WORKING PAPER
ALFRED P. SLOAN SCHOOL OF MANAGEMENT

Decompostion Algorithms for Analyzing Transient Phenomena in Multi-class Queuing Networks in Air Transportation

Dimitris Bertsimas, Amedeo Odoni
and
Michael D. Peterson

WP# - 3540-93 MSA January, 1993

MASSACHUSETTS
INSTITUTE OF TECHNOLOGY
50 MEMORIAL DRIVE
CAMBRIDGE, MASSACHUSETTS 02139

Decompostion Algorithms for Analyzing Transient Phenomena in Multi-class Queuing Networks in Air Transportation

Dimitris Bertsimas, Amedeo Odoni
and
Michael D. Peterson

WP# - 3540-93 MSA January, 1993

Decomposition Algorithms for Analyzing Transient Phenomena in Multi-class Queuing Networks in Air Transportation

Michael D. Peterson[*] Dimitris J. Bertsimas[†] Amedeo R. Odoni[‡]

January 8, 1993

Abstract

In a previous paper (Peterson, Bertsimas, and Odoni 1992), we studied the phenomenon of transient congestion in landings at a hub airport and developed a recursive approach for computing moments of queue lengths and waiting times. In this paper we extend our approach to a network, developing two approximations based on the method used for the single hub. We present computational results for a simple 2-hub network and indicate the usefulness of the approach in analyzing the interaction between hubs. Although our motivation is drawn from air transportation, our method is applicable to all multi-class queuing networks where service capacity at a station may be modeled as a Markov or semi-Markov process. Our method represents a new approach for analyzing transient congestion phenomena in such networks.

Airport congestion and delay have grown significantly over the last decade. By 1986 ground delays at domestic airports averaged 2000 hours per day, the equivalent of grounding the entire fleet of Delta Airlines at that time (250 aircraft) for one day (Donoghue 1986). In 1990, 21 airports in the U.S. exceeded 20,000 hours of delay, with 12 more projected to exceed this total by 1997 (National Transportation Research Board 1991). This amounts to

[*] School of Public and Environmental Affairs, Indiana University, Bloomington, Indiana

[†] Sloan School of Management, Massachusetts Institute of Technology, Cambridge, Massachusetts

[‡] Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts

multi-class queuing networks.

Our model provides important qualitative insight on the interaction between hubs and improves our understanding of hub-and-spoke systems. With our model we can address a number of interesting questions, such as:

- To what degree do network effects alter the results obtained from the study of a single hub?
- What network effects are produced by delay propagation?
- What effect does the degree of connectivity between hubs have on system congestion?
- What are the effects of *hub isolation* strategies — strategies in which a hub's connectivity to others is reduced — on schedule reliability?

The paper is organized as follows. In Section 1 we review briefly the methodology of the algorithm for a single queue and describe the queuing network context of the present problem. In Sections 2 and 3 we outline two decomposition approaches which exploit this algorithm. Section 2 describes a relatively simple method in which downstream arrivals are adjusted according to expected upstream waiting times. Section 3 describes a more involved approach which uses second moment information about delays to give a stochastic description of downstream arrival rates. In Section 4 we employ these approximation methods together with a simple simulation procedure on a 2-hub network. We find that under moderate traffic conditions, the interactions between hubs do not alter demand patterns significantly, so that the predictions of waiting times produced by the different approaches are very close. Under heavy traffic conditions with closely spaced banks, higher waiting times act to smooth the demand pattern significantly over the day. In this latter situation, the two recursive algorithms deviate further from simulation than in the former case. We also find that isolation of a problem hub protects other hubs from schedule disruption at the cost of further disruption at the source of the delays. We provide concluding remarks in Section 5.

1 The Basic Model

Incoming aircraft at an airport require service at three stations: a landing runway, a gate, and a departure runway. The landing operation in particular is subject to wide variations

$$\tilde{p}_{ii}(m) \triangleq \Pr\left((i,m) \rightarrow (i,m+1)\right) = \Pr\left[T_i \geq m+1 \mid T_i \geq m\right]. \tag{1}$$

We next define the following random variables:

$$\begin{aligned}
Q_k &\triangleq \text{Queue length at end of interval } k, \\
W_k &\triangleq \text{Waiting time at end of interval } k, \\
C_k &\triangleq \text{Capacity state at end of interval } k, \\
A_k &\triangleq \text{Age of current capacity state at end of interval } k, \\
T_i &\triangleq \text{Random lifetime of capacity state } i.
\end{aligned}$$

For mean queue length we introduce the notation

$$\begin{aligned}
&\mathcal{Q}_k(l,i,m,q) \triangleq E\left[Q_k \mid Q_l = q, C_l = i, A_l = m\right] \\
&\quad k = 1,\ldots,K, \quad i = 1,\ldots,S, \quad m = 1,\ldots,M, \\
&\quad l \leq k, \quad q = 1,\ldots,q_{\max}(k,i),
\end{aligned} \tag{2}$$

where $q_{\max}(k,i)$ is the maximum attainable queue length at the end of period k, given that at that time the capacity state is i. This obeys the recursion

$$q_{\max}(k,i) = \left[q_{\max}(k-1) + \lambda_k - \mu_i\right]^+ \tag{3}$$

where $q_{\max}(k) \triangleq \max_i q_{\max}(k,i)$ and $x^+ = \max(x,0)$. Similarly, for waiting times we employ the notation

$$\mathcal{W}_k(l,i,m,q) \triangleq E\left[W_k \mid Q_l = q, C_l = i, A_l = m\right]. \tag{4}$$

We write the second moment analogs of (2) and (4) as $\mathcal{Q}_k^2(l,i,m,q)$ and $\mathcal{W}_k^2(l,i,m,q)$, respectively.

