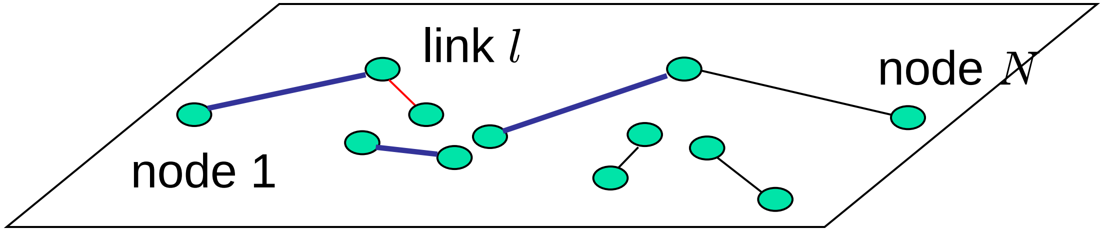
Achieving Both High Throughput and Low Delay with CSMA-Like Algorithms: A Virtual Multi-Channel Approach

Xiaojun Lin, Associate Professor
School of Electrical and Computer Engineering
Purdue University, West Lafayette
http://min.ecn.purdue.edu/~linx

Joint work with Po-Kai Huang

Distributed Optimization of Large-Scale Wireless Networks

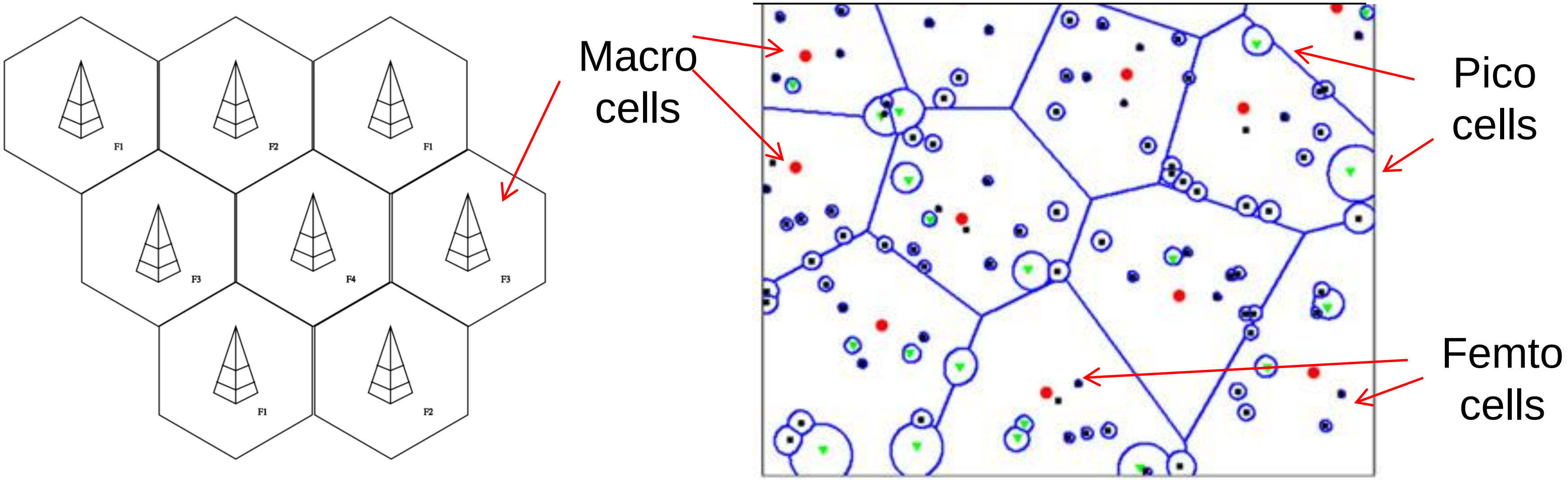


- A large number of potential wireless transmissions
- Neighboring transmissions interfere with each other
- Goals:
  - Maximize system capacity and other important QoS metrics (such as delay) subject to limited spectrum
  - Implement in a fully distributed manner
  - Automatically adapt to changing topology and traffic loads
- Useful in the context of ad hoc wireless networks

…… Also Increasingly Important for Cellular Systems

Homogeneous Cellular Networks (Past)

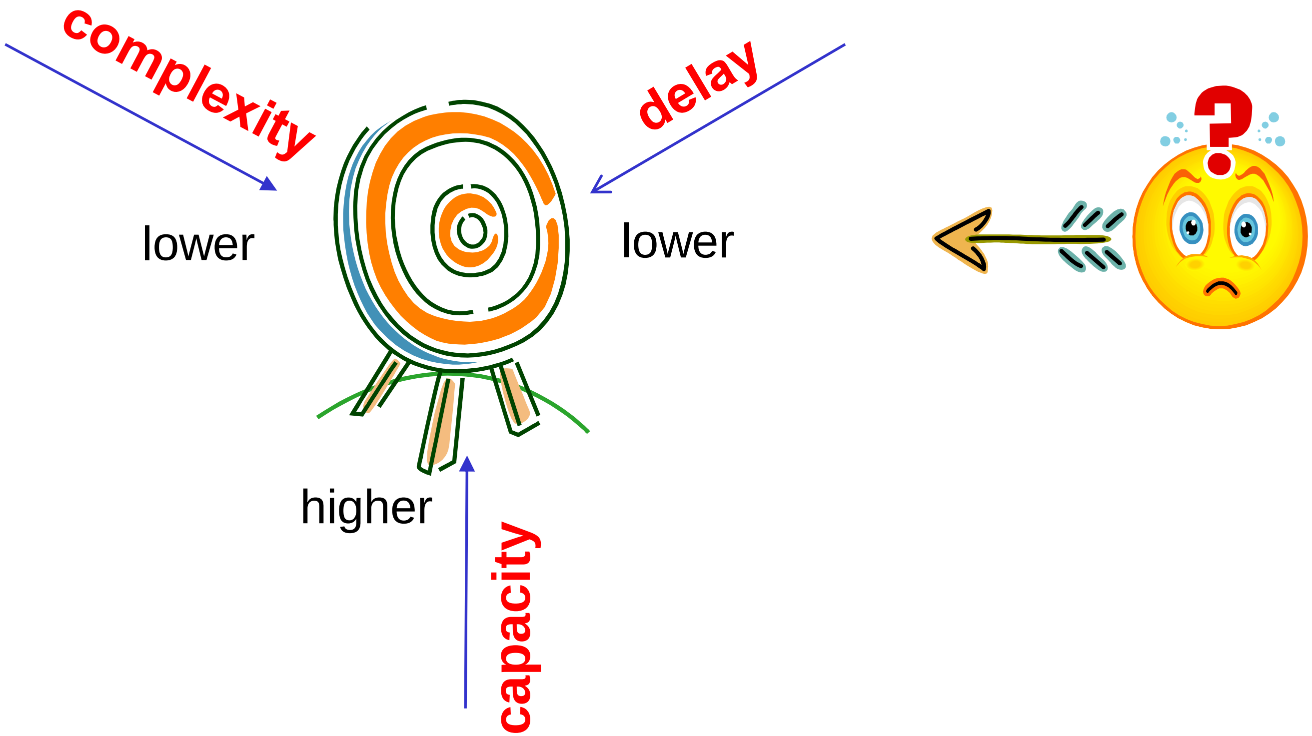
Heterogeneous Cellular Networks (Now and the Future)



(Source: Prof. Jeffery Andrews, UT Austin)

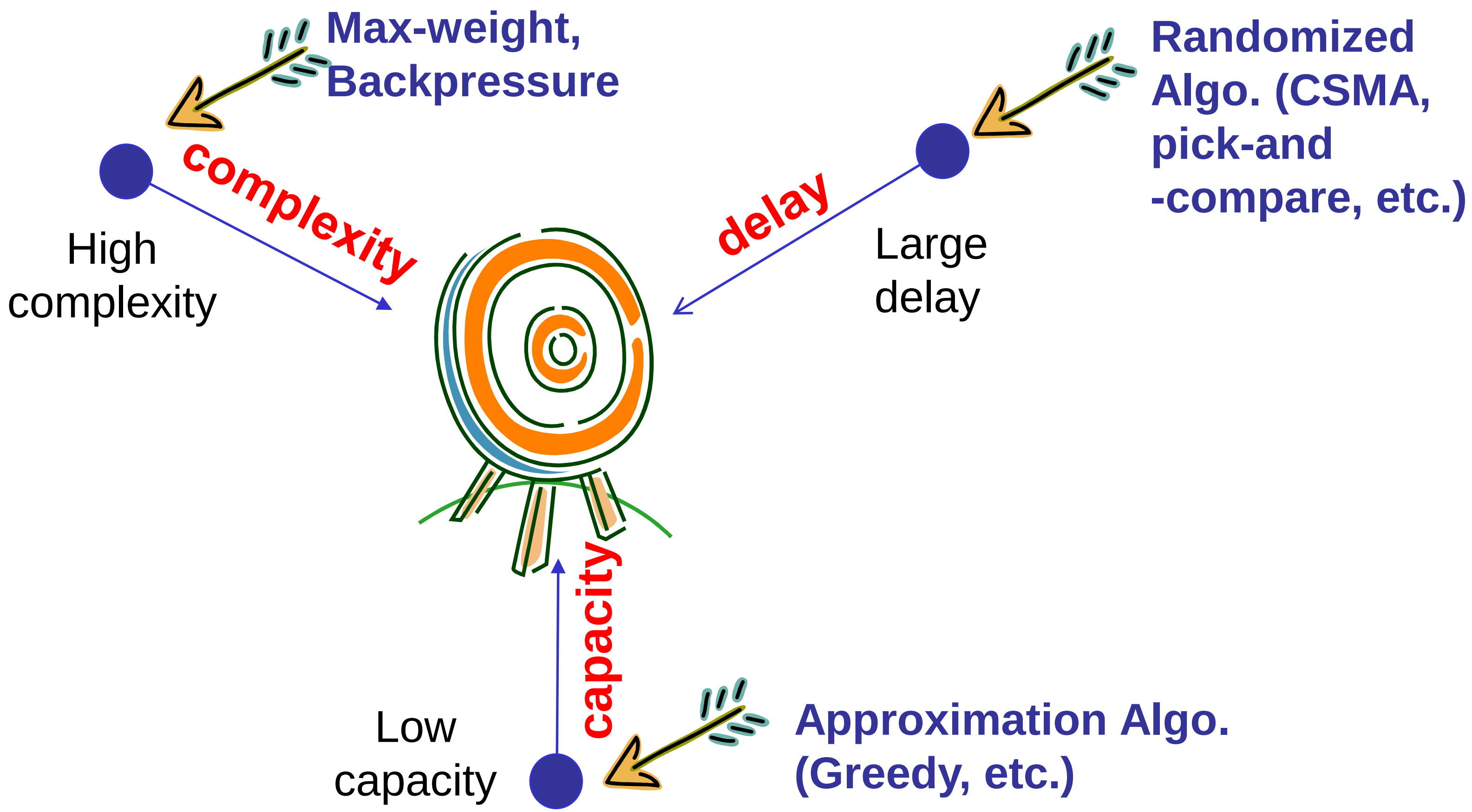
- Cellular topology also becomes more ad hoc
- Are analytical techniques and control algorithms for distributed optimization of ad hoc network algorithms good enough to manage heterogeneous cellular networks?

