0	1	0	0

Fig. 2 View-Shed (visibility matrix) for the hypothetical customer market

3 Tower location heuristics

Math programming models have the ability to examine the change gradient for new variable combinations brought about by unique variable profits or costs for each decision variable. If profits and costs are relatively stable for the customers and towers, there may be easier ways to derive a solution. This is the case for the tower location problem. To address the solution process, we present a basic greedy heuristic.

3.1 The basic greedy heuristic

To deal with the sparse matrix issue, we switch to a set approach within which we store, for each tower, only the list of customers it can "see." This avoids storing the numerous 0 values for non-LOS tower/customer combinations as in Fig. 2. Let VS*j* be the set of visible customers for each tower *j* and *Nj* be the number of customers the tower can see (the element count of VS*j*). Further, let AP be the average monthly profit provided by each customer, AC*pe* be the average monthly cost of providing customer premise equipment, AC*t* be the average monthly cost of a tower, and B the monthly budget limit. The solution could be derived by the following set of steps:

1. Do until all towers have been checked
   a. Choose the tower with the largest value for *Nj* (the tower that will "cover" the most customers). If two sets/towers are tied for the largest *N*, choose the set with the most unique customers (the set with the most customers not duplicated by any other remaining set not under consideration). If there is a tie for the most unique customers, choose a tower arbitrarily. If AP · *Nj* − AC*t* › 0, proceed to step b. Otherwise, stop as all remaining towers with an equal or lower *Nj* will also be unprofitable.
   b. If *Nj* · AC*pe* + AC*t* ≤ B, add the tower *j* to the list of selected towers and proceed to step c; otherwise, return to step 1.
   c. Update the visibility sets for all remaining towers by removing all occurrences of the covered customers. Decrease B by *Nj* · AC*pe* + AC*t* and store the new budget limit back in B.
2. Next tower (return to step 1).

This heuristic does not incorporate the possible stipulation that all customers must be covered. If this were in effect, rule 1 would need to be modified to proceed with the tower which has the largest *Nj* whether or not it is profitable.

For the example problem presented earlier, the visibility sets would be:

VS*a* = {6,7,8,9,10,11,12,13}
VS*b* = {1,2,3,4,5,6,7,8,9,10}
VS*c* = {4,5,6,19,21,22,25,26,27,28}
VS*d* = {20,22,23,24,27,29,30}
VS*e* = {15,16,17,18,20}
VS*f* = {10,11,12,14,15,19}

In step 1, sets VS*b* and VS*c* are tied with *Nb* = *Nc* = 10. However, set VS*c* has no more than two values duplicated within any other set while VS*b* has five customers in common with set VS*a*. Therefore, we would choose set VS*c*. Using the same figures as before (average customer profit level = $80 and average tower cost = $500), $80(10)–$500 = $300 which is greater than zero. We would proceed to step 2 and given the infinite budget limit assumed previously, we would select tower c. Next, we would update all remaining VS*j* by removing covered customers, resulting in:

VS*a* = {7,8,9,10,11,12,13}
VS*b* = {1,2,3,7,8,9,10}
VS*d* = {20,23,24,29,30}
VS*e* = {15,16,17,18,20}
VS*f* = {10,11,12,14,15,19}

and return to step 1. Now, VS*a* and VS*b* are tied with *Na* = *Nb* = 7. VS*b* has the least number of duplicates, so we would choose VS*b* and move to step 2. $80(7)–$500 = $60, so we would add tower b to our list. Updating the remaining visibility sets results in:

VS*a* = {11,12,13}
VS*d* = {20,23,24,29,30}
VS*e* = {15,16,17,18,20}
VS*f* = {11,12,14,15,19}

Returning to step 1, the VS*d*, VS*e*, and VS*f* are all tied with five customers. Sets VS*d* and VS*e* have no duplicates, so we would select one arbitrarily (we selected set VS*d)* and move to step 2. Now, $80(5)–$500 = −$100. Since, in this example, all of the remaining towers/sets have five or fewer potential new customers, even with an unlimited budget none of them would result in positive revenue for the firm; therefore, we would stop. Our final solution is towers b and c with a total profit of $300 + $60 = $360. This matches one of the two optimal solutions derived from the linear programming formulation.