Let $(x \wedge y)$ denote $\min(x,y)$. The quantities $\mathcal{Q}_k(l,i,m,q)$, $\mathcal{Q}_k^2(l,i,m,q)$, $\mathcal{W}_k(l,i,m,q)$, and $\mathcal{W}_k^2(l,i,m,q)$ can be calculated recursively, (Peterson, Bertsimas, and Odoni 1992). We repeat here the basic equations:

$$\begin{aligned}
\mathcal{Q}_k(l,i,m,q) &= \sum_{j \neq i} \tilde{p}_{ij}(m)\mathcal{Q}_k\left(l+1,j,1,(q+\lambda_{l+1}-\mu_j)^+\right) + \\
&\quad \tilde{p}_{ii}(m)\mathcal{Q}_k\left(l+1,i,m+1,(q+\lambda_{l+1}-\mu_i)^+\right), && (5) \\
\mathcal{Q}_k^2(l,i,m,q) &= \sum_{j \neq i} \tilde{p}_{ij}(m)\mathcal{Q}_k^2\left(l+1,j,1,(q+\lambda_{l+1}-\mu_j)^+\right) + \\
&\quad \tilde{p}_{ii}(m)\mathcal{Q}_k^2\left(l+1,i,m+1,(q+\lambda_{l+1}-\mu_i)^+\right), && (6)
\end{aligned}$$

where

$$
\begin{aligned}
i_m^v &\triangleq m\text{th stop on itinerary of aircraft } v,\\
t_m^v &\triangleq \text{scheduled arrival time at } m\text{th stop for aircraft } v,\\
s_m^v &\triangleq \text{slack time between stops } m-1 \text{ and } m \text{ for aircraft } v.
\end{aligned}
$$

Aircraft *slack* between stops $m-1$ and m is the amount of time available to the aircraft at stop $m-1$ beyond the minimal time necessary to turn the aircraft around. In the network schedules are no longer exogenous and deterministic, as delays at one airport affect the schedules at others. In the terminology of queuing theory, the system is a multi-class queuing network, with the classes being the different aircraft with their individual itineraries. Service capacity at each airport is an autocorrelated stochastic process described by a semi-Markov process or Markov chain. Thus our task is to describe the transient behavior of a multi-class queuing network with autocorrelated service rates at each node. This high degree of complexity suggests that approximation methods are necessary.

2 A Simple Decomposition Approach

A first approximation approach is based on the following idea. Suppose that at the start of the day, one knows the schedules for all aircraft operating in the network. Under the assumption that delays are zero at the outset of the day, the schedule for the initial period of the day is fixed. Hence the first period demands are fixed, and mean queue lengths and waiting times for each airport during this period may be determined by applying equations (5) - (14) to each airport. The resulting expected waiting times for period 1 are estimates of the delay encountered by all aircraft scheduled to land in this period. Taking into account the slack which these aircraft have in their schedules and updating future arrival streams accordingly, one then fixes demand for the next period, calculates the resulting new expected waiting times, and so forth.

More formally, let d^v represent the current cumulative delay for aircraft v — i.e. as aircraft v proceeds through its itinerary, d^v is the current amount by which it is behind schedule. Further define the terms

$$
\begin{aligned}
\mathcal{A}(n,k) &\triangleq \text{set of aircraft scheduled to land at } n \text{ in period } k,\\
E\left[W_k^n\right] &\triangleq \text{mean waiting time for an aircraft arriving at } i \text{ at end of period } k,
\end{aligned}
$$

First Decomposition Algorithm for Air Network Congestion

```
Initialize:
For $k = 1$ to $K$
    For $n = 1$ to $N$
        $\mathcal{A}(n, k) = \phi$
**** first itinerary stops are deterministic since not affected by earlier delays *****
For $n = 1$ to $N$
    For $v = 1$ to $V$
        $\mathcal{A}(n, \kappa(t_1^v)) = \mathcal{A}(n, \kappa(t_1^v)) \cup v$
Set $d^v = 0 \ \forall \ v$.

Main loop:
For $k = 1$ to $K$
    For $n = 1$ to $N$
        Set $\lambda_k^n = |\mathcal{A}(n, k)|$.
        Using the recursive method at each airport, calculate $E\left[W_k^1\right], \ldots, E\left[W_k^N\right]$.
        For $v \in \mathcal{A}(n, k)$:
        ***** find the part of the itinerary corresponding to this stop *****
            Find $m : (n_m^v, t_m^v, s_m^v) \in \mathcal{I}(v)$ and $\kappa(t_m^v + d^v) = k$
            Set $n = n_m$, $t = t_m + d^v$, $s = s_m$, $n' = n_{m+1}$, $t' = t_{m+1}$, $s' = s_{m+1}$.
            Set $\alpha = \kappa(t) - t/(\Delta t)$.
        ***** calculate propagated delay *****
            Set $d_{m+1}^v = \left[d^v + \alpha E\left[W_{\kappa(t)-1}^n\right] + (1-\alpha) E\left[W_{\kappa(t)}^n\right] - s'\right]^+$.
        ***** determine next arrival period and update data structure *****
            Set $\mathcal{A}(n', \kappa(t' + d^v)) = \mathcal{A}(n', \kappa(t' + d^v)) \cup v$.
END.
```

Figure 1: Decomposition algorithm for network based on deterministic updating scheme

where U is the complexity of the single hub recursive algorithm for waiting time moments with deterministic input.

