Throughput of Random Access without Message Passing

Alexandre Proutiere
Microsoft Research

Joint work with **Yung Yi** and **Mung Chiang**
CISS 2008, March 2008

Outline

1. Throughput optimality via distributed scheduling in wireless networks - Is there a problem there?

Outline

1. Throughput optimality via distributed scheduling in wireless networks - Is there a problem there?

May be ...

Outline

1. Throughput optimality via distributed scheduling in wireless networks - Is there a problem there?

May be ...

2. What can we achieve using slight modifications of today's random access algorithms?

Outline

1. Throughput optimality via distributed scheduling in wireless networks - Is there a problem there?

May be ...

2. What can we achieve using slight modifications of today's random access algorithms?

A lot!*

Without message passing
Without accounting for queue lengths

---

*Il va falloir ecouter mon talk jusqu'a la fin pour le savoir...

Outline

1. Throughput optimality via distributed scheduling in wireless networks - Is there a problem there?

Not really.

2. What can we achieve using slight modifications of today's random access algorithms?

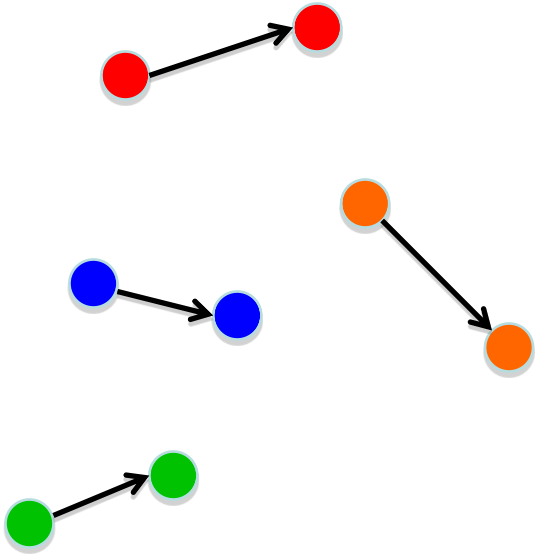
A lot!*

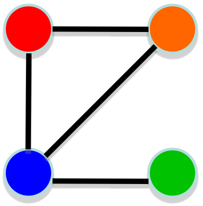
Without message passing
Without accounting for queue lengths

Why this question?
a. Message passing is not always easy (hidden terminals - security)
b. We should not work on improving efficient systems

*Il va falloir ecouter mon talk jusqu'a la fin pour le savoir...

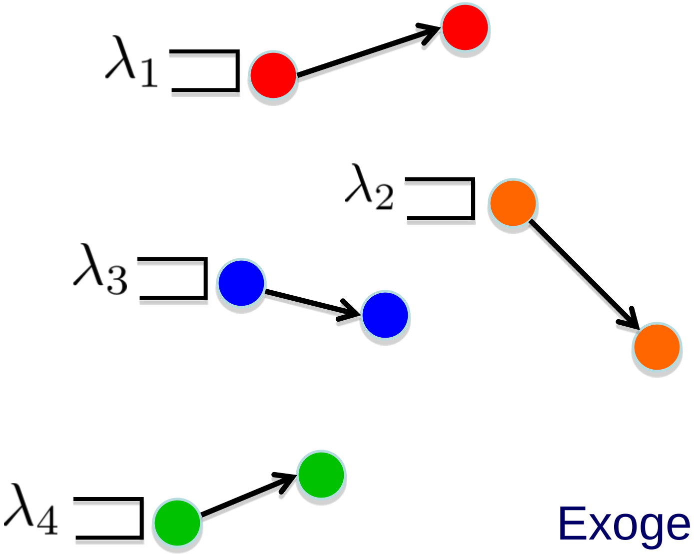
Scheduling in wireless networks

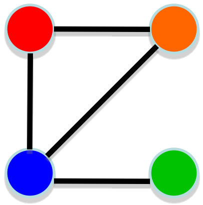




Non-oriented interference graph

Scheduling in wireless networks

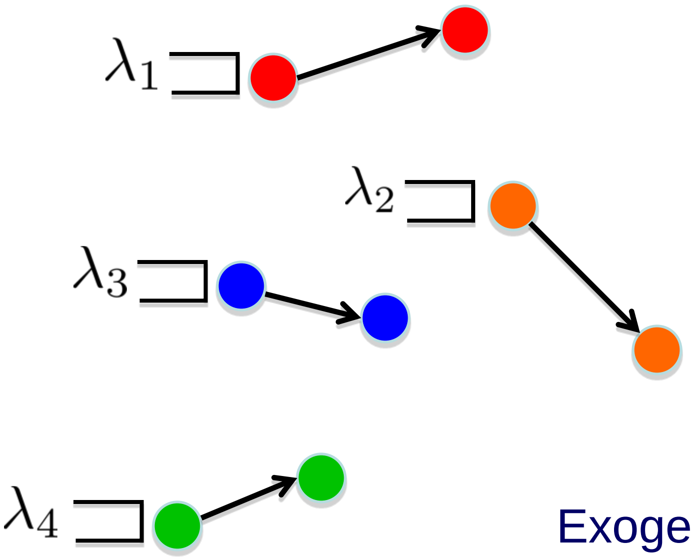


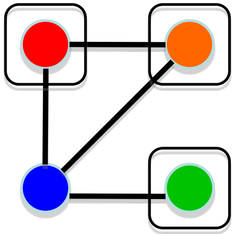


Non-oriented interference graph

Exogeneous Markovian packet arrivals at various links
Arrival intensity at link l: λ_l

