Linear Programming for Emergency Broadcast Systems*

Michael C. Ferris Todd S. Munson

December 2, 1998

This short note describes an application of linear programming embedded in emergency broadcast system hardware. In many communities, large sirens sound when tornados, or other disasters, threaten. These warning systems save many lives each year. However, the current system is typically ineffective for several reasons. Each siren covers a broad area and is difficult or even impossible to hear in many locations. Furthermore, individuals are unable to distinguish between different disaster types. The action taken during a tornado warning is very different from that taken when a chemical spill occurs. Finally, some disasters only affect a small area, while the siren blankets the entire community.

As a result, Alert Systems [1] has developed a system to overcome these limitations. In a 911-center, a simple graphical interface displays a map of the controlled area. The operator outlines a polygon containing the current danger area using a mouse pointer device. The coordinates of the polygon's vertices and a relevant message are then transmitted as a radio signal on a controlled frequency to receivers located throughout the community. If a receiver, which knows its own location, is within the transmitted polygon, it must start beeping and display the short message detailing the nature of the disaster transmitted with the coordinates. If it is outside the affected area, the system remains silent.

This problem is easily seen to be a linear program. In essence, it is just a feasibility problem; determine if the location $b = (b_1, b_2)$ of the receiver is within the polygon defined by the vertices $x^1, \ldots, x^n$. More simply stated, can b be represented as a convex combination of the vertices $x^1, \ldots, x^n$? To solve this problem we introduce artificial variables e_1, e_2 and set up the

*This research was supported in part by Alert Systems, Inc. Authors address: Computer Sciences Department, University of Wisconsin, 1210 West Dayton Street, Madison, Wisconsin 53706

linear program:

$$\begin{array}{ll} \min_{\lambda,e} & e_1 + e_2 \\ \text{subject to} & \sum_{i=1}^n x^i \lambda_i + De = b \\ & \sum_{i=1}^n \lambda_i = 1, \lambda_i \geq 0, e_1, e_2 \geq 0 \end{array}$$

where D is a diagonal matrix with diagonal entries $D_{ii} = \text{sign}(b_i - x_i^1), i = 1, 2$. We choose these values for D in order to guarantee that the linear program has a feasible point, $(\lambda = (1, 0, \ldots, 0), e = |b - x^1|)$. Furthermore, the problem is bounded below by 0. Hence, it has an optimal solution which can be found by the simplex method. If the optimal value is 0, the receiver is within the polygon and the signal is activated; otherwise one of the errors e_i is positive and the receiver is outside the polygon and remains inactive.

End of story – not quite. The remaining difficulty is that the receiver must be mass produced. Hence, the manufacturer has decided to use a processing unit that has been restricted so that only integer arithmetic is allowed. While we could emulate floating point operations on this processor, we might encounter problems with numerical error. We do not consider this option because the system must be stable for all allowable inputs (we have real people depending upon the outcome). Another option is to store all of the values as rational numbers. By writing routines to perform the operations required using rationals, the code can be executed using exact arithmetic. However, a much simpler technique which only uses integer variables and the addition, subtraction, and multiplication operations is developed below.

The manufacturer suggested forcing the coordinates x^i to lie on an integer grid and regarding the location of the system as integer. The issue is now to implement the simplex method using only integer arithmetic. Given a canonical form linear program

$$\begin{array}{ll} \min_y & c^T y \\ \text{subject to} & Ay = b \\ & y \geq 0 \end{array}$$

we assume that an initial basic feasible solution is known with the variables partitioned into basics, B, and non-basics, N. This is easy to ensure in the above application. The steps of the revised simplex method [2] are then:

1. $y_B = A_{\cdot B}^{-1} b$.

2. $r_N^T = c_N^T - c_B^T A_{\cdot B}^{-1} A_{\cdot N}$; if $r_N \geq 0$ stop optimal.