Three Important Goals for Distributed Optimization of Large-Scale Wireless Networks



- All three dimensions are highly critical
- **The Key Question**: How to achieve both ***high-capacity*** and ***low delay*** with ***low-complexity algorithms?***

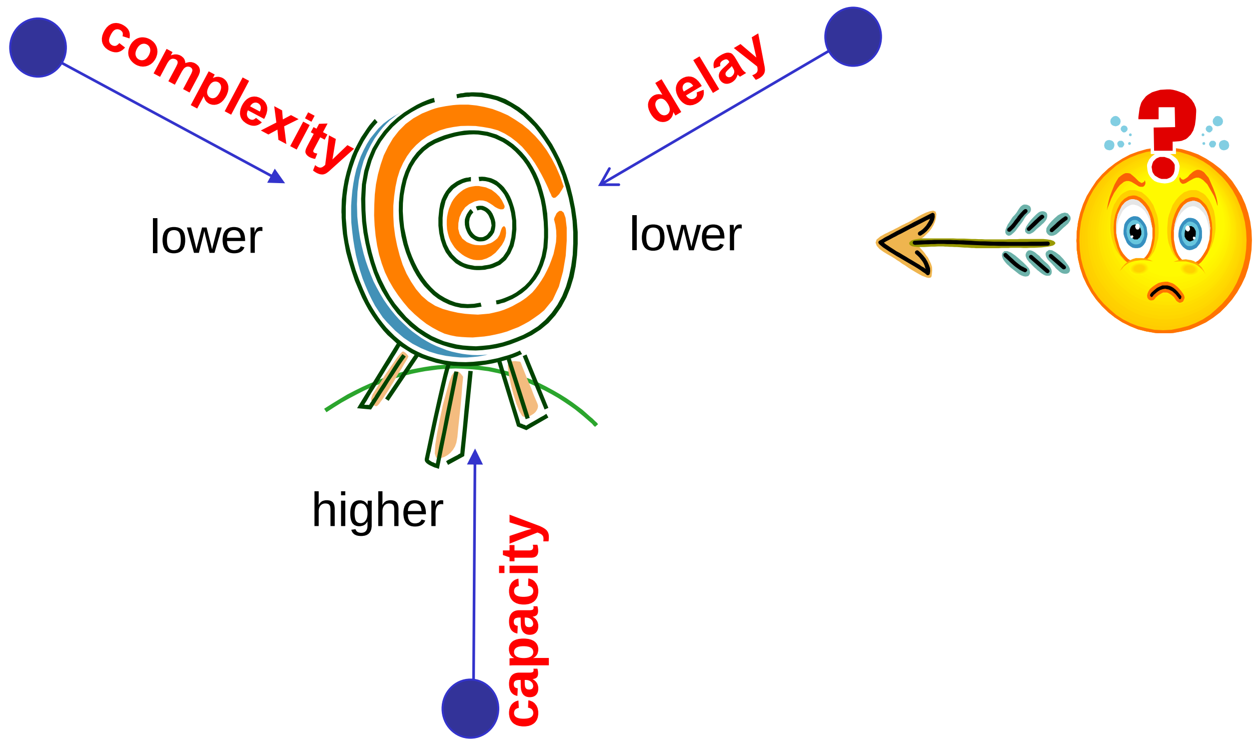
Unsatisfactory State-of-the-Art



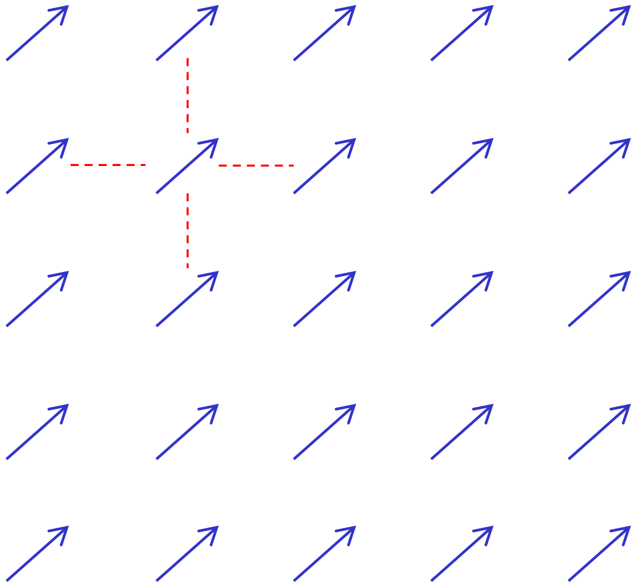
- No existing algorithms have been able to achieve all three goals at the same time!

A Conjecture on the Capacity-Delay-Complexity Tradeoff

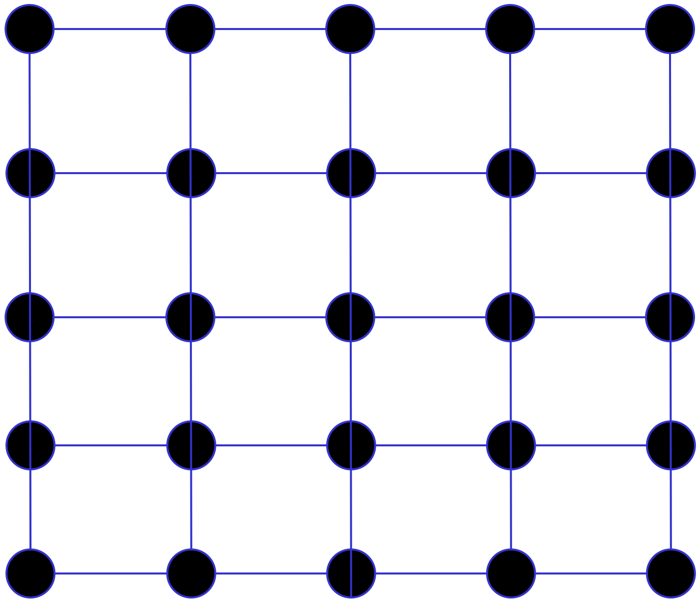
There exists worst-case topology such that, even to achieve a diminishingly small fraction of the optimal capacity, either the complexity or the delay must grow exponentially with the network size [Shah, Tse & Tsitsiklis, 2011].



Why So Difficult? (A Motivating Example)

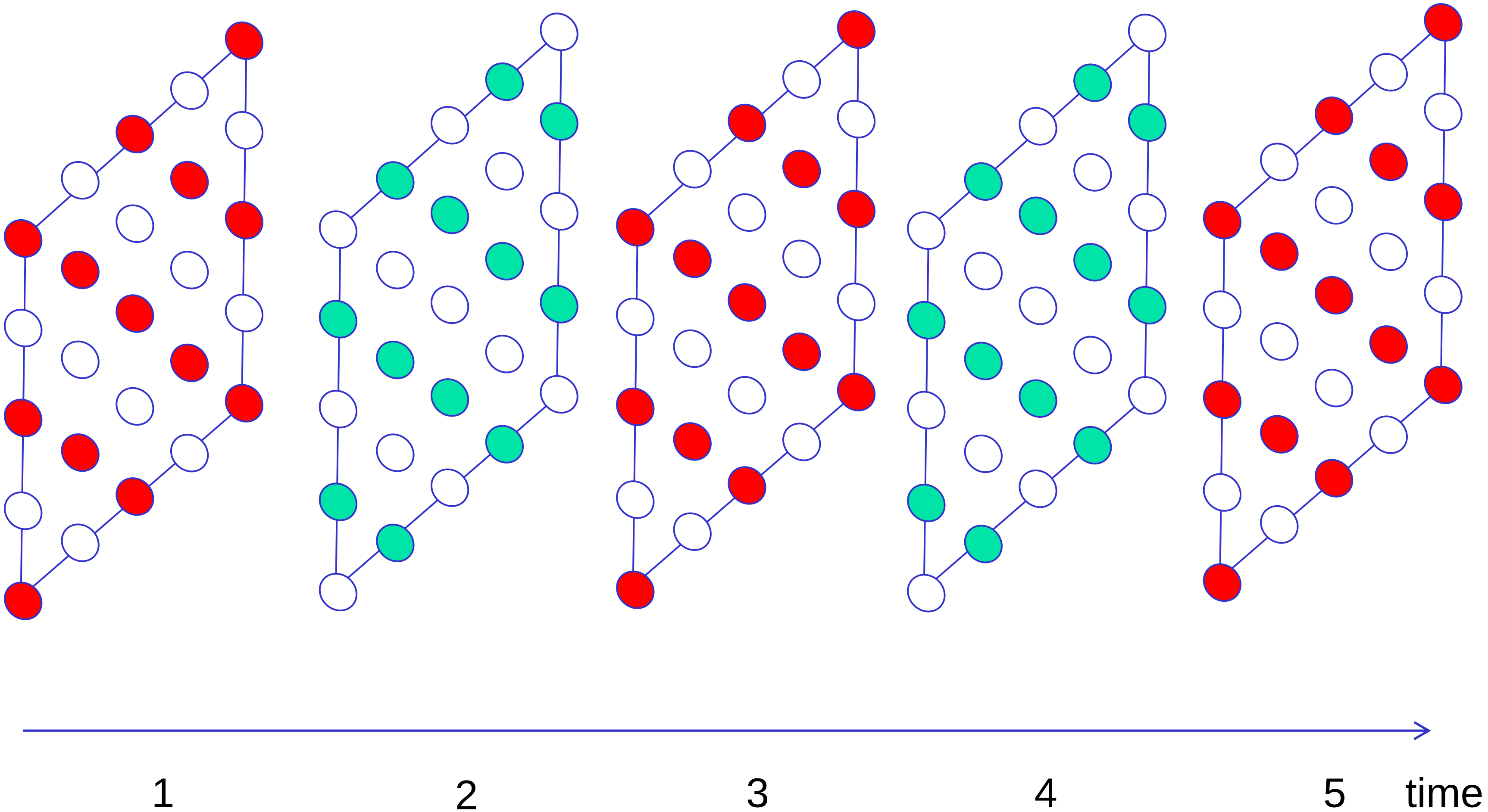


25 unit-capacity links. Each interferes with its 4 neighbors



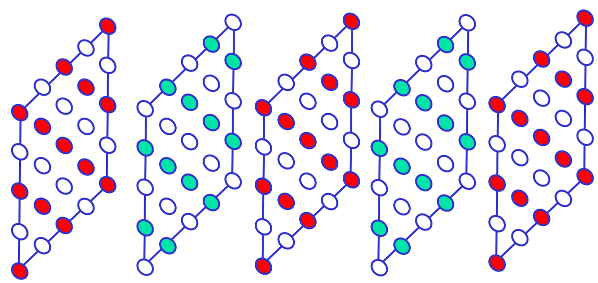
The corresponding ***conflict graph***: each vertex represents a link, each edge represents interfering links