PROOF:
The choice of data structure means that the inner updating loop (the disaggregation procedure) requires only $O(V)$ time. Hence the bottleneck of the algorithm consists of repeated calls to a subroutine for computing expected waiting times. Because for each time period k the algorithm must recalculate all of the preceding expected waiting times, overall complexity is $O(KNU)$. □ In Peterson, Bertsimas,

and Odoni (1992) it was shown that for a Markov model of capacity, the complexity of the recursive algorithm for a single hub is $O(S^2K^2Q_{\max})$. Thus if the Markov capacity model is specified with S capacity states, overall complexity for Algorithm I is $O(NS^2K^3Q_{\max})$.

The presence of the additional factor K arises from the fact that the recursion at each hub is restarted from time 0 at each new period. Thus in the first global iteration the algorithm finds $E[W_1^1]\ldots E[W_1^N]$, in the second it finds $E[W_1^1]\ldots E[W_1^N]$ and $E[W_2^1]\ldots E[W_2^N]$, and so forth. This duplication of effort could be avoided if it were possible to store within the single hub algorithm the end conditions of iteration k as initial conditions for iteration $k+1$. However, even for the simpler Markov capacity model, this would mean storing the joint probabilities for queue length and capacity. Since computing these probabilities requires $O(Q_{\max})$ times as much effort as for the expectation alone (see Peterson, Bertsimas, and Odoni 1992), there is no benefit to doing so unless the probabilities themselves are desired for some other reason.

A more practical improvement is to have the recursion restart only every m periods, where m is the minimum number of periods any aircraft has between scheduled stops. Under this scheme, the algorithm is run for the first m periods, arrivals are updated, then the algorithm is run for the first $2m$ periods, and so on. Whereas in the original implementation, the number of iterations performed within the recursive algorithm is

$$1 + 2 + \ldots + K = K(K+1)/2,$$

under this new scheme it is

$$m + 2m + 3m + Gm + K' = G(G+1)m/2 + K'$$

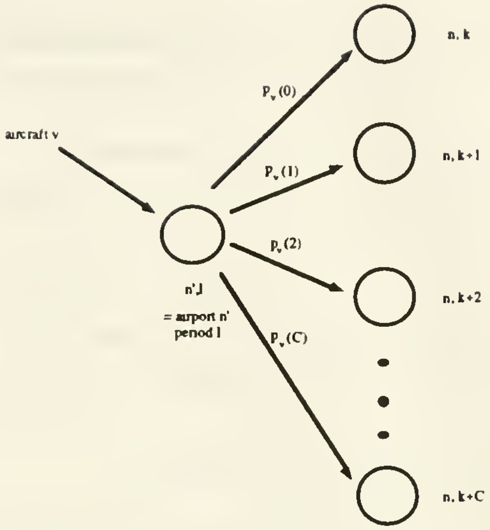

Figure 3: The traffic splitting phenomenon: alternative future aircraft paths depend upon delay encountered. The numbers $\{p_v\}$ indicate probabilities.

In order to complete the updating scheme, the algorithm must translate the probabilistic information on individual aircraft into information on future arrival rates. Define the stochastic arrival quantities

$$\Lambda(n,k) \triangleq \text{number of arrivals at airport } n \text{ in period } k.$$

The goal of this step of the procedure is to specify an approximate probability law for these random variables. For some user-specified number R (representing the number of possible values taken by the random variables) the algorithm estimates numbers $\gamma_k^n(1), \ldots, \gamma_k^n(R)$ and $\lambda_k^n(1), \ldots, \lambda_k^n(R)$ which obey the relationships

$$\begin{aligned} \Pr\{\Lambda(n,k) = \lambda_k^n(1)\} &= \gamma_k^n(1), \\ \Pr\{\Lambda(n,k) = \lambda_k^n(2)\} &= \gamma_k^n(2), \\ &\vdots \\ \Pr\{\Lambda(n,k) = \lambda_k^n(R)\} &= \gamma_k^n(R). \end{aligned} \tag{18}$$

where

$$\sum_i \gamma_k^n(i) = 1 \tag{19}$$

This simplified description of variability in the arrival rates is easily incorporated into the

2. Translation of these density functions into probabilistic descriptions of future arrival periods for each aircraft, as given in the parameters $p_v(0), \ldots, p_v(C)$ and $k_v(0), \ldots, k_v(C)$.

3. Translation of the individual aircraft parameters $p_v(0), \ldots, p_v(C)$ and $k_v(0), \ldots, k_v(C)$ into simple discrete distributions for the random variables $\Lambda(n, k)$.

4. Updating of aircraft itineraries and airport arrival lists.

The fourth of these procedures was described in Section 2. The first three are described in further detail in what follows, and a summary of the algorithm is given in Figure 8.

3.1 Obtaining waiting time densities

Estimation of the densities $f(w)$ cannot be done on the basis of the recursive algorithm alone, since this procedure gives only the first two moments of the distribution. Knowledge of the third moment would give enough information to determine a unique 2-point discrete distribution by solving the nonlinear system

$$\begin{aligned} p_1 w_1 + p_2 w_2 &= E[W] \\ p_1 w_1^2 + p_2 w_2^2 &= E\left[W^2\right] \\ p_1 w_1^3 + p_2 w_2^3 &= E\left[W^3\right] \\ p_1 + p_2 &= 1 \\ p_1, p_2, w_1, w_2 &\geq 0. \end{aligned} \tag{22}$$

for the values p_1, p_2, w_1, and w_2. However, this system is not guaranteed to have any solution because of the positivity requirement.

