Decentralized learning in control and optimization for networks and dynamic games

Part II: distributed optimization

Alexandre Proutiere

KTH

Outline

- Internet congestion control
  - Distributed optimization with separable objective function
- Two miracles in resource allocation in wireless networks
  - Distributed optimization with un-separable objective function, and without message passing
  - Power control
  - Carrier Sensing Multiple Access
- Parallel computations
  - Joint consensus and gradient descent methods
  - Just gradient descent
- Colorings
  - Combinatorial optimization: a sampling approach
- Distributed gradient free optimization

Problem classes

- Separable utilities, coupling constraints

$$\text{minimize } \textstyle\sum_{i=1}^{n} f_i(x_i)$$
$$\text{subject to } x \in \Omega$$

Examples: Internet congestion control, channel allocation in wireless networks

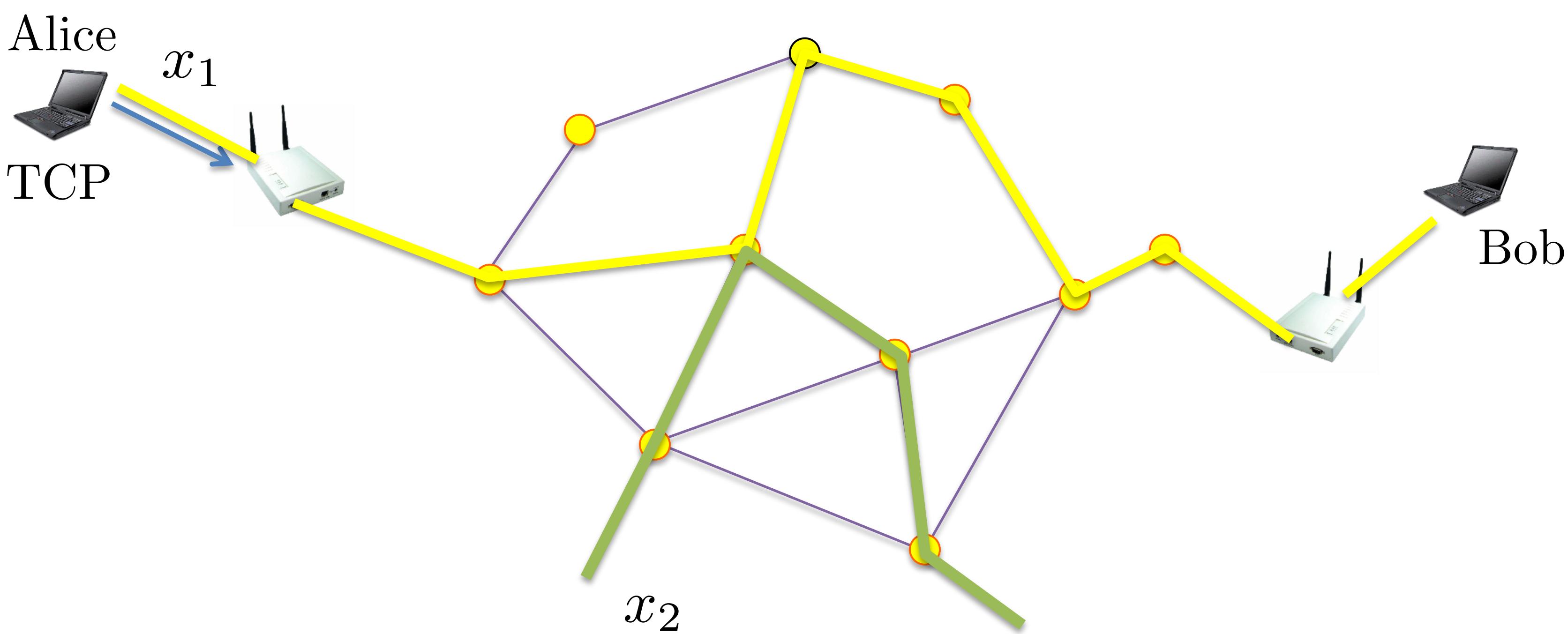
- Non-separable utilities

$$\text{minimize } \textstyle\sum_{i=1}^{n} f_i(x)$$
$$\text{subject to } x \in \Omega$$