Schedules to Achieve the Maximum Capacity



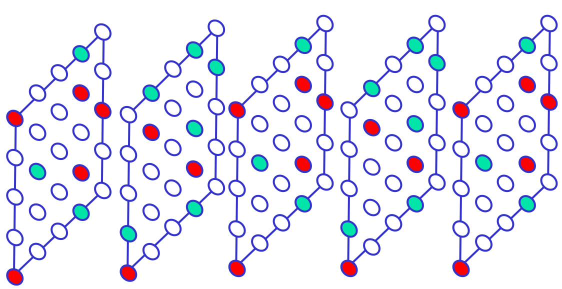
The Difficulty …

- However, computing the optimal schedule in each time-slot in general incurs extremely high complexity

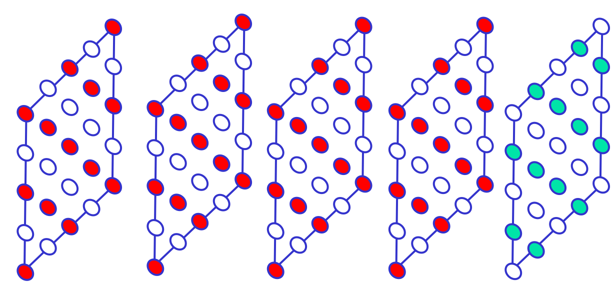


Max-weight Backpressure

- For practical purpose, one either has to reduce the quality of schedule …
- Or, reduce the frequency of computing a new schedule

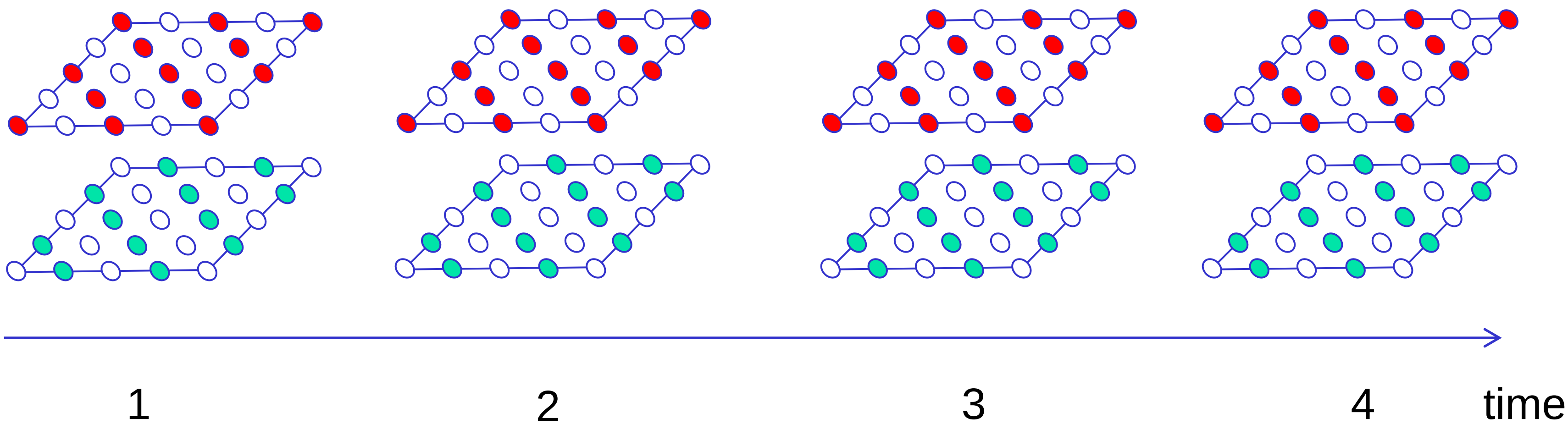


Greedy



CSMA

What If We Divide the Same Frequency Band into Two Channels?



- **A fixed multi-channel schedule will lead to both high throughput and low delay**
- Since we only need to compute the schedule once, we may be able to design algorithms that require **low complexity** in each time slot
- ***Open Question***: Can this simple idea be generalized?

Our Contribution: Virtual-Multi-Channel (VMC-) CSMA

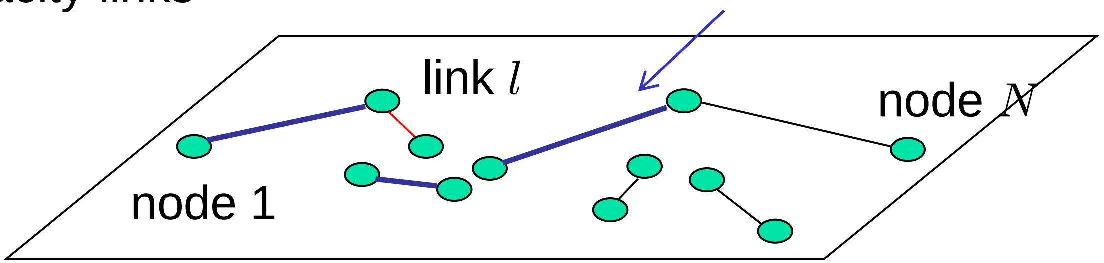
- Using the concept of virtual channels, VMC-CSMA extends the above idea to both ***single-channel*** and ***multi-channel systems*** with arbitrary topologies
- Like CSMA, VMC-CSMA ***distributively*** computes the near-optimal schedule across all virtual channels
- VMC-CSMA can provably achieve arbitrarily ***close-to-optimal system utility*** with ***complexity*** that grows ***logarithmically*** with the network size
- Both the ***packet delay*** and ***HOL (head-of-line) waiting time*** at every link can be tightly bounded.

Outline

- *System Model and Related Work*
- Virtual-Multi-Channel CSMA
- Capacity, Delay and Complexity
- Simulation Results
- Conclusion and Discussions

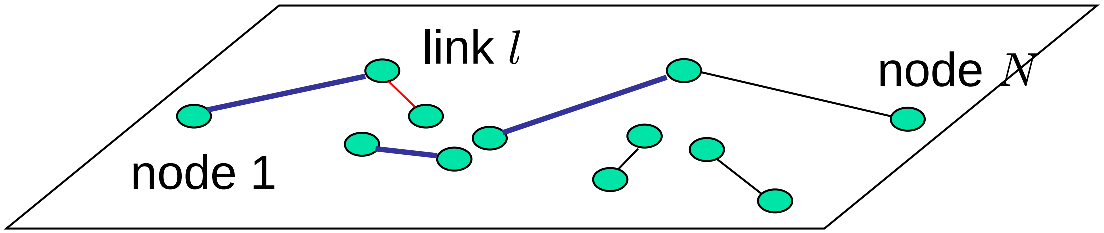
System Model: A Single-Hop Wireless Network with an Ad Hoc Topology

N nodes
L unit-capacity links



- Denote a schedule by $\vec{V} = [V_1, V_2, ...V_L]$
  - V_l = 1 if link l is chosen to be active
- Feasible schedule: no active links interfere with each other
- r_l: long-term average service-rate of link l
- Capacity region Ω: the set of $[r_l]$ that the network can support
  - Ω equals to the convex hull of all feasible schedules [Tassiulas & Ephremides '92]

Utility Maximization



- Each link l has a utility function $U_l(r_l)$
  - The utility function is positive, non-decreasing, and concave
  - Also accounts for fairness
- **Goal:** Develop ***low-complexity*** and ***low-delay*** algorithms to ***maximize*** the total system utility subject to capacity constraints

$$\max \sum_{l=1}^{L} U_l(r_l), \qquad \text{subject to } [r_l] \in$$

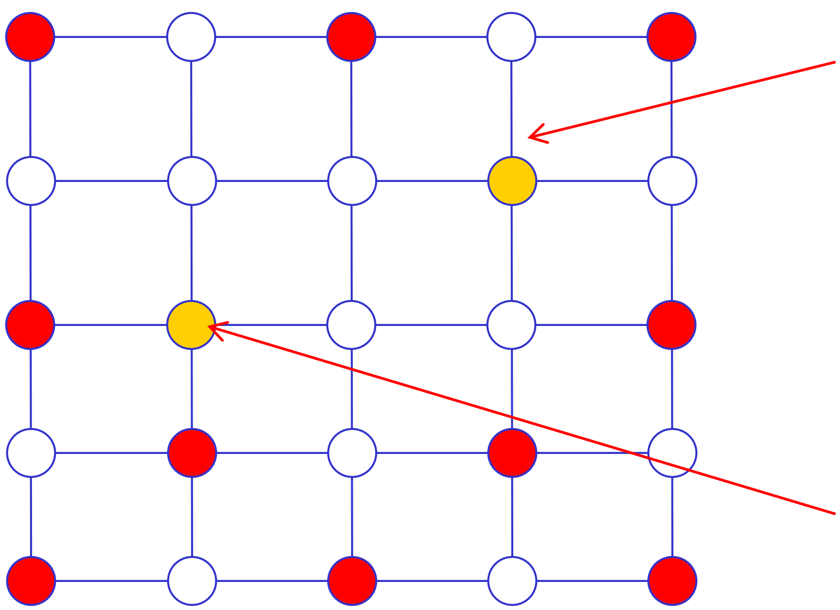
Related Work

The utility maximization problem itself has been extensively studied in the literature.