An alternative method is suggested by Monte Carlo methods (see e.g. Hammersley and Handscomb, 1964).. Consider a simple simulation for a single airport in which capacity, period by period, is determined in Monte Carlo fashion from the Markov chain or semi-Markov process. From the simulation we obtain the matrix of observations

$$\mathbf{W} = \{W_k^m\},$$

where W_k^m is the waiting time at the end of period k for the mth simulation. Ordering the observations, we obtain histograms for the waiting times for each period, like the one illustrated in Figure 4 for a constant arrival rate ($\rho \approx 0.85$, $\lambda = 60$ per hour). Note the

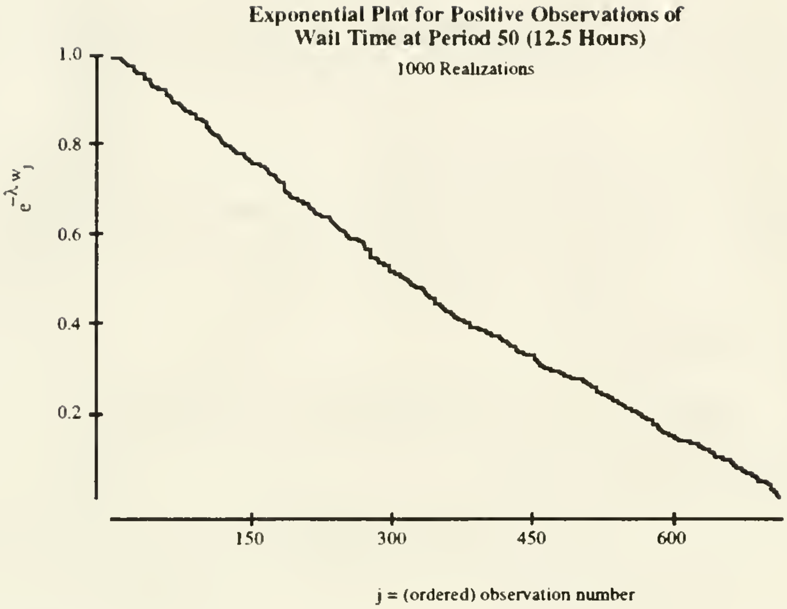

Figure 5: Test for exponential distribution of positive waiting time realizations

ν must be determined by solving the pair of equations (omitting subscripts)

$$\begin{aligned} \delta w_{\min} + (1-\delta)\int_{w_{\min}}^{\infty} w\nu e^{-\nu(w-w_{\min})}\,dw &= E[W] \\ \delta\left(w_{\min}\right)^2 + (1-\delta)\int_{w_{\min}}^{\infty} w^2\nu e^{-\nu(w-w_{\min})}\,dw &= E[W^2] \end{aligned} \qquad (24)$$

In terms of the mean $\overline{w}$ and variance σ^2 we obtain the solution (omitting subscripts)

$$\delta = \frac{\sigma^2 - (\overline{w} - w_{\min})^2}{\sigma^2 + (\overline{w} - w_{\min})^2} \qquad (25)$$

$$\nu = \frac{2(\overline{w} - w_{\min})}{\sigma^2 + (\overline{w} - w_{\min})^2} \qquad (26)$$

Note that δ is always less than 1 and will be nonnegative provided that

$$\frac{\sigma^2}{(\overline{w} - w_{\min})^2} \geq 1.$$

In the typical case where $w_{\min}$ is zero, this is equivalent to the condition that the coefficient of variation for waiting times exceeds 1. Only in rare instances of the tests presented shortly was this condition found not to hold. In those cases, the parameter δ was set to 0 and the entire distribution was assumed to be exponential.

For practical reasons, it is necessary to choose some upper bound C on the number of periods of delay to allow. Hence

$$p_v(C) = \int_{w(C-1)}^{\infty} f(w; \mu, \sigma^2) dw,$$

Together with the numbers $\{k_v(c)\}$, the probabilities $\{p_v(c)\}$ then constitute a probabilistic description of the next period in which aircraft v will demand to land.

3.3 Characterizing arrivals

In order to translate the numbers $\{p_v(c)\}$ into a probabilistic description of the future demand rates $\Lambda(n, k)$, define the random variable

$$X_{n'l,nk}(v) \triangleq \begin{cases} 1 & \text{if } v \in \mathcal{A}(n', l) \text{ is delayed such that its} \\ & \text{next stop will be } n \text{ at period } k \\ 0 & \text{otherwise} \end{cases}$$

This random variable denotes the "contribution" of an arrival at one place and time to the arrival rate at a future place and time. Note that if the next stop of $v \in \mathcal{A}(n', l)$ is n, then

$$\Pr\{X_{n'l,nk}(v) = 1\} = p_v(k-l).$$

In words, for aircraft $v \in \mathcal{A}(n', l)$, the probability that it will contribute to the landing demand at airport n during period k (assuming that n is its next scheduled stop) is $p_v(k-l)$.

