DISCRETE-TIME STOCHASTIC AUTOMATA
NETWORKS AND THEIR EFFICIENT ANALYSIS

Charles University of Prague
Prague, Czech Republic, June 26-30, 2000

Tuğrul Dayar
Department of Computer Engineering
Bilkent University (tugrul@cs.bilkent.edu.tr)

(joint work with Oleg Gusak and Jean-Michel Fourneau)
Discrete-Time Stochastic Automata Networks (SANs)

- SANs provide a methodology for modeling Markovian systems with interacting components (Plateau ’85).
- A discrete-time system of $N$ components can be modeled by a single stochastic automaton for each component.
- When there are $E$ synchronizing events in the system, automaton $k$ (denoted by $\mathcal{A}^{(k)}$) has the corresponding transition probability matrix $P_e^{(k)}$ that represents the contribution of $\mathcal{A}^{(k)}$ to synchronization $e \in \{0, 1, \ldots, E - 1\}$;
- The DTMC corresponding to the global system is given by
  \[
  P_{n \times n} = \sum_{e=0}^{E-1} \bigotimes_{k=0}^{N-1} P_e^{(k)} \quad \text{(descriptor).} \tag{1}
  \]
- Assuming that $\mathcal{A}^{(k)}$ has $n_k$ states, $n = \prod_{k=0}^{N-1} n_k$.
- We represent the state of automaton $\mathcal{A}^{(k)}$ by $s\mathcal{A}^{(k)}$. 


Discrete-Time SANs (continued)

- When transition probabilities in $P_e^{(k)}$ are functions of the global state of the system rather than only $sA^{(k)}$, tensor products become generalized tensor products (Plateau & Fourneau ’91).

- Benefits:
  - Each component can be modeled much easier compared to the global system due to state space reduction.
  - Space required to store the description of components is minimal compared to the case in which transitions from each global state are stored explicitly.

- Drawback:
  - The matrices $P_e^{(k)}$ are relatively dense compared to their continuous-time counterparts implying large # of flops in the generalized descriptor-vector multiply algorithm (Fernandes, Plateau, Stewart ’98) ⇒ difficulty in analysis.
Work

- We model an application from mobile communications as a discrete-time SAN.
- We prove a result on lumpability (Kemeny & Snell ’60) for a class of discrete-time SANs which simplifies the analysis considerably.
- We introduce an efficient iterative aggregation-disaggregation (IAD) algorithm for the same class of discrete-time SANs.
- We provide the results of numerical experiments on the model.
A Wireless ATM System

Basic Model

- Application arises in wireless asynchronous transfer mode (ATM) networks.
- Two types of service integrated over a time-division multiple access (TDMA) system in a mobile communication environment:
  - constant bit rate (CBR) service for two types of voice calls (handover calls from neighboring cells and new calls),
  - available bit rate (ABR) service for data transfer.
- Single cell and single carrier frequency is modeled.
3 (\(= E\)) synchronizing events

\(e_0\): 0 CBR arrivals with prob. \(\gamma_0 = (1 - p_n)(1 - p_h)\),

\(e_1\): 1 CBR arrival with prob. \(\gamma_1 = p_n(1 - p_h) + p_h(1 - p_n)\),

\(e_2\): 2 CBR arrivals with prob. \(\gamma_2 = p_n p_h\),

where

\(p_n\): CBR new call arrival during a TDMA frame,
\(p_h\): CBR handover arrival during a TDMA frame.

3 (\(= N\)) automata

\(A^{(0)}\): models data arrival process to data buffer (2 states: on, off),

\(A^{(1)}\): models TDMA frame of \(C\) slots (\(C + 1\) states: 0, 1, \ldots, \(C\)),

\(A^{(2)}\): models data buffer of size \(B\) (\(B + 1\) states: 0, 1, \ldots, \(B\)).

State space size: \(n = 2(C + 1)(B + 1)\).
Arrival process of data and service process of CBR calls is quite general and subsumes the model in (Véque & Ben-Othman ’98).

When compared to the model in (Véque & Ben-Othman ’98), our SAN model is scalable since its $E$ is independent of $C$.

We model the system as a discrete-time SAN in which state changes occur at TDMA frame boundaries.

Data packet and CBR call arrivals to the system are assumed to happen at the beginning of a frame.

Data packet transmissions and CBR calls are assumed to terminate at the end of the frame.

Since each data packet is transmitted in a single slot, in a particular state of the system we never see slots occupied by data packets.
Data arrival process

\[ P_{e_0}^{(0)} = P_{e_1}^{(0)} = P_{e_2}^{(0)} = \begin{bmatrix} 1 - \beta & \beta \\ \alpha & 1 - \alpha \end{bmatrix}, \]

where

\( \alpha \): prob. of process moving from on-state to off-state,
\( \beta \): prob. of process moving from off-state to on-state.

