

Stable and Utility-Maximizing Scheduling in Stochastic Processing Networks

Libin Jiang, Jean Walrand

UC Berkeley

Allerton conference, UIUC

Oct 1, 2009

Outline

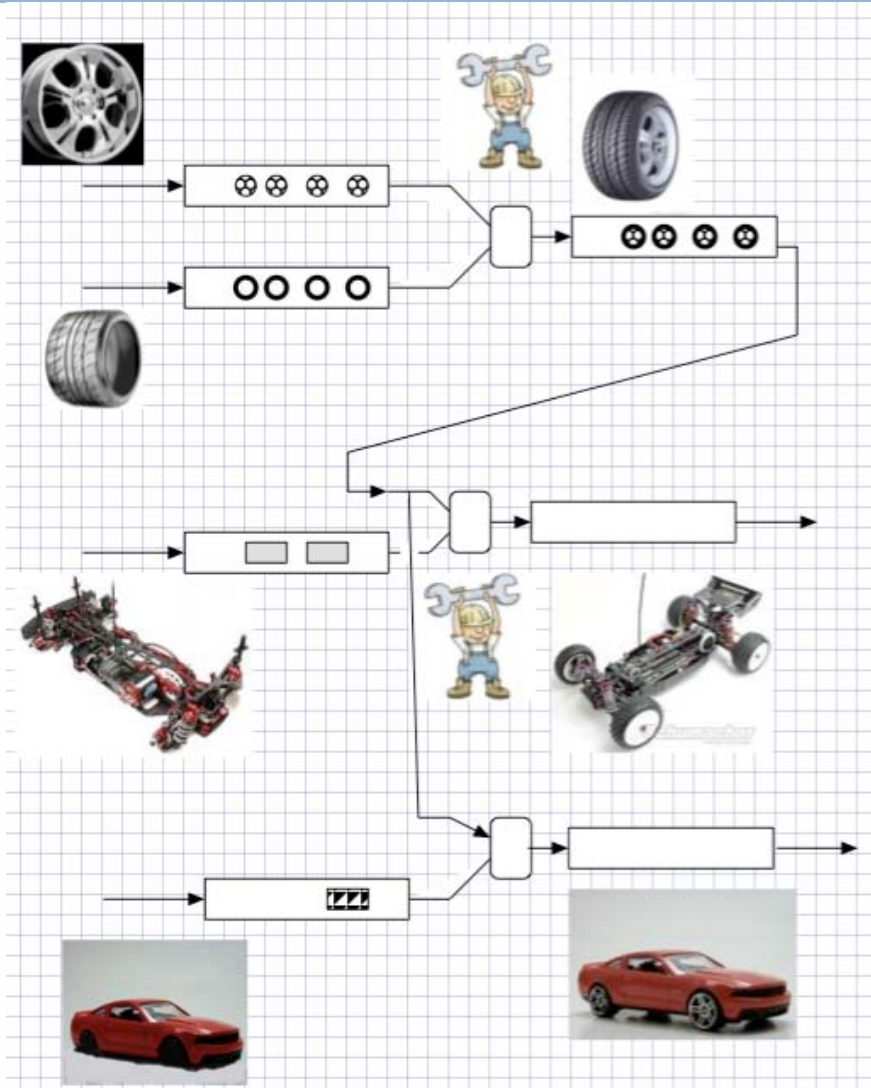
- Introduction of SPNs
- Instability of MaxWeight scheduling (possibly any work-conserving policy)
- Deficit Maximum Weight (DMW) scheduling
- DMW + congestion control

Stochastic Processing Networks (SPNs)

- Examples: Manufacturing, assembly plant, service network, hospital, ...
- General situation: **tasks (activities)** require *parts* and *resources* to produce new parts
- Tasks compete for resources: a scheduling problem

[Harrison'00], [Harrison-Williams'05], [Dai-Lin'05]

