# Control of Fork-Join Networks in Heavy-Traffic

Asaf Zviran

Based on MSc work under the guidance of
Rami Atar (EE) and Avishai Mandelbaum (IE&M)

Industrial Engineering and Management
Technion

May 2011

# Outline

# Introduction

# Main Idea and Motivation

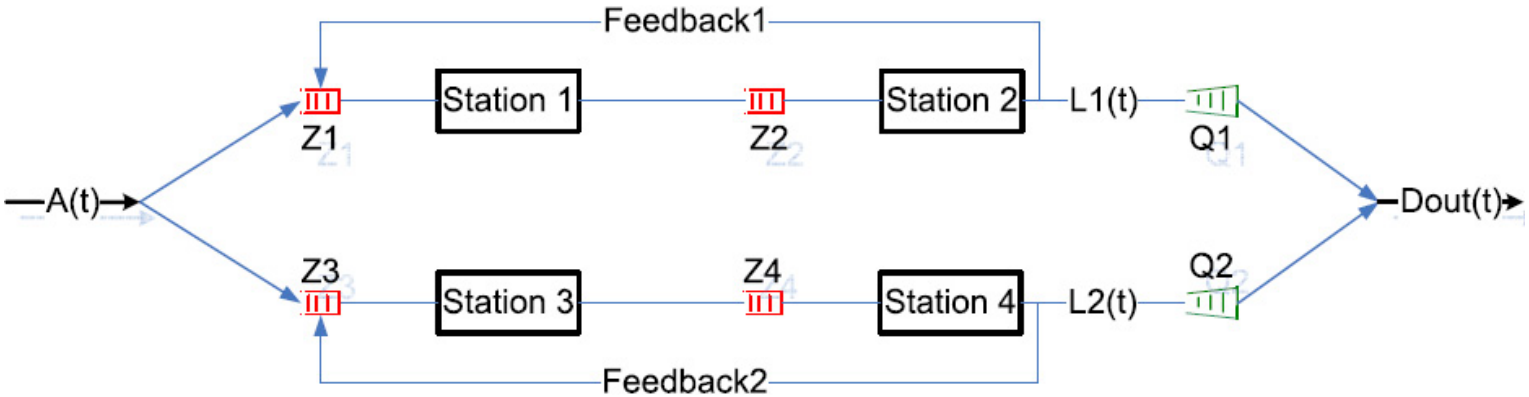
Parallel processing systems are commonly encountered in many human ventures. A Fork-Join network is considered as a typical model of parallel processing systems with arrival and departure synchronization. Our generalized fork-join model allows probabilistic feedback.

**Main idea**: A simple global adaptive scheduling control is used to increase throughput and reduce synchronization delays.

**Motivation: Parallel Processing Application**

- Communication through distributed channel.
- Data streaming through web/cellular application.
- Large scale parallel computing and/or multi-core utilization.
- Multi-Project scheduling problem.
- Health-care systems (service networks).
- Molecular Biology: Transcriptional networks / Ribosome scheduling.