Load offered to data buffer: \( \lambda = \beta / (\alpha + \beta) \).
If square coefficient of variation of process is \( S_C \), then

\[
\beta = 2\lambda(1 - \lambda) / (S_C + 1 - \lambda), \quad \alpha = \beta(1 - \lambda) / \lambda.
\]

In on-state, \( i \in \{0, 1, 2, 3\} \) data packets arrive with prob. \( p_{di} \), \( \sum_i p_{di} = 1 \).
Mean arrival rate of data packets: \( \rho = \sum_{i=0}^{3} i \times p_{di} \).
Global mean arrival rate of data packets: \( \Gamma = \lambda \rho \).
TDMA frame

- We do not consider multiple CBR handover or new call arrivals during a TDMA frame since the associated probabilities with these events are small.

- Each CBR call takes up a single slot of a TDMA frame but may span multiple TDMA frames whereas each data packet is small enough to be served in a single TDMA slot.

- If $sA^{(1)} = i$, the current TDMA frame has $i$ active CBR connections.

- When all the slots are full, incoming CBR calls are rejected.

- # of CBR calls that may terminate in the current TDMA frame depends on $sA^{(1)}$, but can be at most $M(\leq C')$

  $\Rightarrow$ modeled as a truncated binomial process with parameter $p_s$. 
TDMA frame (continued)

The contribution of $\mathcal{A}^{(1)}$ to synchronization $e_a$, $a \in \{0, 1, 2\}$, is given by the matrix $P_{e_a}^{(1)}$ of order $(C + 1)$ with $ij$th element

$$p_{ij} = \begin{cases} 
\gamma_a \binom{i + a}{i + a - j} p_s^{i + a - j} (1 - p_s)^j, & i + a \leq C, \ 0 \leq i + a - j < M \\
\gamma_a \binom{C}{C - j} p_s^{C - j} (1 - p_s)^j, & i + a > C, \ 0 \leq i + a - j < M \\
\gamma_a \sum_{k=j}^{C} \binom{i + a}{i + a - k} p_s^{i + a - k} (1 - p_s)^k, & i + a \leq C, \ 0 \leq i + a - j = M \\
\gamma_a \sum_{k=j}^{C} \binom{C}{C - k} p_s^{C - k} (1 - p_s)^k, & i + a > C, \ 0 \leq i + a - j = M \\
0, & \text{otherwise}
\end{cases}$$

for $i, j = 0, 1, \ldots, C$. 
Data buffer

- Data is queued in a FIFO buffer of size $B$ and has the least priority.
- If $s \mathcal{A}^{(2)} = i$, the buffer has $i$ data packets.
- If the number of arriving data packets exceeds the free space in the buffer plus the number of free slots in the current TDMA frame, then the excess packets are dropped.
Data buffer (continued)

\[
P_{e_0}^{(2)} =
\begin{bmatrix}
g_1 & p_{-1} & p_{-2} & p_{-3} \\
g_1 & p_0 & p_{-1} & p_{-2} & p_{-3} \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
g_1 & p_{C-1} & p_{C-2} & p_{C-3} & p_{C-4} & \cdots & p_{-3} \\
p_C & p_{C-1} & p_{C-2} & p_{C-3} & \cdots & p_{-2} & p_{-3} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
p_C & p_{C-1} & p_{C-2} & p_{C-3} & \cdots & p_{-2} & g_2 \\
p_C & p_{C-1} & p_{C-2} & \cdots & p_{-1} & g_2 \\
p_C & p_{C-1} & \cdots & p_0 & g_2 \\
p_C & \cdots & p_1 & g_2 
\end{bmatrix}
\]
Data buffer (continued)

- $P_{e_1}^{(2)}$ and $P_{e_2}^{(2)}$ have the same (functional) nonzero structure as that of $P_{e_0}^{(2)}$, but different contents.

- For synchronization $e_a$, the probability $p_l$ is obtained from

$$p_l = \begin{cases} 
p_d(FS(a) - l), & sA^{(0)} = 1, \ 0 \leq (FS(a) - l) \leq 3 \\
1, & sA^{(0)} = 0, \ FS(a) = l \\
0, & \text{otherwise}
\end{cases}, \quad (2)$$

where $FS(a) = [C - (sA^{(1)} + a)]^+$ denotes the # of free slots in the current TDMA frame taking into account up to $a \in \{0, 1, 2\}$ possible CBR arrivals. The values $g_1$ and $g_2$ are given by

$$g_1 = \sum_{l=sA^{(2)}}^C p_l, \quad g_2 = \sum_{l=B-sA^{(2)}}^3 p(-l).$$
A SAN model for VBR traffic

- Variable bit rate (VBR) service for two types of calls (handover calls from neighboring cells and new calls).

- Probabilities of VBR new call and handover arrivals during a TDMA frame have geometrical distributions with parameters $p_{vn}$ and $p_{vh}$, respectively.

- We do not consider multiple VBR handover or new call arrivals during a TDMA frame as for CBR calls.

- A VBR connection is characterized by a state of high intensity and a state of low intensity in which the transmission rate is reduced $\Rightarrow$ some slots will be empty.

- A VBR connection moves from high-state to low-state with prob. $\alpha_v$ and from low-state to high-state with prob. $\beta_v$.

- When a VBR connection is set up, it is in high-state.