Scheduling in wireless networks



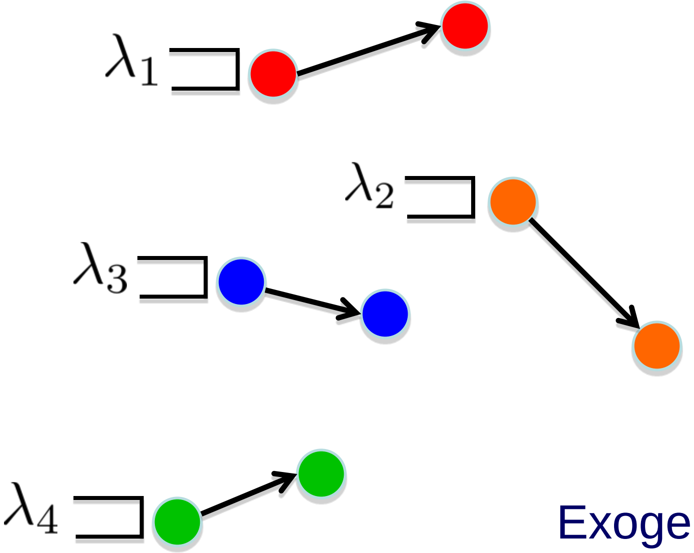


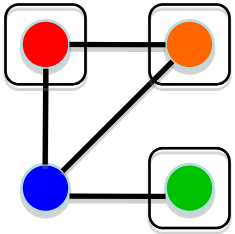
Non-oriented interference graph

Exogeneous Markovian packet arrivals at various links
Arrival intensity at link l: λ_l

Schedules: sets of active links, not necessarily non-interfering

Scheduling in wireless networks





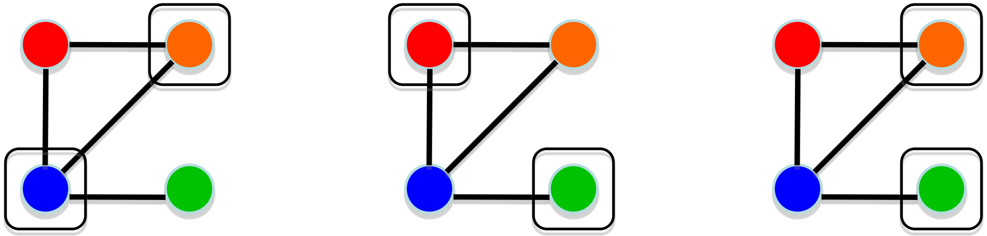
Non-oriented interference graph

Exogeneous Markovian packet arrivals at various links
Arrival intensity at link l: λ_l

Schedules: sets of active links, not necessarily non-interfering

How to select the right schedule at each time in a distributed manner so as to maximize the stability region of the queues?

Maximum Throughput region



Set of maximal schedules $\mathcal{M}$

Pareto-boundary of the maximum stability region

$$\partial\Gamma = \left\{ \gamma : \exists \tau \in [0,1]^M, \sum_{m \in \mathcal{M}} \tau_m = 1, \forall l, \gamma_l = \sum_{m \in \mathcal{M} : l \in m} \tau_m \right\}$$

Max-Weight scheduling scheme – Tassiulas/Ephremides

$$m(t) = \arg\max_{m \in \mathcal{M}} \sum_{l \in m} Q_l(t)$$

Distributed solutions?

1. Distributed implementations of the MW scheduling scheme
    - Modiano-Shah-Zussman: Gossip algorithms (ACM Sigmetrics’06)
    - Sanghavi-Bui-Srikant: Low overhead algos (ACM Sigmetrics’07)
    - Yi-Chiang: Low overhead, k-hop interference (ICC’08)

Require message passing, and other *complex* procedures

Distributed solutions?

1. Distributed implementations of the MW scheduling scheme
    - Modiano-Shah-Zussman: Gossip algorithms (ACM Sigmetrics'06)
    - Sanghavi-Bui-Srikant: Low overhead algos (ACM Sigmetrics'07)
    - Yi-Chiang: Low overhead, k-hop interference (ICC'08)

    Require message passing, and other *complex* procedures

2. Maximal scheduling
    - Chaporkar-Kar-Sarkar'05, Wu-Srikant-Perkins'07: 1/Kmax throughput optimality

Distributed solutions?

1. Distributed implementations of the MW scheduling scheme
    - Modiano-Shah-Zussman: Gossip algorithms (ACM Sigmetrics’06)
    - Sanghavi-Bui-Srikant: Low overhead algos (ACM Sigmetrics’07)
    - Yi-Chiang: Low overhead, k-hop interference (ICC’08)

   Require message passing, and other *complex* procedures

2. Maximal scheduling
    - Chaporkar-Kar-Sarkar’05, Wu-Srikant-Perkins’07: 1/Kmax throughput optimality

3. Queue-based random access
    - Lin-Rasool’06, Joo-Shrof’07: local message passing and ½-throughput optimality

Distributed solutions?