The random variables $X_{n'l,nk}(v)$ provide the necessary connection between aircraft and arrival rates. Then

$$\Lambda(n, k) = \sum_{n'=1}^{N} \sum_{l<k} \sum_{v=1}^{V} X_{n'l,nk}(v). \tag{30}$$

In words, this says that the arrival rate at (n, k) is the sum of all contributions from previous points in the itineraries (see Figure 6). Thus the random variables $\{\Lambda\}$ are sums of Bernoulli random variables. Defining

$$NL(v, k) \triangleq \text{next destination of aircraft } v \text{ after period } k$$

the expectation is easily obtained as

$$\begin{aligned} E[\Lambda(n, k)] &= \sum_{n'=1}^{N} \sum_{l<k} \sum_{v=1}^{V} E[X_{n'l,nk}(v)]. \\ &= \sum_{n'=1}^{N} \sum_{l<k} \sum_{v:NL(v,l)=n} p_v(k-l) \end{aligned} \tag{31}$$

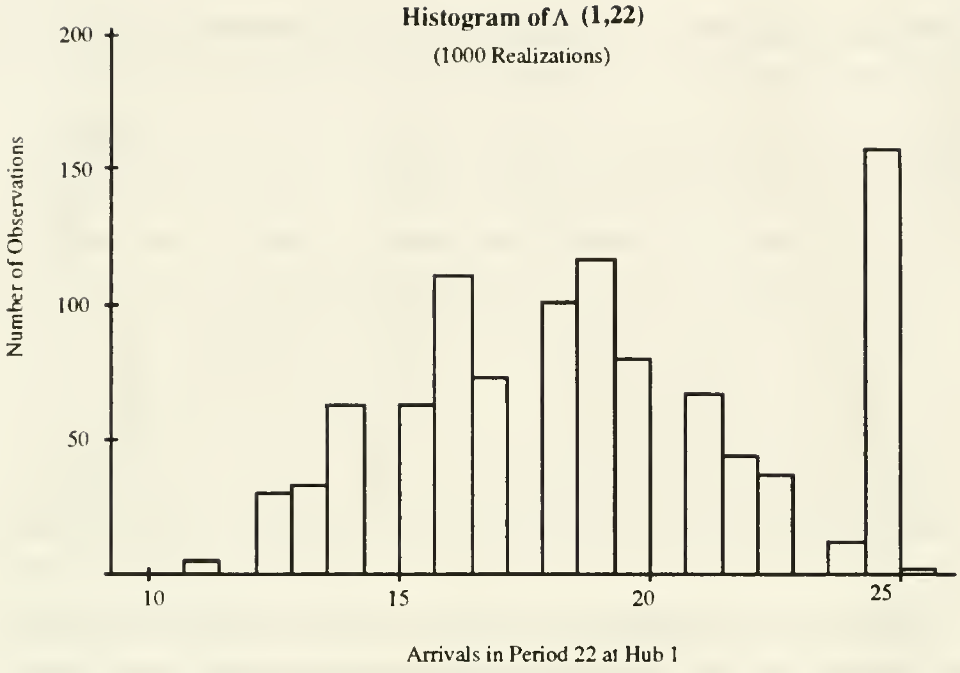

Figure 7: Histogram of $\Lambda(1,22)$ obtained from simulation. Unusual skewness patterns such as this one may occur in the early part of the day when the contributing prior arrivals are still largely deterministic.

considerable degree of insensitivity to the demand rate distribution. We retain the normality assumption while acknowledging its imperfections.

Although Algorithm 2 involves considerably more modeling work than Algorithm 1, its computational complexity is not significantly higher, as our next result indicates.

Theorem 2 *The complexity of Algorithm 2 is $O(RKNU)$, where R is the user-specified number of values used in the approximate distribution for the arrival rates and U is the complexity of the single hub recursive algorithm for waiting time moments with deterministic input. If the Markov capacity model is specified with S capacity states, overall complexity is $O(RNS^2K^3Q_{\max})$.*

PROOF:

Within the principal loop, the bottleneck operation remains that of calculating the waiting time moments in the recursion. Because the arrival stream is specified probabilistically rather than deterministically, there is an additional factor R equal to the number of values specified for each arrival rate distribution. The result follows. □

Both Algorithms 1 and 2 are suitable for any kind of network. Without the streamlining suggested at the end of Section 2, running times are somewhat high. For example, on a simple 2-airport network with $K = 80$ periods at each airport, Algorithm 1 takes about one hour on a DEC-3100 workstation while Algorithm 2 takes about three hours (with $R = 3$). With the reduction in calls to the recursion achieved by the streamlining procedure, there is roughly tenfold improvement in these figures. Even with this improvement, modeling a full-size network of a large airline (400+ nodes) is a daunting problem. On the other hand, the problem is well suited to parallel computation, with different processors handling the individual nodes and a central processor controlling the bookkeeping of aggregation and disaggregation

In the present context, further simplification is possible. Consider a single carrier trying to understand congestion in its own hub-and-spoke network. From this carrier's perspective, delays at its *hubs* have far greater implications for disruption of its schedule than delays at its *spokes*. This observation suggests a simplification: reduce the hub-and-spoke network to a network of hubs. That is, keep track only of aircraft belonging to the hub carrier, treat other arrivals as fixed, and treat all congestion delays other than those emanating from the hubs as *neglibible*. In the resulting network, we incorporate spoke information in setting aircraft itineraries. As before, these consist of ordered triples $\{(i_m, t_m, s_m)\}$, but now each i_m refers to a *hub* airport and each s_m reflects the total slack available to an aircraft between successive visits to hubs, including the slack available at an intervening spoke. External aircraft add to demand and congestion in the system, but their arrival schedules are considered fixed. All internal flights in the collapsed network appear to take place between hubs, but flight times vary widely to reflect the fact that in reality, the aircraft have intermediate spoke stops.