- Algorithms based on ***max-weight*** (and back-pressure for multi-hop) [Tassiulas & Ephremides '92, Neely & Modiano '03, Lin & Shroff '04, and many others]
  - Provably optimal but of exponentially-high complexity
- ***Approximation algorithms*** with provable efficiency ratios [by Lin, Shroff, Srikant, Prashant, Sarkar, Zussman, Modinan, Joo, and many others]
  - Incur lower complexity but can only guarantee a small fraction of the optimal capacity
- ***Randomized algorithms***: CSMA [Liew et al '09, Jiang & Walrand '10, Marbach et al '10, Shin & Shah '10] or pick-and-compare [Tassiulas & Ephremides '98]
  - Provably optimal and of low complexity
  - But they suffer from large delay

Standard CSMA Algorithm: Update Phase

- Choose a random decision schedule (which is feasible) from a set S [Ni & Srikant '09]
- Update the transmission schedule of each link belonging to the decision schedule



$$\mathbf{P}[V_l = 1] = \frac{\exp(\alpha Q_l(t))}{1 + \exp(\alpha Q_l(t))}$$

$$\mathbf{P}[V_l = 0] = \frac{1}{1 + \exp(\alpha Q_l(t))}$$

$V_l = 0$

Standard CSMA Algorithm

- **Rate Control:** The injection rate of each link is determined by

$$r_l(t) = \arg\max_{r \geq 0} U_l(r) - \beta r Q_l(t)$$

  - Inject $A_l(t)$ number of packets such that

$$\boldsymbol{E}[A_l(t)] = r_l(t)$$

- **Queue-length Update:**

$$Q_l(t+1) = [Q_l(t) + A_l(t) - V_l(t)]^+$$

Intuition Behind the Standard CSMA Algorithm

- Suppose that the (relative) queue lengths at all links change very slowly compared to the schedule update
  - Known as the ***time-scale separation*** assumption
- The update rule will lead to the following stationary distribution

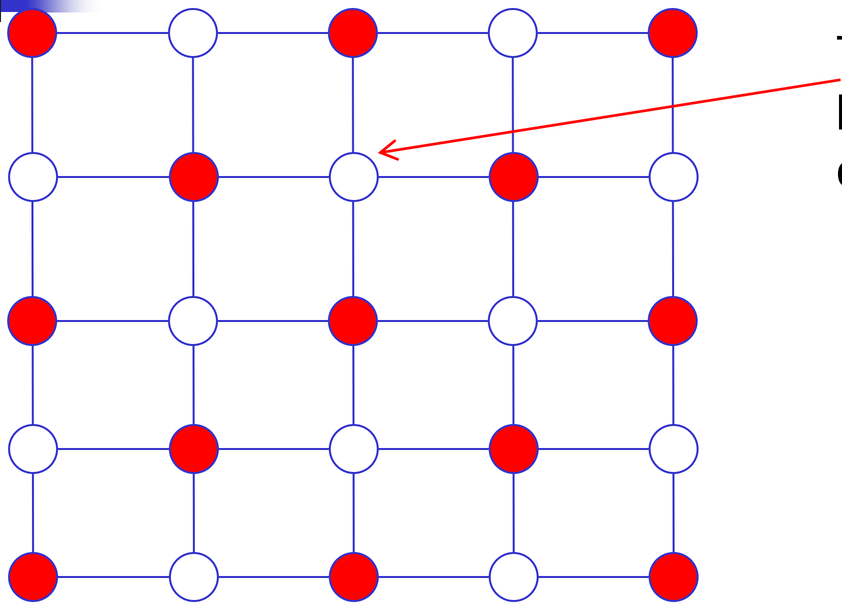
$$\mathbf{P}[\vec{V}(t) = \vec{V}] \quad \propto \quad \exp\left[\alpha \sum_{l=1}^{L} Q_l(t) V_l\right]$$

Weight of the schedule

  - As α increases, the schedule with the max-weight will be reached with probability close to 1

- **The standard CSMA algorithm computes the max-weight schedule with fully distributed and low-complexity control**

The Starvation Problem: An Example



To turn on link l: all four neighboring links must be off (a small probability event when α is large!)

$$\mathbf{P}[V_l = 1] = \frac{\exp(\alpha Q_l(t))}{1 + \exp(\alpha Q_l(t))}$$

$$\mathbf{P}[V_l = 0] = \frac{1}{1 + \exp(\alpha Q_l(t))}$$

- The starvation problem: the standard CSMA algorithm will be "stuck" into one of the max-weight schedules for a long time.
  - Lead to large delay!

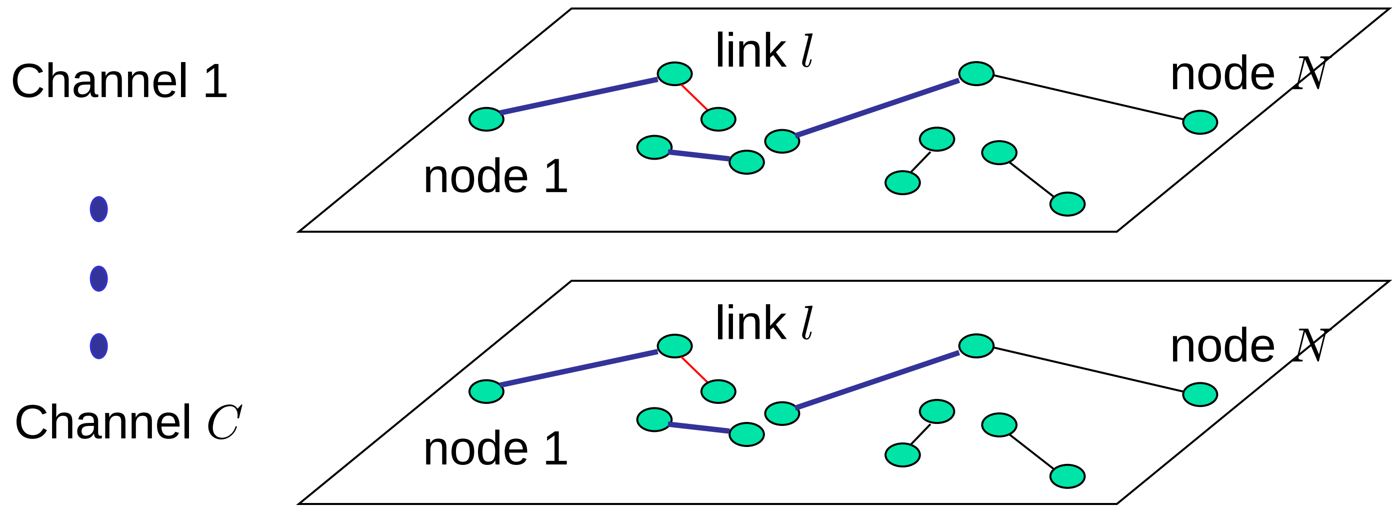
Improvements to the CSMA algorithms

- Lower capacity
  - [Jiang et al '11, Subramanian & Alanyali '11]: show reduced mixing time when the offered load is small
  - [Lam et al '12]: each link uses one channel in a multi-channel system
- Partitioning approach
  - [Shah & Shin '10]: divide the network into finite-size partitions and run CSMA in each partition
  - To approach closer to the optimal capacity, the partition size must be large, which again leads to large delay
- Fine tuning the update rules
  - [Lee et al '12]: Tuning between Glauber dynamics to metropolis algorithm
  - Unlikely to alter the exponential growth of delay
- Restricted topology
  - [Li & Eryilmaz '12]: complete graph
  - [Lotfinezhad & Marbach '11]: regular grid

Outline

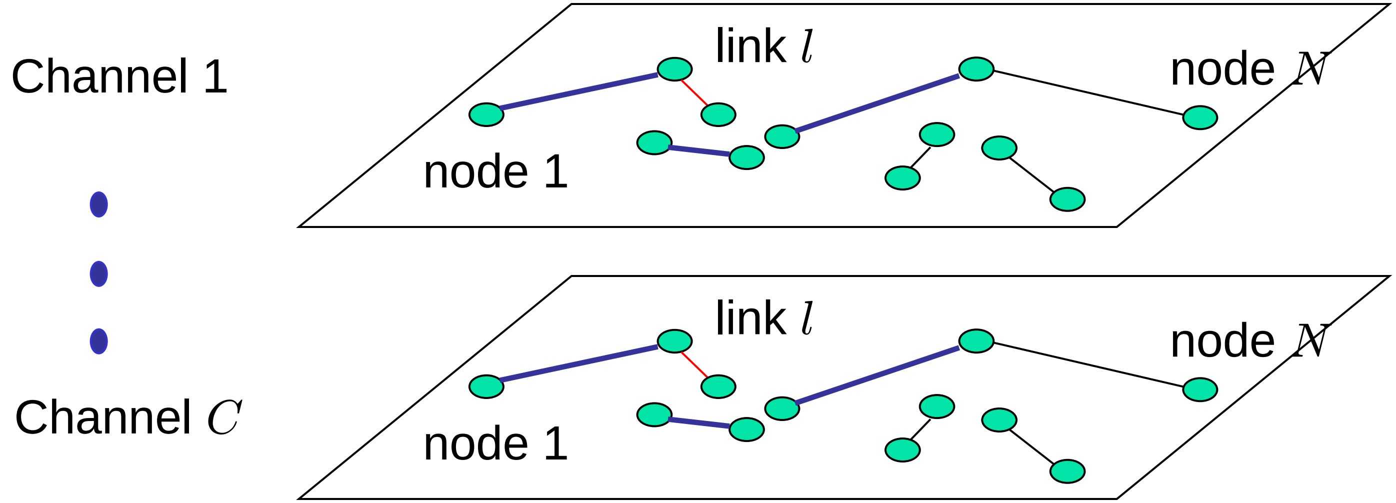
- System Model and Related Work
- *Virtual-Multi-Channel CSMA*
- Capacity, Delay and Complexity
- Simulation Results
- Conclusion and Discussions

Using Multiple Channels



- C channels, each with $1/C$ of the bandwidth
- There is a feasible schedule $\vec{V}^k = [V_l^k]$ computed for each channel k. We call $\vec{V} = [\vec{V}^k]$ the ***global schedule.***
- $x_l(\vec{V}) = \sum_{k=1}^{C} V_l^k$: total number of channels on which link l is active.
- $r_l(\vec{V}) = x_l(\vec{V})/C = \sum_{k=1}^{V} V_l^k/C$: average rate of link l

