DATABASE QUERY INTELLIGENCE DEPLOYMENT IN THE LOCAL ACCESS AND TRANSPORT NETWORKS.

Yakov M. Lurye
Bell Communications Research
290 W. Mt. Pleasant Avenue
Livingston, New Jersey 07039 USA

ABSTRACT.

This paper discusses the strategies for the deployment of database query (DQ) capabilities in the telephone network. Currently, database query software is provided in a limited number of tandem switches that serve as concentrators for the query traffic originated in Central Offices. With the offering of new DQ services the increased DQ traffic volumes could justify a gradual transition to an architecture in which Central Office switches are selectively provided with the DQ feature software.

1. INTRODUCTION.

In order to offer a wide range of Intelligent Network (IN) services, telephone companies must provide the network switches with database query (DQ) capabilities. The most common example of a service that requires DQ support is 800 service. Other examples include Private Virtual Network (PVN) service, and Alternate Billing Service (ABS).

To perform a query (e.g., an 800 number translation request) in the Central Office (CO) switch, it must be connected to the Common Channel Signaling network, and equipped with a special software package. Corresponding costs could be quite significant, especially for the analog switches. An alternative is to provide DQ capabilities in a limited number of tandem switches. An originating CO recognizes the DQ call and routes it to a tandem. The tandem performs the query, and completes the call to its destination (Figure 1).

While the tandem DQ service architecture minimizes the feature deployment expenditures, there are certain penalties involved:

- In comparison with a non-DQ call to the same destination, the DQ call may tie up an additional trunk;
- Concentration of the DQ traffic could significantly accelerate the tandem exhaust.

In this paper we describe a simple heuristic algorithm for the gradual deployment of DQ intelligence within a Local Access and Transport Area (LATA). The deployment program takes into account the service demand, switch utilization, and the costs associated with switch replacements/upgrades. Using these data, and associated trunking costs, the planner determines a locally optimal DQ deployment date for each CO. As a next step, he adjusts this schedule in order to reflect the tandem exhaust considerations.
2. PROBLEM FORMULATION.

Consider an (access) tandem sector comprised of a tandem (node 0), and N subtending COs. For each CO \( i \in 1, \ldots, N \) the planner must determine the dates for the DQ deployment \( t_i \). The tandem is provided with the DQ feature at the beginning of the planning interval \( (t_0 = 0) \).

Traffic growth, and the DQ processing overhead could cause the switches' exhaust. The exhaust situation for node \( i \) is resolved or prevented by increasing the switch capacity at the moment \( T_i \). The capacity increase is sufficient to prevent the secondary exhaust within the planning interval \( (0, T) \).

The limitations imposed by the CO switch capacity constraints on a feasible schedule \( t_i \), \( T_i \) could be described as follows:

\[
R_i(t) + Q_i(t) \left(1 + b_i \cdot y_i(t)\right) \leq c_i + (C_i - c_i) \ Y_i(t) \quad i = 1, \ldots, N, \quad t \leq T
\]

\[
y_i(t) = \begin{cases} 0 \quad t < t_i \\ 1 \quad t \geq t_i \end{cases} \quad Y_i(t) = \begin{cases} 0 \quad t < T_i \\ 1 \quad t \geq T_i \end{cases}
\]

Here \( R_i(t) \), \( Q_i(t) \), \( i = 1, \ldots, N \) are respectively the (non-decreasing) non-DQ, and DQ load forecasts for CO \( i \) in the year \( t \) measured by the number of average busy hour calls; \( b_i \) is the per-query processing overhead for the queries performed in the CO; \( y_i(t) \), \( Y_i(t) \) are the "status" indicators ("dumb" /"smart" CO, old/upgraded switch), and \( c_i \), \( C_i \) are, respectively, the switch capacity before, and after the upgrade.
The cost of providing the CO \( i \) with a DQ feature is \( p_i \), and the CO switch replacement cost is \( P_i \). The planner must also take into account the trunk savings from the improved DQ call routing - a portion of the DQ traffic, \( k_i Q_i(t) \), will be routed over direct links, bypassing the tandem. The rest of the DQ calls still go through the tandem. However, the tandem treats such calls as normal non-DQ traffic.

The trunk savings for the first year of DQ software deployment, \( S_i(t_1) \), are roughly proportional to the volume of rerouted calls, \( k_i Q_i(t_1) \). For the following years the savings \( S_i(t) \) depend on the DQ traffic growth [1].