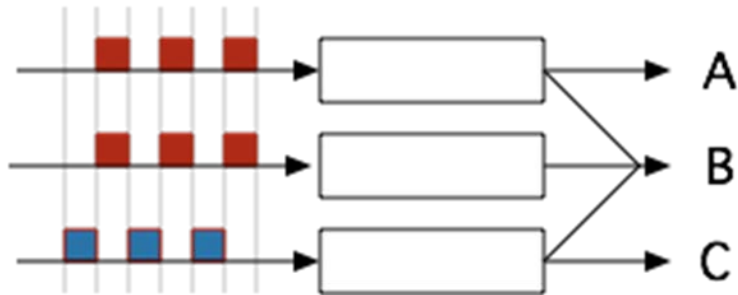
SPN: Example



Goal: Scheduling tasks and ordering parts to maximize the utility of the production

SPN: Basic Problem

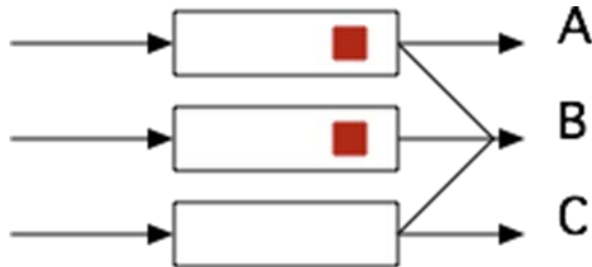
Time: 5 4 3 2 1 0



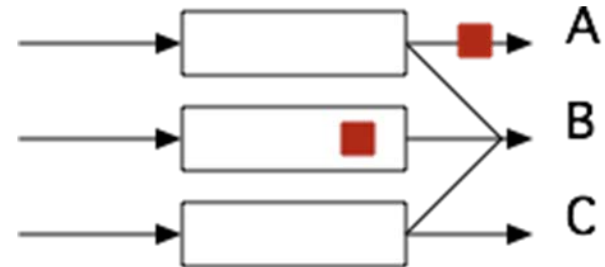
Task A requires a part from queue 1
Task B requires a part from all queues
Task C requires a part from queue 3