Examples: power control and scheduling in wireless networks

Internet congestion control

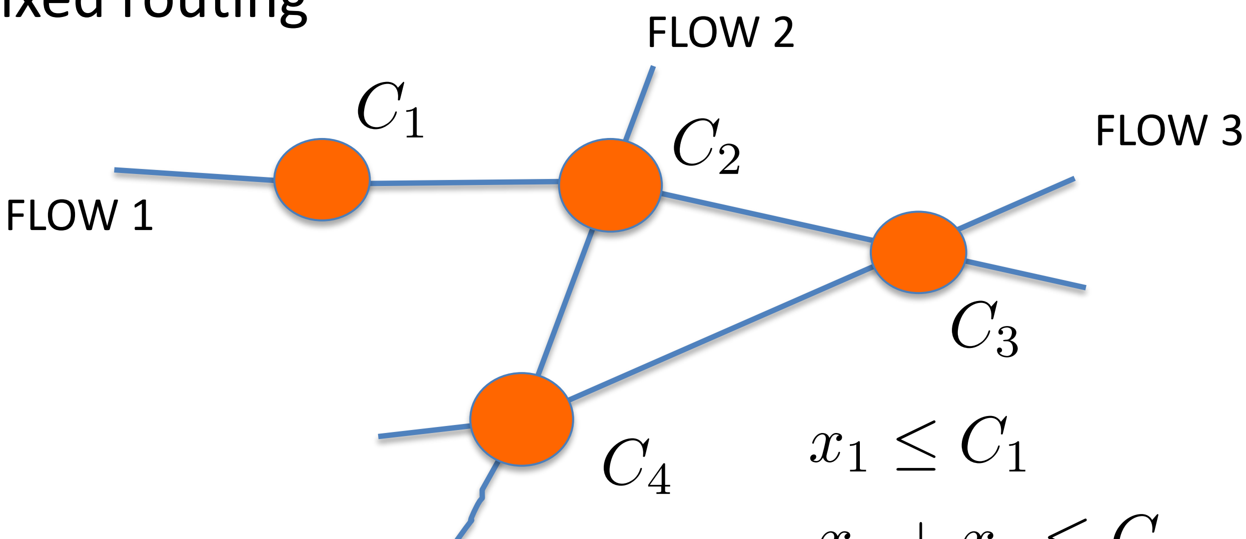
Internet congestion control



Objective of TCP: adapt the rates of sources to fairly and efficiently share network resources

A simple model

- Resources: a set of *L* links shared by a fixed population of *n* connections or data flows
- Fixed routing



$$x_1 \leq C_1$$

$$x_1 + x_2 \leq C_2$$

$$x_1 + x_3 \leq C_3$$

$$x_2 + x_3 \leq C_4$$

Network Utility Maximization

- The goal is to design distributed protocols converging to the solution of:

$$\begin{aligned}&\text{maximize } \textstyle\sum_{i=1}^{n} U_i(x_i)\\&\text{subject to } Rx \le C\end{aligned}$$

Previous example:

$$C = \begin{pmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \end{pmatrix} \qquad R = \begin{array}{c} \text{FLOWS} \\ \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \end{array} \text{LINKS}$$

Network Utility Maximization

- Utility functions:
  - Proportional fairness: $U_i(\cdot) = \log(\cdot)$
  - α-fairness: $U_i(\cdot) = (\cdot)^{(1-\alpha)}/(1-\alpha)$
  - Max-min fairness: $\alpha = \infty$

Why NUM?

- Seems reasonable with a fixed number of flows (good trade-of between efficiency and fairness)
- Distributed implementation
- Optimal accounting for dynamics in the population of flows
  - On route *i*, Poisson flow arrivals with intensity λ_i
  - Flow sizes: exponential distribution with mean σ_i
  - Load on route *i*: $\rho_i = \lambda_i \times \sigma_i$
  - Separation of time-scales assumption: the congestion control protocol is much faster than the flow population dynamics. When the flow population is $n = (n_i, i \in \mathcal{R})$, the rates of flows on the various routes solve

$$\begin{array}{l} \text{maximize } \sum_{i \in \mathcal{R}} n_i U_i(x_i) \\ \text{subject to } Rx \leq C \end{array}$$

Optimality of NUM-based protocols

- The Markov process capturing the numbers of active flows on the various routes is always stable (whenever possible)

Theorem* Let *N(t)* be the numbers of active flows at time *t* on the various routes. Then, under α-fair allocation, *N(t)* is ergodic if and only if:

$$R\rho < C$$

* Impact of fairness on Internet performance, **Bonald-Massoulie**, ACM Sigmetrics, 2001.

