A Decision Support System for Locating Facilities and Routing Traffic on a Factory Site

Jean J.M. Derks Frits C.R. Spieksma

University of Limburg
Department of Mathematics
P.O. Box 616
6200 MD Maastricht (The Netherlands)

Abstract

In this note we deal with a practical problem arising at ENCI Netherlands BV. ENCI is a major producer of cement. Its factory site is located south of Maastricht where an important ingredient of cement, marl, is digged from a quarry.

The following situation occurs at the side of ENCI. There are different kinds of traffic on the site and the vehicles of each kind of traffic follow a fixed route on the site. The current set of routes is not satisfactory for ENCI. Indeed, some facilities and roads become congested on a daily basis. In order to solve this problem a decision support system called RACE is developed.

1 Introduction

In this note we report on a practical problem arising at ENCI Netherlands BV. ENCI is a major producer of cement. One of its factory sites is located south of Maastricht where an important ingredient of cement, marl, is digged from a quarry.

The following situation occurs at this site of ENCI. There are different kinds of traffic on the site, referred to as *streams*. The vehicles (or *units*) of each stream follow a fixed *route* on the site. Each stream has a certain *intensity* (see Section 2), which varies during the day, but follows roughly the same pattern each day. Further, on the site different installations, referred to as *facilities* are present. Some of these facilities have been located at a fixed position on the site, others have not been located yet.

A stream may have to visit one or more facilities. More precisely, for each stream there is a facility which serves as *destination* for the units of that stream. At a destination the units of the particular stream spend some time (for example, loading or unloading) before they continue their route on the site. Thus, each stream enters the factory site, goes to its destination and leaves the factory site, while visiting certain prespecified facilities (possibly in a given order).

The current set of routes is not satisfactory for ENCI. Indeed, some facilities and roads become congested on a daily basis. The problem which ENCI faces can thus be described as follows:
can we find locations for the facilities and can we construct a set of routes such that congestion on the site is minimized?

In order to solve this problem we have developed a decision support system called RACE (Routing Advice for Cement-factory ENCI). It was stipulated by ENCI that RACE should be able to operate in an interactive fashion. Thus, solutions were to be computed fast (within seconds) and it should be possible to modify a solution and/or demand that a solution fulfills certain properties.

In Section 2 we give a more precise description of the problem, and in Section 3 we describe the decision support system RACE. Section 4 is devoted to the specific instance at hand and Section 5 contains the conclusion.

For an overview of research concerning facility location and/or routing problems, we refer to Balakrishnan, Ward and Wong (1987), Drezner (1995), Love, Morris and Wesolowsky (1988), Madsen (1983) and Srivastava (1993).

2 The problem

In this section we give a more formal description of the problem presented in the introduction.

Given is a directed graph $G = (V, A)$ such that each arc $a \in A$ corresponds to a road segment on the factory site and such that each vertex $v \in V$ corresponds to:

- a crossing of roads on the site, or
- a potential location for a facility, or
- an entrance/exit of the site.

Thus, informally writing, the graph G represents the factory site. To each arc $a \in A$ and to each vertex $v \in V$ a number is assigned, denoted by $\text{cap}(a)$ ($\text{cap}(v)$), which represents the maximum throughput per hour expressed in so-called *car-equivalents.* Thus, the number $\text{cap}(a)$ ($\text{cap}(v)$) equals the maximum number of cars which can go along an arc a (or vertex v) without causing congestion. It may depend on the physical situation at the site (width of roads, presence of curves, etcetera).

Given are also K facilities. Each facility j has a number associated to it, called $\text{capfac}(j)$, which captures the maximum number of car-equivalents that facility j can handle in a single hour.

Finally, M streams are given (recall that a stream refers to a kind of traffic). Each stream i has a weight w_i, expressing the number of car- equivalents corresponding to a unit of stream i. Also, to each stream a facility $d(i) \in \{1, \ldots, K\}$ is assigned, which serves as *destination* for that stream. Moreover, for each stream a set of facilities, including the destination, is specified, denoted by $F(i)$, which have to be visited by the route of that stream (possibly in a given order). And in addition to all this, the *intensity* of each stream is known, that is, for each hour-interval t it is known how many units of a stream enter (leave) the factory site, denoted by $\text{in}(i, t)$ ($\text{out}(i, t)$).