Use MaxWeight scheduling

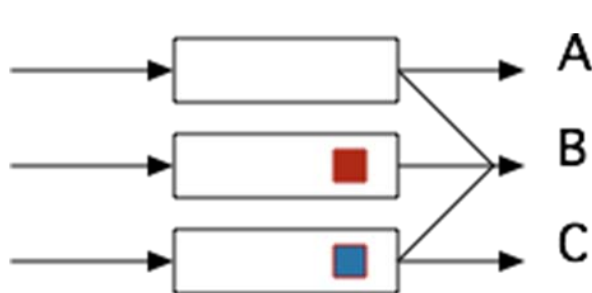
Time 0



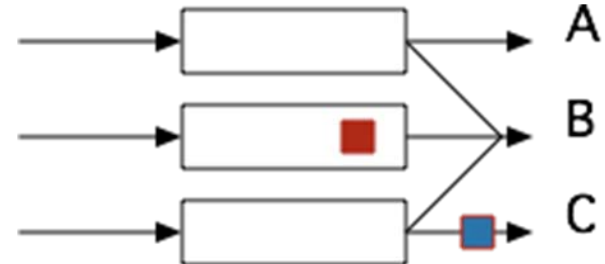
Time 1-



Time 1

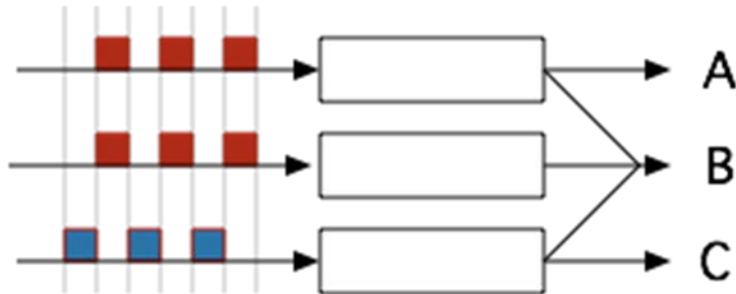


Time 2-



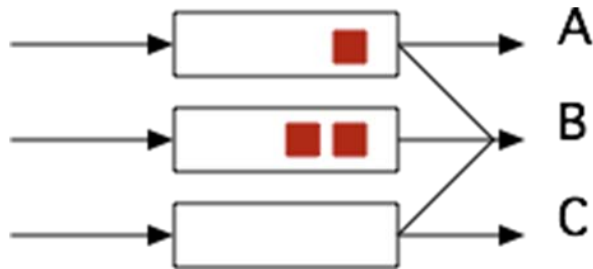
SPN: Basic Problem

Time: 5 4 3 2 1 0

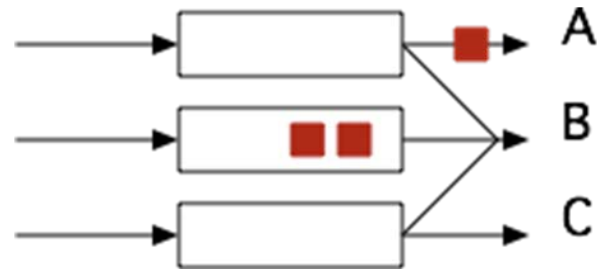


Task A requires a part from queue 1
Task B requires a part from all queues
Task C requires a part from queue 3

Time 2



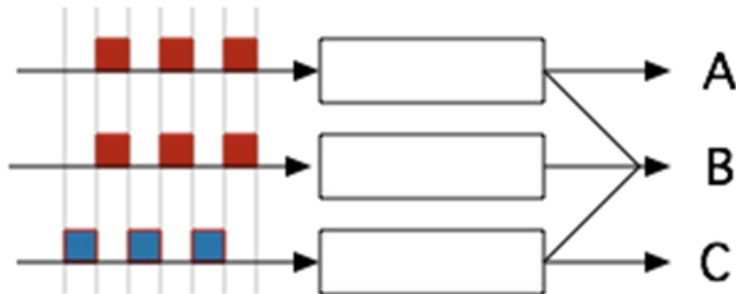
Time 3-



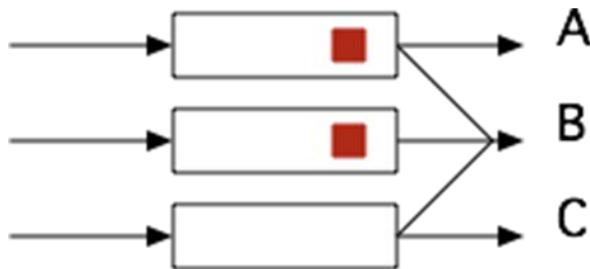
No work-conserving policy can stabilize the system!

SPN: Basic Problem

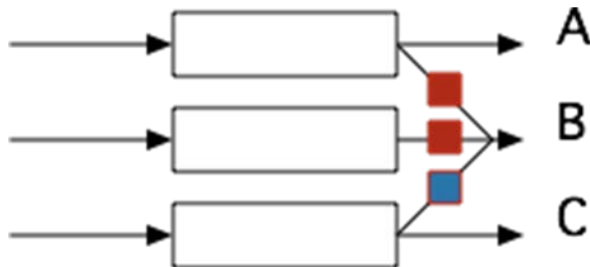
Time: 5 4 3 2 1 0



Time 0

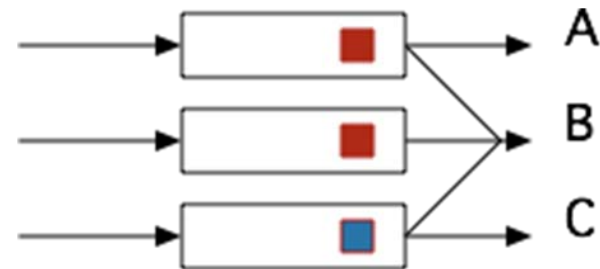


Time 2-



Task A requires a part from queue 1
Task B requires a part from all queues
Task C requires a part from queue 3

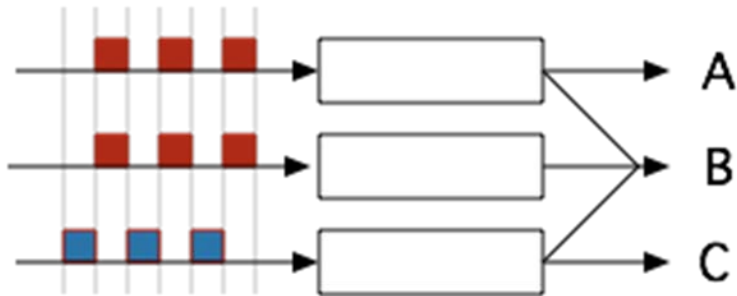
Time 1: Do not serve



Modified scheduling is stable.