Decomposition

- Lagrangean:

$$L(x,\mu) = \sum_{i=1}^{n} (U_i(x_i) - x_i \sum_{l:R_{li}=1} \mu_l) + \sum_l \mu_l C_l$$

- Dual function:

$$q(\mu) = \sum_{i=1}^{n} \underbrace{\max_{x_i} (U_i(x_i) - x_i \sum_{l:R_{li}=1} \mu_l)}_{\text{Source sub-problems}} + \sum_l \mu_l C_l$$

Dual decomposition

- Link price update: for each link *l*

$$\mu_l(k+1) = \left[\mu_l(k) + \beta\Big(\sum_{i:R_{li}=1} x_i(k) - C_l\Big)\right]^+$$

- Source rate update:

$$x_i(k+1) = \arg\max_{x_i}\Big(U_i(x_i) - x_i \sum_{l:R_{li}=1} \mu_l\Big)$$

Convergence of dual GD algorithm

- The gradient of the dual function is lipschitz
  - Assume that $-U_i''(x_i) \geq 1/g > 0$
  - Let *L* and *S* be the length of the longest route and maximum number of sources using a given link, respectively

Lemma* We have:

$$\|\nabla q(\mu) - \nabla q(\mu')\|_2 \leq gLS\|\mu - \mu'\|_2$$

- … which ensures convergence of the algorithm

* Optimization flow control-I: Basic algorithm and convergence, **Low-Lapsley**, ACM/IEEE trans. on Networking, 1999.

Primal decomposition

- Source rate update:

$$x_i(k+1) = x_i(k) + \beta \left(U_i'(x_i(k)) - \sum_{l:R_{li}=1} \mu_l \right)$$

- Price update:

$$\mu_l(k+1) = p_l(\sum_{i:R_{li}=1} x_i(k+1))$$

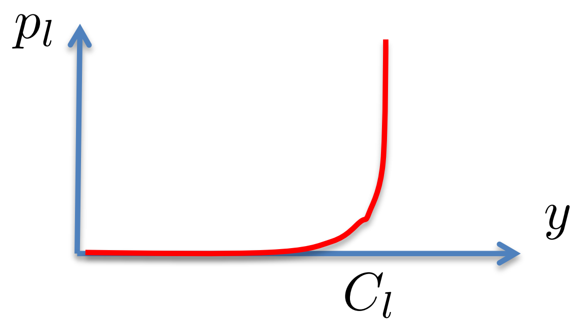
p_l : barrier function (to be defined later)

Convergence of the primal algorithm

Theorem* For appropriate choice of β, the primal algorithm converges to a solution of:

$$\max \sum_i U_i(x_i) - \sum_l \int_0^{\sum_{i:R_{li}=1} x_i} p_l(y)dy$$

- The barrier functions are increasing, and can be chosen so that we obtain a good approximation of the initial NUM problem



* Rate control for communication networks: shadow prices, proportional fairness and stability, **Kelly-Maulloo-Tan**, J. Oper. Res. Soc., 1998.

Does TCP scale?

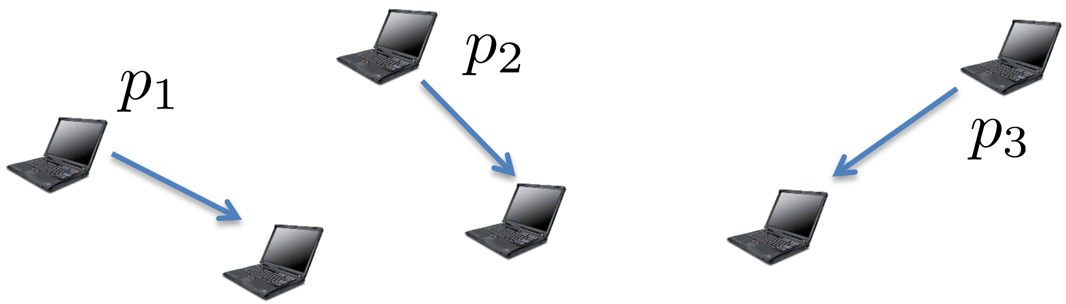
- It seems that it does not!
  - Dual algorithm: The lipschitz constant of the dual function depends on the number of flows. Decreasing convergence rate with the network size
  - Primal algorithm: for proportional fairness, the gradient seems unbounded when the number of flows increases
  - Other issues: see **Low-Paganini-Doyle**, IEEE Control Syst. Magazine, 2002
- The picture is more complicated
- Flow-level dynamics without separation of time-scales
  - Under the dual algorithm, the system is stable at flow-level whenever possible, see **Lin-Shroff-Srikant**, IEEE trans. IT, 2008