Assuming the discount factor \( \omega \), the adjusted expenses \( \pi_i(t_1, T_1) \) associated with the schedule \( (t_1, T_1) \) for the CO \( i \) can be approximately estimated as follows:

\[
\pi_i(t_1, T_1) = \omega^{t_1} p_i + \omega^{T_1} P_i - \sum_{t \geq t_1} \omega^t S_i(t)
\]

The CO activities could be planned independently, except for the fact that the DQ feature deployment in COs postpones the tandem exhaust. The tandem load consists of three components: (a) non-DQ traffic \( R_0(t) \); (b) "translated" DQ calls from "smart" COs that are still routed via the tandem; (c) DQ traffic from the "dumb" COs. The tandem exhaust date is determined by the CO feature deployment schedule \((t_1, \ldots, t_N)\):

\[
T_0(t_1, \ldots, t_N) = \min \left\{ t \mid R_0(t) + \sum_{i=1}^{N} (1 - k_i) Q_i(t) y_i(t) + (1 + b_0) \sum_{i=1}^{N} Q_i(t) (1 - y_i(t)) > c_0 \right\}
\]

Here \( c_0 \) is the initial tandem capacity, and \( b_0 \) is the per-query overhead for the queries performed in the tandem. At exhaust the tandem capacity is increased at a cost \( P_0 \). As with COs, the assumption is that the capacity increase is sufficient to prevent the secondary exhaust within the planning interval.

The least-cost schedule could be determined by minimizing the total cost of CO activities, and tandem capacity increase:

\[
\min_{t_i, T_i, i \in 1, \ldots, N} \left\{ \sum_{i=1}^{N} \pi_i(t_i, T_i) + P_0 \omega^{T_0(t_1, \ldots, t_N)} \mid (t_i, T_i) \in D_i \right\}
\]

The notation \((t_i, T_i) \in D_i\) indicates that the schedule \((t_i, T_i)\) for CO \( i \) must satisfy the feasibility constraints (2.1).

The tandem exhaust date \( T_0 \) depends on the dates of DQ feature deployment \( t_i \), but not on the dates of CO switch replacement \( T_i \). Late DQ deployment in COs can only accelerate the tandem exhaust. Thus, there is no reason to delay the CO upgrade beyond the locally optimal schedule. Using these considerations, (2.2) could be rewritten in the following form:

\[
\min_{t_i, i \in 1, \ldots, N} \left\{ \sum_{i=1}^{N} \pi_i(t_i) + \pi_0 (T_0(t_1, \ldots, t_N)) \mid t_i \leq t_i^* \right\}
\]

where \( \pi_i(t_i) = \min \left\{ \pi_i(t_i, T_i) \mid (t_i, T_i) \in D_i \right\} \), \( \pi_0(t) = P_0 \omega^t \),

4.4B.6.3
and \( t_i^* = \arg \min_{t_i} \pi_i(t_i) \) is the locally optimal DQ deployment date for the CO \( i \).

\[
\pi_i(t_i), \text{ K}\$
\]

**Feature Deployment Year, \( t_i \)**

**Figure 2. Adjusted cost of CO activities (example).**

(Feature deployment, capacity increase expenditures minus trunk savings).

(a) Low feature cost, large trunk savings \( (t_i^* = 0) \).
(b) Low trunk savings, high feature cost.
(c) "Intermediate" case.

### 3. Equivalent Combinatorial Problem.

In many practical cases the individual penalty functions \( \pi_i(t_i) \) could be considered non-decreasing for \( t_i \leq t_i^* \) (Figure 2). It can be shown that in this case the optimal schedule has the following property:

Let \( I_k = (i_1, \ldots, i_k) \) be the ordered list of COs provided with the query capabilities prior to the tandem capacity increase (ordered by the deployment dates). We will use the notation \( I_n \) for the first \( n \) offices on this list \( (I_0 = \emptyset) \).

In an optimal schedule each EO is provided with the DQ capabilities either at a locally optimal date, or at the moment of tandem exhaust. The deployment dates for COs in the list \( I_k \) are given by an expression:

\[
\hat{t}_n = \min \left( t_n^*, T(I_{n-1}) \right), \quad n = 1, \ldots, k, \tag{3.1}
\]

where \( T(I_n) \) is the moment of the first unresolved tandem exhaust in a schedule \( (\tau_1, \ldots, \tau_N: \tau_i = t_i, i \in I_n, \tau_i = t_i^*, i \notin I_n) \).

The remainder of COs are provided with the DQ software according to the locally optimal schedule: \( t_j = t_j^*, j \notin I_k \).
Problem (2.4) could thus be reduced to a combinatorial problem of determining the ordered list of COs that should be provided with the DQ feature prior to the tandem capacity increase:

$$\min I_k ((t^*) = \sum_{n=1}^{k} \bar{\pi}_n (t^*_n) + \pi_0 (T(I_k)) + \sum_{j \in \mathcal{I}_k} (t_j)$$

(3.2)

where $t^*_n$, $T(I_n)$ are calculated iteratively, using (3.1).