DMW stabilizes the system

Time: 5 4 3 2 1 0



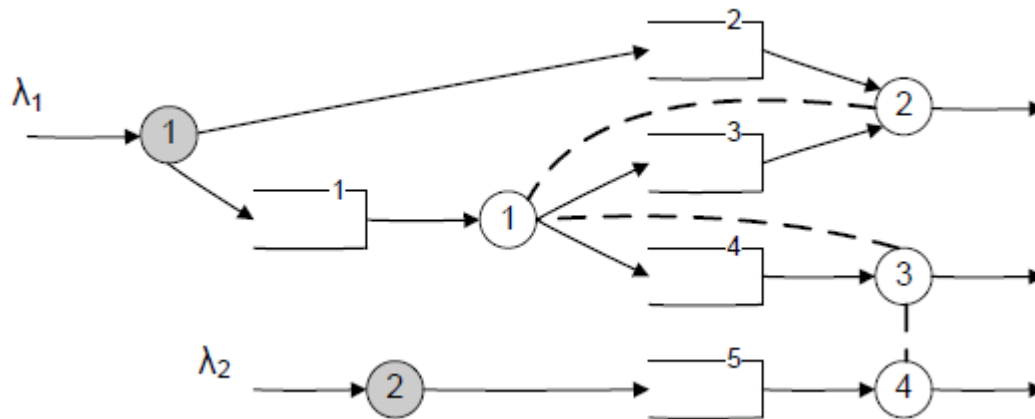
q_i = virtual backlog at queue i .

Q_i = actual backlog at queue i .

time	0	1-	1	2-	2	3-
q_1, Q_1	1, 1	0, 0	0, 0	0, 0	1, 1	0, 0
q_2, Q_2	1, 1	0, 0	0, 0	0, 0	1, 1	0, 0
q_3, Q_3	0, 0	-1, 0	0, 1	0, 1	0, 1	-1, 0
Activity	Arrival	B	Arrival	None	Arrival	B
Note:		Null activity				Actual activity

Repeats forever

Model



Shaded circle: IA

Circle: SA

Dashed line: Conflict

- M Input Activities (IA)
 - ▣ Ordering parts as input
- N Service Activities (SA)
 - ▣ Each SA consumes parts (from a set of queues), produce parts (for another set of queues) and/or products (that leave the network)
- K queues
- Each IA is the source of a “flow”, such that the inputs of a flow can be **exactly** consumed. M flows.

Model

- Time slotted
- Arrival vector $\mathbf{a}(t) \in \{0, 1\}^M$
- Service vector $\mathbf{x}(t) \in \{0, 1\}^N$
- *Scheduling problem*: With given arrival processes, choose $\mathbf{x}(t)$ to stabilize the queues
- *Scheduling and congestion control problem*: choose $\mathbf{a}(t)$ and $\mathbf{x}(t)$ to maximize utility

DMW: Deficit Maximum Weight Scheduling

- Actual queues $Q(t)$, **virtual queues $q(t)$**
 - ▣ Schedule decided by MW **w/o considering feasibility**

$$\mathbf{x}^*(t) = \arg \max_{\mathbf{x} \in \mathcal{X}} \mathbf{d}^T(t) \mathbf{x}(t)$$