- However, a VBR connection can terminate in any state.
• When the arrival process is in low-state:
  – it moves from the state in which the slot allocated to the VBR connection is busy to the state in which the slot is empty with prob. \( p_{empty} \),
  – it moves from the state in which the slot allocated to the VBR connection is empty to the state in which the slot is busy with prob. \( p_{busy} \).

• When a VBR connection changes its state from high to low, it enters the busy state.

• The \# of VBR calls that may terminate in a given TDMA frame depends on the \# of active VBR calls.

• The duration of each VBR call is assumed to be a geometric process with parameter \( p_{vs} \).
We model each slot reserved for VBR traffic in the current TDMA frame by a single automaton of 4 states. Let there be $V$ such slots and let $A^{(k)}$ be the automaton corresponding to the $k$th VBR slot. Then

\[ sA^{(k)} = 0: \text{ idle}, \quad sA^{(k)} = 1: \text{ high}, \]
\[ sA^{(k)} = 2: \text{ low and busy}, \quad sA^{(k)} = 3: \text{ low and empty}. \]

3 synchronizing events

$f_0$: 0 VBR arrivals with prob. $\tilde{\gamma}_0 = (1 - p_{vn})(1 - p_{vh})$,

$f_1$: 1 VBR arrival with prob. $\tilde{\gamma}_1 = p_{vn}(1 - p_{vh}) + p_{vh}(1 - p_{vn})$,

$f_2$: 2 VBR arrivals with prob. $\tilde{\gamma}_2 = p_{vn}p_{vh}$.

Transition probability matrix $P_{f_b}$ that corresponds to synchronizing event $f_b$, $b \in \{0, 1, 2\}$, is given by

\[
P_{f_b} = \tilde{\gamma}_b \bigotimes_{k=0}^{V-1} P_{f_b}^{(k)} = \tilde{\gamma}_b P_{f_b}^{(0)} \bigotimes_{k=1}^{V-1} \left( \bigotimes_{k=1}^{V-1} P_{f_b}^{(k)} \right).
\]
We define

\[ P_{fb}^{(k)} = \begin{cases} \gamma_b P_b^{(k)}, & k = 0 \\ P_b^{(k)}, & 0 < k < V \end{cases} \]

where \( P_b^{(k)} \) is the transition probability matrix describing the evolution of the \( k \)th VBR slot when there are \( b \) VBR arrivals.

Then

\[
P_0^{(k)} = \begin{bmatrix} 1 & p_{vs} \tilde{p}_{vs} \alpha_v & \tilde{p}_{vs} \alpha_v \\ p_{vs} & \bar{p}_{vs} \beta_v & \bar{p}_{vs} \beta_v \bar{p}_{empty} & \bar{p}_{vs} \beta_v \bar{p}_{empty} \\ p_{vs} & \bar{p}_{vs} \beta_v & \bar{p}_{vs} \beta_v \bar{p}_{busy} & \bar{p}_{vs} \beta_v \bar{p}_{busy} \end{bmatrix},
\]

where \( \bar{q} = 1 - q \) for \( q \in [0, 1] \).
\[
P_1^{(k)} = \begin{bmatrix}
1 - g_3(k)\bar{p}_{vs} & g_3(k)\bar{p}_{vs} \bar{\alpha}_v & g_3(k)\bar{p}_{vs} \alpha_v \\
p_{vs} & \bar{p}_{vs} \bar{\alpha}_v & \bar{p}_{vs} \alpha_v \\
p_{vs} & \bar{p}_{vs} \beta_v & \bar{p}_{vs} \bar{\beta}_v \bar{p}_{empty} & \bar{p}_{vs} \bar{\beta}_v \bar{p}_{empty} \\
p_{vs} & \bar{p}_{vs} \beta_v & \bar{p}_{vs} \bar{\beta}_v \bar{p}_{busy} & \bar{p}_{vs} \bar{\beta}_v \bar{p}_{busy}
\end{bmatrix},
\]

where

\[
g_3(k) = \begin{cases}
1, & s\mathcal{A}^{(k)} = 0 \text{ and } \sum_{l=0}^{k-1} \tilde{s}\mathcal{A}^{(l)} = k \\
0, & \text{otherwise}
\end{cases},
\]

\[
\tilde{s}\mathcal{A}^{(l)} = \begin{cases}
0, & s\mathcal{A}^{(l)} = 0 \\
1, & \text{otherwise}
\end{cases}.
\]

To obtain \(P_2^{(k)}\), replace \(g_3(k)\) in \(P_1^{(k)}\) with:

\[
g_4(k) = \begin{cases}
1, & s\mathcal{A}^{(k)} = 0 \text{ and } (\sum_{l=0}^{k-1} \tilde{s}\mathcal{A}^{(l)} = k \text{ or } \sum_{l=0}^{k-1} \tilde{s}\mathcal{A}^{(l)} = k - 1) \\
0, & \text{otherwise}
\end{cases}.
\]
Combined SAN model

- TDMA frame consists of $C$ slots reserved for CBR traffic and $V$ slots reserved for VBR traffic.