Outline

- Internet congestion control
  - Distributed optimization with separable objective function
- Two miracles in resource allocation in wireless networks
  - Distributed optimization with un-separable objective function, and without message passing
  - Power control
  - Carrier Sensing Multiple Access
- Parallel computations
  - Joint consensus and gradient descent methods
  - Just gradient descent
- Colorings
  - Combinatorial optimization: a sampling approach
- Distributed gradient free optimization

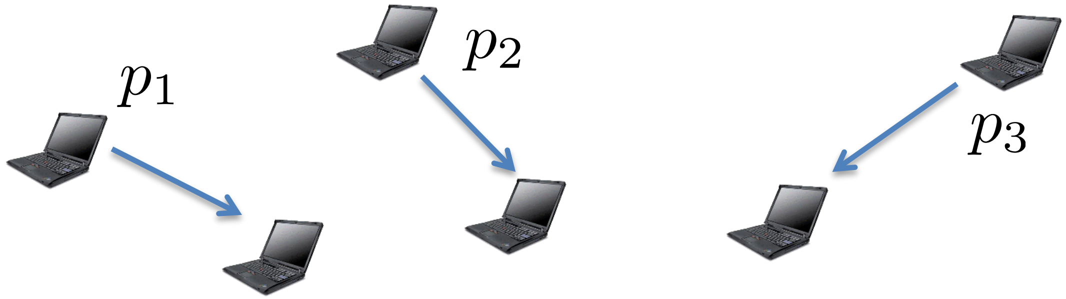
Power control

Objective



- Interfering links
- Target SINR for link i: γ_i
- Are these targets feasible?
- Is there a distributed algorithm answering the question?

Power control algorithm



- SINR at link-i receiver: $\dfrac{p_i G_{ii}}{N + \sum_{j \neq i} p_j G_{ji}}$
- Power updates*: $p_i[k+1] = p_i[k] \times \dfrac{\gamma_i}{SINR_i(k)}$

Distributed: each link measure its SINR only.

* Performance of optimum transmitter power control in cellular radio systems, **Zander**, IEEE trans. Vehicular Tech., 1992.

Fixed point iteration

- Power vector $p = (p_1, \ldots, p_n)$
- Iteration: $p[k+1] = I(p[k])$
- Interference function: $I_i(p) = \gamma_i \times \dfrac{\sum_{j \neq i} p_j G_{ji}}{G_{ii} + N}$
- A new analysis via contractive functions*

* Contractive interference functions and rates of convergence of DPC, **Feyzmahdavian-Johansson-Charalambous**, arxiv and ICC, 2012.

Contractive interference function

- Contractive interference function:

1. $I : \mathbb{R}_+^n \to \mathbb{R}_+^n$, $I(p) > 0$;
2. Monotonicity: $p \geq p' \implies I(p) \geq I(p')$;
3. Contractivity: $\exists c \in (0,1)$, $\exists v > 0$, $\forall \epsilon > 0$,

$$I(p + \epsilon v) \leq I(p) + c\epsilon v.$$

- Weighted maximum norm:

$$\|x\|_{\infty,v} = \max_i \frac{|x_i|}{v_i}$$

Convergence of DPC

Theorem A contractive interference function *I* has a unique fixed point $p^\star$. The sequence $p[k+1] = I(p[k])$ converges to the fixed point, and:

$$\|p[k] - p^\star\|_{\infty,v} \le c^k \|p[0] - p^\star\|_{\infty,v}$$

- Application: $I_i(p) = \gamma_i \times \frac{\sum_{j \neq i} p_j G_{ji}}{G_{ii} + N}$

or equivalently $I(p) = Mp + N.1$

$$M_{ij} = 1_{j \neq i} \frac{\gamma_i G_{ji}}{G_{ii}}$$

Convergence of DPC

Theorem If $\|M\|_{\infty,v} = \sup_{x \neq 0} \frac{\|Mx\|_{\infty,v}}{\|x\|_{\infty,v}} < 1$

then *I* is *c*-contractive with $c = \|M\|_{\infty,v}$

Theorem If $M \in \mathbb{R}_{+}^{n \times n}$
there exists a vector $v > 0$ such that $\|M\|_{\infty,v} < 1$
iff the spectral radius of *M* is strictly less than 1.