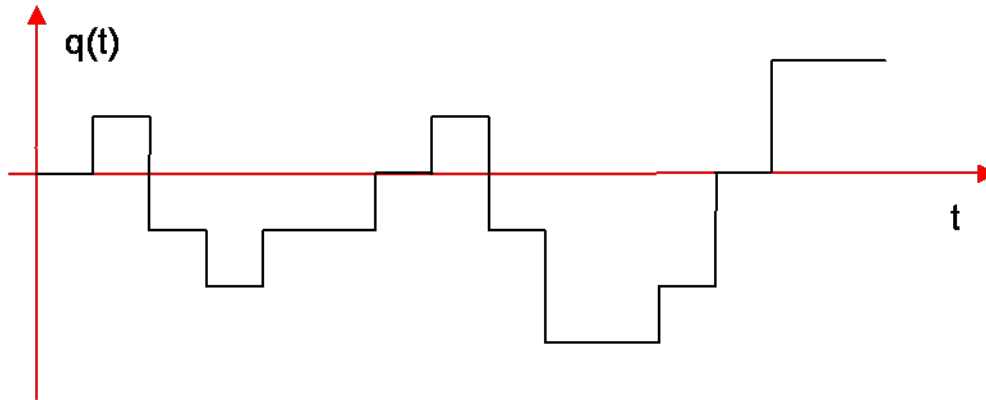
- ▣ Allow $q(t)$ to be negative
- Queue dynamics: $q(0) = Q(0) = 0$

$$q_k(t+1) = q_k(t) - \mu_{out,k}(t) + \mu_{in,k}(t), \forall k$$

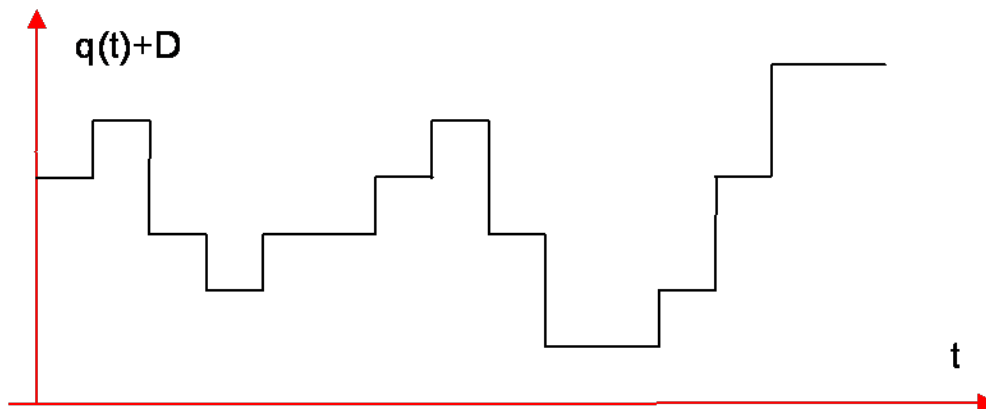
$$Q_k(t+1) = [Q_k(t) - \mu_{out,k}(t)]_+ + \mu_{in,k}(t), \forall k$$

- ▣ If Q_k “underflows”, then activate a “null SA” which produces “fictitious parts”
- “Deficit” $D_k(t) = Q_k(t) - q_k(t)$

Overcoming starvation (idea)

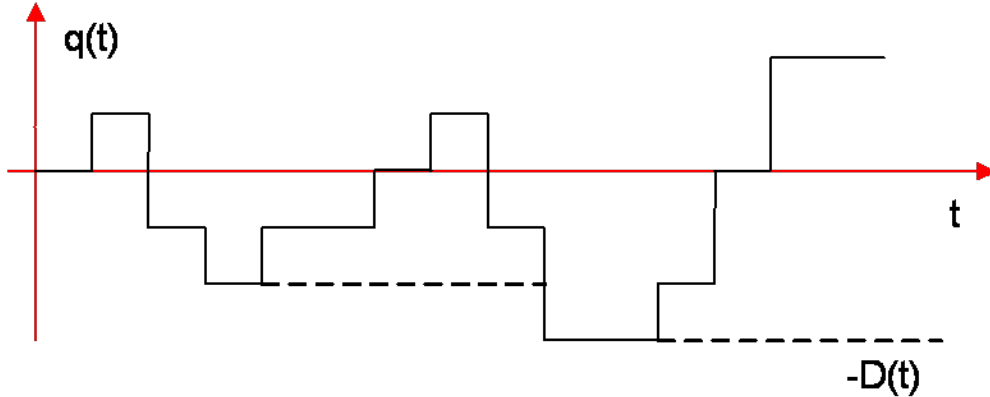


If $q(t)$ bounded, then...