# Network Model



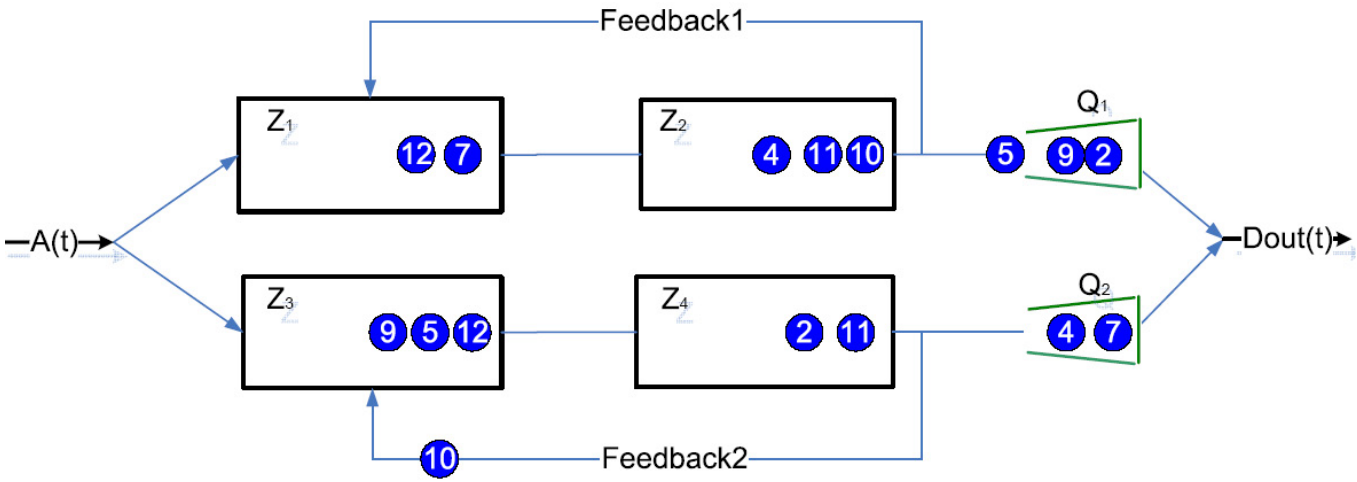

In this network a job arriving to the system "forks" to tasks processed simultaneously in two parallel processing routes, each route consisting of two service stations in tandem, followed by a probabilistic feedback.

The completed task in each route waits in the synchronization queues ($Q_1 \& Q_2$) until tasks of both routes are completed.

Complete job "joins" and depart from the system only after the completion of the tasks associated with it.

# Departure Synchronization

In our model, tasks are associated uniquely with customers. They are hence non-exchangeable in the sense that one can not join tasks associated with different customers.




Conclusion- Customers' disorder increase Idle-time in the join nodes and decrease throughput.

In contrast to Assembly network with exchangeable customers, in which

**Complementarity Condition:** $Q_1(t) \wedge Q_2(t) = 0$.

# Methodology: Asymptotic Analysis

We shall work in the conventional Heavy-Traffic regime.

The precise formulation of Heavy-Traffic limits requires the construction of a "sequence of systems", indexed by $n = 1, 2, \ldots$

Assume that the following relations hold:

- Average arrival rate: $\lambda^n = \lambda \cdot n + \hat{\lambda} \cdot \sqrt{n} + o(\sqrt{n})$.
- Average service rates: $\mu_j^n = \mu_j \cdot n + \hat{\mu}_j \cdot \sqrt{n} + o(\sqrt{n})$.
- Heavy Traffic Condition - Define the traffic intensity at station j to be $\rho_j^n$; it is assumed that there exists deterministic numbers $-\infty < \theta_j < \infty$, such that $n^{\frac{1}{2}}(\rho_j^n - 1) \longrightarrow_n \theta_j$, as $n \longrightarrow \infty$, for each station j.

Notation - Throughout the presentation, we shall use the scaling

$$\hat{Q}_i^n(t) = \frac{Q_i^n(t)}{\sqrt{n}}.$$

# Existing Research

Heavy-Traffic Analysis of Fork-Join Networks:

- Processing Networks With Parallel and Sequential Tasks (Single-Class Feedforward Networks): Viên Nguyen (1993).
- The Trouble With Diversity: Fork-Join Networks With Heterogeneous Customer Population (Multi-Class Feedforward Networks): Viên Nguyen (1994).
- Heavy traffic analysis of state-dependent fork-join queues with triggers : Leite and Fragoso (2008).

Scheduling Policies in Fork-Join Networks:

- Non-Preemptive Priorities in Simple Fork-Join Queues (Multi-Class Feedforward Networks): Halfin and Avi-Itzhak[1] (1991).
- Multi-Project Scheduling and Control: Cohen, Mandelbaum and Shtub (2004).
- Generalized parallel-server fork-join queues with dynamic task scheduling: Squillante, Zhang, Sivasubramaniam and Gautam (2008).

