

Approaching Throughput-Optimality in Distributed CSMA: Collisions and Stability

Libin Jiang, Jean Walrand

UC Berkeley

Mobihoc S³ Workshop'09

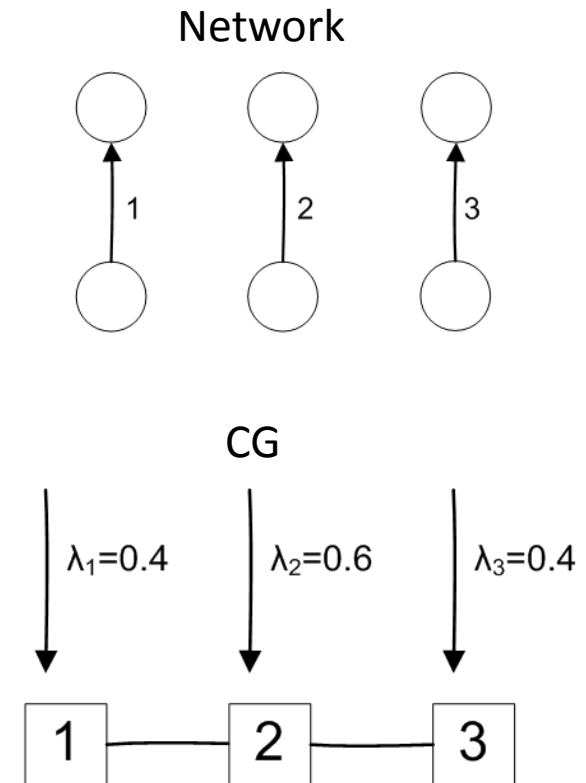
May 18, 2009

Outline

- Throughput-Optimality
- Traditional Max-Weight Scheduling (MWS)
- Distributed CSMA scheduling
 - ▣ Idealized CSMA
 - ▣ CSMA with collisions
 - ▣ Simulations

Throughput-Optimality Objective

- K interfering queues (or links)
 - Conflict graph (CG)
 - Independent set (IS) $v^i, i=1,2,\dots,N$
- Arrival rates vector λ is **feasible** if
$$\lambda \in \bar{\mathcal{C}} := \{\lambda' \mid \lambda' = \sum_{i=1}^N (\bar{p}_i \cdot v^i)\}$$
where $\bar{\mathbf{p}}$ is a probability distribution



IS: (1 0 1),
(0 1 0),
(1 0 0)

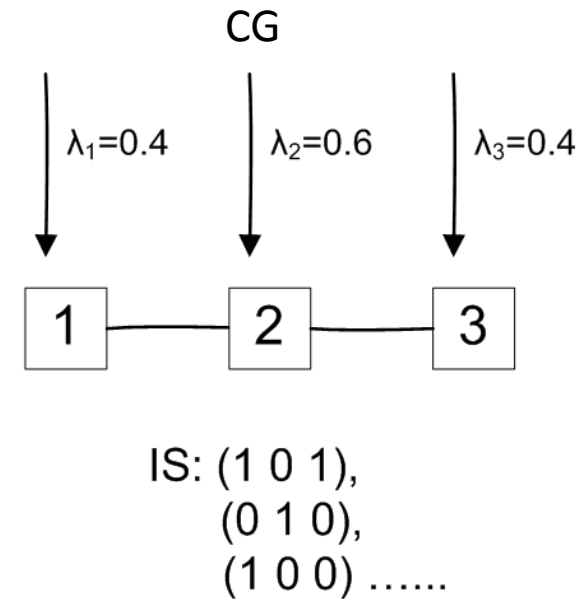
Throughput-Optimality Objective

- K interfering queues (or links)
 - Conflict graph (CG)
 - Independent set (IS) $v^i, i=1,2,\dots,N$
- Arrival rates vector λ is **feasible** if

$$\lambda \in \bar{\mathcal{C}} := \left\{ \lambda' \mid \lambda' = \sum_{i=1}^N (\bar{p}_i \cdot v^i) \right\}$$