By ignoring congestion at the spokes of the system and concentrating only on the hubs, we can reduce the size of a large airline's network from 400+ nodes to perhaps 5 or 6. These changes reduce the model's realism, but the reduced model should capture essential behavior. Since one of the main goals is to improve our understanding of interactions between *hubs* (e.g. the issue of isolating Chicago), this simplification seems to be further justified. The testing and analysis presented in the next section is conducted on a simple 2-hub network

case #	no. banks	bank space	p	ρ	slack	initial capacities
1	(DFW)	—	0	0.5	15-20 mins.	low/high
2a	12	15 mins.	0.5	0.9	5 mins.	steady state
2b	12	15 mins.	0.5	0.9	500 mins.	steady state
3	—	—	0.5	0.8	5 mins.	steady state
4a	10	30 mins.	0	0.7	5 mins.	low/high
4b	10	30 mins.	1	0.7	5 mins.	low/high
5a	10	30 mins.	0.5	0.7	5 mins.	steady state
5b	10	30 mins.	0.5	0.7	10 mins.	steady state
5c	10	30 mins.	0.5	0.7	15 mins.	steady state
5d	10	30 mins.	0.5	0.7	20 mins.	steady state

Table 1: Test run information. Note that traffic intensities ρ are based on that part of the schedule which does not include the runout period at the end of the day. 'Steady state' indicates that initial capacities occur according to the steady state probabilities of the Markov chain.

$1 - p$. A value of $p = 1$ implies a fully connected network (all flights alternate between the hubs), while a 0 value means a totally disconnected network.

Cases 1,2, and 3 are concerned with validation and an initial exploration of the nature of the network effects. In each case we test the models against a simulation procedure which generates period-by-period capacities at each hub using Monte Carlo methods. The simulation works in exactly the same fashion as the two approximation procedures, except that arrival rates are adjusted by simulated waiting times rather than by expected values or some approximate distribution of waiting time.

The demand and capacity data for both the hubs in case 1 closely resemble those at Dallas-Fort Worth. We have, however, collapsed the capacity state space to three states with steady state probabilities 0.07, 0.10, and 0.83. The simulated arrival pattern, together with the actual one for DFW, is shown in Figure 10. In case 1, slacks for the individual aircraft take values in the range of 15-20 minutes between stops at hubs, depending on the distance to the intervening spoke.

Case 2 provides an instance of higher traffic intensity than is present in case 1. Here internal aircraft are grouped into identically timed banks of 30-minutes duration at each *isolation*. by considering an instance in which the two hubs have no aircraft in common ($p = 0$, the disconnected case) and an instance in which they have all aircraft in common ($p = 1$, the fully connected case). In case 5 we examine four instances in which aircraft slack is varied.

4.2 Results and Discussion

Considering cases 1-5 together, we note that model parameters should have a noticeable effect on the mechanics of the network. For example, in the DFW case (#1), waiting times are of the same order as aircraft slack, and there is substantial separation between major traffic peaks. For these reasons, we expect delay propagation to be relatively low and have a less disruptive effect on the schedule. In case 2, on the other hand, major peaks are much closer together, traffic intensity is sharply higher, and delay propagation should be more important.

Using a DEC-3100 workstation we performed computations for test cases 1-5, using both of our recursion-based approximations as well the the simple simulation procedure discussed above. Our investigation is primarily motivated by the desire to develop qualitative insight into the transient phenomena of the network. Accordingly, the following set of questions will guide our discussion of the results:

- To what degree do network effects alter the results obtained from the study of a single hub?
- What are the network effects produced by delay propagation and under what circumstances do they become important?
- How closely do the network approximation results match those of simulation? Where do they differ?
- What is the effect of congestion at one hub on demand and congestion at the other?
- How is this effect altered by the amount of slack in aircraft schedules?
- What is the effect of isolating a congested hub by not allowing its flights to connect with the other hub?

preservation of the peaked delay pattern, both of which indicate that the effects induced by delay propagation (the "network effects") are relatively minor. Because there is ample space between major banks and slack values are close to the mean waiting times, the general peaked pattern is preserved. The result suggests that *when space between banks is adequate to ensure a moderate traffic intensity and when mean waiting times are not substantially greater than aircraft slack, network effects are outweighed by the "deterministic" congestion effects resulting from the banked structure of arrivals at hubs.*

Behavior Under Heavier Traffic and Closer Spacing

The relative weakness of the network effect in the preceding example obscures differences between the network approximations and the simulation. A more revealing picture is provided by case 2a (Figure 12). Here expected waiting times (30-40 minutes) are quite high relative to aircraft slack (5 minutes), and there is only a 15-minute gap between successive banks. While the early part of the day shows a close fit between the simulation and the algorithms, there is a noticeable disparity in the middle part of the day, when alterations in the arrival stream become significant. Relative to simulation, both algorithms tend to overestimate delay during the middle part of the day, with the difference as high as 30% for certain periods. This same effect is present in case 1 to a much lesser degree.

For a given hub and algorithm we define a standard error in the predictions relative to simulation. Let X_k denote the waiting time value predicted by algorithm for period k and Y_k denote the corresponding value for the simulation. Then the standard error s is given by

$$s = \sqrt{\frac{\sum_{k=1}^{K} (X_k - Y_k)^2}{K}}.$$

This provides a measure of how far apart the simulation and algorithm results are. For Algorithm 1, these values are 4.5 and 2.6 minutes at the two hubs, while the corresponding numbers for Algorithm 2 are 4.5 and 2.3. The numbers represent an average error of 10-20%, with worse fits in the middle part of the day.