In fact, assuming that we charge a uniform rate to our customers, if the cost of each customer's premise equipment is the same, and the cost of each tower is the same, then the average cost will, of course, be equal to the actual cost. In this situation, the heuristic will perform quite well.

While the rate we charge our customers will normally be uniform and the customer equipment will be the same for each customer resulting in uniform cost, it is not likely that the costs of placing towers will always be identical. Factors such as tower size (taller towers are commonly more expensive), amortized land costs and site preparation, land lease rates, electrical supply costs, etc. will likely affect the monthly cost of each tower differently. As costs become more variable, it seems logical that a heuristic which directly incorporates cost would be superior. In the next section, we present a ratio heuristic which reflects costs.

3.2 Ratio heuristic

Ratio heuristics have been commonly applied to selection problems such as the one presented here. Kochenberger et al. [7] proposed a greedy heuristic based on a pseudo-utility ratio where items with the highest ratio were added to a knapsack until the knapsack is full. Alternatively, Senju and Toyda [10] presented a heuristic where the knapsack was populated with all available items and then removed one at a time based on their ratio until feasibility was reached. We have developed a similar heuristic based on the ratio of profit of a potential tower (as determined by customer coverage count) to the cost of the tower. Once potential towers are ranked based on this ratio, a tower is chosen, the ratios are updated, and the process is repeated until the budget limit is reached.

Using the notation from the first heuristic, CT*j* as the cost of tower *j*, and *Rj* to stand for the ratio of profit to tower cost for tower *j*, the ratio heuristic can be stated as follows:

1. For each available tower:
a. Calculate *Nj* (the number of customers the tower will cover) for all available towers.
b. Calculate the ratio of profit to tower cost for each tower *j* as $Rj = (Nj * \mathrm{AP})/\mathrm{CT}j$. Select the tower with the largest ratio. If there is a tie for the largest ratio, choose a tower arbitrarily from among the tied towers. For the selected tower *j*, if $\mathrm{AP} * Nj - \mathrm{CT}j > 0$ proceed to step c. Otherwise, remove this tower from the list of available towers and return to step 1 (its unique customer coverage can only go down as other towers are assigned, so if it is not profitable now it never will be).
c. If $Nj * \mathrm{AC}pe + \mathrm{CT}j \leq \mathrm{B}$, add tower *j* to the list of selected towers, remove tower *j* from the list of available towers, and proceed to step d. Otherwise, return to step 1.
d. Update the visibility sets for all remaining (available) towers by removing all occurrences of the covered customers. Decrease B by $Nj * \mathrm{AC}pe + \mathrm{CT}j$ and store the new budget limit back in B.
2. Next tower (return to step 1).

While consideration of specific tower costs is reasonable, it does not always guarantee dominance over the basic greedy heuristic. Consider the case where the number of towers (or the budget) is low resulting in a small solution space. Choosing the tower with the best ratio may use just enough of the budget to prevent any further choices, whereas choice based entirely on the coverage count might have resulted in a selection which had a higher coverage still within the budget, resulting in better overall coverage and therefore a higher profit. Clearly, as the size of the solution space increases, we would expect the ratio heuristic performance to dominate the simple heuristic.

Along with an increase in the size of the solution space comes a greater possibility that a local optima will be overlooked. In these cases, a guided search technique may be superior to either of the two heuristics presented thus far. In the next section, we present a genetic algorithm (GA) metaheuristic for tower placement.

3.3 A GA metaheuristic for tower placement

GAs were first suggested by Holland [5], and have since been applied to numerous problem domains, including optimization. While there are numerous design issues in GA implementation, a first and critical decision is that of a representation scheme. In GA, artificial strings called chromosomes (containing members called genes) are used to represent populations of possible solution values. Through the operations of reproduction, crossover, and mutation, these chromosomes will combine and persist based on some fitness value. While optimality is not assured, GAs have proven to provide very good solutions if allowed to evolve through sufficient generations.