Summarizing, the input to our problem consists of

- A directed graph $G = (V, A)$ with for each $a \in A$ a capacity $\text{cap}(a)$ and for each $v \in V$ a capacity $\text{cap}(v)$.
- K facilities with for each facility $j \in \{1, \ldots, K\}$ a capacity $\text{capfac}(j)$.

- M streams with for each stream $i \in \{1,\ldots,M\}$ a weight w_i, a destination $d(i) \in \{1,\ldots,K\}$, a set of facilities $F(i) \subseteq \{1,\ldots,K\}$ and incoming intensities $\mathrm{in}(i,t)$ as well as outgoing intensities $\mathrm{out}(i,t)$ for each hour-interval $t = 1,\ldots,24$.

A feasible solution S is found by specifying:

- for each facility a vertex $v \in V$ (let $v(i)$ be the vertex to which $d(i)$ (the destination of stream i) $i = 1,\ldots,M$ is assigned), and
- for each stream i two sets of arcs A^i_{in} and A^i_{out} such that A^i_{in} is a path from an entrance vertex to vertex $v(i)$ (the *ingoing* path) and A^i_{out} is a path from vertex $v(i)$ to an exit vertex (the *outgoing* path) while each vertex to which a facility in $F(i)$ is assigned to, is visited.

Obviously, the quality of a solution depends on the amount of congestion present in it. Let us therefore first formulate a condition describing congestion. Given a solution, congestion on arc $a \in A$ in time-interval t occurs if and only if the sum over the streams containing arc a of the weighted intensities exceeds the capacity of arc a, or equivalently if

$$\sum_{i=1}^{M} w_i[in(i,t)x^{in}_{ai} + out(i,t)x^{out}_{ai}] > cap(a), \tag{1}$$

where $x^{in}_{ai}(x^{out}_{ai}) = 1$ if $a \in A^i_{in}(A^i_{out})$, and 0 otherwise.

A similar expression can be computed describing a condition for congestion at a vertex $v \in V$, provided we make the following adjustment. The capacity of a vertex can be influenced by the capacity of a facility assigned to it. To model this we use the following inequality. Congestion on vertex $v \in V$ in time-interval t occurs if and only if

$$\sum_{i=1}^{M} \sum_{a:a=(v',v)} w_i[in(i,t)x^{in}_{ai} + out(i,t)x^{out}_{ai}] > \min(cap(v), \sum_j z_{jv} capfac(j)), \tag{2}$$

where $z_{jv} = 1$ if facility j is assigned to vertex v, and 0 otherwise.

Of course, there are several objective functions possible, which reflect the amount of congestion. We simply choose to minimize the number of times that congestion occurs. Thus, given a solution S, we let

$$f(S) = |\{(a,t) : \text{ inequality (1) holds }\}| + |\{(v,t) : \text{ inequality (2) holds }\}|$$

There are a number of assumptions we make when using this formulation. Let us try to explicitly address some of them.

1. We assume in this formulation that considering a single day (divided into 24 1-hour intervals) is representative for the situation at ENCI on any day.
2. We assume that the estimates of the data in(i,t) and out(i,t) for all i,t (which we use in Section 4) are not influenced by the solution currently used by ENCI for the routing problem.
3. We assume that the two segments (or directions) of a two-way road can be treated independently of each other.
4. We assume that consecutive time-intervals can be treated independently of each other; thus, congestion in time-interval t has no influence on time-interval $t+1$.
5. We assume that no loops occur in A_i^{in} or A_i^{out} for all i.

Finally, notice that finding an optimal solution for this problem seems quite hard computationally. For instance, even if it is known how the facilities are assigned to the vertices, the problem of constructing a set of routes contains the k-disjoint path problem (see Garey and Johnson (1988)).