- $N = 3 + V$ automata: first 3 due to basic SAN model, last $V$ dedicated to VBR arrivals.

  $$n = 2(C + 1)(B + 1)4^V.$$  

- Set of synchronizing events: $E_{CBR} \times E_{VBR}$, where $E_{CBR} = \{e_0, e_1, e_2\}$ and $E_{VBR} = \{f_0, f_1, f_2\}$

- Synchronizing event $s_{ab}$: $a$ CBR arrivals and $b$ VBR arrivals,

  $$P_{s_{ab}}^{(k)} = \begin{cases}  
P^{(k)}_{e_a}, & 0 \leq k < 3 \\
  P^{(k-3)}_{f_b}, & 3 \leq k \leq V + 2 \end{cases}.$$
• Replace each $C$ in $P_{c_a}^{(2)}$ and $g_1$ by $(C + V)$.

• Redefine $p_l$ in equation (2) and the function $FS(a)$ as:

$$p_l = \begin{cases} 
p_{d(FS(a,b)-l)}, & sA^{(0)} = 1, \quad 0 \leq (FS(a,b) - l) \leq 3 \\
1, & sA^{(0)} = 0, \quad FS(a,b) = l \\
0, & \text{otherwise}
\end{cases}$$

where $a$ and $b$ are the indices of synchronizing event $s_{ab}$,

$$FS(a,b) = [C-(sA^{(1)}+a)]^+ + [V-(\sum_{k=3}^{V+2} \delta(sA^{(k)} = 1) + \sum_{k=3}^{V+2} \delta(sA^{(k)} = 2)+b)]^+$$

and $\delta$ denotes the Kronecker delta.

• The underlying DTMC of the complete system is given by

$$P = \sum_{a=0}^{2} \sum_{b=0}^{2} \bigotimes_{k=0}^{V+2} P_{s_{ab}}^{(k)}.$$
Performance Measures of Interest

\[ P_{cdrop} = \| \pi_{sA(1)} = C \|_1 \]  

\[ P_{cblock} = \| \pi_{sA(1)} \geq C - 1 \|_1 \]  

\[ P_{vdrop} = \| \pi_{sA(k)} \neq 0, \forall k \in \{3, 4, \ldots, V+2\} \|_1 \]  

\[ P_{vblock} = \| \pi_{sA(k)} = 0 \text{ for only one } k \in \{3, 4, \ldots, V+2\} \|_1 \]  

\[ P_a = \frac{\rho P[0 \text{ empty slots}]+(p_{d2}+2p_{d3})P[1 \text{ empty slot}]+p_{d3}P[2 \text{ empty slots}]}{\rho} \]  

(data packets)

where

\[ P[0 \text{ empty slots}] = \| \pi_{FS(0,0)} = 0 \wedge sA(2) = B \|_1, \]

\[ P[1 \text{ empty slot}] = \| \pi_{FS(0,0)} = 1 \wedge sA(2) = B \|_1 + \| \pi_{FS(0,0)} = 0 \wedge sA(2) = B - 1 \|_1, \]

\[ P[2 \text{ empty slots}] = \| \pi_{FS(0,0)} = 2 \wedge sA(2) = B \|_1 + \| \pi_{FS(0,0)} = 1 \wedge sA(2) = B - 1 \|_1 + \| \pi_{FS(0,0)} = 0 \wedge sA(2) = B - 2 \|_1. \]
Difficulty in Analysis

Let $\lambda = 0.1$, $S_C = 1$, $(p_{d0}, p_{d1}, p_{d2}, p_{d3}) = (0.05, 0.1, 0.25, 0.6)$ (amounting to an avg. of $\rho = 2.5$ packet arrivals during a TDMA frame), $(p_n, p_h, p_s) = C(5 \times 10^{-6}, 10^{-5}, 5 \times 10^{-6})$, $\lambda_v = 0.5$, $S_C v = 10$, $(p_{empty}, p_{busy}) = (0.9, 0.1)$, $(p_{vn}, p_{vh}, p_{vs}) = V(5 \times 10^{-6}, 10^{-5}, 5 \times 10^{-6})$.

- For $(C, V, B) = (8, 2, 15)$ with $M = 2$, we have $n = 4,608$ and $nz = 1,618,620$ (# of nonzeros larger than $10^{-16}$ is 1,174,657).

- For $(C, V, B) = (12, 3, 15)$ with $M = 3$, we have $n = 26,624$ and $nz = 39,042,922$ (# of nonzeros larger than $10^{-16}$ 19,979,730).

- For $(C, V, B) = (16, 4, 15)$ with $M = 4$, we have $n = 139,264$, but are not able to determine $nz$ in a reasonable amount of time.