Searching for the Right Global Schedule



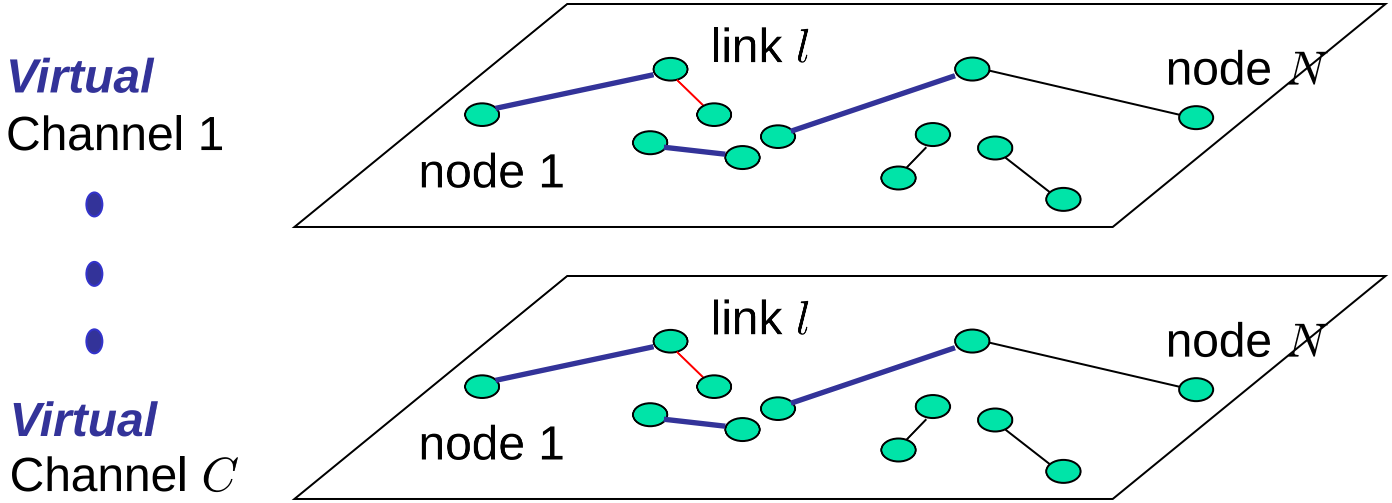
- Our goal is to find one global schedule that solve the following optimization problem:

$$\max \quad \sum_{l=1}^{L} U_l(r_l) = \sum_{l=1}^{L} U_l\left(\sum_k V_l^k / C\right)$$

$$\text{subject to} \quad \vec{V}^k \text{ is a feasible schedule for all channels } k$$

- Intuitively, the solution should approach the optimal system utility (without channelization) when C is large

Single-Channel Systems: Virtual Channels



- At each time, a random virtual channel k is chosen uniformly from 1 to C
- All links then use $\vec{V}^k$ to determine their transmissions
- $r_l(\vec{V}) = x_l(\vec{V})/C = \sum_{k=1}^{V} V_l^k/C$: the probability that link l is served, independently across time.
  - ***Key for achieving good throughput and low delay.***

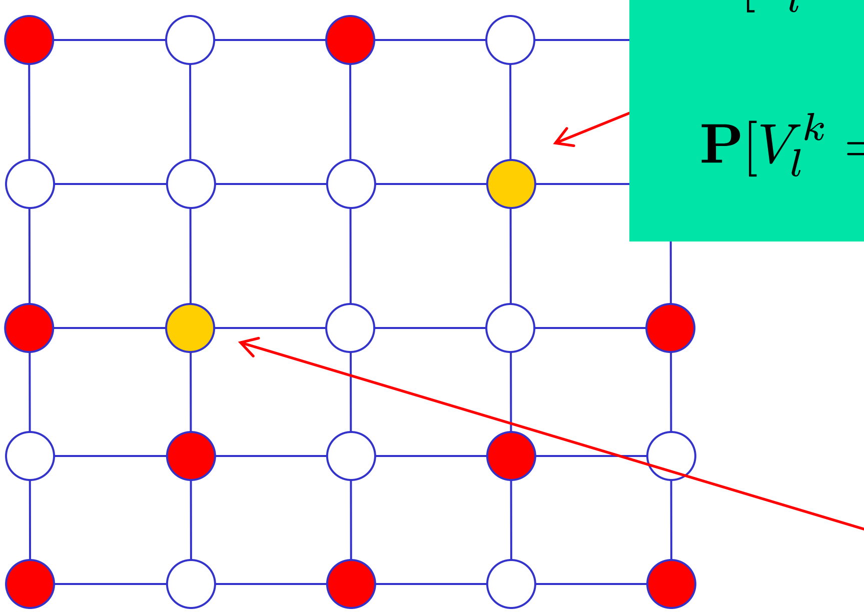
VMC-CSMA Algorithm: Update Phase

- Choose a random decision schedule (which is feasible) from a set S
- For each link in the decision schedule, update all C channels
- Broadcast the update to neighbors

virtual channel k

$$\mathbf{P}[V_l^k = 1] = \frac{\exp[\alpha U_l(r_l + \frac{1}{C})]}{\exp(\alpha U_l(r_l)] + \exp[\alpha U_l(r_l + \frac{1}{C})]}$$

$$\mathbf{P}[V_l^k = 0] = \frac{\exp[\alpha U_l(r_l)]}{\exp(\alpha U_l(r_l)] + \exp[\alpha U_l(r_l + \frac{1}{C})]}$$



$V_l^k = 0$

VMC-CSMA Algorithm

- **Transmission Phase:** A common virtual-channel $k(t)$ is chosen by all links in the network uniformly at random from 1 to C, and each link l transmits a packet if $V_l^{k(t)} = 1$
- **Rate Control:** a new packet is injected to link l only if a packet is served at link l.
  - The number of packets in the buffer of link l is always 1
  - Known as window-based flow control (with window size = 1)

Key Differences from Standard CSMA: Update Phase

- VMC-CSMA: If all interfering links in $I(l)$ are not using channel k, set

$$\mathbf{P}[V_l^k = 1] = \frac{\exp[\alpha U_l(r_l + \frac{1}{C})]}{\exp(\alpha U_l(r_l)] + \exp[\alpha U_l(r_l + \frac{1}{C})]}$$

$$\mathbf{P}[V_l^k = 0] = \frac{\exp[\alpha U_l(r_l)]}{\exp(\alpha U_l(r_l)] + \exp[\alpha U_l(r_l + \frac{1}{C})]}$$

- Standard CSMA: If all interfering links in $I(l)$ are inactive, set

$$\mathbf{P}[V_l = 1] = \frac{\exp(\alpha Q_l(t))}{1 + \exp(\alpha Q_l(t))}$$

$$\mathbf{P}[V_l = 0] = \frac{1}{1 + \exp(\alpha Q_l(t))}$$

Key Differences from Standard CSMA: Update Phase

- VMC-CSMA: If all interfering links in $I(l)$ are not using channel k, set

$$\mathbf{P}[V_l^k = 1] = \frac{\exp[\alpha U_l(r_l + \frac{1}{C})]}{\exp(\alpha U_l(r_l)] + \exp[\alpha U_l(r_l + \frac{1}{C})]}$$

$$= \frac{\exp[\alpha U_l(r_l + \frac{1}{C}) - \alpha U_l(r_l)]}{1 + \exp[\alpha U_l(r_l + \frac{1}{C}) - \alpha U_l(r_l)]}$$

$$\approx \frac{\exp[\alpha U_l'(r_l)\frac{1}{C}]}{1 + \exp[\alpha U_l'(r_l)\frac{1}{C}]}$$

  - Recall that $U'_l(r)$ is decreasing in r.
- **Key to VMC-CSMA:** The larger r_l is, the less likely the link l will turn on a new virtual channel.
  - The decisions at different channels are coordinated
  - **Avoid the starvation problem!**

Key Differences from Standard CSMA: Rate Control

- VMC-CSMA: window-based flow control
  - a new packet is injected to link l only if a packet is served at link l

- Standard CSMA: rate is chosen to maximize net utility

$$r_l(t) = \arg\max_{r \geq 0} U_l(r) - \beta r Q_l(t)$$

- **Key to VMC-CSMA:** Since the scheduling decision has already accounted for the utility, there is no need to do so with rate control!
  - **Further reduce the backlog and delay**

Outline

- System Model and Related Work
- Virtual-Multi-Channel CSMA
- *Capacity, Delay and Complexity*
- Simulation Results
- Conclusion and Discussions

Provably-High Capacity

Lemma 1:

- Under the VMC-CSMA algorithm, the global schedule forms a Markov chain with stationary distribution given by

$$\mathbf{P}[\vec{V}(t) = \vec{V}] = \frac{1}{Z} \exp \left[\alpha \sum_{l=1}^{L} U_l(r_l(\vec{V})) \right]$$

  - where Z is a normalizing constant.
- Proved by checking that the local balance equation.

- ***Implication:*** As α increases, the probability of reaching the global schedule with the largest utility will approach 1.

How Large C Needs to be?

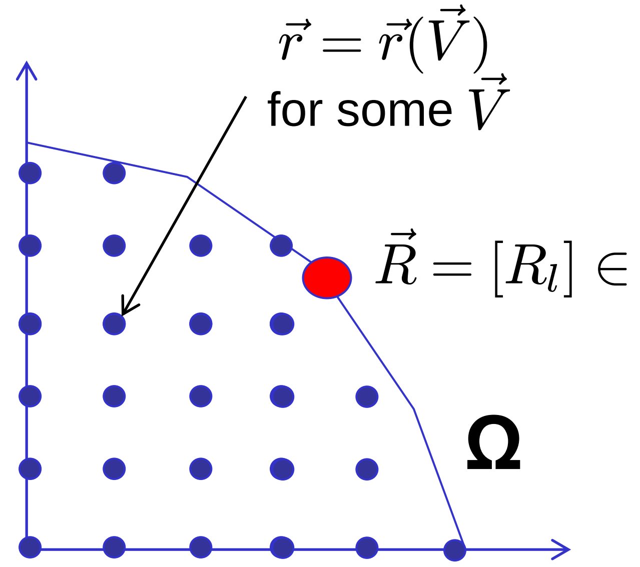
Lemma 2:

- If ϵ <= 0.1 and

$$C \geq \frac{2 \log L}{3\epsilon^2},$$

then for any $[R_l] \in \Omega$, there exists a feasible global schedule $\vec{V}$ such that

$$r_l(\vec{V}) \geq R_l - \epsilon, \text{ for all links } l.$$



C grows very slowly (log L) with the network size!