In the location literature, both Aytug and Saydam [2] and Jaramillo et al. [6] utilized GAs to solve location problems. The interested reader can consult those sources for an in-depth discussion of different representation possibilities. We will adopt the straightforward approach of letting binary values for each gene represent the choice to locate or not locate a potential tower. Thus, let T be the chromosome representing tower decisions where each gene T*j* assumes a binary value representing a tower decision for tower *j*. Further, let P*i* be the profit of customer *i*, C*j* be the cost associated with tower *j*, and VS*j* represent the visibility set for tower *j* (we will once again use the set approach to avoid storage of the large, sparse matrix). The fitness function may then be written as:

$$\mathrm{Max} \sum_i \mathrm{P}i\ \mathrm{Z}i - \sum_j \mathrm{C}j\ \mathrm{T}j.$$

where

$$\mathrm{Z}i = \{1 \text{ if } i \ \varepsilon\ \mathrm{VS}j \text{ and } \mathrm{T}j = 1 \text{ for at least one } j;\ 0 \text{ otherwise}\}$$

Using our earlier example customer market, our chromosome would be $\mathrm{T} = (\mathrm{T}_1,\mathrm{T}_2,\mathrm{T}_3,\mathrm{T}_4,\mathrm{T}_5,\mathrm{T}_6)$ for the six possible tower locations.

GA Notation
e: elite percentage
p_m: probability of mutation
ps: population size
NU: number of desired unique solutions
CL: crossover attempt limit

The GA begins with an initial population of strings which are all created randomly with the exception of the two strings which are created using the heuristics. The random strings are created by randomly selecting tower placements one at a time until all possible placements have been attempted. If a selected tower placement does not violate the budget constraint, it is selected and the resulting bit is set to 1. Otherwise, it is not selected and the bit is set to 0. Between generations the elite percentage (e) is utilized to determine how many of the most fit strings will survive unchanged into the next generation. The roulette wheel crossover mechanism selects potential reproductive parental strings based on their relative fitness values. Each string has a probability of selection which is directly proportional to the ratio of its fitness value divided by the sum of the fitness values of the entire population. The fittest strings are thereby given the highest probability of selection. Having selected two parent strings via the roulette wheel process described above, a single crossover point is randomly selected. In the example depicted in Fig. 3, point number two, which falls between towers two and three, was selected. The genetic material on the left side of the crossover point in the first parent is then directly inherited by child 1 and similarly for the second parent and child 2. In our example (see Fig. 4), the first set of tower placements which are inherited by child 1 are towers 1 and 2. Based on the chosen crossover point and the genetic material of the parents, two children strings are created. This reproduction process has created two new offspring for the next generation. However, before being added into the next population, the new offspring are given an opportunity to mutate based on the pre-defined probability of mutation operator (p_m). A string which is selected for mutation will have the binary selection bit for a randomly selected tower placement bit swapped from 0 to 1 or from 1 to 0. In the example below (see Figs. 5 and 6), it is assumed that the second child has been selected for mutation and tower six has been randomly selected as a mutation candidate. After mutation, the strings that pass the feasibility test are added to the next population. Those that do not are discarded. This entire process in repeated from generation to generation until a predefined number of unique solutions have been created or the crossover attempt limit has been exceeded.