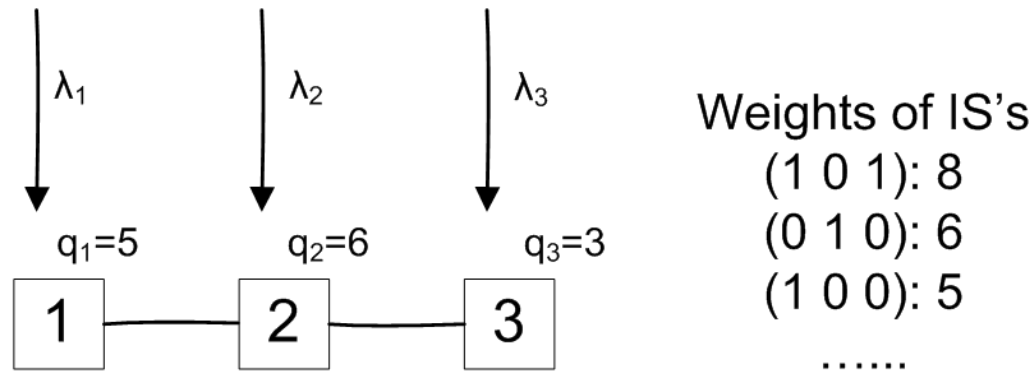
where $\bar{\mathbf{p}}$ is a probability distribution

- Scheduling: which queues to serve (at each time instance)
- A scheduling algorithm is **throughput-optimal** if it supports any **strictly feasible** $\lambda \in \mathcal{C}$



Traditional Max-Weight Scheduling

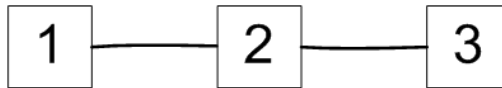
- Time slotted (sync tx)
- In each slot, schedule the IS with the maximum **weight** (sum of queue lengths)



- Stabilize all queues [Tassiulas, Ephremides,'92]
- Centralized, hard to implement in wireless network. High complexity.

Distributed CSMA

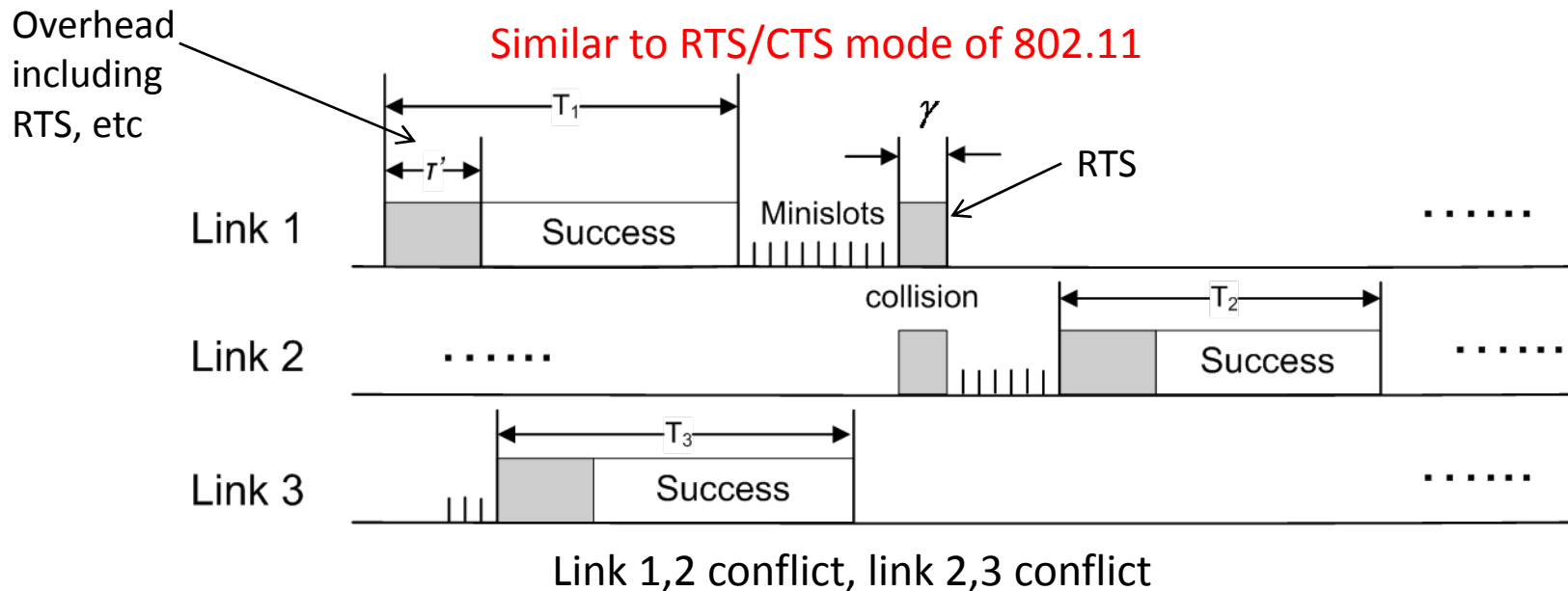
- Surprisingly, adaptive CSMA/CA can achieve throughput-optimality **distributively**



