Stability and performance in cyclic network

Anne Bouillard (Nokia Bell Labs France)

FORMATS 2019
Performances in networks

Objective: deterministic performance guarantees

Compute the maximum time it takes for a packet to cross the system (Worst-case delay)
Network calculus

Real data

Real input traffic → abstraction
network element → abstraction

↓

Delay / backlog → pessimism

(min,plus) functions

arrival curve
service curve

↓ (min,plus)-operators

Upper bound on the delay / backlog

Two kinds of pessimism

1. The abstraction
2. The (min,plus) operations
Network calculus

- Theory developed in the 1990’s by R.L. Cruz, then developed and popularized by C.S. Chang and J.-Y. Le Boudec.
- Filtering theory in the (min,plus) algebra.
- Applications:
  - Internet: video transmission (VoD),
  - Load-balancing in switches [Birkhoff-von Neumann switches, C.S. Chang]
  - Embedded systems: AFDX (Avionics Full Duplex) [Rockwell-Collins software used to certify A380], Networks-on-chip
State of the art and contribution: Feed-forward networks

Many recent results for computing tight bounds feed-forward networks:

- PBOO/PMOO phenomena [Schmitt et al 2008]
- Linear programming solutions [B. et al, 2010]
  - tight bounds
  - the problem is NP-hard
  - polynomial for tandem networks
- Exhaustive search / pay segregation only once [Bondorff et al, 2016]
  - good heuristics to approximate the worst-case performance bounds
- Neural networks [Geyer, 2018]
  - learning the good heuristic
State of the art and contribution: Feed-forward networks

Many recent results for computing tight bounds feed-forward networks:

- PBOO/PMOO phenomena \cite{Schmitt2008}
- Linear programming solutions \cite{B2010}
  - tight bounds
  - the problem is NP-hard
  - polynomial for tandem networks
- Exhaustive search / pay segregation only once \cite{Bondorff2016}
  - good heuristics to approximate the worst-case performance bounds
- Neural networks \cite{Geyer2018}
  - learning the good heuristic
- Tight bounds for tree-networks for the backlog of a subsets of flows and delay
State of the art and contribution: Cyclic networks

Few results in networks with cyclic dependencies

- Computing good stability conditions and performance guarantees for network with cyclic dependencies is an open issue
- Obtaining such guarantees would enable more flexible design of systems, with fewer switches.
State of the art and contribution: Cyclic networks

Few results in networks with cyclic dependencies

- Computing good stability conditions and performance guarantees for network with cyclic dependencies is an open issue
- Obtaining such guarantees would enable more flexible design of systems, with fewer switches.

- Flow-based bounds: fix-point/stopped time method
  - "classical" [Cruz 1994]
  - PMOO [Amari et Mifdaoui, 2017]

- Backlog-based bounds: "stability" of the ring [Tassiulas, Georgiadis, 1996]
  Additional assumption: the traffic is upper-bounded in each link.

- instability results from adversarial method [Andrews, 2001]
State of the art and contribution: Cyclic networks

Few results in networks with cyclic dependencies

- Computing good stability conditions and performance guarantees for network with cyclic dependencies is an open issue
- Obtaining such guarantees would enable more flexible design of systems, with fewer switches.

- Flow-based bounds: fix-point/stopped time method
  - ”classical” [Cruz 1994]
  - PMOO [Amari et Mifdaoui, 2017]

- Backlog-based bounds: ”stability” of the ring [Tassiulas, Georgiadis, 1996]
  Additional assumption: the traffic is upper-bounded in each link.

- instability results from adversarial method [Andrews, 2001]

- Improve the fix-point method to combine flow and backlog-based bounds
Network calculus framework

Networks with cyclic dependencies

Computing performance bounds in feed-forward networks

Performances in cyclic network

Conclusion
Cumulative processes

- A : $\mathbb{R}_+ \rightarrow \mathbb{R}_{\text{min+}}$: process of the cumulative arrivals, non-decreasing function
- B : $\mathbb{R}_+ \rightarrow \mathbb{R}_{\text{min+}}$: process of the cumulative departures, non-decreasing function
- Causality constraint: $A \geq B$
Arrival and service curves