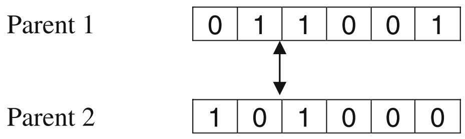

Fig. 3 Selected parents prior to crossover

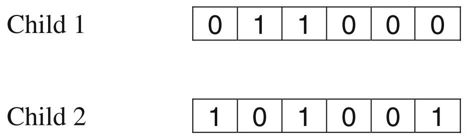

Fig. 4 Resulting offspring

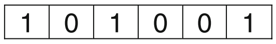

Fig. 5 Child 2 prior to mutation

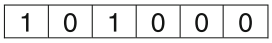

Fig. 6 Child 2 after mutation

GA—Algorithm
Step 1: Initialize
Step 2: Apply each heuristic and insert the resulting solutions as the first strings in the initial GA population
Step 3: Fill out the initial population by creating ($ps - 2$) random chromosomes. A random chromosome is created by selecting tower placements randomly until all placement possibilities have been considered. Those tower placements that would violate the budget constraint are not selected.
Step 4: Set t = 1 and c = 0.
Step 5: Sort the strings in descending order of their relative fitness values
Step 6: Populate the elite list by selecting the best ($e * ps$) strings based on their relative fitness values. These strings are added to the next population.
Step 7: Utilizing the roulette wheel selection method, select two parent strings for reproduction and cross them over. Set c = c + 1.
Step 8: Mutate the resulting children based on the mutation probability.
Step 9: Add the children strings to the next population ad test them for uniqueness. For each child that is unique, set t = t + 1.
Step 10: If t $\geq$ NU or c $\geq$ CL, calculate the fitness value for those strings in the new population and terminate reporting the best solution so far.
Step 11: If the number of chromosomes in the next population $\geq ps$, goto step 5; otherwise, goto step 8.

3.3.1 GA parameter selection

As discussed by Aytug et al. [1], one of the biggest concerns with respect to the application of these 'black box' type algorithms, such as neural networks and genetic algorithms, is the absence of theoretical guidance with respect to the methods by which the parameter settings should be selected. A few researchers have attempted to construct systems to automatically determine optimal GA parameter values, such as the fuzzy knowledge-based system developed by Laoufi et al. [8] for an economical

Table 1 Design of experiment—GA parameter values

Factors	
Population size	15–30–50
Elite list (%)	15–25–40
Mutation probability (%)	0.5–1–5

power dispatch solution for electric power generation. However, these systems have not been proven to be generalizable to other domains. We used the accepted method of performing numerous pilot runs with different settings for the elitism and mutation rates. To enhance our ability to evaluate the performance of the genetic algorithm, we conducted a three level full factorial design. The parameter levels, as detailed in Table 1, represent parameter ranges which performed well during the numerous pilot runs. The GA results presented in Table 5 represent the average GA performance across all possible parameter combinations as described in the design of experiments for each individual problem set.

4 Problem set development

As long as the variance in tower costs is relatively low, we are very optimistic that the proposed greedy heuristic will perform well. With an increase in problem size and/or cost variance, the ratio heuristic should perform better than the basic heuristic. However, as the variance and/or problem size increases even further, we anticipate that, as a result of the increased complexity of the solution space, the GA with its guided random search capabilities should emerge as the best heuristic. This last point is no surprise, as it is well established that metaheuristics such as GA should perform better than plain heuristics. We are simply trying to establish the problem characteristics under which the GA dominance will become evident. This leads us to two basic research questions: (1) as the variance in tower costs increases, how will the performance of the greedy heuristic and the ratio heuristic compare, and (2) for larger problem sizes and solution spaces, will the GA dominate both of the simple step-wise selection heuristics as expected?

To test the effectiveness of the two proposed heuristics and the GA metaheuristic, we developed three problem sets of different sizes and complexities. Each problem set, which was created by hand to ensure that the view shed matrices for each tower location were accurate so as not to compromise the legitimacy of the test, consists of 90 problems as detailed below in Tables 2, 3, and 4, resulting in 270 total problem sets. While more data is always better, the need to create the visibility matrix manually in order to assure the integrity of the problem limited the number of possible problem sets. We are confident that 270 problem sets provide an adequate cross-comparison of the heuristics' performance.

For each problem set, the number of potential tower locations varies from 12 to 18 and the number of potential customers varies from 30 to 120. The tower location budget and the tower costs were all generated randomly. The budget varies uniformly between $2,000 and $3,000 for each problem. Towers of two separate sizes are considered. The variance level of the uniform distributions from which the tower placement costs were selected increases from problem set 1 (Low Variance—LV) to problem set 2 (MV) to problem set 3 (HV) and the associated expense levels increases within each data set from the small to the medium to the high problems. In problem set 1 (Table 2), it is assumed that the placement costs for the larger towers are strictly more than those for the smaller towers. Although this may be the case in some situations, as a result of the plethora of placement factors such as site preparation, terrain inclination, availability of necessary power, etc., there are certainly many situations where these costs may overlap. In an effort to model this scenario, we developed problem sets 2 and 3. In problem set 2 (Table 3), we do not force the strict ordering of placement costs which increases the variance substantially. In problem set 3 (Table 4), the tower placement costs are selected from the same distribution for the small and large towers (thus, the single column for tower costs in that table). This increases the problem variance even further. Based on our research questions, we expect the greedy heuristic to perform well relative to the others in problem set 1, then falter as we move to problem sets 2 and 3. Likewise, as we move to problem set 3, we expect the GA to be dominant.