1. Distributed implementations of the MW scheduling scheme
    - Modiano-Shah-Zussman: Gossip algorithms (ACM Sigmetrics’06)
    - Sanghavi-Bui-Srikant: Low overhead algos (ACM Sigmetrics’07)
    - Yi-Chiang: Low overhead, k-hop interference (ICC’08)

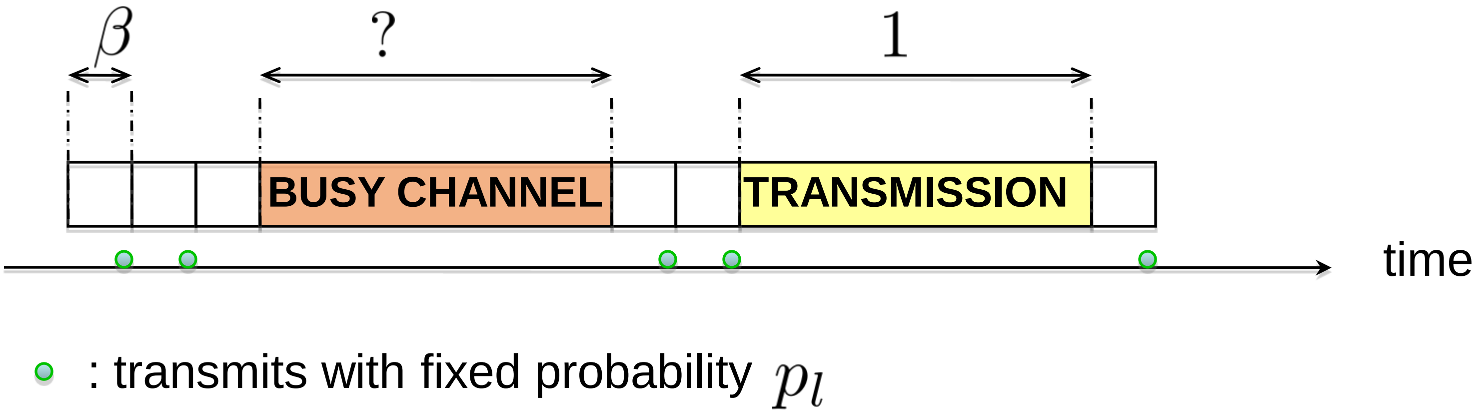
   Require message passing, and other *complex* procedures

2. Maximal scheduling
    - Chaporkar-Kar-Sarkar’05, Wu-Srikant-Perkins’07: 1/Kmax throughput optimality

3. Queue-based random access
    - Lin-Rasool’06, Joo-Shrof’07: local message passing and ½-throughput optimality

4. Do (almost) nothing! Non-adaptive CSMA is enough to guarantee (1-x)-throughput optimality … x?

Throughput of non-adaptive CSMA algos

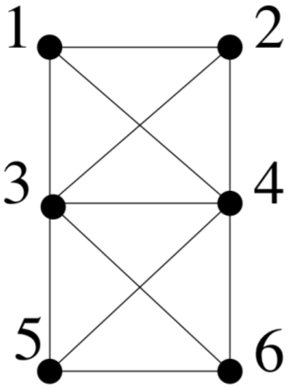


- Identifying the stability region of the above algorithms is as difficult as characterizing the stability region of ALOHA systems: an open problem since the 70's

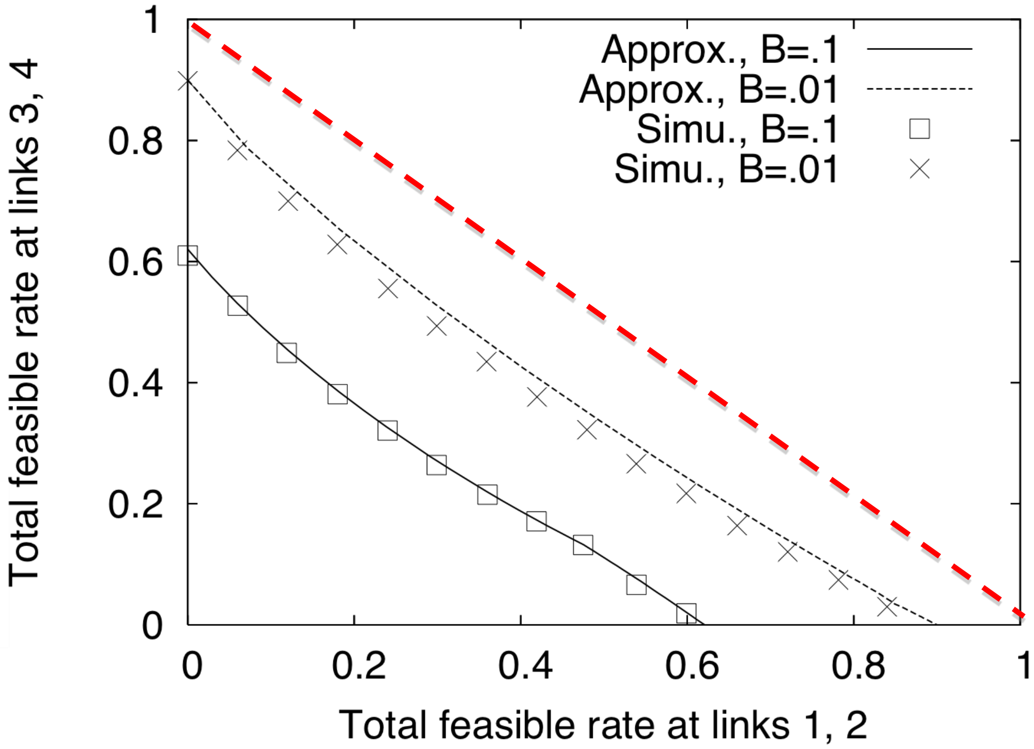
- An asymptotically exact approximate stability region has been proposed in:

Performance of Random Medium Access Control
An asymptotic approach
C. Bordenave, D. McDonald, A. Proutiere
ACM Sigmetrics 2008

Throughput of non-adaptive CSMA algos



$$\lambda_1 = \lambda_2 = \lambda_5 = \lambda_6$$
$$\lambda_3 = \lambda_4$$

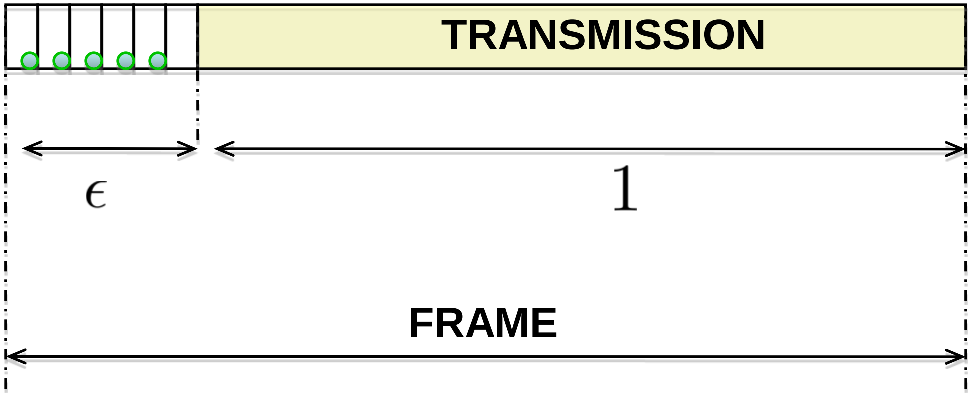


When the packet size becomes large, non-adative CSMA algorithms seem to achieve high throughput

Can we formalize this observation? Or can we modify the algorithms so as to get explicit throughput guarantees?

Weighted-Fair Maximal scheduling

- By letting the packet size grows large we can implement maximal scheduling
- Synchronous systems: users share the notion of frames



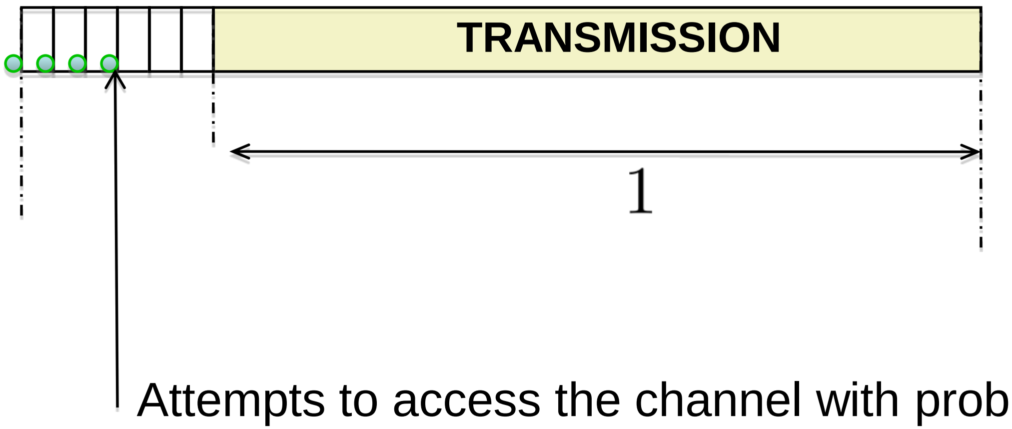
$$\beta \times M = \epsilon$$

● : transmits with fixed probability p_l

Weighted-Fair Maximal scheduling

- By letting the packet size grows large we can implement maximal scheduling

- Synchronous systems: users share the notion of frames

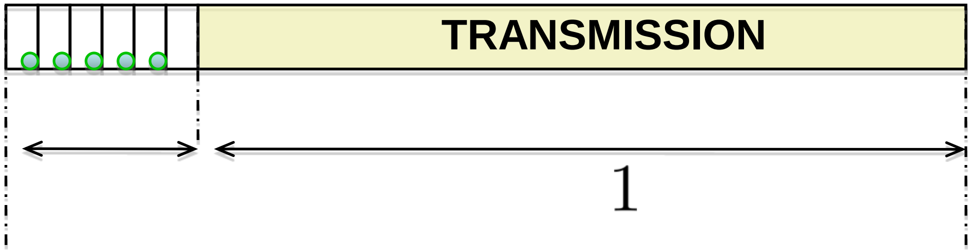


$$\beta \times M = \epsilon$$

Attempts to access the channel with probability p_l
Until it wins the contention or one of the neighbor does

Weighted-Fair Maximal scheduling

- By letting the packet size grows large we can implement maximal scheduling
- Asynchronous systems: each user has its own slot process



● : transmits with fixed probability p_l

Synchronous systems

Lemma: Let $\mathcal{A}$ the set of active links, and m_p the schedule used in a frame. Then, when $\beta \to 0$ with $\beta \times M = \epsilon$,

$$\forall m \subset \mathcal{A}, \quad P_\beta[m_p = m] \to \tau_p(m) 1_{\{m \in \mathcal{M}_\mathcal{A}\}},$$

where $\mathcal{M}_\mathcal{A}$ is the set of maximal schedule in $\mathcal{A}$.