Hence, in this problem, we not only have state space explosion, but we also have a relatively dense global DTMC hindering performance analysis by conventional techniques.
Lumpability

Theorem 1 A discrete-time SAN of $N$ automata and $E$ synchronizing events whose automata are reordered and renumbered so that $A^{(k)}[A^{(0)}, A^{(1)}, \ldots, A^{(k-1)}]$, $k \in \{1, \ldots, N - 1\}$, and that has the descriptor in equation (1) with equal row sums in each $P_e^{(k)}$ for $k = 0, 1, \ldots, N - 1$ and $e = 0, 1, \ldots, E - 1$ is lumpable with respect to the partitioning

$$P_{n \times n} = \begin{pmatrix} P_{11} & P_{12} & \ldots & P_{1K} \\ P_{21} & P_{22} & \ldots & P_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ P_{K1} & P_{K2} & \ldots & P_{KK} \end{pmatrix}$$

in which all the blocks $P_{ij}$ are square, of order $\prod_{k=m}^{N-1} n_k$, and $K = \prod_{k=0}^{m-1} n_k$ for any $m \in \{1, 2, \ldots, N - 1\}$. 
Hence, for the given ordering of automata, there are \((N - 1)\) different lumpable partitionings which vary between

- one that has \(n_0\) diagonal blocks of order \(\prod_{k=1}^{N-1} n_k\)
- one that has \(\prod_{k=0}^{N-2} n_k\) diagonal blocks of order \(n_{N-1}\).

For the wireless ATM problem, such an ordering implies that

- the automaton corresponding to the data buffer be placed in the last position
- the automata corresponding to VBR traffic be placed in any position other than the last as long as they are ordered according to increasing index among themselves

\[\Rightarrow\text{ one possibility is } A^{(1)}, A^{(3)}, A^{(4)}, \ldots, A^{(V+2)}, A^{(0)}, A^{(2)}.\]

IAD (Stewart, Stewart, McAllister ’94) exhibits fast convergence if the degree of coupling, \(\|F\|_\infty\), is small compared to 1, where \(P = F + \text{diag}(P_{11}, P_{22}, \ldots, P_{KK})\).
Cyclic Dependencies and Lumpability

**Definition 1** Let $G(\mathcal{V}, \mathcal{E})$ be the directed graph (digraph) corresponding to a discrete-time SAN in which the vertex $v_k \in \mathcal{V}$ represents $A^{(k)}$ and the edge $(v_k, v_l) \in \mathcal{E}$ if transitions in $A^{(k)}$ depend on the state of $A^{(l)}$ (i.e., $A^{(k)}[A^{(l)}]$). Then the SAN is said to contain cyclic functional dependencies if and only if the digraph has at least one strongly connected component (SCC) composed of multiple automata.

**Theorem 2** A discrete-time SAN of $N$ automata $A^{(k)}$, $k = 0, 1, \ldots, N - 1$, and $E$ synchronizing events that contains cyclic functional dependencies among its automata is lumpable if the digraph corresponding to the SAN has more than one SCC and each $P_e^{(k)}$ has equal row sums for $k = 0, 1, \ldots, N - 1$ and $e = 0, 1, \ldots, E - 1$. 
**Algorithm 1. IAD algorithm for discrete-time SANs**

1. Let $\pi^{(0)} = (\pi_1^{(0)}, \pi_2^{(0)}, \ldots, \pi_K^{(0)})$ be an initial approximation of $\pi$; $it = 1$.

2. Aggregation (compute and solve lumped matrix $L$ once):
   (a) Compute $L$ of order $K$ with $ij$th element $l_{ij} = \max(P_{ij}u)$.
   (b) Solve the singular system $\tau(I - L) = 0$ subject to $\|\tau\|_1 = 1$
        for $\tau = (\tau_1, \tau_2, \ldots, \tau_K)$.

3. Disaggregation (block Gauss-Seidel, BGS, iteration using $\tau$):
   (a) Compute $z^{(it)} = (\tau_1 \frac{\pi_1^{(it-1)}}{\|\pi_1^{(it-1)}\|_1}, \tau_2 \frac{\pi_2^{(it-1)}}{\|\pi_2^{(it-1)}\|_1}, \ldots, \tau_K \frac{\pi_K^{(it-1)}}{\|\pi_K^{(it-1)}\|_1})$.
   (b) Solve the $K$ nonsingular systems of which the $i$th is given by
       $\pi_i^{(it)}(I - P_{ii}) = b_i^{(it)}$ for $\pi_i^{(it)}$, $i = 1, 2, \ldots, K$, where $b_i^{(it)} = \sum_{j>i} z_j^{(it)} P_{ji} + \sum_{j<i} \pi_j^{(it)} P_{ji}$.

4. Test $\pi^{(it)}$ for convergence. If the desired accuracy is attained,
   then stop and take $\pi^{(it)}$ as the stationary probability vector of $P$.
   Else set $it = it + 1$ and go to step 3.
Implementation

Aggregation:

- $L$ of order $K = \prod_{k=0}^{m-1} n_k$ is computed at the outset and solved once for its stationary vector $\tau$.

- $L$ is also lumpable if $m > 1$.

- When $K$ is on the order of hundreds, one may use Gaussian elimination (GE), or the method of Grassmann-Taksar-Heyman (GTH) if $L$ is relatively ill-conditioned.

- Else one may use IAD with a lumped matrix of order $\prod_{k=0}^{m'-1} n_k$, where $1 < m' < m$, or IAD with a nearly completely decomposable, NCD (Meyer ’89), partitioning if $L$ is relatively ill-conditioned.