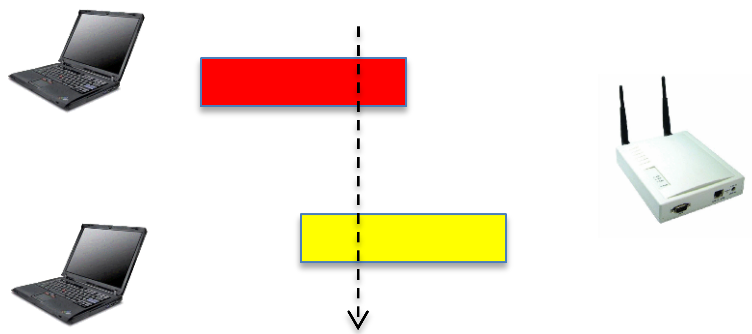
- As a consequence, if the spectral radius of *M* is < 1, then the DPC algorithm converges

Utility optimal CSMA

Distributed MAC protocols

- How to share an interfered channel in a distributed way?

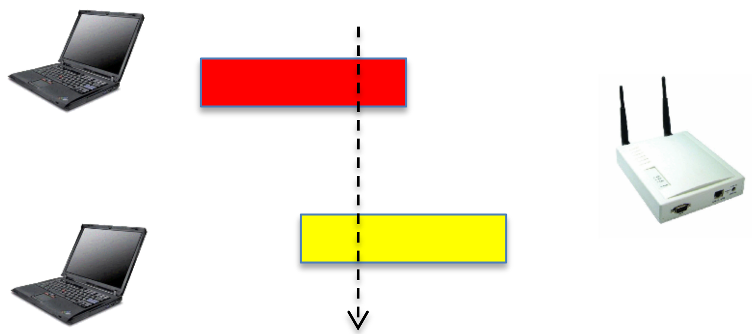
Distributed MAC protocols



Collision: both transmissions fail

- How to share an interfered channel in a distributed manner?