- We realize a maximal scheduling algorithm with random selection of schedules
- The distribution of schedules determined by a random packing process
- How to tune the schedule distribution?

Minimum-degree greedy algorithm

- A sequence of systems such that:

$$\lim_{\beta \to 0} p_l = 1, \quad \lim_{\beta \to 0} \frac{\beta}{\min_l (1 - p_l)} = 0$$

$$\lim_{\beta \to 0} \frac{\prod_{j \neq k : A_{jk}=1} (1 - p_j)}{\prod_{j \neq l : A_{jl}=1} (1 - p_j)} = b_{kl}$$

- At the limit, realizes the following random packing algorithm:

```
Step 1   $\mathcal{S} = \emptyset, \mathcal{R} = \mathcal{A}$

Step 2.  With probability $(\sum_{k \in \mathcal{R}} b_{kl})^{-1}$ do:
         $\mathcal{S} = \mathcal{S} \cup \{l\}$ and $\mathcal{R} = \mathcal{R} \setminus \{k : A_{kl} = 1\}$

Step 3.  Apply Step 2 until $\mathcal{P} = \emptyset$
```

Minimum-degree greedy algorithm

- A sequence of systems such that:

$$\lim_{\beta\to 0} p_l = 1, \quad \boxed{\lim_{\beta\to 0} \frac{\beta}{\min_l(1-p_l)} = 0}$$

$$\lim_{\beta\to 0} \frac{\prod_{j\neq k:A_{jk}=1}(1-p_j)}{\prod_{j\neq l:A_{jl}=1}(1-p_j)} = b_{kl}$$

Ensures that the schedule is built before the end of the contention period with high proba

Minimum-degree greedy algorithm

- A sequence of systems such that:

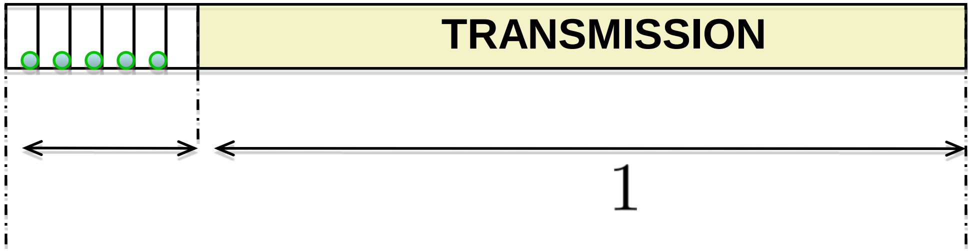
$$\lim_{\beta\to 0} p_l = 1, \quad \lim_{\beta\to 0} \frac{\beta}{\min_l (1-p_l)} = 0$$

$$\lim_{\beta\to 0} \frac{\prod_{j\neq k: A_{jk}=1}(1-p_j)}{\prod_{j\neq l: A_{jl}=1}(1-p_j)} = b_{kl}$$

- Why minimum degree algorithm? [Sequentially build the schedule randomly selecting the link with minimum number of neighbors]

Example: $p_l = 1 - c_\beta, \quad c_\beta \to 0$

Asynchronous systems



● : transmits with fixed probability p_l

- For fixed packet size and transmission probabilities, system modelled as a *non-reversible* loss network

  (Bordenave-McDonald-Proutiere, ACM Sigmetrics'08)

- Reversibility recovered when almost no collisions:

$$\forall l, \lim_{\beta \to 0} p_l = 0, \quad \lim_{\beta \to 0} \frac{\beta}{\min_{l \in \mathcal{L}} p_l} = 0$$

$$\forall l, k \in \mathcal{L}, \quad \lim_{\beta \to 0} \frac{p_k}{p_l} = \frac{\nu_k}{\nu_l} \in [0, +\infty]$$

Asynchronous systems

Lemma: Fix the set of active links $\mathcal{A}$, and denote by $m_p(t, \mathcal{A})$ the schedule used at time t. The process $(m_p(t, \mathcal{A}), t \geq 0)$ converges in law when $\beta \to 0$ to a continuous-time reversible process with stationary distribution $\tau_\nu(m, \mathcal{A})$ such that:

$$\tau_\nu(m, \mathcal{A}) \sim 1_{\{m \in \mathcal{M}_\mathcal{A}\}} \prod_{l \in m} \nu_l.$$

- Example: with identical transmission probabilities, the schedule stationary distribution is uniform on the maximum size schedules

Asynchronous systems:throughput

- When the set of active links is fixed, convergence to the right schedule distribution
- Does it imply that the system has the same throughput as that of Maximum Size scheduling? … not sure, but …

Asynchronous systems:throughput

- When the set of active links is fixed, convergence to the right schedule distribution
- Does it imply that the system has the same throughput as that of Maximum Size scheduling? … not sure, but …
- Consider the following modifications:

Rule 1. a link stops transmitting when its queue is less than K.T, restart transmitting when its queue is greater than 2K.T
[Ensures that the set of active links remain identical for a long time]

Asynchronous systems:throughput

- When the set of active links is fixed, convergence to the right schedule distribution
- Does it imply that the system has the same throughput as that of Maximum Size scheduling? … not sure, but …
- Consider the following modifications:

Rule 1. a link stops transmitting when its queue is less than K.T, restart transmitting when its queue is greater than 2K.T
[Ensures that the set of active links remain identical for a long time]

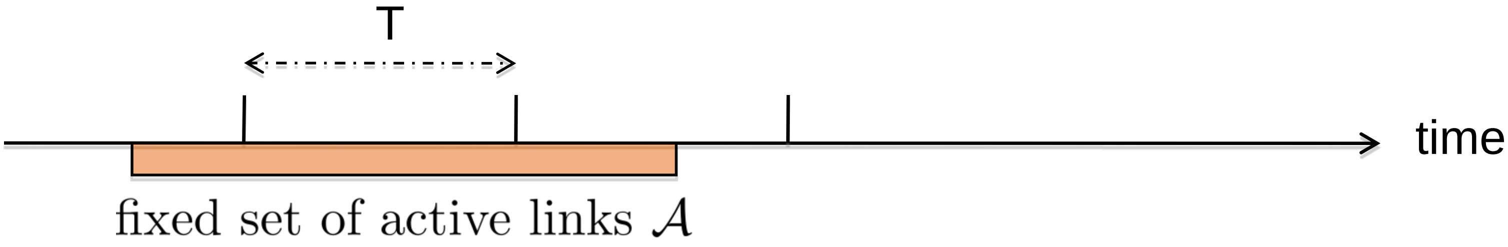
Rule 2. Periodic reset every T’ [To use convergence on compacts]

Asynchronous systems:throughput

- When the set of active links is fixed, convergence to the right schedule distribution
- Does it imply that the system has the same throughput as that of Maximum Size scheduling? … not sure, but …
- Consider the following modifications:

Rule 1. a link stops transmitting when its queue is less than K.T, restart transmitting when its queue is greater than 2K.T
[Ensures that the set of active links remain identical for a long time]

Rule 2. Periodic reset every T' [To use convergence on compacts]



$$E[r_l(\mathcal{A})] \geq E[r_l(\mathcal{A}, \mathrm{MaxSize})] - \epsilon$$

Asynchronous systems:throughput

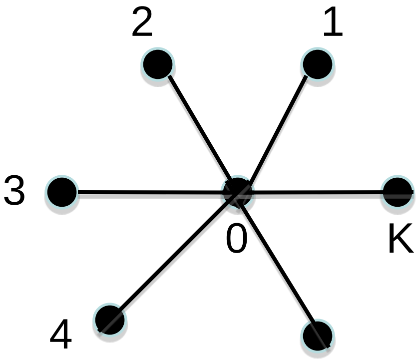
Proposition. In asynchronous systems, for any $\epsilon > 0$, there exists a scheme based on non-adaptive random access algorithm with throughput $(1-\epsilon) \times \Gamma_{MS}$.

Throughput of MaxSize scheduling

- MaxSize scheduling is not throughput optimal [Zhang-Shen-Keslassy-McKeown’03]
- Throughput guarantees of maximal scheduling: 1/Kmax

  Chaporkar-Kar-Sarkar’05
  Wu-Srikant-Perkins’07

  1/Kmax: (a) worst network topology and size, (b) worst maximal scheduling algorithm, (c) worst relative mean arrival rates