- Sufficient space must be allocated to store $L$. 
Implementation (continued)

Disaggregation:

- \( P_{ij} = \sum_{e=0}^{E-1} \xi_{ij}^{(e)} T_i^{(e)} \), where \( \xi_{ij}^{(e)} = \prod_{l=0}^{m-1} P_e^{(l)}(sA^{(l)}, s'A^{(l)}) \)
  and \( T_i^{(e)} = \bigotimes_{k=m}^{N-1} P_e^{(k)} \) (from Theorem 1)

We remark that the prob. \( P_e^{(l)}(sA^{(l)}, s'A^{(l)}) \) may be a function of \([sA^{(0)}, sA^{(1)}, \ldots, sA^{(l-1)}]\) for \( l \in \{1, 2, \ldots, m - 1\} \), but is still fixed for the particular mapping \( i \leftrightarrow (sA^{(0)}, sA^{(1)}, \ldots, sA^{(m-1)}) \).

Then

\[
\begin{align*}
b_i^{(it)} &= \sum_{j>i} \sum_{e=0}^{E-1} \xi_{ji}^{(e)} \xi_{ji}^{(e)} T_j^{(e)} + \sum_{j<i} \sum_{e=0}^{E-1} \pi_j^{(it)} \xi_{ji}^{(e)} T_j^{(e)} \bigg) \bigg) + \sum_{j<i} \sum_{e=0}^{E-1} \xi_{ji}^{(e)} \pi_j^{(it)} T_j^{(e)} \bigg) \bigg) \\
&= \sum_{j>i} \sum_{e=0}^{E-1} \xi_{ji}^{(e)} \xi_{ji}^{(e)} T_j^{(e)} + \sum_{j<i} \sum_{e=0}^{E-1} \xi_{ji}^{(e)} \pi_j^{(it)} T_j^{(e)} \bigg) \bigg) \\
&\text{for } i = 1, 2, \ldots, K.
\end{align*}
\]
Since $T_j^{(e)}$ is composed of $(N - m)$ tensor products, the vector-matrix multiplications $z_j^{(it)} T_j^{(e)}$ and $\pi_j^{(it)} T_j^{(e)}$ turn out to be expensive operations. They are performed a total of $K(K - 1)E$ times during each iteration ⇒ bottleneck of iterative solver.

Subvectors $z_j^{(it)} T_j^{(e)}$ and $\pi_j^{(it)} T_j^{(e)}$ in the two summations appear in the computation of multiple $b_i^{(it)}$. Therefore, at iteration $it$, these subvectors of length $\prod_{k=m}^{N-1} n_k$ can be computed and stored when they are encountered for the first time for a specific pair of $j$ and $e$, and then they can be scaled by $\xi_j^{(e)}$ whenever necessary.

This improvement comes at the expense of $E(K - 2)$ vectors of length $\prod_{k=m}^{N-1} n_k$, that is roughly $E$ vectors of length $n$.

In summary, the proposed solver is limited by $\max(K^2, (E + 2)n)$ amount of double precision storage assuming that $L$ is stored in two dimensions. The 2 vectors of length $n$ are used to store the previous and current approximations of the solution.
Numerical Results

- Algorithm 1 is implemented in C++ as part of the s/w package PEPS (Plateau, Fourneau, Lee ’88).
- The solver is timed on a Pentium III with 128 MB of RAM under Linux although most problems could fit into 64 MB.
- We order the automata as $A^{(3)}, A^{(4)}, \ldots, A^{(V+2)}, A^{(1)}, A^{(0)}, A^{(2)}$ and choose $m = N - 2$
  ⇒ $L$ to be solved in step 2 is of order $K = 4^V(C + 1)$ and each of the $K$ n.s. systems to be solved in step 3.b is of order $2(B + 1)$.
- We use a tolerance of $10^{-8}$ on the appr. error $\|\pi^{(it)} - \pi^{(it-1)}\|_2$ in step 4 (we remark that $\|\pi^{(it)} - \pi^{(it)} P\|_2 < \|\pi^{(it)} - \pi^{(it-1)}\|_2$ upon convergence in all experiments).
- Regarding the solver for $L$, we use GTH as discussed in (Dayar & Stewart ’96) when $K$ is on the order of hundreds.
- When $L$ is of considerable size and density with # of nonzeros on
the order of millions, we solve it using sparse IAD as discussed in (Dayar & Stewart ’96) with a tolerance of $10^{-12}$ on its appr. error. In doing this, when $L$ is NCD with a small $\|F\|_\infty$, hence ill-conditioned, we employ an NCD partitioning. Otherwise we take advantage of the fact that $L$ is also lumpable and try to use a balanced partitioning by separating the first $(N - 2)$ automata in the chosen ordering into two subsets.

- For all combinations of the integer parameters we considered, there is sufficient space to factorize in sparse format the $K$ diagonal blocks in step 3.b
  $\Rightarrow$ we use sparse forward and back substitutions to solve the $K$ nonsingular systems at each iteration of Algorithm 1.