3. Choose $j = N(s)$ s.t. $r_j < 0$.

4. Calculate $d = A_{\cdot B}^{-1} A_{\cdot j}$. If $d \leq 0$, stop unbounded.

5. Choose r s.t. $\frac{y_{B(r)}}{d_r} = \min\{\frac{y_{B(i)}}{d_i} \mid d_i > 0\}$.

6. Swap $B(r)$ and $N(s)$ and goto 1.

In the following discussion, we assume that A, b, and c consist of integer data. Using the fact [3] that

$$A_{\cdot B}^{-1} = \frac{1}{\det(A_{\cdot B})} \mathrm{adj}(A_{\cdot B})$$

where $\mathrm{adj}(\cdot)$ denotes the adjugate matrix, and some algebraic manipulation, we can refine the revised simplex method to use only integer arithmetic. Note that the determinant and adjugate can be evaluated in integer arithmetic using only additions and multiplications. For example, in step 1, we could evaluate

$$y_B = \frac{1}{\det(A_{\cdot B})} \mathrm{adj}(A_{\cdot B}) b.$$

We will now outline why the required division here is redundant in the application.

For step 2, we note that the reduced costs, r_N^T, are invariant under multiplication by a positive constant. Letting $\delta = \mathrm{sign}(\det(A_{\cdot B}))$, it is clear that that $\delta \det(A_{\cdot B})$ is a positive constant. We now calculate our reduced costs as follows:

$$R_N^T := \delta \det(A_{\cdot B}) r_N^T = \delta(\det(A_{\cdot B}) c_N - c_B^T \mathrm{adj}(A_{\cdot B}) A_{\cdot N}).$$

If $R_N^T \geq 0$, then we can stop at an optimal solution. Otherwise choose $j = N(s)$ s.t. $R_N^T < 0$.

Since

$$d = \frac{1}{\det(A_{\cdot B})} \mathrm{adj}(A_{\cdot B}) A_{\cdot j},$$

step 4 becomes: if $\delta \mathrm{adj}(A_{\cdot B}) A_{\cdot j} \leq 0$ stop unbounded. Otherwise, we need to perform the ratio test of step 5. Looking at $\frac{y_{B(r)}}{d_r}$ we immediately note that the quantity $\frac{1}{\det(A_{\cdot B})}$ factors out of each term. We implicitly store the resultant $\frac{(\mathrm{adj}(A_{\cdot B})b)_r}{(\mathrm{adj}(A_{\cdot B})A_{\cdot j})_r}$ as a rational number. We find the minimum of all the eligible values by calculating a common denominator, and comparing the integer numerator.

To summarize, our refined revised simplex method is as follows:

1. Calculate δ and R_N^T as above. If $R_N^T \geq 0$, stop optimal. Otherwise choose $j = N(s)$ s.t. $R_N^T < 0$.

2. If $\delta \mathrm{adj}(A_{\cdot B})A_{\cdot j} \leq 0$ stop unbounded. Let $I = \{i \mid \delta(\mathrm{adj}(A_{\cdot B})A_{\cdot j})_i > 0\}$.

3. While $|I| > 1$ do

   (a) Let $\bar{i}, \hat{i} \in I$.

   (b) Let $\Delta := \mathrm{sign}((\mathrm{adj}(A_{\cdot B})A_{\cdot j})_{\bar{i}}(\mathrm{adj}(A_{\cdot B})A_{\cdot j})_{\hat{i}})$.

   (c) If

   $$\Delta(\mathrm{adj}(A_{\cdot B})b)_{\bar{i}}(\mathrm{adj}(A_{\cdot B})A_{\cdot j})_{\hat{i}} \leq \Delta(\mathrm{adj}(A_{\cdot B})b)_{\hat{i}}(\mathrm{adj}(A_{\cdot B})A_{\cdot j})_{\bar{i}}$$

   then $I = I \setminus \hat{i}$.

   (d) Otherwise, $I = I \setminus \bar{i}$.

4. Let r be the remaining element in I. Swap $B(r)$ and $N(s)$ and goto 1.

Note in particular that no divisions are required. The solution values y_B are not needed since for the application considered we only need to test whether the optimal solution of the linear program is zero or positive. Furthermore, for the production version of the code we have included the smallest subscript anti-cycling rule [2] to prevent the simplex method from failing to terminate because of cycling.

For large problems, this method is impractical and not recommended. However, for the feasibility problem that Alert Systems needs to solve where typically n is less than 10, the above method works extremely well. Furthermore, by looking at our specific grid size, we can symbolically evaluate all of the information needed and determine the maximum magnitude of the integers required. This information helps in the actual implementation of the algorithm in hardware.

References

[1] pageALERT. http://www.alertsys.com/.

[2] V. Chvátal. *Linear Programming.* W. H. Freeman and Company, New York, 1983.

[3] R. A. Horn and C. R. Johnson. *Matrix Anaysis.* Cambridge University Press, Cambridge, England, 1985.