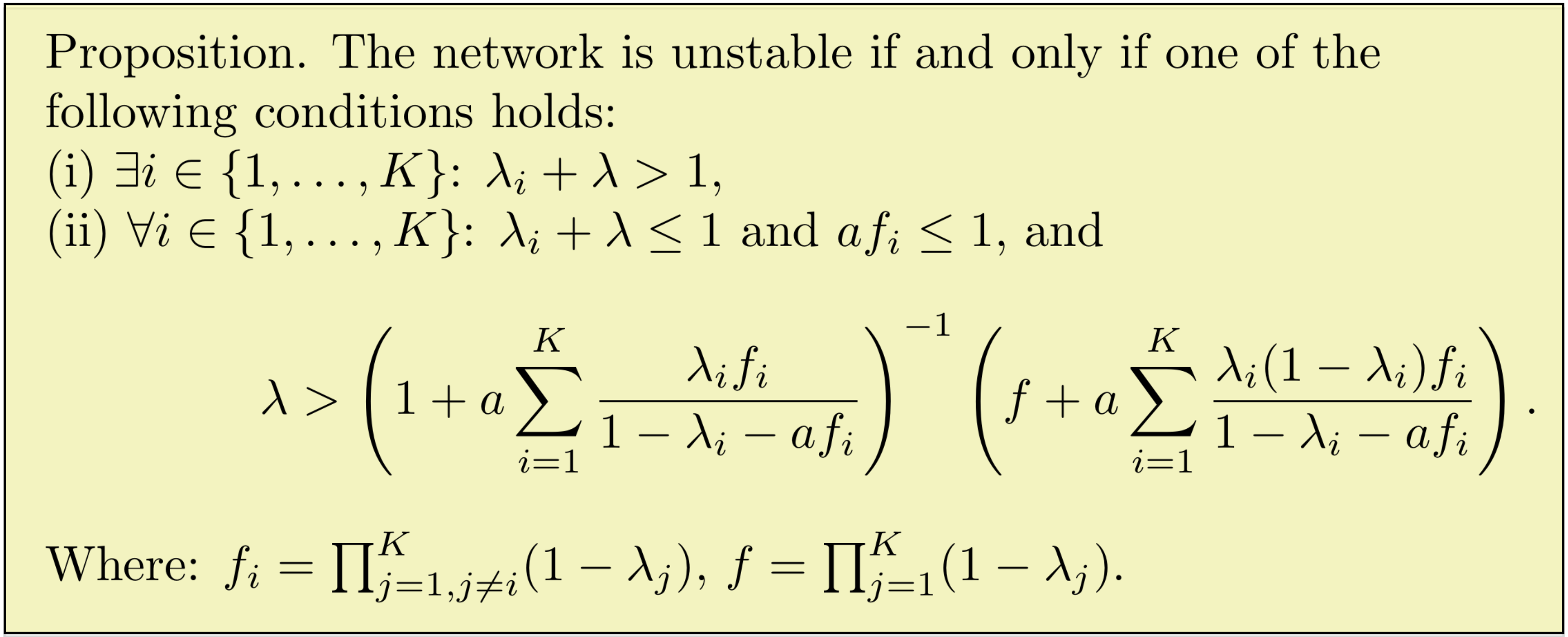
K = Kmax

Worst maximal scheduling: Max Size + tie breaking (priority given to external link)

Throughput = 1/Kmax only when Kmax tends to infinity

Throughput of MaxSize scheduling

- MaxSize scheduling with probabilistic tie breaking (a=proba to use schedule {0} when 2 active links only, inlcuding 0)

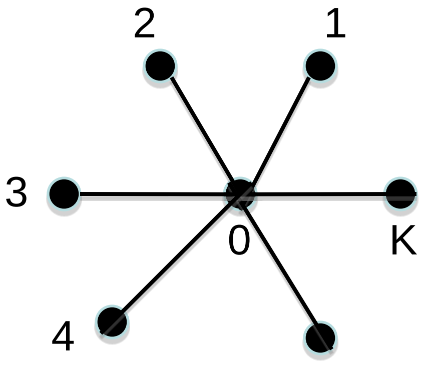


Proposition. The network is unstable if and only if one of the following conditions holds:
(i) $\exists i \in \{1, \ldots, K\}$: $\lambda_i + \lambda > 1$,
(ii) $\forall i \in \{1, \ldots, K\}$: $\lambda_i + \lambda \leq 1$ and $a f_i \leq 1$, and

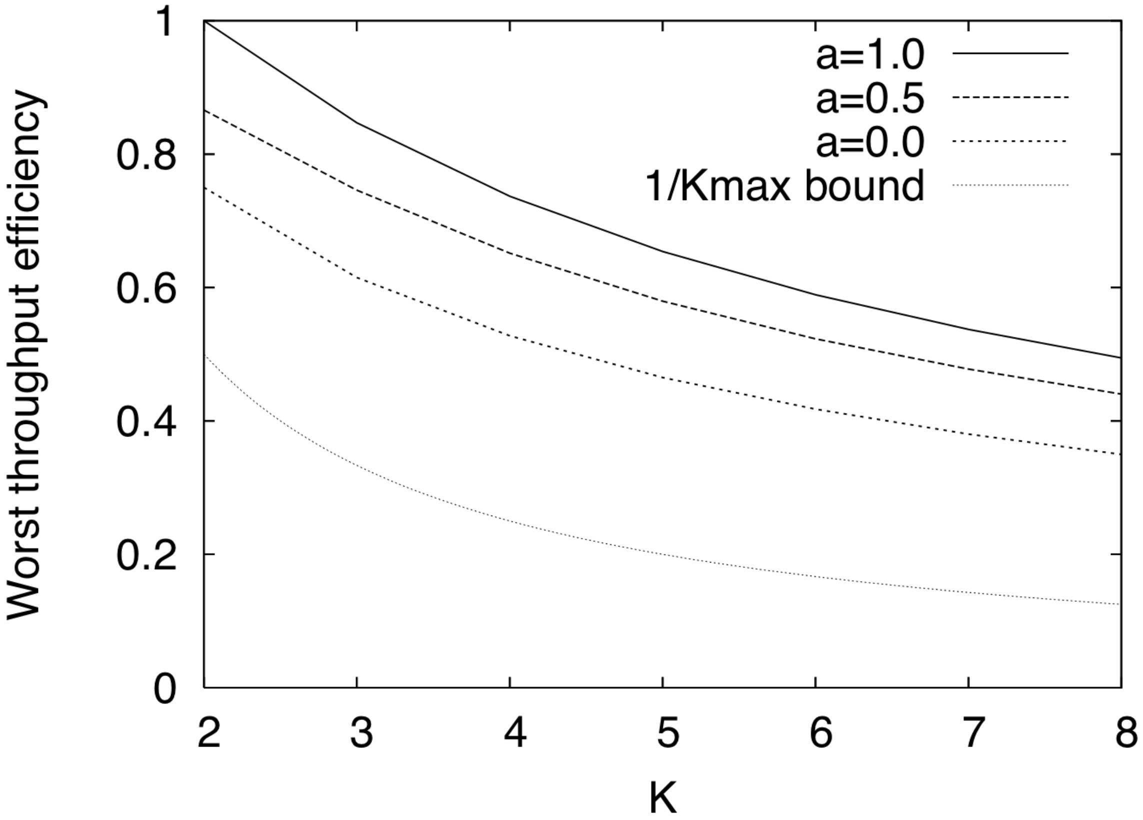
$$\lambda > \left(1 + a \sum_{i=1}^{K} \frac{\lambda_i f_i}{1 - \lambda_i - a f_i}\right)^{-1} \left(f + a \sum_{i=1}^{K} \frac{\lambda_i (1-\lambda_i) f_i}{1 - \lambda_i - a f_i}\right).$$

Where: $f_i = \prod_{j=1, j\neq i}^{K} (1-\lambda_j)$, $f = \prod_{j=1}^{K} (1-\lambda_j)$.