- ϵ = 0.1, L = 30 ⇒ C>= 226
- ϵ = 0.1, L = 1000 ⇒ C>= 461

Provably-High Capacity

Proposition 3:

- Suppose that $[R_l^*]$ is the solution to the following utility-maximization problem

$$\max \sum_{l=1}^{L} U_l(r_l) \qquad \text{subject to } [r_l] \in$$

where Ω is the optimal capacity region of the system (without channelization).

- For any ϵ <= 0.1, choose C > $\frac{2 \log L}{3\epsilon^2}$
- For any γ > 0, for sufficiently large α, the following holds

$$\mathbf{P}\left[\sum_{l=1}^{L} U_l(r_l(\vec{V}(t))) \geq \sum_{l=1}^{L} U_l(R_l^* - \epsilon)\right] \geq 1 - \gamma$$

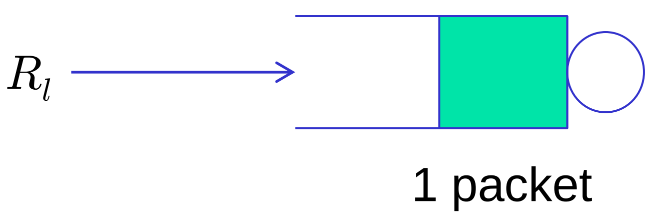
- **VMC-CSMA will attain near-optimal utility *with probability close to 1.***

Low Delay: Average Packet Delay

- Packet delay: from the time that a packet is injected to the buffer to the time that the packet is transmitted

Corollary 4:

- Let R_l denote the average rate of link l under VMC-CSMA, i.e., $$R_l = \sum_{\vec{V}} \mathbf{P}(\vec{V}) r_l(\vec{V})$$
- Then the average packet delay of link l is $1/R_l$



Little's Law: $1 = R_l \times W$

- Further, note that $$\sum_{l=1}^{L} U_l(R_l) \geq (1-\gamma) \sum_{l=1}^{L} U_l(R_l^* - \epsilon)$$

Low Delay: Another Notion of Delay

- However, packet delay does not fully capture the effect of the potential starvation problem.

  - Example: packet delays are

1,1,1…., 1, 1000

999 packets Last packet

- Despite the long starvation period (of 1000 time-slots), the average packet delay is 1.99, which is deceivingly low!

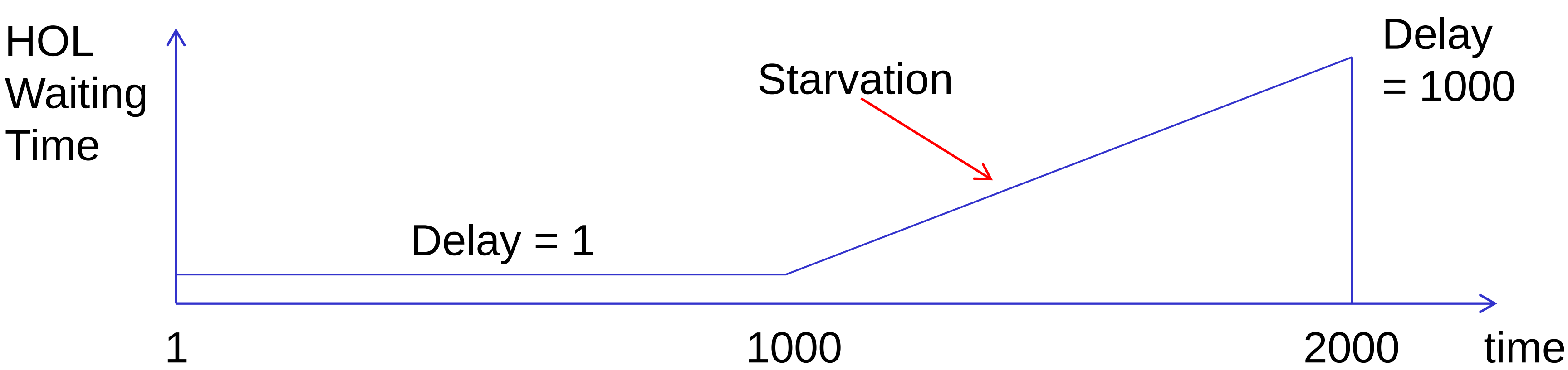
Head-of-Line (HOL) Waiting Time

- HOL waiting time: At each given time, the amount of time that the HOL packet has waited in the buffer.
- Example: packet delays are

1,1,1…., 1, 1000

999 packets. Last packet

- Average HOL waiting time is around 250.



Head-of-Line (HOL) Waiting Time

Proposition 5:

- Under the VMC-CSMA algorithm, for a fixed integer d > 0 the following holds for each link l,

$$\mathbf{P}[\text{HOL waiting time} \geq d] \cdot \ (1 - r_l^{\min})^d + \gamma,$$

$r_l^{\min}$ is the worst rate for link l among all global schedules with the maximum utility

γ approaches zero as α increases

$$r_l^{\min} = \min\left\{r_l(\vec{V}) : U(\vec{r}(\vec{V})) = U(\vec{r}(\vec{V}^{\max}))\right\}$$

- **Implication:** HOL waiting time decays exponentially fast.
- **Key to VMC-CSMA*:*** service is independent across time

Complexity and Overhead

Both complexity and overhead are linear in the number of virtual channels

- Each link needs to update the schedule in C virtual channels
  - Can be carried out in parallel
- Each link only needs its own schedule and that of its neighboring links
  - Only needs to exchange C-bit control messages with neighbors

- The number of virtual channels is O(log L), which grows very **slowly** with the size of the network.
- One control message can exchange the schedules at all virtual channels
  - **Overhead is low** even when C is ~1000 (=125 bytes)

Relationship to [Shah et. al. '11]

- In [Shah, Tse & Tsitsiklis '11], the authors show the following ***impossibility result***:
  - There exists worst-case network topology such that, even to attain a diminishingly small fraction of the optimal capacity, either the delay or the complexity must grow exponentially with the network size

- Our results seem to suggest that it is possible to attain both high capacity and low delay with low-complexity algorithms
  - However, our results do not contradict [Shah et. al. '11] due to two differences

Devavrat Shah, David N. C. Tse, and John N. Tsitsiklis, "**Hardness of Low Delay Network Scheduling**", *IEEE Transactions on Information Theory*, vol. 57, no. 12, 2011

Relationship to [Shah et. al. '11]

- ***Steady-state*** delay versus ***transient*** delay
  - The delay/backlog in [Shah et. al. '11] is defined as

$$\sup_{t \geq 0} \mathbf{E}\left[\sum_{l=1}^{L} Q_l(t)\right]$$

    - which accounts for ***transient*** delay before the system reaches steady-state

  - In contrast, we focus on ***steady-state*** packet delay and/or HOL waiting time
    - The time to reach steady-state may still be exponential in the worst-case, although it seems to be uncommon for practical topologies

Relationship to [Shah et. al. ’11]

- ***Closed-loop*** versus ***open-loop***
  - [Shah et. al. ’11] studies an ***open-loop*** system where the packet injection rate is not controlled.
  - We focus on a ***closed-loop*** system where the packet injection rate can be reduced when the backlog is large

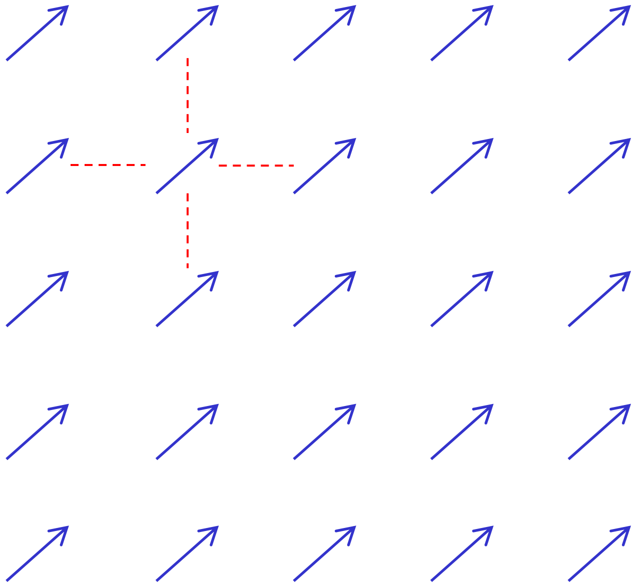
- Despite these differences, a low value of delay as we defined in our setting is useful in practice.

- The impossibility results in [Shah et. al. ’11] may not prevent us to develop low-complexity, low-delay and high-capacity algorithms that are useful in practice.

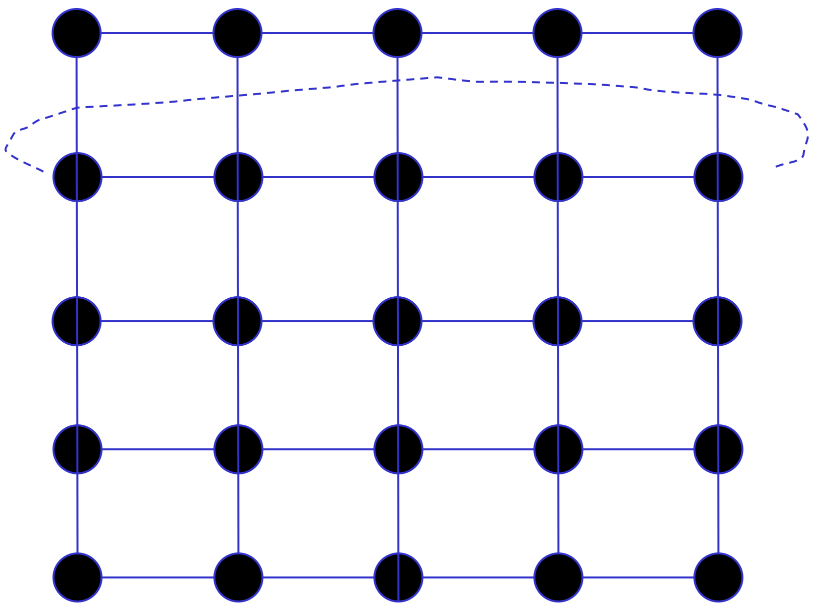
Outline

- System Model and Related Work
- Virtual-Multi-Channel CSMA
- Capacity, Delay and Complexity
- *Simulation Results*
- Conclusion and Discussions