3 A description of RACE

In order to deal with the problem described in the previous section, a decision support system called RACE is developed. RACE consists of 2 parts: an algorithmic part and an interface. In Subsection 3.1 a local search method is described which heuristically solves (part of) the problem, and Subsection 3.2 describes shortly how RACE operates.

3.1 A local search method

As indicated in Section 2, the problem naturally decomposes into the subproblems:

(A) where to locate the facilities, and

(B) how to construct a set of routes.

We refer to a solution to subproblem (A) as a *scenario* and to a solution to subproblem (B) as a *routing*. In this subsection, we describe how RACE deals with subproblem (B); the problem of finding a scenario is solved specifically for our instance and described in Section 4.

We are going to use *hierarchical decomposition* in order to solve subproblem (B). Thus, as is often done when solving complicated problems (see, for instance, in VLSI-design (Korte (1989)) or in automated manufacturing (Crama et al. (1990))), we assume a scenario is given and next solve subproblem (B) using a simple local search algorithm. The local search algorithm can informally be described as follows.

Given is an arbitrary solution, which serves as start-solution. We generate all directed paths $P_1, \ldots, P_k$ in G which start at an entrance vertex or end at an exit vertex and store these paths in a list (vertices which serve as destinations of (some of) the streams are considered as entrance and exit vertices). This is carried out by a simple depth-first search labeling procedure (see for example Ahuja, Magnanti and Orlin (1993)) ensuring that all directed paths in G are generated. An iteration of the algorithm goes now as follows: we take a path P_l from the list, and we consider assigning it to stream 1, first as an ingoing path, next as an outgoing path. If P_l is a feasible path for stream 1 and if this decreases the objective function compared to the current solution, we modify the current solution by assigning P_l to stream 1. Next, we repeat this procedure by considering to assign path P_l to stream $2, 3, \ldots, M$. Then we apply a similar step for paths $P_{l+1}, \ldots, P_k$. When all paths on the list are considered, we proceed by simply considering $P_1, P_2, \ldots$ again. The algorithm stops when k paths are considered while no changes in the solution have occurred. A stepwise description is as follows:

```
Step 0: Find an arbitrary solution $S$; let $l := 0$; changed:=TRUE.

Step 1: $l := l + 1$.
        If $l = k + 1$ then
        begin
            If changed=FALSE then STOP else
            begin
                changed:=FALSE;
                $l := 1$;
            end
        end

Step 2: For $i := 1$ to $M$ do
        begin
            If $P_l$ is feasible for stream $i$ as an ingoing path then
            construct the solution $S'$ from $S$ by assigning $P_l$ as an ingoing
            path to stream $i$;
            If $f(S') < f(S)$ then $S := S'$, changed:=TRUE;
            If $P_l$ is feasible for stream $i$ as an outgoing path then
            construct the solution $S'$ from $S$ by assigning $P_l$ as an outgoing
            path to stream $i$;
            If $f(S') < f(S)$ then $S := S'$, changed:=TRUE;
        end
        Go to Step 1.
```

3.2 The interface

Since RACE is to operate in an interactive fashion, the development of the interface was an important issue. The interface built is menu-driven and based on a graphical representation of the factory site. By selecting the appropriate menu's, the user is able to modify all input data, including the graph representing the factory site. Further, the user is able to construct a scenario by assigning the facilities to vertices. Finally when computing a routing, the user is able to fix a route for certain streams, and optimize over the remaining ones.