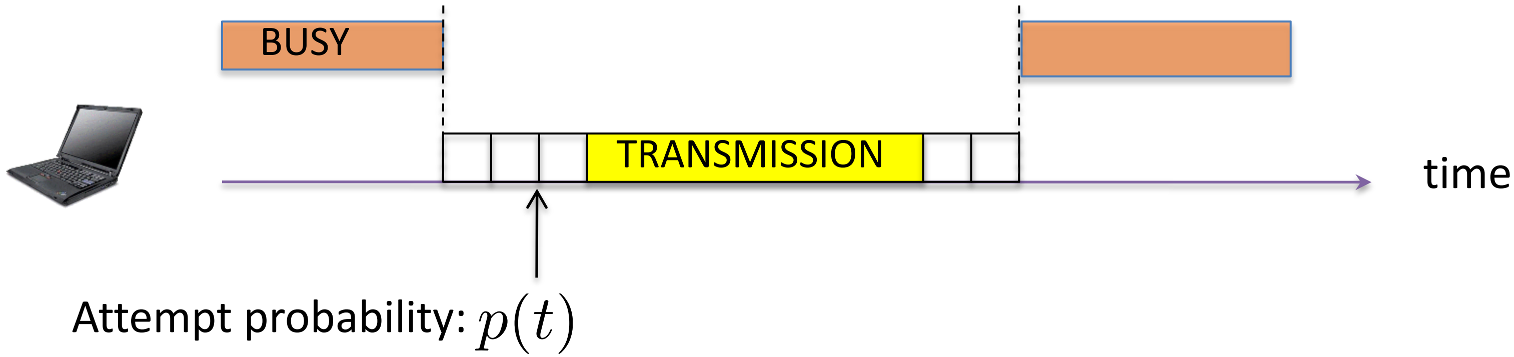
Distributed MAC protocols



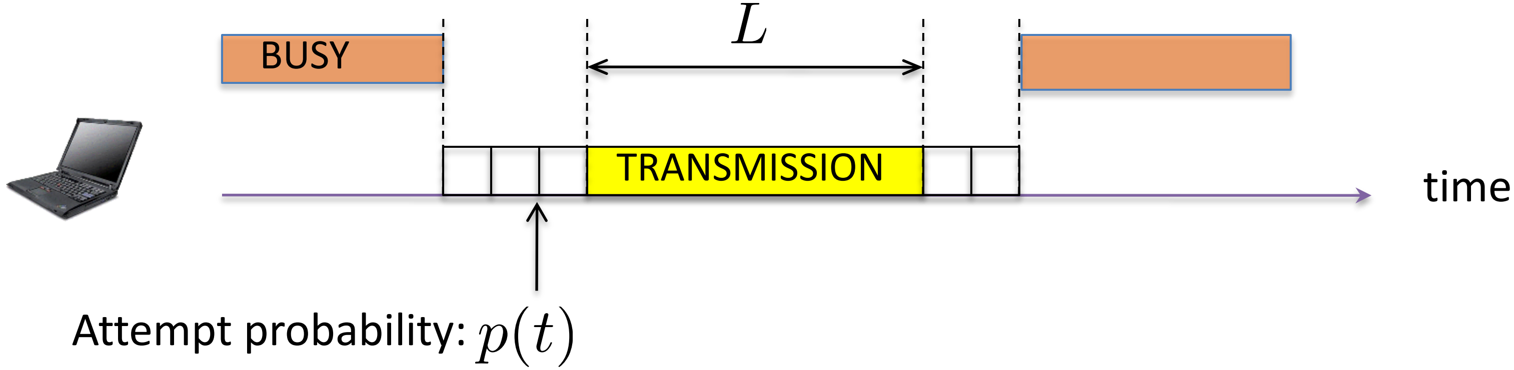
Collision: both transmissions fail

- How to share an interfered channel in a distributed manner?
  - Randomize! Wait a random time before transmitting
  - Be polite: listen before you talk, CSMA (Carrier Sense Multiple Access)

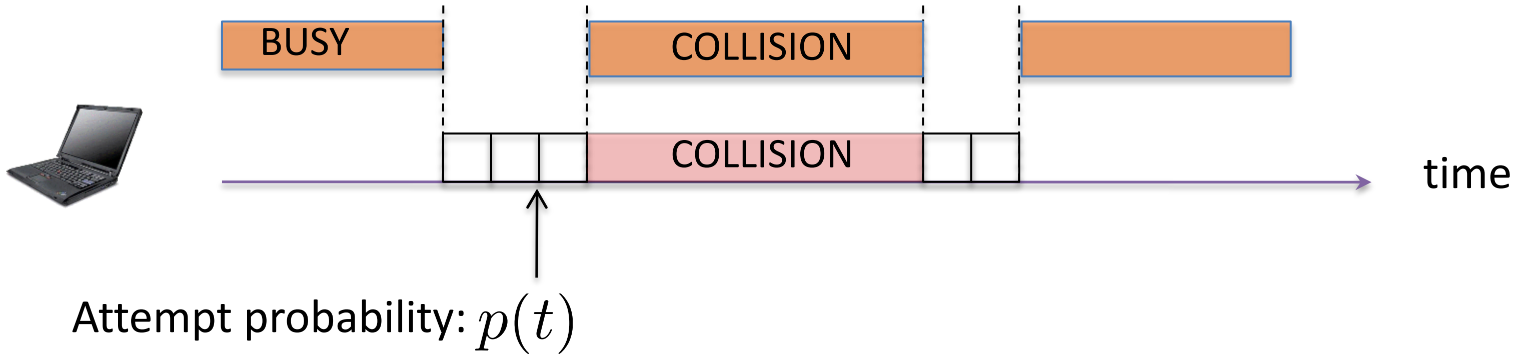
Distributed MAC protocols



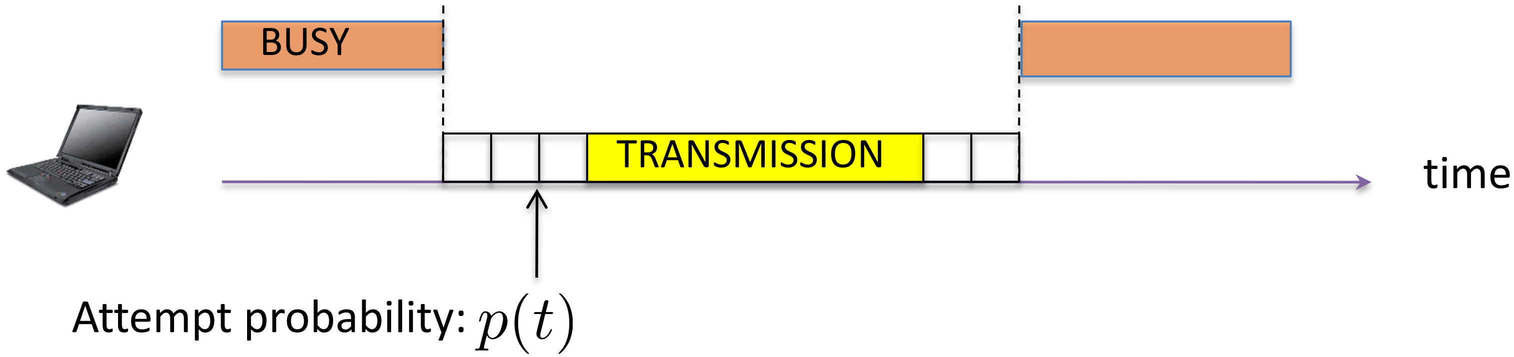
Distributed MAC protocols



Distributed MAC protocols



Distributed MAC protocols



Non-adaptive MAC (Aloha)