For small-size sectors problem (3.2) could be solved by direct enumeration. Several heuristic planning rules have been proposed for the large-scale cases, e.g.:

1. At the moment of tandem exhaust, provide the DQ capability in an office generating the largest volume of the DQ traffic. The intent is to delay the tandem exhaust, and to immediately capture the trunk savings from the direct routing of the large DQ traffic volumes (Gansert, [2]).

2. At the moment of tandem exhaust, provide the DQ capability in an office with the closest "locally optimal" deployment date.

We have tested another heuristic planning algorithm briefly described below:

A. CO Candidate Selection. Let $I_n$ be the ordered list of COs already provided with the DQ capabilities. The next candidate office $i_{n+1}$ is selected using the following rules:

A.1. In a "non-exhaust" situation, follow the locally optimal schedule:

If $\min t^*_j < T(I_n)$, then $i_{n+1} = \arg \min_{j \in \mathcal{I}_n} t^*_j$

A.2. Otherwise, select the candidate CO by solving the problem

$$\min_{i_{n+1} \in \mathcal{I}_n} \left\{ \Pi(I_{n+1}) - \Pi(I_n) \right\} = \min_{i_{n+1} \in \mathcal{I}_n} \left\{ \bar{\pi}_{i_{n+1}} (t^*_{i_{n+1}}) - \bar{\pi}_{i_{n+1}} (t^*_{i_{n+1}}) + \pi_0 (T(I_{n+1})) - \pi_0 (T(I_n)) \right\}$$

B. "Cutoff" Condition. If

$$\min_{i_{n+1} \in \mathcal{I}_n} \Pi(I_{n+1}) - \Pi(I_n) > 0,$$

increase the tandem capacity at the moment $T(I_n)$. Follow the locally optimal schedule for the remaining COs.

Note that algorithm 1 tends to minimize $\pi_0(T(I_{n+1})) - \pi_0(T(I_n))$, while algorithm 2 is oriented towards decreasing $\bar{\pi}_{i_{n+1}} (t^*_{i_{n+1}}) - \bar{\pi}_{i_{n+1}} (t^*_{i_{n+1}})$. The proposed algorithm, to a certain degree, combines these goals.

Since $\Pi(I_n) < \min \Pi(I_{n+1})$ is a necessary, but not sufficient condition for the optimality of a sequence $I_n$, it is possible to construct an example, in which $\Pi(I_n) < \min \Pi(I_{n+1})$, but $\Pi(I_{n+2}) < \Pi(I_n)$. This problem could be partly
alleviated by allowing additional "look-ahead" iterations of step A, and strengthening the cutoff condition:

$$\Pi(I_n) < \min_{r \geq n} \Pi(I_r).$$

Our limited modeling experience with the small-size sectors indicates that except for the specially constructed cases, the modified algorithm provides satisfactory results:

<table>
<thead>
<tr>
<th>Performance of the &quot;look-ahead&quot; algorithm.</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Problem (3.2); test sector size: 6 nodes).</td>
</tr>
<tr>
<td>Error less than 5% 70% cases</td>
</tr>
<tr>
<td>Error less than 10% 90% cases</td>
</tr>
</tbody>
</table>

4.1. Future Work.

The "almost separable" form of the problems (2.3), (3.2) suggests an attempt to test a heuristic two-stage planning approach (Pervozvansky, [3]), that is somewhat similar to dynamic programming. In the case of problem (2.3) the idea is to replace it by a a modified problem:

$$\min \left\{ \sum_{i \in d} \pi_i(0) + E(I) \right\}$$

Here $I$ is the list of COs scheduled for the immediate DQ feature deployment, and $E(I)$ is the estimated cost of the future activities for the rest of the sector nodes. The estimate for $E(I)$ is produced by direct modeling of the future deployment schedule. The modeling algorithm could be based on a reasonably simple operative planning rule (e.g., one of the rules described in the previous section).

Since the two-stage approach determines only the list of immediately needed activities, the planning algorithm could incorporate the updates in the traffic load forecasts. The next logical step is to study a stochastic analog of (4.1) that would directly take into account the uncertainty in the long term forecasts $Q_i(t), R_i(t)$.

ACKNOWLEDGEMENTS.

I wish to thank J. Gansert, and S. Roltsch for the numerous useful discussions.

REFERENCES