Arrival curve

A is constrained by the function $\alpha$ if

$$\forall 0 \leq s \leq t, \quad A(t) - A(s) \leq \alpha(t - s).$$
Arrival and service curves

Arrival curve

A is constrained by the function $\alpha$ if

$$\forall 0 \leq s \leq t,
A(t) - A(s) \leq \alpha(t - s).$$

data

$\alpha$

A

time

A

B
Arrival and service curves

A is constrained by the function $\alpha$ if
\[ \forall 0 \leq s \leq t, \quad A(t) - A(s) \leq \alpha(t - s). \]
Arrival and service curves

**Arrival curve**
A is constrained by the function \( \alpha \) if \( \forall 0 \leq s \leq t, \)
\[
A(t) - A(s) \leq \alpha(t - s).
\]

**Strict service curve**
A network element guarantees \( \beta \) for A if, while system not empty, B satisfies
\[
B(t) \geq B(s) + \beta(t - s).
\]
**Arrival and service curves**

A is constrained by the function $\alpha$ if $\forall 0 \leq s \leq t$, $A(t) - A(s) \leq \alpha(t - s)$.

A network element guarantees $\beta$ for $A$ if, while the system is not empty, $B(t) \geq B(s) + \beta(t - s)$.

**Arrival curve**

**Strict service curve**
From constraints to performance bounds

**Maximum backlog:**
\[ B_{\text{max}} = \sup_{t \geq 0} A(t) - B(t) \]

**Maximum delay:**
\[ D_{\text{max}} = \inf \{ d \mid \forall t \in \mathbb{R}_+, B(t + d) \geq A(t) \} \]

**Performance bounds**
- \[ B_{\text{max}} \leq \alpha \otimes \beta(0) = v(\alpha, \beta) = \sup \{ \alpha(t) - \beta(t) \mid t \geq 0 \} \]
- \[ D_{\text{max}} \leq h(\alpha, \beta) = \inf \{ \forall t \geq 0, d \geq 0 \mid \alpha(t) \leq \beta(t + d) \} \] (for FIFO per flow).
Hypotheses

- \( m \) token-bucket arrival curves: \( \alpha_i(t) = b_i + r_i t \);
- \( n \) rate-latency strict service curves: \( \beta^{(j)}(t) = R_j(t - T_j)_+ \).
Network calculus framework

Networks with cyclic dependencies

Computing performance bounds in feed-forward networks

Performances in cyclic network

Conclusion
Stability in cyclic networks

Consider a server offering a strict service curve $\beta : t \mapsto R(t - T)_+$ and a flow crossing it, with arrival curve $\alpha : t \mapsto b + rt$.

- This server is said unstable if its worst-case backlog is unbounded: $R < r$;
- This server is said critical if its worst-case backlog is bounded, but the lengths of its backlogged periods are not bounded bounded: $R = r$;
- This server is said stable if the length of its backlogged periods is bounded: $R > r$.

**Definition (Global stability)**

A network is globally stable if for all its servers, the length of the maximal backlogged period is bounded.
Fix-point method

(service curves and arrival curves of exogenous arrivals are constants of the problem)

\[ \alpha^{(3)}_2 = H^{(1)}_1(\alpha_1, \alpha_2) \]

\[ \alpha^{(4)}_2 = H^{(3)}_1(\alpha^{(3)}_1, \alpha^{(3)}_3) \]

We write this equation for each output flow at each server and obtain a system

\[ \alpha = H(\alpha) \]

**Lemma**

If the system is stable, then there exists a family \( \alpha = (\alpha_{i,j})_{i,j} \) of arrival curves for the flows \( (F^{(j)}_i) \) such that \( \alpha \leq H(\alpha) \).