[1] Emeritus professor of IE&M

# Control Problem Formulation

# Optimality Criteria

**Definitions**

- Exact Optimality: Maximum throughput in the sense of maximum achievable departures on any finite time-interval $[0, T]$.
- Asymptotic Optimality: Policy $\gamma$ is asymptotically optimal if for any other policy $\beta$ and for any fixed T,

$$\hat{D}_{out}^{n,\gamma}(T) \geq \hat{D}_{out}^{n,\beta}(T) - \epsilon(n), \quad \epsilon(n) \to 0, \text{ in probability .}$$

**Proposition**

*These conditions are equivalent to the definitions above*

- *Exact Optimality:* $Q_1(T) \wedge Q_2(T) = 0$, *a.s.*,
- *Asymptotic Optimality:* $\hat{Q}_1^n(T) \wedge \hat{Q}_2^n(T) \to 0$, *in probability ;*

*for any fixed T.*

Note: These conditions indicate an asymptotic equivalence between fork-join and assembly networks in Heavy-Traffic.

# Generalization

Note that both the exact and asymptotic conditions may be generalized to any number of parallel processing routes. For M processing routes:

## Proposition

*Equivalent conditions:*

- *Exact Optimality:* $\bigwedge_{i\in\{1,..,M\}}(Q_i(T)) = 0$, *a.s.;*
- *Asymptotic Optimality:* $\bigwedge_{i\in\{1,..,M\}}(\hat{Q}_i^n(T)) \to 0$, *in probability ;*

*for any fixed $T$.*

According to the following relations

- $M \cdot N(t) = \sum_j (Z_j(t)) + \sum_{i=1}^{M}(Q_i(t))$;
- $\sum_{i=1}^{M} Q_i(t) = \sum_{i=1}^{M} (L_i(t) - \bigwedge_{i\in\{1,..,M\}}(L_i(t))) + M \cdot \bigwedge_{i\in\{1,..,M\}}(Q_i(t))$;