Simulation Results

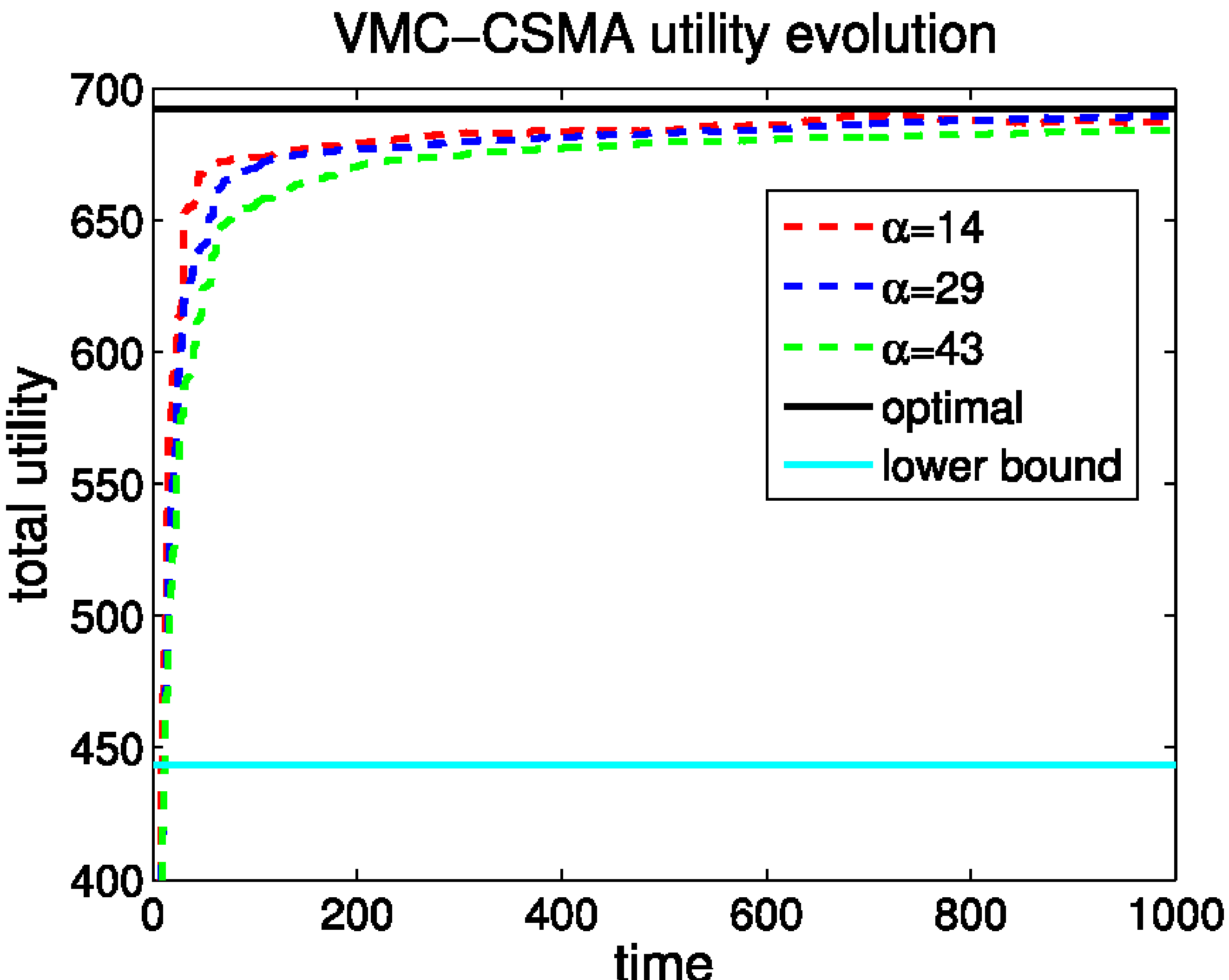


n x n torus



The corresponding ***conflict graph***:

Simulation Results: 8x8 torus

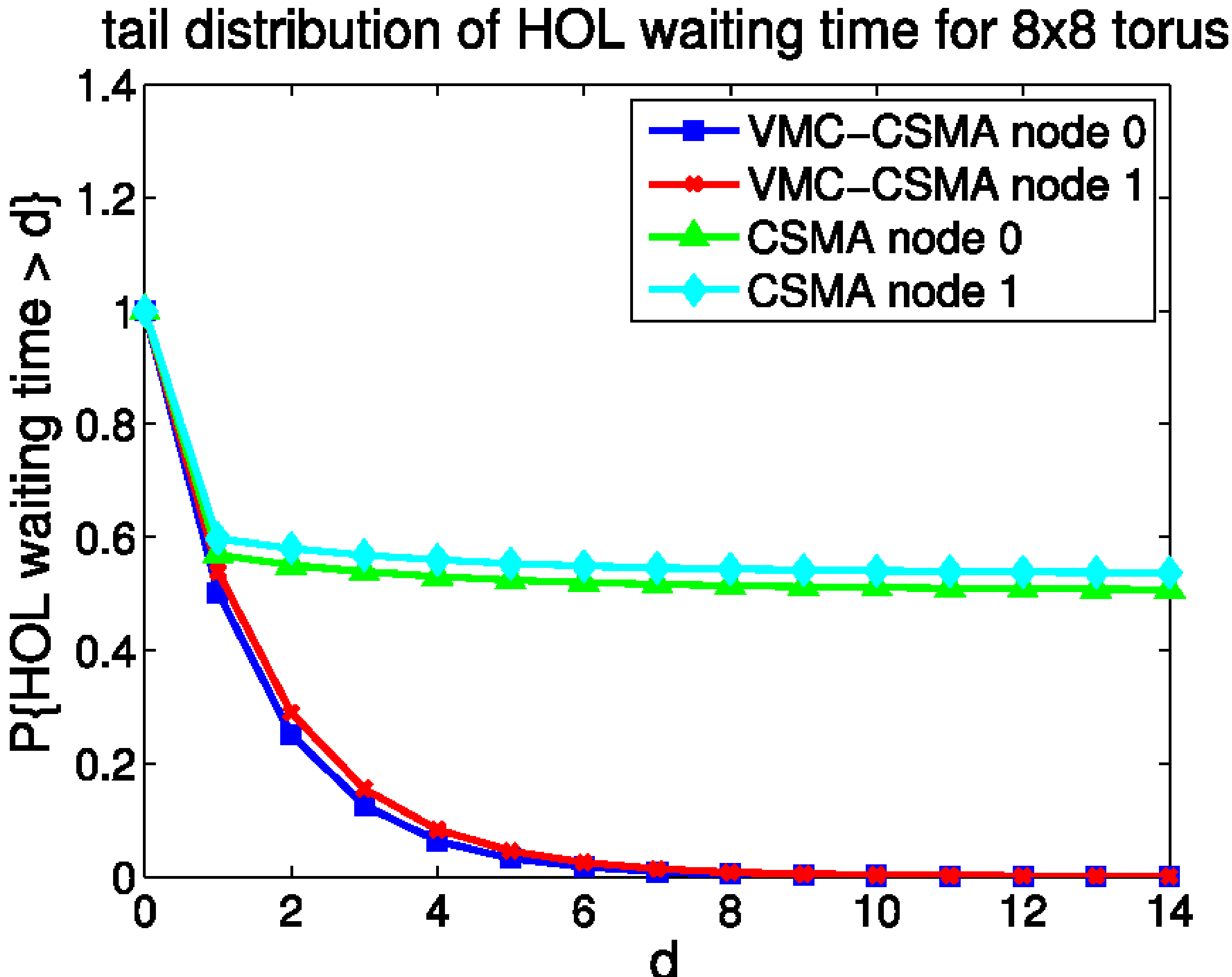


C=30
ϵ = 0.3

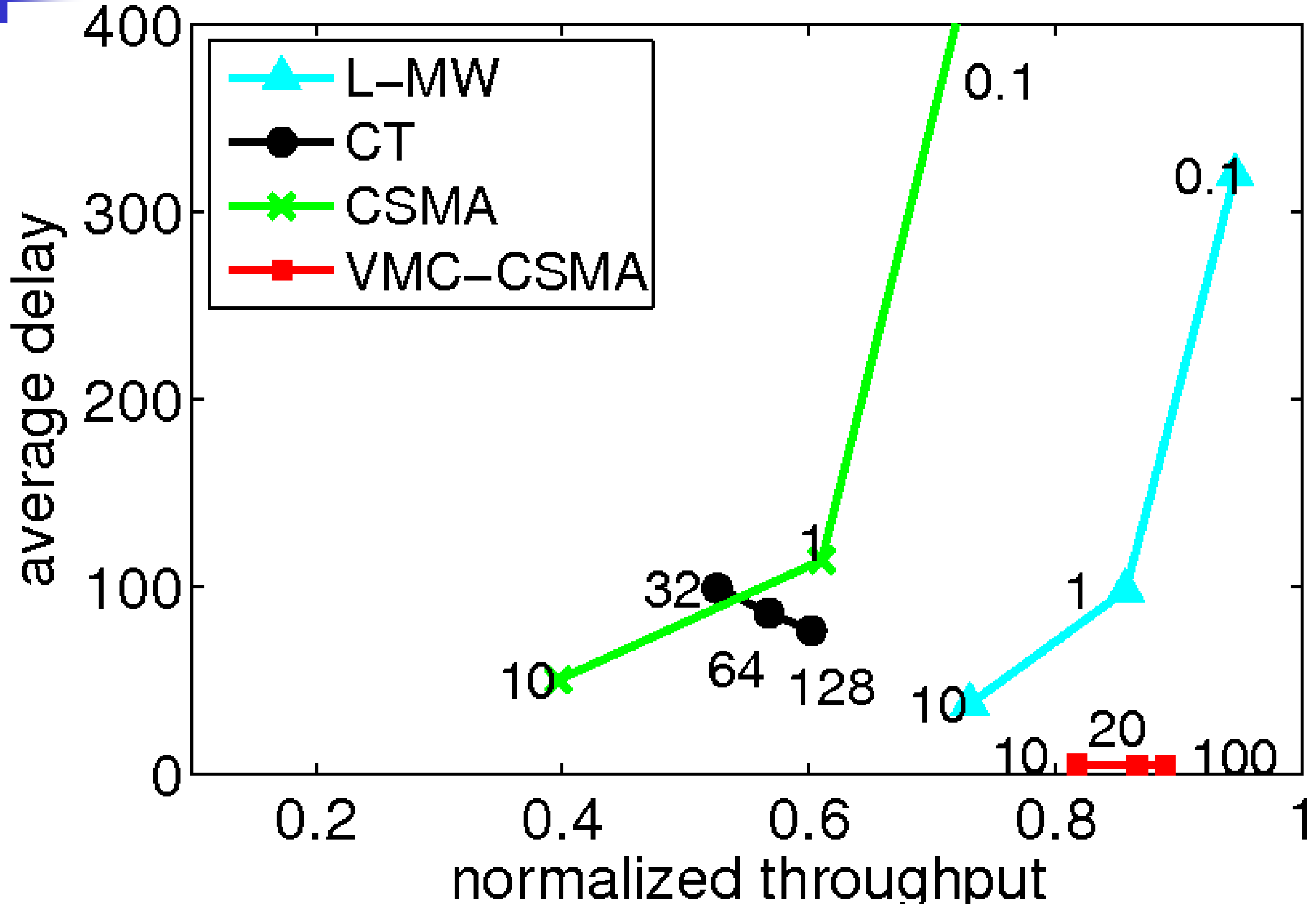
Simulation Results: 8x8 torus

	Throughput	Packet delay	HOL waiting time
VMC-CSMA	0.479	2.09	2.10
CSMA	0.427	**159**	**372.8**