If $Q(t)=q(t)+D$, then
never starved

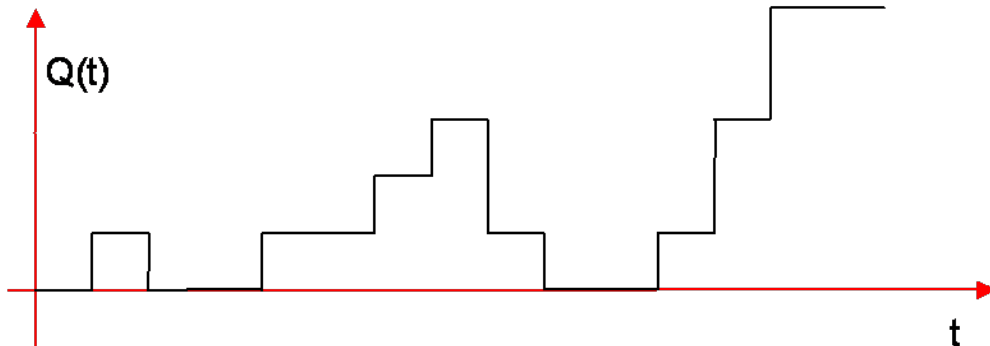
DMW finds proper deficits



For convenience...

$$q(t+1) = q(t) - \mu_{out}(t) + \mu_{in}(t)$$

$$Q(t+1) = [Q(t) - \mu_{out}(t) + \mu_{in}(t)]_+$$



$$D(t) = Q(t) - q(t) = -\min_{\tau \leq t} (q(\tau))$$

- *Prop. 1:* If $q(t)$ is bounded, then both $Q(t)$ and $D(t)$ are bounded \rightarrow Only a finite number of null SAs occur \rightarrow long-term throughput not affected.

DMW scheduling (Main properties)

- **Thm 1:** If the arrival process is *smooth enough*, then $\mathbf{q}(t)$ is uniformly bounded \rightarrow DMW is throughput-optimum.
- A mild condition: there exists $T > 0$ so that $\tilde{\mathbf{a}}_l := \sum_{\tau=l \cdot T}^{(l+1)T-1} \mathbf{a}(\tau) / T$ ($l = 0, 1, 2, \dots$) satisfy that $\tilde{\mathbf{a}}_l + \sigma \mathbf{1}$ and $\tilde{\mathbf{a}}_l - \sigma \mathbf{1}$ are feasible for some $\sigma > 0$.

- **Proof:**

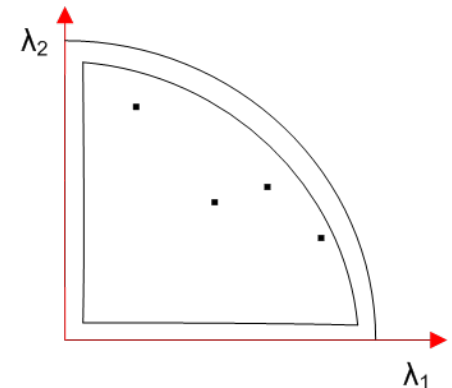
- Lyapunov function $L(\mathbf{q}(t)) := \|\mathbf{q}(t)\|^2$

- Consider $L(\cdot)$ in a larger time scale

$$L(\mathbf{q}(l \cdot T)), l = 0, 1, 2, \dots$$

- Show that $L(\mathbf{q}((l+1) \cdot T)) - L(\mathbf{q}(l \cdot T)) \leq -\delta \|\mathbf{q}(l \cdot T)\| + c$

- $\mathbf{q}(t)$ uniformly bounded



Analysis differs from MWS
 (i) $\mathbf{q}(t)$ can be negative
 (ii) SAs are multi-input multi-output

DMW scheduling (Main properties)