- Constant transmission probability

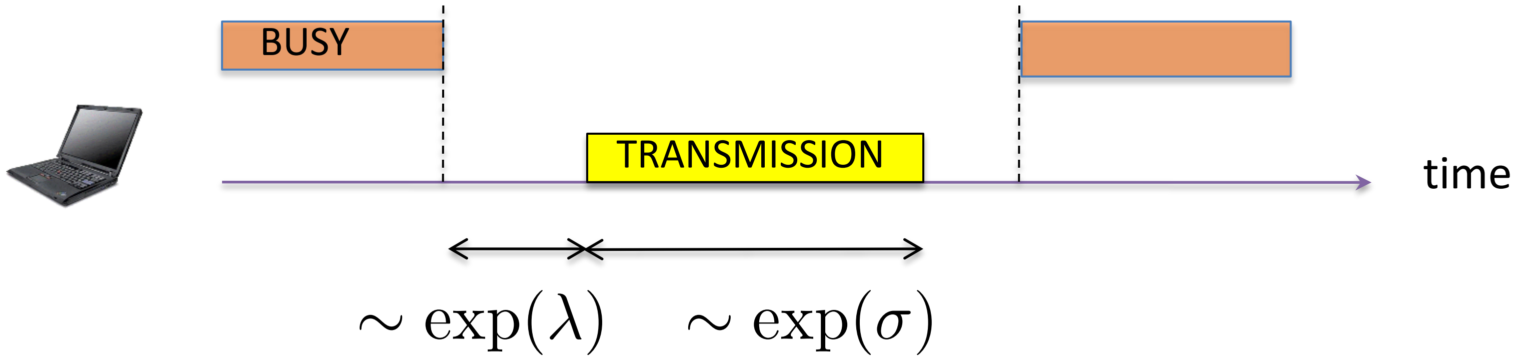
$$\forall t, \quad p(t) = p_0$$

Adaptive MAC

- Adaptive transmission probability
- e.g. exponential back-off

$$\begin{aligned} \text{success:} &\quad p(t) \rightarrow p_0 \\ \text{collision:} &\quad p(t) \rightarrow p(t)/2 \end{aligned}$$

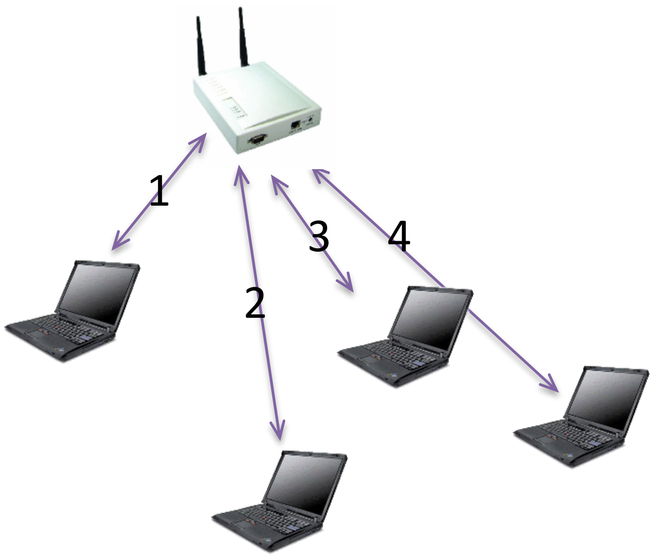
CSMA: a simple model

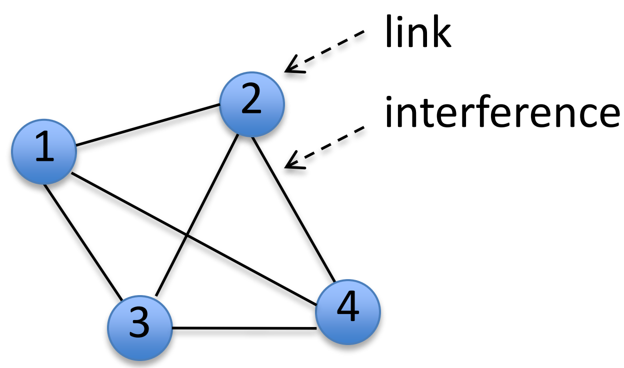


- Each transmitter runs a Poisson clock
- When the clock ticks, if the channel is idle, start transmitting
- No collision