Throughput of MaxSize scheduling



K = Kmax



Beyond MaxSize scheduling?

- Can asynchronous systems realize MaxWeight algorithm?
  Idea: simulated anealing

$$p_l = 1 - \exp(-Q_l/T)$$

queue length temperature

Hajek, *Cooling schedules for optimal annealing*, Maths of Op. Research, 1988

Beyond MaxSize scheduling?

- Can asynchronous systems realize MaxWeight algorithm?
  Idea: simulated anealing

$$p_l = 1 - \exp(-\underset{\text{queue length}}{Q_l}/\underset{\text{temperature}}{T})$$

Hajek, *Cooling schedules for optimal annealing*, Maths of Op. Research, 1988

When the temperature is low enough:

$$\tau(m) \to C \exp(\sum_{l \in m} Q_l/T)$$

$$P\left[\tau(m) = \tau_{MW}(m)\right] \to 1$$

Beyond MaxSize scheduling?

- Can asynchronous systems realize MaxWeight algorithm?
  Idea: simulated anealing

$$p_l = 1 - \exp(-Q_l/T)$$

queue length temperature

Hajek, *Cooling schedules for optimal annealing*, Maths of Op. Research, 1988

When the temperature is low enough:

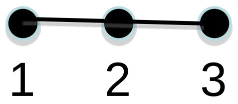
$$\tau(m) \to C \exp(\sum_{l \in m} Q_l/T)$$

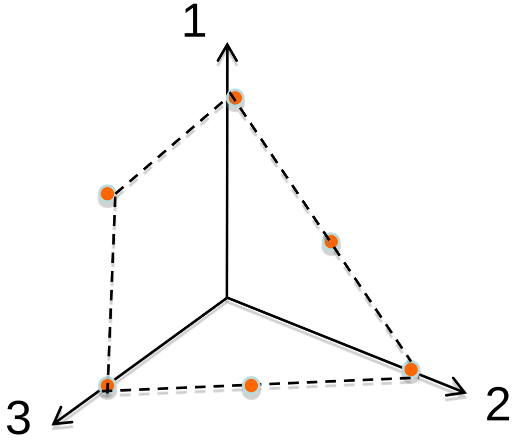
$$P\left[\tau(m) = \tau_{MW}(m)\right] \to 1$$

- Issue: we cannot implement this! Because the time to get a schedule is roughly:

$$\beta \times \exp(Q_l/T)$$

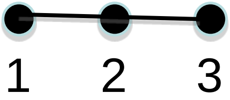
Moving/Adding points in the rate region

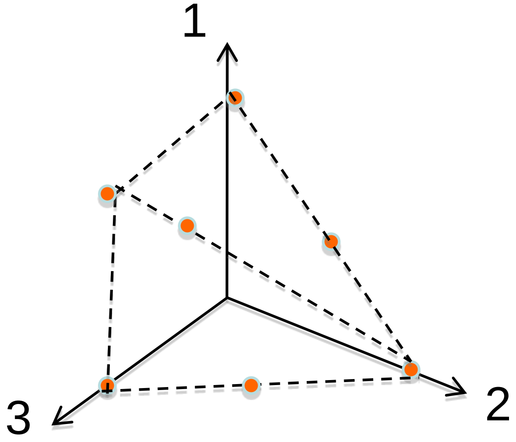




MaxSize scheduling

Moving/Adding points in the rate region

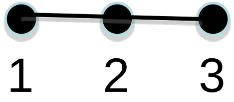


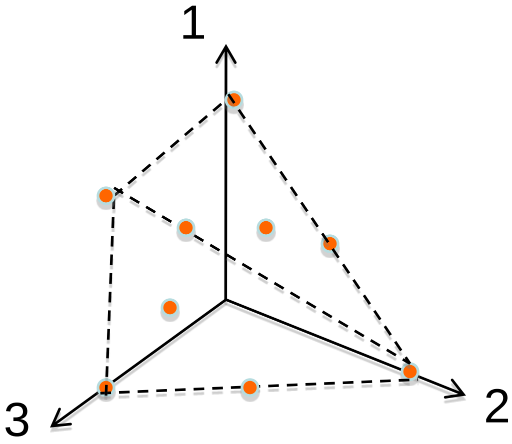


Random scheduling

- All traffic intensity below a point of the rate region is stabilized
- Randomly choosing maximal scheduling helps!

Moving/Adding points in the rate region





Random scheduling

- All traffic intensity below a point of the rate region is stabilized
- Randomly choosing maximal scheduling helps!
- Adding points (playing with the trans. probas) helps more!

Conclusions

- Random access is difficult to analyze …

- … but performs well, and we are close to provide throughput optimal schemes based on simple CSMA algorithms

- More complicated schemes (distributed MW schemes) have no future

Thank you!