Take the best arrival curves, they will satisfy every inequality.
Fix-point method

(service curves and arrival curves of exogenous arrivals are constants of the problem)

\[ \alpha_2^{(3)} = H_2^{(1)}(\alpha_1, \alpha_2) \]
\[ \alpha_2^{(4)} = H_2^{(3)}(\alpha_3^{(3)}, \alpha_3^{(3)}) \ldots \]

We write this equation for each output flow at each server and obtain a system

\[ \alpha = H(\alpha) \]

- If service curves are rate-latency and arrival curves token bucket, this is a linear equation: \( b = Mb + N \).
Network calculus framework

Networks with cyclic dependencies

Computing performance bounds in feed-forward networks

Performances in cyclic network

Conclusion
Generic (min,plus) method

1. In the topological order of the servers, for each flow crossing the server:

   **Residual service curve:**
   \[ \beta^{(j)}_1 = (\beta^{(j)} - \alpha^{(j)}_C)_+ \]

   **Output arrival curve:**
   \[ \alpha^{(j+1)}_1 = \alpha^{(j)}_1 \odot \beta^{(j)}_1 \]

2. For the flow of interest

   **End-to-end service curve:**
   \[ \beta = \beta^{(1)}_1 \ast \beta^{(2)}_1 \ast \cdots \ast \beta^{(k)}_1 \]

3. Delay bound: \( h(\alpha_1, \beta) \),
   Backlog bound: \( v(\alpha_1, \beta) \)
Theorem

Consider a tandem network of $n$ servers. The worst-case delay is linear in the bursts and latencies:

$$D = \sum_{j \in N_n} \lambda_j T_j + \sum_{i \in N_m} \mu_i b_i$$

where the coefficients $\lambda_j$ and $\mu_i$ depend only on the arrival and service rates and can be effectively computed in time $O(n^2 + m)$. 
 Tight worst-case delays and backlog for tree networks

This theorem can be adapted to backlog at server n and for tree-topologies:

\[
B = \sum_{j \in N_n} \rho_j T_j + \sum_{i \in N_m} \varphi_i b_i + \sum_{i \in N_p} b_i^* 
\]

where the coefficients \( \rho_j \) and \( \varphi_i < 1 \) depend only on the arrival and service rates and can be effectively computed in time \( O(n^2 + m + p) \).
Sketch of the proof

1. Find properties of a worst-case scenario for the network
   - SDF (Shortest-to-destination first) service policy
   - one backlogged period for each server
   - minimum service
   - maximum arrivals

2. Backward induction on the servers
   - Only the dates of the start of backlogged periods need to be computed
   - they depend only on the amount of data transmitted at those time.
Comparison of the approaches

1. (min,plus):
   - Efficient algorithms (linear in the total length of the flows)
   - Pessimistic performance bounds (as soon as two servers and two flows)
   - Linear in $T_j$ and $b_i$

2. Our approach:
   - Quadratic algorithm in tree network
   - Tight delay bound
   - Linear in $T_j$ and $b_i$
Network calculus framework

Networks with cyclic dependencies

Computing performance bounds in feed-forward networks

Performances in cyclic network

Conclusion
Flow-based decomposition of the network

- (min,plus) approach: at each server for each flow

\[ b_{i,k+1} \leq b_{i,k} + \frac{r_i}{R_j - \sum_{p \in Fl(j) \setminus \{i\}} r_p} \left( \sum_{s \in S_j \setminus \{(i,k)\}} b_s + R_j T_j \right). \]
Flow-based decomposition of the network

- \((\min,\text{plus})\) approach: at each server for each flow

\[
b_{i,k+1} \leq b_{i,k} + \frac{r_i}{R_j - \sum_{p \in \text{Fl}(j) \setminus \{i\}} r_p} \left( \sum_{s \in S_j \setminus \{(i,k)\}} b_s + R_j T_j \right).
\]

- Our approach, with a tree decomposition:

\[
b_{i,k+1} \leq \sum_{s \in S} \varphi_{s}^{i,k+1} b_s + \sum_{\{j|j \sim j_1\}} \rho_{j}^{i,k+1} T_j,
\]
Flow-based decomposition of the network

Linear problem:

Maximize $Qb + C$

such that $b \leq Mb + N$.