Case 3, in which demand is allowed to be continuous over the day (no banks), also shows a discrepancy between the approximations and the simulation during the middle part of the day, as Figure 13 indicates. The traffic intensity for this case is higher than case 1 but lower than case 2. The difference between the algorithms and simulation exceeds 20% for a large part of the day at hub 2, and the standard errors are approximately 15% of the delays: 2.2

minutes at hub 1 (for both algorithms) and 2.4 and 2.6 minutes (Algorithm 1 and Algorithm 2) at hub 2. Thus in all cases, the simulation produces lower waiting time estimates during the middle part of the day than both of the approximation algorithms. We next consider the likely explanation for the discrepancy.

The Network Effect: Demand Smoothing

Consider the waiting time profiles for cases 1 and 2a (Figures 11 and 12). Evidently, waiting time profiles are much smoother in the latter than in the former. With only a 15-minute separation between banks, the relatively high waiting times combine with low aircraft slack to overwhelm the bank structure. Thus we see that when traffic intensity is very high and aircraft slack is low, the order of the network's schedule breaks down. Further evidence of this effect is given in the top half of Figure 14, where we have plotted the original demand profiles at Hub #1 together with those which are produced as a result of delayed arrivals under scenario 2a. The original schedule is labeled "slack = 500", corresponding to the artificial situation where aircraft slack is large enough to eliminate propagation completely. We see by comparison with the situation "slack = 5" that propagated delays smooth the demand pattern substantially, with large numbers of aircraft shifted to late periods. The sharp peak structure of the original demand is considerably altered.

This smoothing phenomenon explains why Algorithms 1 and 2 overestimate delays consistently in the middle part of the day. In the actual process, an aircraft scheduled at a given period may experience a delay ranging from zero up to 3 hours or more. In cases of high waiting times, the aircraft's next arrival time will be considerably later than was scheduled, and its contribution to later demand is pushed back by a significant number of periods. Thus over a large number of simulations with heavy traffic, a noticeable fraction of arrivals are pushed back to the later part of the day, when there is no scheduled traffic. Because capacity is more than adequate then, the result of this traffic shift is to reduce overall waiting times. Ideally, the computational algorithms should reflect this shifting and smoothing of demand. However, as was remarked earlier, to do a thorough job they would have to keep track of the thousands of potential paths which aircraft may follow as a result of delay, a seemingly impossible computational burden. To limit the state space to manageable size, both algorithms update aircraft *schedules* according to one number, expected waiting time. The result is that both algorithms tend not to shift aircraft to the very late part of the day sufficiently but rather to concentrate demand more in the middle, resulting in higher predicted waits.

The Effect of Demand Smoothing on Waiting Times

The phenomenon of demand shifting and smoothing explains certain observations which seem counter-intuitive at first. An example of such a result is the fact that higher aircraft slack can *increase* expected *queuing times* at the hubs. Cases 2a and 2b illustrate this. Both have heavy traffic organized into narrowly separated banks. The difference is that in case 2b, artificially high slack prevents the network effect of demand smoothing, whereas case 2a allows the demand to become smoother over time as aircraft are pushed back to the end of the day. In the high slack case, the higher concentration of demand produces *higher* queuing delays, as we see in the bottom half of Figure 14. Because slack preserves the *schedule*, it also preserves the peaked pattern in that schedule, which produces queues. Note that in case 2b there is a closer fit between the algorithms and the simulation, because the high slack means that the schedule becomes essentially deterministic.

Assessing the Network Approximations

The results of cases 1-3 suggest the circumstances under which network effects become important, and they also indicate that under these circumstances, the network approximations developed in this paper tend to overestimate waiting times during the busy period of the day. Case 1 suggests that for networks of airports like DFW, waiting times on average are probably not high enough *on average* to create significant network effects: the deterministic part of the schedule (i.e. the bank structure) predominates. However, as cases 2 and 3 illustrate, the situation changes when traffic becomes heavier and spacing between major banks is decreased. This situation may only describe a few airports at present in this country (e.g. O'Hare), but it represents a future scenario which is quite possible. In the cases of heavier traffic, lower slack, and less separation, the performance of the algorithms worsens as they tend not to capture the true spreading of demand which is the major network effect.

Slack, Connectivity, and Hub Isolation

We consider next the effect of network *connectivity* and aircraft slack on *cumulative aircraft delay*. We distinguish between this latter measure and that of the *waiting times* at the hubs.

The latter correspond to the waiting times of aircraft at the various stations in the network, while the former is really the sum of such waiting times with aircraft slack subtracted. We measure network connectivity in terms of the percentage of flights having operations at both hubs in the network. Case 4 illustrates two opposing extremes of connectivity: a fully disconnected network (case 4a), where each hub has its own set of aircraft; and a fully connected network (case 4b), where all flights alternate between the two hubs in between visits to spokes.

Case 4a models the idea of hub isolation referred to earlier. Because the network is completely disconnected ($p=0$), scheduled bank times at one hub cannot be disrupted by late arrivals from the other. In contrast, case 4b ensures that aircraft encountering delays at one hub have the maximum chance to disrupt the schedule at the second, since that is their next destination after the intervening spoke. Case 4's scenario thus allows us to explore the network effects of low capacity at a single location. To do this, in both case 4a and case 4b we take the initial state of the first hub to be 1 (lowest capacity) and that of the second hub to be 3 (highest capacity). The phenomenon of interest is the propagation of delays created at the first to the banks of the second.