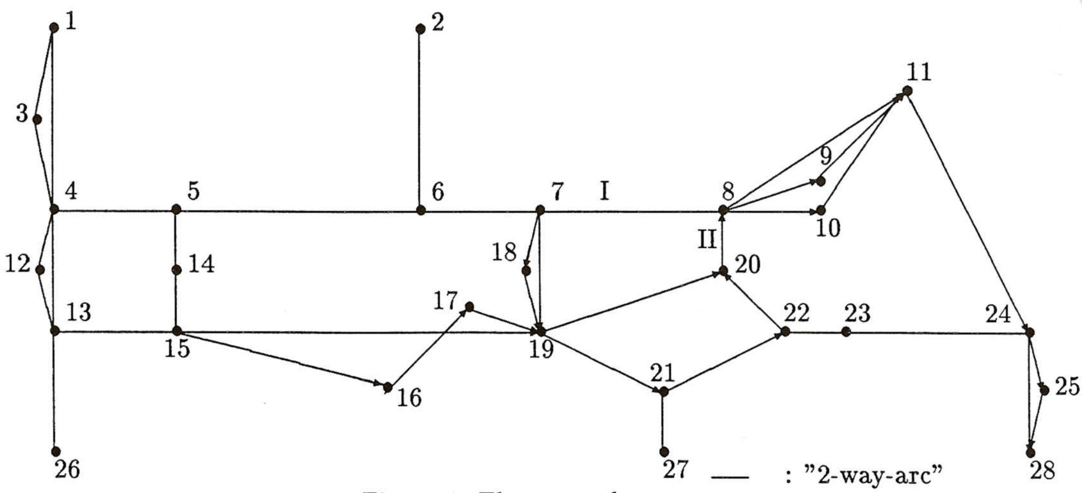

Figure 1: The network

4 The ENCI-instance

In this section we describe the ENCI-instance. The graph which represents the factory site is depicted in Figure 1. Recall that each arc and each vertex has a capacity (which is not listed here). Nodes 26, 27 and 28 in Figure 1 are the three entrance/exit vertices.

Further, in Table 1, 16 facilities are listed, each with their capacity in car-equivalents. Table 2 gives 13 streams, with for each stream i its weight w_i, destination $v(i)$ and the set $F(i)$. For reasons of confidentiality the intensities are not given here.

The current location of the facilities and the set of routes is such that congestion occurs as listed in Table 3, in total 14 times. This corresponds with daily practice.

Finding a feasible scenario turns out to be a complicated task. Indeed, the scenario currently in use is infeasible since a number of facilities (body-washer, underbodywasher and pallet-intake) is not present at the site yet. Constraints of different nature play a role: environmental regulations (for instance concerning the tankstation), safety laws (for instance concerning entrance/exit vertices), future building plans etcetera all have to be satisfied. In order to cope with these constraints, a number of different scenario's were constructed in cooperation with ENCI. As an example, we give in Fig-

Facility	Capacity
1) Bridge South	90
2) Tankstation	60
3) Loading South	25
4) Loading North	30
5) Packing South	12
6) Packing North	12
7) Reception	150
8) Guard	150
9) Underbodywasher	90
10) Bodywasher	150
11) Pallet-intake	72
12) Distribution	80
13) Office nr.1	∞
14) Warehouses	∞
15) Office nr.2	∞
16) Raw material center	∞

Table 1: Facilities

Stream	Weight	Destination (see Table 1)	$F(\cdot)$ (see Table 1)
1) Cementtype 1	5	12	3,12,10
2) Cementtype 2	5	12	4,12,10
3) Cementtype 3	3	12	11,5,12
4) Cementtype 4	3	12	11,6,12
5) Cementtype 5	3	12	11,5,6,12
6) Raw materialtype 1	3	16	1,16,9,1
7) Raw materialtype 2	3	16	1,16,9,1
8) Raw materialtype 3	3	16	16,9
9) Personnel office 1	1	13	8,13,8
10) Personnel office 2	1	13	8,13,8
11) Visitors	1	13	7,8,13,8
12) Suppliers	1	14	8,14,8
13) Tanking	5	2	2

Table 2: Streams

	vertices (Fig. 1)				arcs (Fig. 1)	
Hour	9	12	18	27	I	II
7-8				x	x	x
8-9	x	x		x		
9-10	x					
10-11	x					
12-13	x		x			
13-14	x			x		
16-17				x		
17-18				x		

Table 3: Current congestion

ure 2 a potential scenario, showing how the facilities listed in Table 2 are assigned to the vertices.

For this scenario, RACE computes a set of routes which decreases the number of times that congestion occurs compared to the current routing (see Table 4).