Simulation Results: 8x8 torus



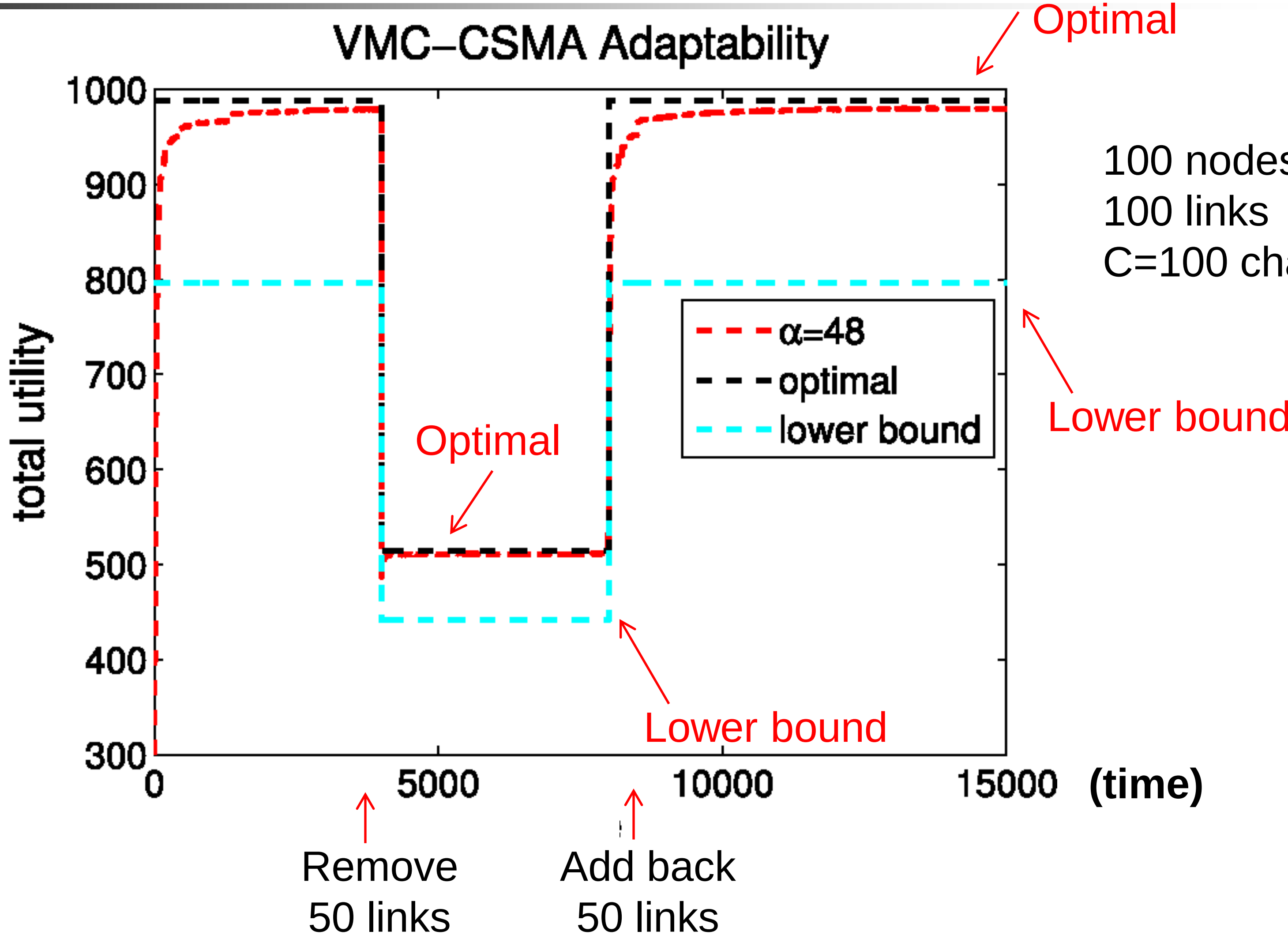
Simulation Results: Random Topology



$$\min_l \frac{r_l}{R_l^*}$$

100 nodes
100 links
C=100 channels

Simulation Results: Adaptivity



100 nodes
100 links
C=100 channels

Outline

- System Model and Related Work
- Virtual-Multi-Channel CSMA
- Capacity, Delay and Complexity
- Simulation Results
- *Conclusion and Discussions*

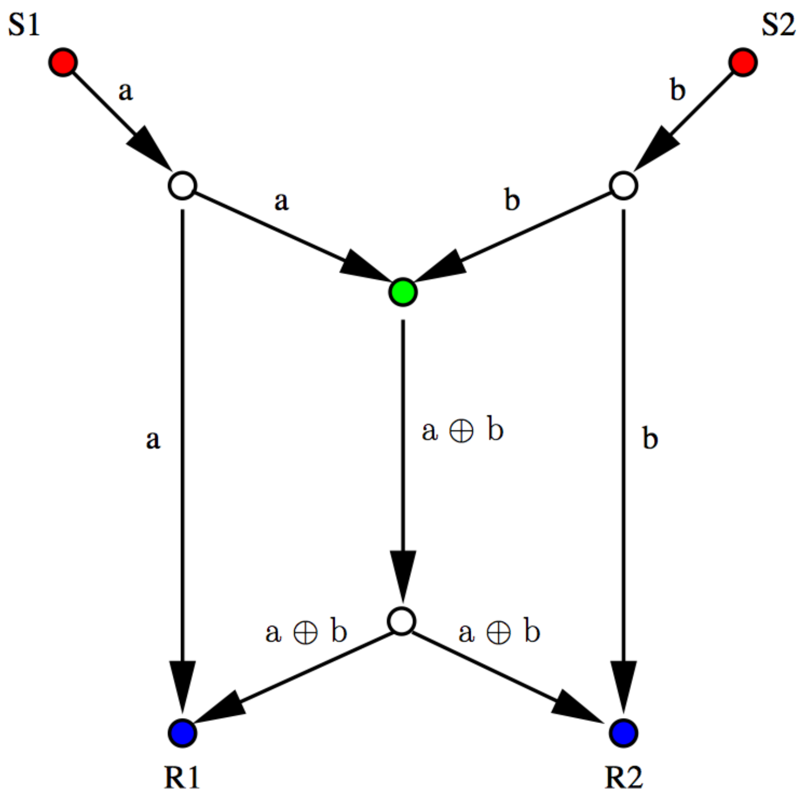
Conclusion

- We have develop a new framework for designing ***low-complexity and distributed*** wireless control algorithms to achieve both ***high capacity*** and ***low delay***

- By exploiting multiple physical- or virtual- channels, the proposed VMC-CSMA algorithm can provably achieve arbitrarily close to the ***optimal system utility*** with ***complexity*** that grows ***logarithmically*** with the network size

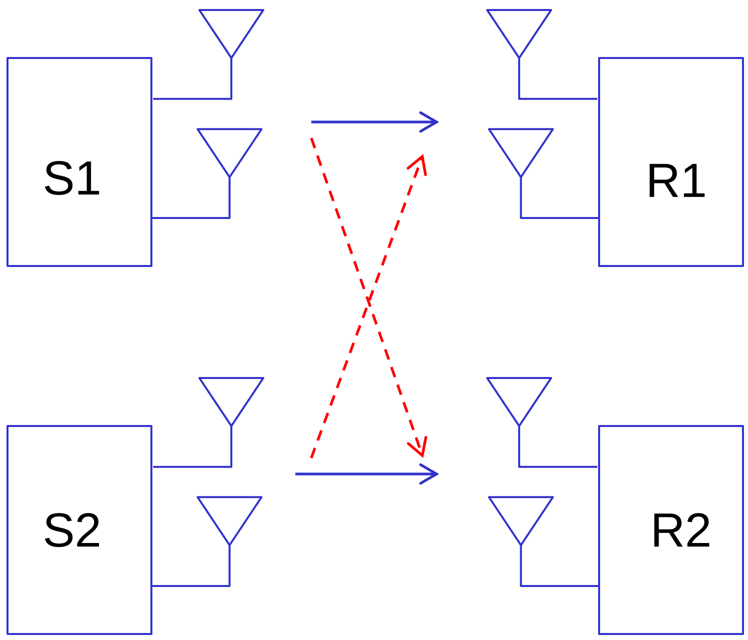
- Both the ***packet delay*** and ***HOL (head-of-line) waiting time*** can be tightly bounded.

Future Work (Advanced Coding and Transmission Mechanisms): Network Coding



- Coded transmissions can further improve capacity
- For wireless systems, we must consider link scheduling and network coding jointly
- Delay will be further increased due to the decoding requirement

Future Work (Advanced Coding and Transmission Mechanisms): MIMO



MIMO/Beam-forming

- Using MIMO, one can further increase the number of concurrent transmissions
- How to distributively assign the various nulling/transmission patterns is again a difficult scheduling problem
- How to account for power control and adaptive coding/modulation?

Future Work

- Incorporate other network control and performance objectives
  - Multi-hop routing
  - Energy efficiency

- Using randomized algorithms for the entire optimization problem may be too slow.

- Instead, we will seek ***new decomposition approach*** that can exploit the structure of the problem to expedite convergence

Thank you!

- Po-Kai Huang and Xiaojun Lin, "Improving the Delay Performance of CSMA Algorithms: A Virtual Multi-Channel Approach," *Technical Report, Purdue University, 2012*

https://engineering.purdue.edu/~linx/papers.html