- (min,plus) approach: at each server for each flow

$$b_{i,k+1} \leq b_{i,k} + \frac{r_i}{R_j - \sum_{p \in Fl(j) \setminus \{i\}} r_p} \left( \sum_{s \in S_j \setminus \{(i,k)\}} b_s + R_j T_j \right).$$

- Our approach, with a tree decomposition:

$$b_{i,k+1} \leq \sum_{s \in S} \varphi_{s,i,k+1} b_s + \sum_{\{j | j \sim j_1\}} \rho_{j,i,k+1} T_j,$$
Comparison on the ring network

Utilization rate
Backlog upper bound

$n = 10$ servers, $R = 100 \text{Mb.s}^{-1}$, $L = 1 \text{ms}$, $b = 1 \text{Mb}$, $r = \frac{uR}{n}$
Backlog-based decomposition

Idea:

- worst-case bounds for flows ending at the same server do not happen at the same time;
- computing worst-case backlog of all flows following an arc might takes this phenomenon into account.

\[
B_a \leq \sum_{s \in S} \varphi_s^a x_s + \sum_{\{j|j \sim j_1\}} \rho_j^a T_j \\
\leq \sum_{a' \in A'} \left( \max_{s \in S_{a'}} \left( \sum_{s \in S_{a'}} x_s \right) \right) + \sum_{i=1}^{m} \varphi_{(i,1)}^a b_i + \sum_{\{j|j \sim j_1\}} \rho_j^a T_j \\
\leq \sum_{a' \in A'} \left( \max_{s \in S_{a'}} \varphi_s^a \right) B_{a'} + \sum_{i=1}^{m} \varphi_{(i,1)}^a b_i + \sum_{\{j|j \sim j_1\}} \rho_j^a T_j.
\]
Ring stability revisited

**Theorem (TG96, LT04)**

“The ring is stable” under assumption for stability of each server
Additional assumption: the traffic is upper-bounded in each link.

**Our approach**

$$B = \sum_{j \in \mathbb{N}_n} \rho_j T_j + \sum_{i \in \mathbb{N}_m} \varphi_i b_i + \sum_{i \in \mathbb{N}_p} b_i^*$$

where the coefficients $\rho_j$ and $\varphi_i < 1$ depend only on the arrival and service rates.

$$B \leq C + \varphi B$$

where $\varphi = \sup \varphi_j^n < 1$ and $B \leq \frac{C}{1 - \varphi}$. 
Comparison on the Ring Network

\[ n = 10 \text{ servers}, R = 100\text{Mb.s}^{-1}, L = 1\text{ms}, b = 1\text{Mb}, r = \frac{uR}{n} \]
Combining Flow-based and Backlog-based bounds

New set of linear constraints:

\[ \mathcal{L} = \left\{ \begin{array}{l}
 b_s \leq \sum_{s' \in S} \varphi^s_{s'} x^s_{s'} + C_s, \quad \forall s \in S \\
 B_a \leq \sum_{s' \in S} \varphi^a_{s'} x^a_{s'}, + C_a, \quad \forall a \in A^r \\
 0 \leq x^s_{s'}, \quad \forall s' \in S, s \in S \cup A^r \\
 \sum_{s' \in a} x^s_{s'} \leq B_a, \quad \forall a \in A^r, s \in S \cup A^r
\end{array} \right\}, \]

Number of constraints:

- Flow-based: \( O(k) \)
- Backlog-based: \( O(a) \)
- Combination: \( O(k + a)^2 \)

where \( k \) is the number of flows is the tree-decomposition and \( a \) the number of arcs removed for this decomposition.
Comparison on the Ring Network

\[ n = 10 \text{ servers, } R = 100 \text{Mb.s}^{-1}, L = 1 \text{ms}, b = 1 \text{Mb}, r = \frac{uR}{n} \]
Two-Ring network

Utilization rate

Backlog bound ($b$)

$LP_{F,B}$

$LP_F$

$LP_B$

(min, plus)
Network calculus framework

Networks with cyclic dependencies

Computing performance bounds in feed-forward networks

Performances in cyclic network

Conclusion
Conclusion

- A new efficient algorithm to compute tight worst-case delays and backlog.
- Application to networks with cyclic dependencies:
  - best stability conditions
  - stability of the ring without additional assumptions
- Implementation in a Python package: https://github.com/nokia/NCBounds

Future work

- Extension to feed-forward networks (we conjecture that a simple generalization can lead to the same approximation with one linear program)
- what is the best decomposition?
- Extension to some service policies (FIFO, GPS for example), maximum service rate...