# Main Result:

# Adaptive Task Scheduling

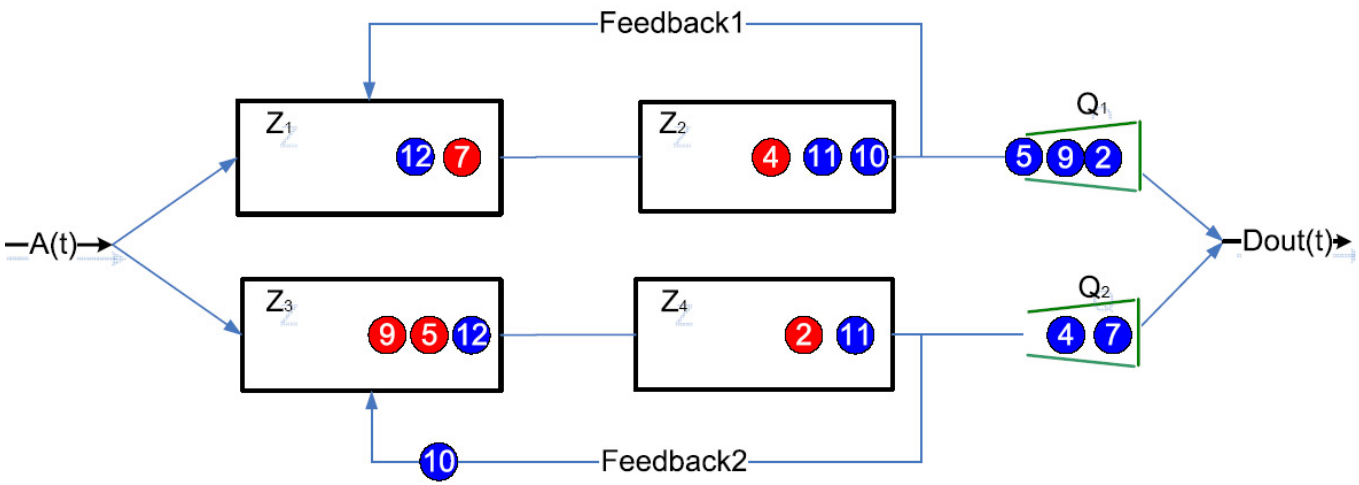
# Control Policy

Definition: At each route, assign preemptive priority to customers whose service has completed in the other route.

The definition of the policy creates a natural division of the customers into two classes:

- LP (Low Priority) Customers: Customers whose service is still incomplete in both routes.
- HP (High Priority) Customers: Customers whose service was completed in one of the routes but is still incomplete in the other.

Define FCFS priority policy within each class. Then the policy is fully defined.



# Asymptotically Optimal Control

## Theorem

*Given the system and control policy defined above, and a fixed $T$:*

$$max_{t\in[0,T]}\{\hat{Z}^{n,H}_{1,2}(t) \wedge \hat{Z}^{n,H}_{3,4}(t)\} \to 0, \quad \textit{in probability},$$

$$\textit{where} \begin{cases} \hat{Z}^{n,H}_{1,2}(t) = \hat{Z}^{n,H}_{1}(t) + \hat{Z}^{n,H}_{2}(t); \\ \hat{Z}^{n,H}_{3,4}(t) = \hat{Z}^{n,H}_{3}(t) + \hat{Z}^{n,H}_{4}(t). \end{cases}$$

Note that

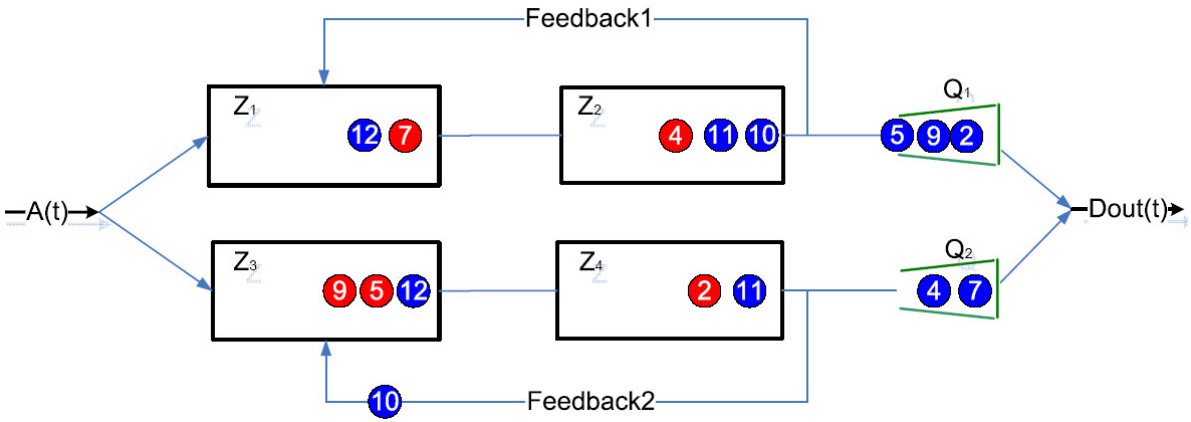
$$\hat{Z}^{n,H}_{1,2}(t) = \hat{Q}^{n}_{2}(t) \quad \text{and} \quad \hat{Z}^{n,H}_{3,4}(t) = \hat{Q}^{n}_{1}(t).$$

We conclude that

The scaled process $\hat{Q}^{n}_{1}(t) \wedge \hat{Q}^{n}_{2}(t)$ converges uniformly to 0, in probability.