- The case of Idealized CSMA [Jiang, Walrand'08]*
 - ▣ w/o collisions (no sensing delay), w/o hidden nodes (HN)
 - ▣ A link reduces its **mean backoff time** when queue length increases, and vice versa
 - ▣ Convergence / stability results (more recently)
- However, collisions not considered

*Related work: [Rajagopalan, Shah'08], [Marbach et al. '07], [Proutiere et al. '08], [Liu et al.'09]

A model of CSMA with collisions



- In a minislot, link k starts a transmission w.p. p_k (fixed) if the medium is idle
- Markov chain

[Ni, Srikant'09] proposed a different algorithm to resolve collision

A model of CSMA with collisions

- State $x \in \{0, 1\}^K$: which links are transmitting in a given minislot
 - ▣ Stationary distribution (**product-form** due to quasi-reversibility)

$$p(x) = C(x) \prod_{k \in S(x)} T_k$$

The set of links that transmit successfully in x

- Re-parametrize T_k by r_k

$$T_k := \tau' + T_0 \cdot \exp(r_k)$$

- $s_k(\mathbf{r})$: service rate of link k

Distributed Tx-time control Algorithm

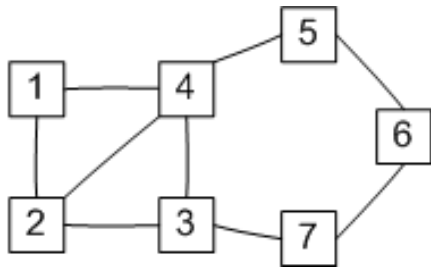
- *Theorem:* for any $\lambda \in \mathcal{C}$, there exists a unique \mathbf{r}^* such that $s_k(\mathbf{r}^*) = \lambda_k, \forall k$. (Properly choosing Tx-times can support λ .)
- Proof: Convex function $L(\mathbf{r})$:
$$\partial L(\mathbf{r}) / \partial r_k = s_k(\mathbf{r}) - \lambda_k . \mathbf{r}^* \text{ solves } \min_{\mathbf{r}} L(\mathbf{r}).$$
- Intuition: Fixed p_k and fixed collision length limit the impact of collisions.

Distributed Tx-time control Algorithm

- *Theorem*: for any $\lambda \in \mathcal{C}$, there exists a unique \mathbf{r}^* such that $s_k(\mathbf{r}^*) = \lambda_k, \forall k$. (Properly choosing Tx-times can support λ .)
- Proof: Convex function $L(\mathbf{r})$:
$$\partial L(\mathbf{r}) / \partial r_k = s_k(\mathbf{r}) - \lambda_k. \mathbf{r}^* \text{ solves } \min_{\mathbf{r}} L(\mathbf{r}).$$
- Distributed algorithm (*stochastic gradient algorithm*)
$$r_k(j+1) = \{r_k(j) + \alpha(j)[\lambda'_k(j) - s'_k(j)]\}_{[r_{min}, r_{max}]}, \forall k$$
- Convergence
 - ▣ If $\mathbf{r}^* \in (r_{min}, r_{max})^K$, then with $\alpha(j) = 1/j$,
$$r_k(j) \rightarrow r_k^*, \forall k$$
- As $r_{min} \rightarrow -\infty, r_{max} \rightarrow \infty$, the “capacity region” of the algorithm $\rightarrow \mathcal{C}$

Simulations

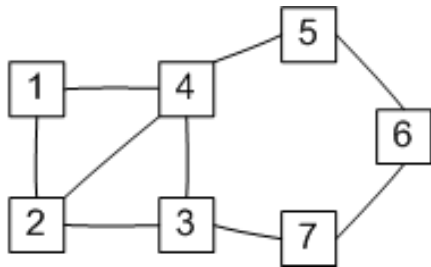
Conflict
graph:



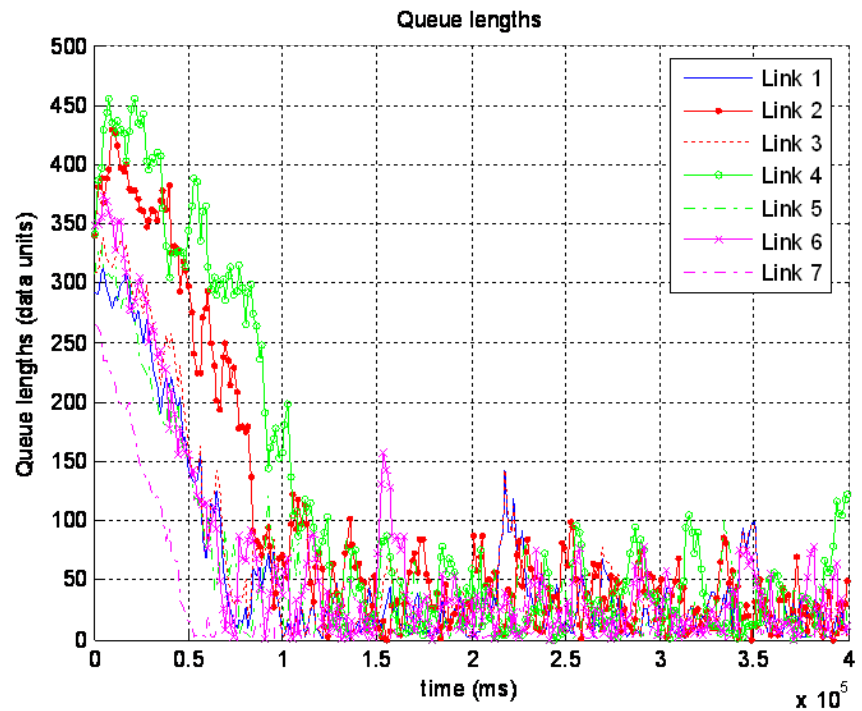
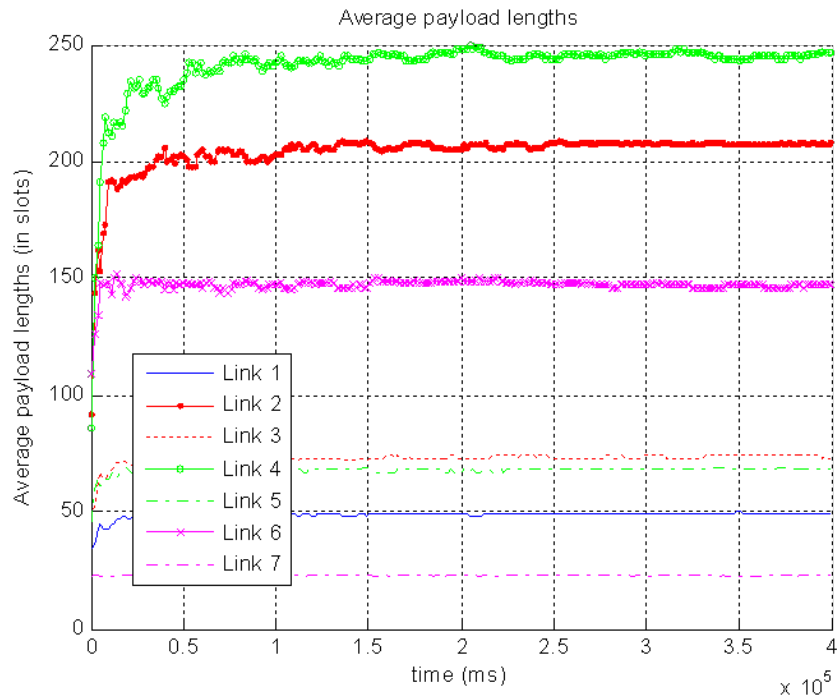
$$\lambda = 0.8 * [0.4, 0.4, 0.4, 0.2, \\ 0.4, 0.6, 0.2]$$

Simulations

Conflict graph:



$$\lambda = 0.8 * [0.4, 0.4, 0.4, 0.2, 0.4, 0.6, 0.2]$$





Thank you!