Our results are summarized in Figure 15, which plots *average cumulative delay per arriving aircraft.* The early banks show zero delay, while later banks reflect delay carried over from previous points in the itinerary. The figure indicates a degradation in performance at hub #1 when it is isolated, as well as the corresponding benefits of isolation at hub #2. Conversely, the fully connected case benefits hub #1 at the expense of #2.

Upon further examination, these results make intuitive sense. Clearly we expect hub #2's schedule to become more reliable when it is disconnected from the disruptions produced by #1. But we also see that hub #1's schedule performance improves when it moves in the opposite direction — from disconnected to fully connected. Examining the situation at hub #1 more closely, we notice that the delays in the connected case seem to lag behind the delays in the disconnected case by about 2 banks (2 hours). This is no coincidence: in the connected case, the minimum time between any aircraft's successive visits to the same hub is 4 hours (4 banks), while in the disconnected case it is only 2. Thus the schedule delays produced by the congestion at hub #1 are felt 2 hours later at that hub in the connected case, producing the 2-hour lag. However, this lag does *not* fully explain the difference in the heights of the two curves in the top half of Figure 15. In the connected case, late aircraft leaving hub #1 have the opportunity of recovering some of the delay through slack at their next stop (uncongested hub #2). This opportunity is not available in the disconnected case, since the next stop is (congested) hub #1, a fact which explains why the corresponding delay is higher even after we take account of the lag.

These results have interesting implications for a strategy of hub isolation. In the case of a hub which is believed to be the source of a large amount of congestion, such a strategy will indeed protect other hubs in the system from the uncertainties and disruptions produced by the problem hub. On the other hand, disruption at that hub itself may worsen since many of its later arrivals will have had an earlier scheduled stop there already.

Cases 5a, 5b, 5c, and 5d illustrate the effect of slack on aircraft lateness. We noted earlier that higher slack preserves demand peaking and may actually increase queuing delays at the hubs. On the other hand, slack reduces each aircraft's *cumulative delay*. Figure 16 illustrates that this second effect predominates in this relatively light traffic. For varying slack values, the figure plots the average cumulative delay per aircraft arriving at each bank of the day, not including any waiting at the current stop. Certainly the figure does not contain any surprises. We include it in order to illustrate the kind of planning for which the models are well suited.

Finally, we note that in a situation of major capacity shortfalls, airlines do not passively accept long delays which disrupt the schedule. Instead, schedulers respond in "real time" by canceling and rerouting flights. The preceding discussion is intended to provide insight into the phenomena of interest and to the strategic issues that airlines must plan for in connection with schedule disruptions due to congestion at their hubs. At the tactical and operational level, airline behavior is in actuality more dynamic.

5 Conclusion

In this paper we have developed two related approximation approaches to the difficult problem of modeling transient queuing behavior in a hub-and-spoke network. We would summarize our major findings as follows:

1. *Importance of traffic splitting phenomenon.* High uncertainty in levels of delay encountered by aircraft is a prominent feature of the network problem. Unfortunately,

accuracy in keeping track of aircraft amid this uncertainty is limited by high computational complexity.

2. *Importance of deterministic effect.* The peaked pattern of demand at hub airports remains a strongly determining factor in predicting waiting times, particularly when major banks are separated by ample lengths of time.

3. *Delay and smoothing.* On the other hand, in cases where banks are narrowly spaced, delay propagation exerts a strong smoothing effect on the demand and waiting time profiles.

4. *Effects of hub isolation.* A policy of isolating a congestion-prone hub clearly does have the effect of improving performance at others. On the other hand, under this policy the isolated hub produces congestion delays which disrupt its own future schedule.

We conclude with some remarks about running times. As we reported earlier, the running times for Algorithms 1 and 2 are high even for the small 2-hub test network: approximately one hour for Algorithm 1 and three hours for Algorithm 2 on a DEC-3100 workstation. These times are particularly poor considering that the running time for the simulation program (5000 samples) is significantly *shorter* — about 10 minutes. In the absence of improvements in the algorithms, this observation favors simulation. However, the implementation of Algorithms 1 and 2 used in our tests is a rather inefficient one. Incorporating the earlier suggestion that the recursion be restarted every m periods rather than at every period would reduce running times by at least a factor of

$$\frac{K+1}{(K/m)+1} \approx m$$

A value $m = 10$ (2 1/2 hours), which is approximately the minimum time a typical aircraft would have between successive visits to hubs, would reduce running times by a factor of 9 (for $K = 80$ periods). This improvement alone would bring the running times of the algorithms into the same range as simulation. The reduction is important for the general problem because the number of simulations necessary cannot be known in advance. However, at least in this test case, the simulation procedures themselves, based on the same ideas of the original Markov and semi-Markov capacity models, offer a third approach to understanding network effects.

ROTH, EMILY. 1981. An Investigation of the Transient Behavior of Stationary Queuing Systems. Ph.D. dissertation. Operations Research Center, Massacusetts Institute of Technology, Cambridge, MA.

WHITT, W. 1983. The Queuing Network Analyzer. *Bell System Technical Journal* **62**:9, 2779-2815.

NATIONAL TRANSPORTATION RESEARCH BOARD. 1991. Winds of Change: Domestic Air Transport Since Deregulation. Transportation Research Board National Research Council Special Report 230. Washington, D.C.