# About the Proof (1)

The High-Priority Arrival ("Birth") Processes in single station perspective is hard to define in the sense of probabilistic distributions. Hence, the lemmas can not be proven by "simple" fluid and diffusion limits. Example

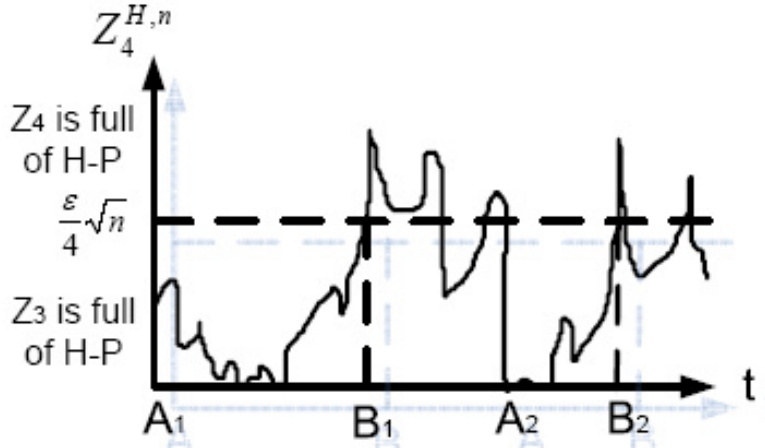



But on the event considered: $Z_{3,4}^{n,H}(s) \geq \frac{\epsilon}{3}\sqrt{n}, \quad \forall s \in (\tau, \sigma)$. Therefore, we note heuristically that

- When $Z_4^H$ is full with HP customers then the idleness process is non-increasing;
- When $Z_3^H$ is full with HP customers then the arrival process to $Z_4^H$ is in the order of n. Hence, the Idleness should be in the order of $n^{-\frac{1}{2}}$;
- What about the transitions between states?

# About the Proof (2)

Illustration of $Z_4^H$ sample-path:




The proof consists of the following steps:

- The number of transitions between states in $[\tau, \sigma)$ equals to the number of down-crossings by $Z_4^H$. We showed that the number of down-crossings is tight.
- In every $[A_i, B_i)$ period, the increase in $\hat{I}_4^{n,H}([A_i, B_i))$ is bounded. Recall that the HP arrival rate in these periods is in the order of n.
- Therefore, $\mathrm{P}(I_4^{n,H}[\tau, \sigma] > n^{-\frac{1}{2}+\delta}) \longrightarrow_n 0$. □

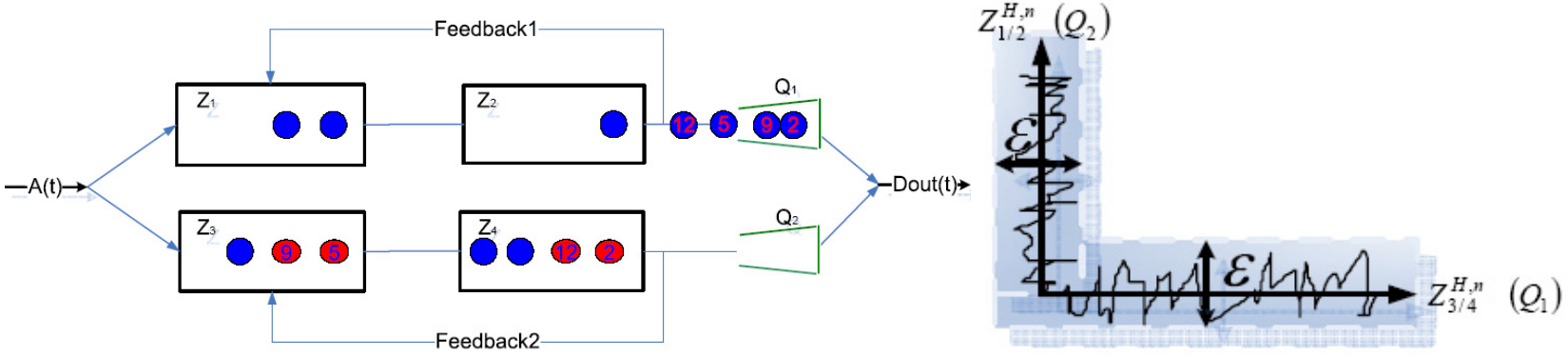
The central ingredient of the proof requires the use of the tightness of the potential service processes.