It is noteworthy to remark that running times of RACE for instances as described in this section are usually within 5 seconds.

5 Conclusions

In this note we describe a practical problem involving locating facilities and routing traffic on the factory site of ENCI Netherlands BV. A DSS called RACE is developed which enables the user to find a routing, and to simulate the effect of different scenario's.

Acknowledgements: We would like to thank Mr. Pluijmen and Mr. Jaspers of ENCI and Mr. Delfgaauw for their support and their willingness to provide us with data, and Maarten Oosten for his comments on an earlier version of this note.

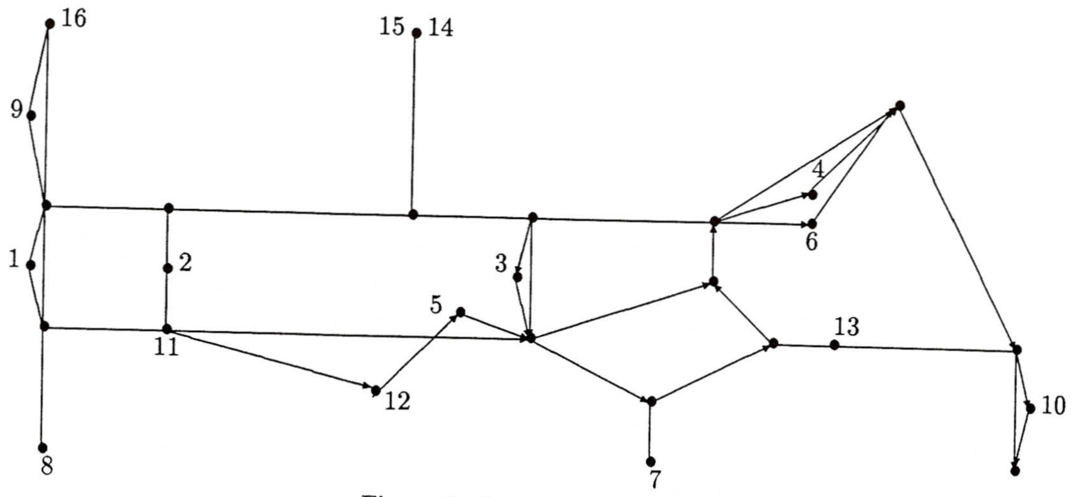

Figure 2: A scenario

	vertices (Fig. 1)				
Hour	9	12	18	26	27
7-8				x	
8-9	x	x			x
9-10	x				
10-11	x				
12-13	x		x		
13-14	x				
16-17				x	

Table 4: Congestion of a solution found by RACE

6 References

Ahuja, R.K., Magnanti, T.L. and J.B. Orlin (1993), "Network Flows", Prentice-Hall, Englewood Cliffs.

Balakrishnan, A., J.E. Ward and R.T. Wong (1987), "Integrated facility location and vehicle routing models: recent work and future prospects", American Journal of Mathematical and Management Sciences 7, 35-61.

Crama, Y., A.W.J. Kolen, A.G. Oerlemans and F.C.R. Spieksma (1990), "Throughput rate optimization in the automated assembly of printed circuit boards", Annals of Operations Research 26, 455-480.

Drezner, Z. (editor) (1995), "Facility location: a survey of applications", Springer, New York.

Garey, M.R. and D.S. Johnson (1979), "Computers and Intractability: A Guide to the Theory of NP-Completeness", Freeman, New York.

Korte, B. (1989), "Applications of combinatorial optimization", in: Mathematical Programming, recent developments and applications, edited by M. Iri and K. Tanabe, KTK Scientific Publishers, Tokyo, 1-55.

Love, R.F., J.G. Morris and G.O. Wesolowsky (1988), "Facilities location, models and methods", North-Holland, New York.

Madsen, O.B.G. (1983), "A survey of methods for solving combined location routing problems", European Journal of Operational Research 12, 295-301.

Srivastava, R. (1993), "Alternate solution procedures for the location-routing problem", OMEGA 21, 497-506.