5 Experimental results

The experimental results are reported in Table 5. As we expected, the basic heuristic often beat the ratio heuristic when variance of tower costs were low. In one case the basic heuristic was 124.77% better than the ratio heuristic. However, as we move to the highest level of variance in tower cost, the ratio heuristic clearly dominates the basic heuristic. Across the three problem sets and three problem sizes looking at the high variance level, in eight of the nine cases the ratio heuristic clearly outperforms the basic heuristic. In the one case where the basic heuristic was better, it was only better by 1.51%.

We also postulated that for larger problem sizes and solution spaces, the GA will dominate both of the simple step-wise selection heuristics. While the dominance of the GA metaheuristic is not surprising, the degree of dominance in our results were even stronger than we expected.

Table 2 Low variance (LV)

Problems	Number of prob.	Number of potential towers	Number of potential customers	Budget	TC low	TC high	Customers profit
12–30–L	10	12	30	2000–3000	200–400	400–600	70
12–30–M	10	12	30	2000–3000	400–600	600–800	70
12–30–H	10	12	30	2000–3000	600–800	800–1000	70
14–60–L	10	14	60	2000–3000	200–400	400–600	70
14–60–M	10	14	60	2000–3000	400–600	600–800	70
14–60–H	10	14	60	2000–3000	600–800	800–1000	70
18–120–L	10	18	120	2000–3000	200–400	400–600	70
18–120–M	10	18	120	2000–3000	400–600	600–800	70
18–120–H	10	18	120	2000–3000	600–800	800–1000	70

Table 3 Medium variance (MV)

Problems	Number of prob.	Number of potential towers	Number of potential customers	Budget	TC low	TC high	Customers profit
12–30–L	10	12	30	2000–3000	100–400	200–500	70
12–30–M	10	12	30	2000–3000	200–600	400–800	70
12–30–H	10	12	30	2000–3000	300–800	500–1000	70
14–60–L	10	14	60	2000–3000	100–400	200–500	70
14–60–M	10	14	60	2000–3000	200–600	400–800	70
14–60–H	10	14	60	2000–3000	300–800	500–1000	70
18–120–L	10	18	120	2000–3000	100–400	200–500	70
18–120–M	10	18	120	2000–3000	200–600	400–800	70
18–120–H	10	18	120	2000–3000	300–800	500–1000	70

Table 4 High variance (HV)

Problems	Number of prob.	Number of potential towers	Number of potential customers	Budget	Tower cost	Customers profit
12–30–L	10	12	30	2000–3000	100–500	70
12–30–M	10	12	30	2000–3000	200–800	70
12–30–H	10	12	30	2000–3000	300–1000	70
14–60–L	10	14	60	2000–3000	100–500	70
14–60–M	10	14	60	2000–3000	200–800	70
14–60–H	10	14	60	2000–3000	300–1000	70
18–120–L	10	18	120	2000–3000	100–500	70
18–120–M	10	18	120	2000–3000	200–800	70
18–120–H	10	18	120	2000–3000	300–1000	70

The GA metaheuristic clearly dominates both of the two selection heuristics across all variance levels, problem sizes, and price structures. This strong performance by the GA clearly shows its potential as a viable heuristic for wireless tower location. The only drawback to the GA solution is the longer computational time. Table 6 shows the average execution time for problems within each problem set for the GA (both of the simple heuristics took less than 1 s for every problem set). While the execution times for the GA are obviously much longer, the longest execution time experienced, which was for the largest problem size of 18 towers and 120 customers, was only

Table 5 Experimental results—avg % improvement of GA and ratio heuristic against the basic greedy heuristic