# High-Priority Dynamics

Recall that $\hat{Z}^{n,H}_{1,2}(t) = \hat{Q}^n_2(t)$ and $\hat{Z}^{n,H}_{3,4}(t) = \hat{Q}^n_1(t)$. We showed that

$$\hat{Q}^n_1(t) \wedge \hat{Q}^n_2(t) \text{ converges uniformly to 0, in probability.}$$

Note the following properties




- Synchronization queues state space is reduced to a one-dimensional state space.
- A critical route determines the customers' departure order. The critical route index is a random variable defined by the service dynamic ($argmin_{i\in\{1,2\}} L_i(t)$).
- An asymptotic equivalence appears between fork-join and assembly network dynamics, in the sense of synchronization queue length and throughput processes.

# Comment about Generalization

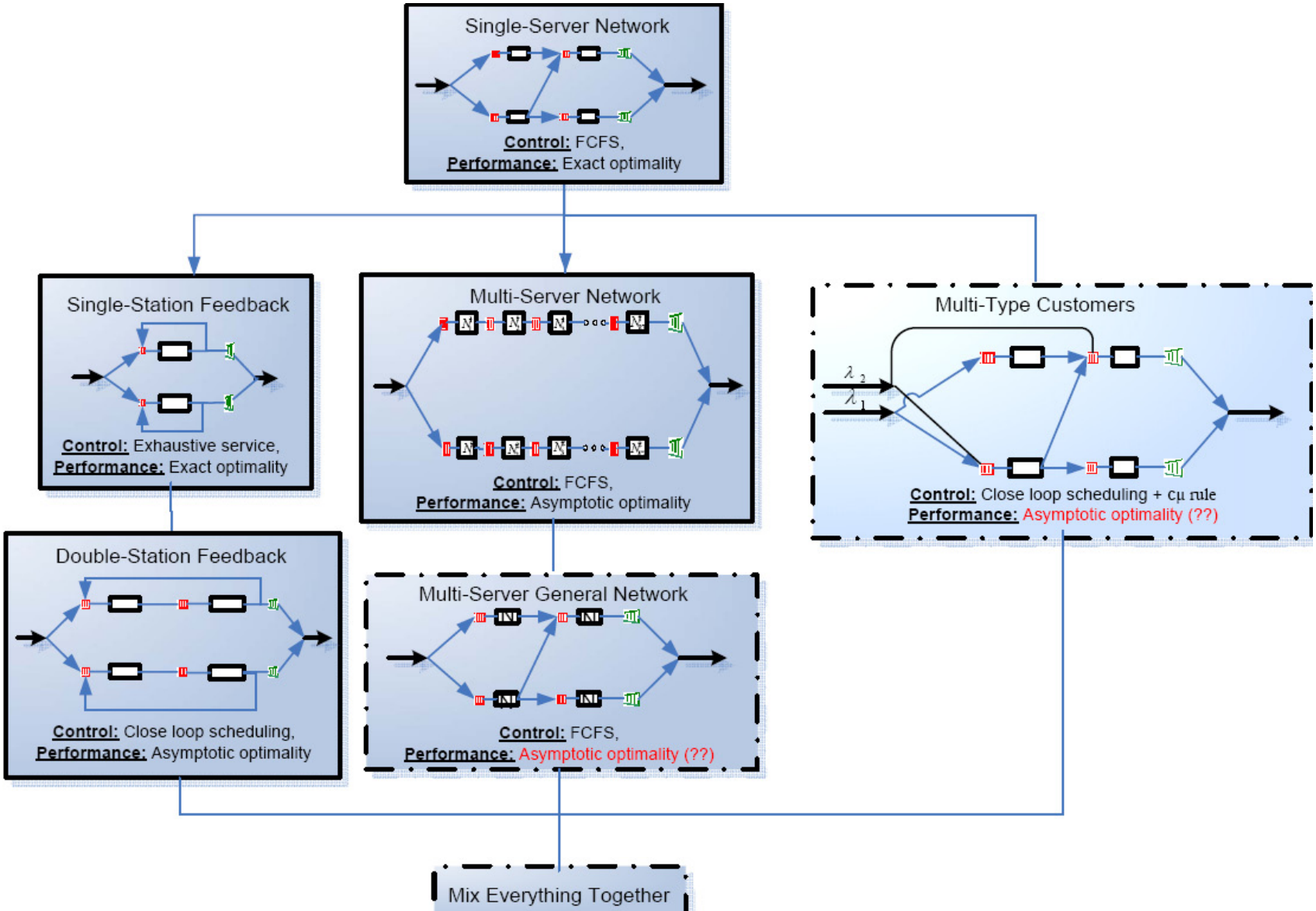
Note that the strategy of the proof is designed to be generalized to general service time distributions and a nonpreemptive discipline. However, these extensions have not been established at this time.