- More random arrivals

$$E[a_m(t)] = \lambda_m, \forall m$$

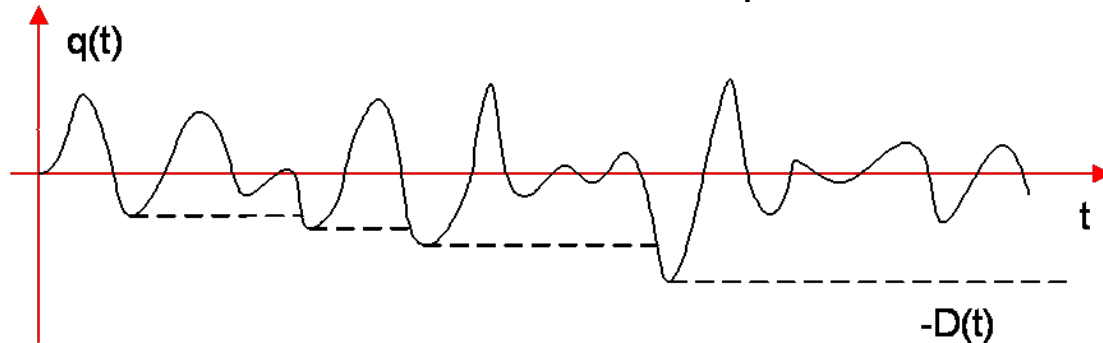
- $L(\mathbf{q}(t))$ has negative drift when it's large enough

$$E[L(\mathbf{q}(t+1)) | \mathbf{q}(t)] - L(\mathbf{q}(t)) \leq -\delta \|\mathbf{q}(t)\| + c$$

- Foster-Lyapunov criterion $\rightarrow \mathbf{q}(t)$ is stable

- **Thm 2: system is “rate-stable”**

- Null SAs become less and less frequent

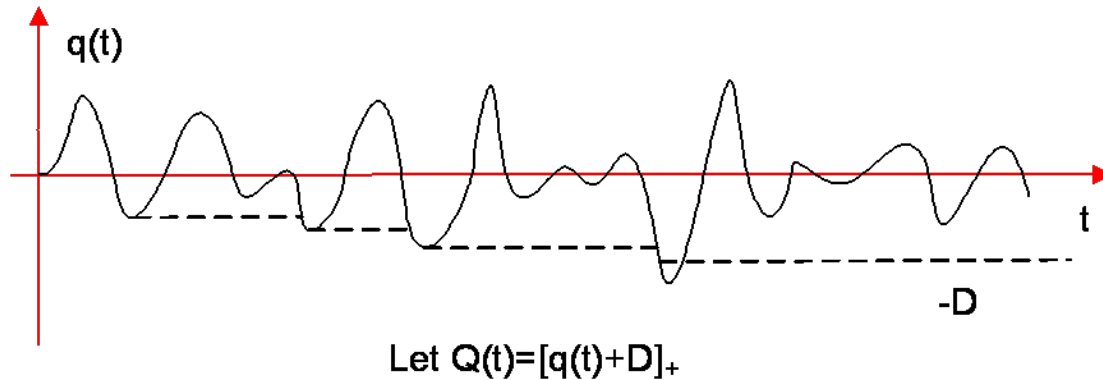


- But $Q(t)$ may slowly drift to infinity

$$Q_k(t) - q_k(t) = D_k(t) \approx -\min_{\tau \leq t} (q_k(\tau)) \rightarrow +\infty$$

DMW scheduling (Main properties)