- If this had not been the case, we would suggest using point GS as discussed in (Uysal & Dayar ’98) with a max. of 100 iterations and a tolerance of $10^{-3}$ on the appr. error for the $K$ nonsingular systems at each iteration of Algorithm 1 (Dayar & Stewart ’00).
1st Set of Experiments

Parameters:
\[(p_n, p_h, p_s) = C(5 \times 10^{-6}, 10^{-5}, 5 \times 10^{-6}), \ M = C.\]

- In Figure 1, we plot \(P_{c\text{block}}\) and \(P_{c\text{drop}}\) versus \(C\).
- \(P_{c\text{drop}}\) and \(P_{c\text{block}}\) are independent of ABR and VBR traffic.
- \(P_{v\text{drop}}\) and \(P_{v\text{block}}\) are independent of ABR and CBR traffic.
- When \(C = V = M\) and the real valued parameters for CBR and VBR traffic are the same, we have \(P_{c\text{drop}} = P_{v\text{drop}}\) and \(P_{c\text{block}} = P_{v\text{block}}\).
- \(P_{c\text{block}} > P_{c\text{drop}}\) and larger \(C\) implies smaller \(P_{c\text{block}}\) and \(P_{c\text{drop}}\).
Figure 1. $P_{c_{\text{block}}}$ (solid) and $P_{c_{\text{drop}}}$ (dashed) vs $C$. 
2nd Set of Experiments

Parameters:

\((C, V, B) \in \{(8, 2, 15), (12, 3, 15), (16, 4, 15)\} \Rightarrow (\text{small}, \text{medium}, \text{large})\)

\((p_{d0}, p_{d1}, p_{d2}, p_{d3}) = (0.05, 0.1, 0.25, 0.6), \lambda \in \{0.1, 0.3, 0.5, 0.7, 0.9\},\)

\(S_C \in \{1, 10\}, (p_n, p_h, p_s) = C(5 \times 10^{-6}, 10^{-5}, 5 \times 10^{-6}), M = V,\)

\(\lambda_v = 0.5, S_{Cv} = 10, (p_{vn}, p_{vh}, p_{vs}) = V(5 \times 10^{-6}, 10^{-5}, 5 \times 10^{-6}),\)

\((p_{empty}, p_{busy}) = (0.9, 0.1).\)

- In Figure 2, we plot \(P_a\) versus \(\lambda\) for the problems \text{small}, \text{medium}, and \text{large} when (a) \(S_C = 1\), (b) \(S_C = 10\).

- \(P_a\) increases with \(\lambda\) and \(S_C\) though the increase with \(S_C\) happens slowly.
Figure 2(a). \(P_a\) vs \(\lambda\) for (\(\diamondsuit, \triangle, \star\)) = (small, medium, large) when \(S_C = 1\).
Figure 2(b). $P_a$ vs $\lambda$ for $(\diamond, \triangle, \ast) = (small, medium, large)$ when $S_C = 10$. 
3rd Set of Experiments

Parameters:
\((C, V, B) = (4, 4, 15), (p_{d0}, p_{d1}, p_{d2}, p_{d3}) = (0.05, 0.1, 0.25, 0.6), \lambda = 0.5, S_C \in \{1, 10\}, (p_n, p_h) = (p_{vn}, p_{vh}) = C(5 \times 10^{-6}, 10^{-5}), M = 4, \lambda_v = 0.5, S_{Cv} = 10, (p_{empty}, p_{busy}) = (0.9, 0.1)\).

- In Figure 3, we plot \(P_a\) versus \(p_{vs} = iV \times 10^{-6}, i \in \{1, 3, 5, 7, 9\}\), for fixed \(p_s = 5C \times 10^{-6}\) using a dashed curve, and we plot \(P_a\) versus \(p_s = iC \times 10^{-6}, i \in \{1, 3, 5, 7, 9\}\), for fixed \(p_{vs} = 5V \times 10^{-6}\) using a solid curve on the same graph when (a) \(S_C = 1\), (b) \(S_C = 10\).

- \(P_a\) depends slightly weaker on \(p_{vs}\) than on \(p_s\) since data packets can use VBR slots when a VBR connection is active but in low-state.
Figure 3(a). $P_a$ vs $p_{vs}$ ($p_s = 5C \times 10^{-6}$, dashed), $P_a$ vs $p_s$ ($p_{vs} = 5V \times 10^{-6}$, solid) for $(C, V, B) = (4, 4, 15)$ and $\lambda = 0.5$ when $S_C = 1$. 
Figure 3(b). $P_a$ vs $p_{vs}$ ($p_s = 5C \times 10^{-6}$, dashed), $P_a$ vs $p_s$ ($p_{vs} = 5V \times 10^{-6}$, solid) for $(C, V, B) = (4, 4, 15)$ and $\lambda = 0.5$ when $S_C = 10$. 
4th Set of Experiments

Parameters:
\((C, V, B) = (4, 4, 15), (p_{d0}, p_{d1}, p_{d2}, p_{d3}) = (0.05, 0.1, 0.25, 0.6), \lambda = 0.5,\)
\(S_C \in \{1, 10\}, (p_n, p_h, p_s) = (p_{vn}, p_{vh}, p_{vs}) = C(5 \times 10^{-6}, 10^{-5}, 5 \times 10^{-6}),\)
\(M = 4, S_{C\_v} = 10.\)