Recall that

- General service time distribution refers to iid service durations.
- Non-Preemptive is a policy where service to a customer can not be interrupted before it is completed.

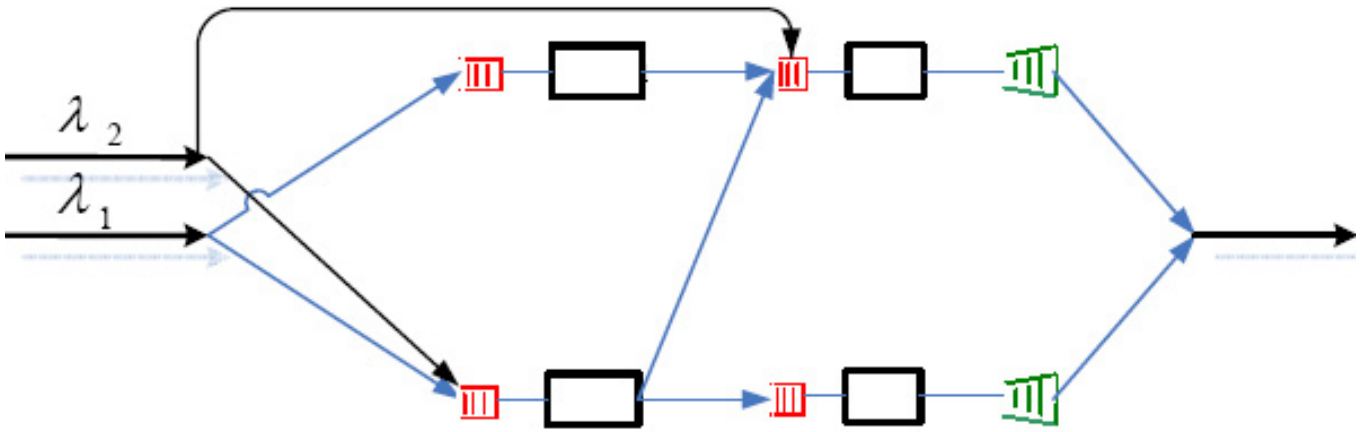
# Research Evolution

# Research Evolution



# Extension: Multi-Type Model

Consider an Heterogeneous customer population, such that different customers may have different precedence constraints, interarrival time distributions and service time distributions, e.g.




Definition: At each route, assign preemptive priority to customers whose service has completed in the other route. Define maximum pressure policy (Jim Dai 2011) within each class.

Will this policy be asymptotically optimal for such model?

# Thank You