- Can force $Q(t)$ to be stable, at a cost of the throughput



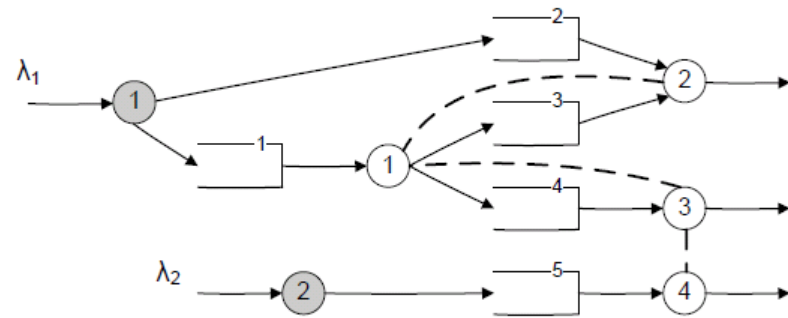
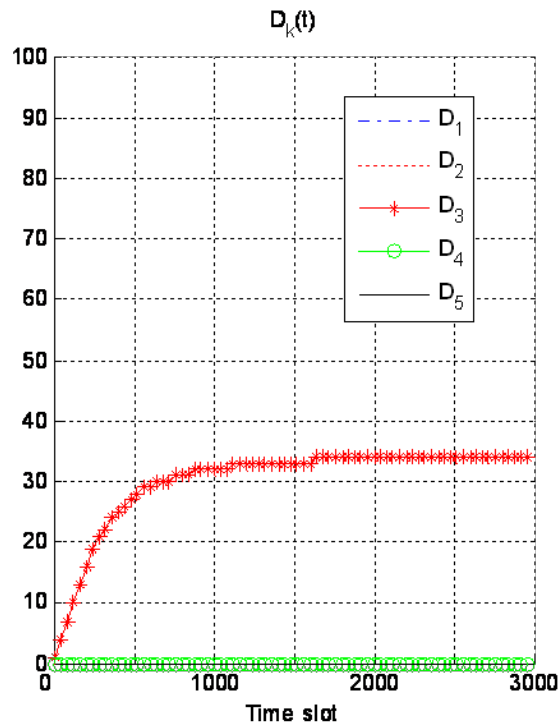
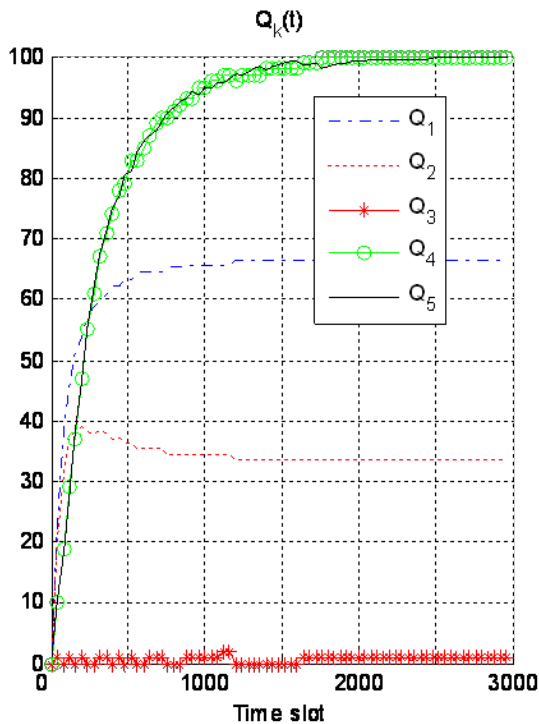
- ▣ Tradeoff between queue lengths and throughput

DMW scheduling + congestion control

□ Congestion control

$$f_m(\mathbf{q}(t)) := \arg \max_{0 \leq f \leq 1} \{V \cdot u_m(f) + \mathbf{q}(t)^T A_m f\}$$

□ **Thm 3:** $q(t)$, $Q(t)$ always uniformly bounded. Approaching maximum utility when $V \rightarrow \infty$



$$u_m(\cdot) = \log(\cdot), V = 50$$

Throughput:

Flow 1: 0.4998

Flow 2: 0.4998

Theoretical optimum: 0.5, 0.5

Discussion

- What if different SAs have different durations?
 - ▣ Slotted system with preemption (or suspension)
 - ▣ If suspension is not allowed
 - Can use long time slots
 - Perform each scheduled SA many times in a long slot. Wasted time negligible.
 - Approximate the maximum throughput (with a cost of delay)
 - ▣ [Dai-Lin'05]: a sufficient condition under which non-preemptive MaxWeight scheduling is throughput-optimum.

Conclusion & Future work

- Starvation (or, queue underflow) causes instability in SPNs
- DMW scheduling for SPNs
 - ▣ Augment the “state” with virtual queues
 - ▣ Throughput optimality
- DMW scheduling with congestion control
 - ▣ Maximal utility
- Future work
 - ▣ Performance improvement
 - ▣ Positive recurrence with random arrivals



Thank you!

Questions and comments?