- In Figure 4, we plot \(P_a\) versus \(\lambda_v\) when
  (a) \((p_{empty}, p_{busy}) = (0.9, 0.1),\) (b) \((p_{empty}, p_{busy}) = (0.5, 0.5).\)
- When the VBR arrival process behaves more like the CBR arrival process as in part (b), \(P_a\) is larger.
- The increase in \(P_a\) with respect to \(\lambda_v\) is smoother for larger \(S_C\).
Figure 4(a). $P_a$ vs $\lambda_v$ when $\lambda = 0.5$, $S_C = 1$ (solid), $S_C = 10$ (dashed) for $(C, V, B) = (4, 4, 15)$ and $S_{C_v} = 10$ when $(p_{empty}, p_{busy}) = (0.9, 0.1)$. 
Figure 4(b). $P_a$ vs $\lambda_v$ when $\lambda = 0.5$, $S_C = 1$ (solid), $S_C = 10$ (dashed) for
$(C, V, B) = (4, 4, 15)$ and $S_{Cv} = 10$ when $(p_{empty}, p_{busy}) = (0.5, 0.5)$. 
Table 1. Timing results in seconds and # of IAD iterations for Figure 2.

<table>
<thead>
<tr>
<th>Problem</th>
<th>$S_C$</th>
<th>$\lambda = 0.1$</th>
<th>$\lambda = 0.3$</th>
<th>$\lambda = 0.5$</th>
<th>$\lambda = 0.7$</th>
<th>$\lambda = 0.9$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Time</td>
<td>#it</td>
<td>Time</td>
<td>#it</td>
<td>Time</td>
<td>#it</td>
</tr>
<tr>
<td>small</td>
<td>1</td>
<td>8 18</td>
<td>9 21</td>
<td>8 17</td>
<td>9 20</td>
<td>11 27</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>8 18</td>
<td>8 19</td>
<td>8 19</td>
<td>15 35</td>
<td>38 89</td>
</tr>
<tr>
<td>medium</td>
<td>1</td>
<td>77 9</td>
<td>85 10</td>
<td>85 10</td>
<td>86 11</td>
<td>138 18</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>146 19</td>
<td>168 22</td>
<td>94 12</td>
<td>309 41</td>
<td>824 110</td>
</tr>
<tr>
<td>large</td>
<td>1</td>
<td>889 4</td>
<td>990 5</td>
<td>803 4</td>
<td>1742 9</td>
<td>3622 19</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>3813 20</td>
<td>4381 23</td>
<td>1005 5</td>
<td>8531 45</td>
<td>23119 122</td>
</tr>
</tbody>
</table>

- The reported times in Table 1 correspond to the iterative part of Algorithm 1 and they exclude the time spent for solving $L$.

- Lumped matrices of the small problems are all the same. The same is true for the lumped matrices of the medium and large problems.

This is simply because the parameters that we alter in the experiments of Figure 2 are only those of $sA^{(0)}$, which happens to be among the last two automata in the chosen order of automata.

Even though $\alpha$ and $\beta$ change, row sums of $P_{ij}$ in equation (3) are the same because $P_{sab}^{(0)}u = u$ for all $a, b \in \{0, 1, 2\}$ (from Theorem 1).
• Among small, medium, and large, the DTMC of only the first can be stored on the target architecture.

• The lumpable partitionings we consider for the problems in Figure 2 are not NCD. However, there exist highly NCD partitionings for each one (with \( \|F\|_\infty \approx 10^{-5} \)), and therefore they are all very ill-conditioned problems.

• Nevertheless, we are fortunate that Algorithm 1 does not require NCD partitionings for convergence (Marek & Mayer ’98).

• We solve \( L (n_L = 144 \text{ and } nz_L = 7,644) \) of the small problem using GTH in nearly 0 seconds.

• We solve \( L (n_L = 832 \text{ and } nz_L = 188,094) \) of the medium problem in 0.5 seconds and 16 iterations using sparse IAD with an NCD partitioning of 4 blocks (with orders varying between 117 and 351) and a degree of coupling \( \approx 10^{-5} \).
• We solve $L$ ($n_L = 4,352$ and $nz_L = 3,980,512$) of the large problem in 48.7 seconds and 22 iterations using sparse IAD with an NCD partitioning of 16 blocks (with orders varying between 17 and 1,377) and a degree of coupling $\approx 10^{-4}$.

• Hence, much larger problems than can fit explicitly into a given architecture may be solved by the proposed approach.

• Regarding the # of iterations taken by Algorithm 1 to convergence, the highest values are for $\lambda \in \{0.7, 0.9\}$ and $S_C = 10$.

They are iteration #s greater than or equal to 35, and are for the cases in which $\alpha$ and $\beta$ are highly unbalanced.

• In general, the solution times are very satisfactory if we keep in mind the # of nonzeros of the underlying DTMC.
Conclusion

- Result on lumpability of discrete-time SANs.

- Efficient IAD algorithm for discrete-time SANs that satisfy the conditions of lumpability.

- Depending on the interaction of automata in the discrete-time SAN, the user has some flexibility in ordering the automata and choosing the partitioning to be used by the IAD algorithm.