Variance level	Problem size					
	12–30		14–60		18–120	
	Ratio Hr (%)	GA (%)	Ratio Hr (%)	GA (%)	Ratio Hr (%)	GA (%)
Low price						
LV	13.77	22.27	−1.30	1.81	−3.83	1.43
MV	21.72	24.38	−1.58	0.68	3.39	5.75
HV	23.00	24.09	−1.51	1.28	7.95	9.57
Med price						
LV	−18.66	5.08	−5.02	0.00	−2.00	0.00
MV	14.09	25.20	−4.47	2.98	−0.74	5.47
HV	26.74	28.81	0.84	4.28	8.67	13.34
High price						
LV	−124.77	4.53	−56.25	0.00	−1.52	4.25
MV	−17.63	24.91	3.87	6.14	−13.13	1.01
HV	30.98	32.37	7.70	8.07	3.18	9.81

Table 6 Avg CPU execution time (in seconds) for the GA

Variance level	Problem size		
	12–30	14–60	18–120
Low price			
LV	8.58	10.40	13.54
MV	8.14	6.57	9.75
HV	8.42	7.33	11.13
Med price			
LV	1.83	0.30	11.28
MV	1.90	3.75	22.56
HV	2.82	5.86	28.38
High price			
LV	1.38	1.56	2.08
MV	0.67	0.43	9.36
HV	1.02	1.02	13.50

Note: *Avg CPU time (in seconds) for the GA*: Running VB.net, Windows XP, 2 GHz processor, 1 GB RAM. Both Heuristics ran each problem in <1 s

28.38 CPU seconds. This is encouraging, especially given that this problem is a design problem which will be run periodically but not in demand to real-time decision situations. Thus, longer execution times can be tolerated.

6 Conclusions

In this research, we have explored improved solution possibilities for the last mile connectivity problem. While math programming has been shown to be a viable solution approach for this problem, the computational aspects of the IP solution methodology are formidable.

We have explored three heuristics which can provide alternative solution approaches to mathematical programming for the last mile wireless tower location problem. Given the NP-hard nature of the problem, alternatives which avoid direct solution of the problem are a necessity as the size of the problem expands. Performance of the two sequential selection heuristics, basic greedy and ratio, were mixed with each showing promise at times. However, as variance in tower costs increases, the ratio heuristic will begin to dominate. For the GA search heuristic, performance was superior across all data sets, cost variances, and problem sizes. While execution times for the GA search are longer than for the other two heuristics, they are reasonable and tolerable given the mode in which the problem will be solved.

References

1. H. Aytug, M. Khouja, F.E. Vergara, Use of genetic algorithms to solve production and operations management problems: a review. Int. J. Prod. Res. **41**(17), 3955–4099 (2003)
2. H. Aytug, C. Saydam, Solving large-scale maximum expected covering location problems by genetic algorithms: a comparative study. Eur. J. Oper. Res. **141**, 480–494 (2002)
3. Frontline. (Frontline's Premium Solver^{TM}, Frontline Systems, Inc., Incline Village, NV), http://www.solver.com, Accessed 20 March 2008
4. M. Garey, D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (W·H. Freeman and Co., New York, NY, 1979)
5. H. Holland, *Adaptation in Natural and Artificial Systems* (University of Michigan Press, Ann Arbor, 1975)
6. J. Jaramillo, J. Bhadury, R. Batta, On the use of genetic algorithms to solve location problems. Comput. Oper. Res. **29**, 761–779 (2002)

7. G. Kochenberger, B. McCarl, F. Wyman, A heuristic for general integer programming. Decision Sci. **5**, 36–44 (1974)
8. A. Laoufi, A. Hazzab, M. Rahli, Economic power dispatch using fuzzy-genetic algorithm. Int. J. Appl. Eng. Res. **1**(3), 409–426 (2006)
9. K. Scheibe, L. Carstensen, T. Rakes, L. Rees, Going the last mile: a spatial decision support system for wireless broadband communications. Decis. Support Syst. **42**(2), 557–570 (2006)
10. S. Senju, Y. Toyoda, An approach to linear programming with 0–1 variables. Manag. Sci. **15**, B196–B207 (1968)