Full interference

- A fully connected interference graph

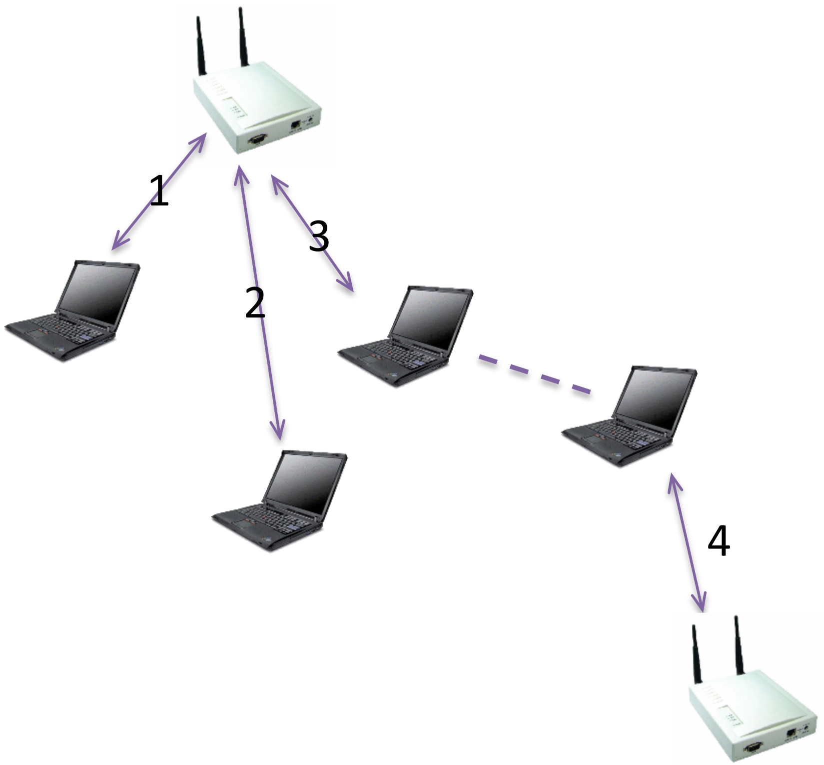


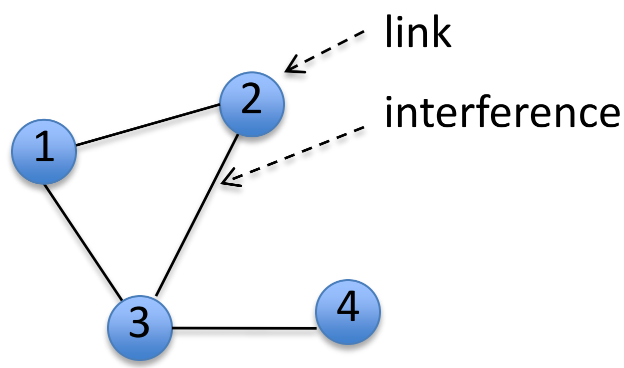


Interference graph

Partial interference

- A general interference graph

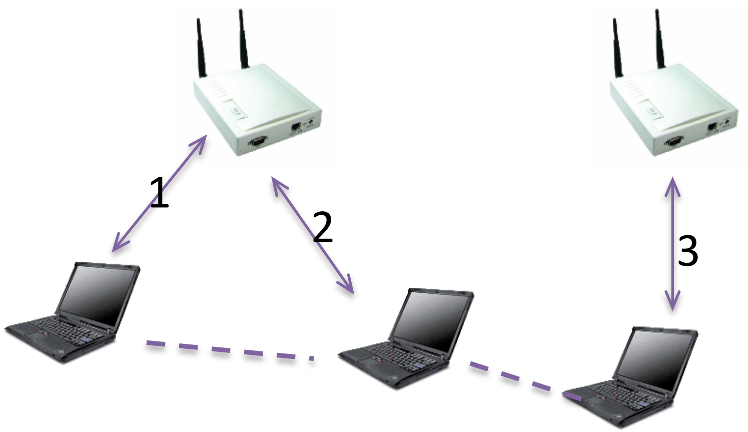


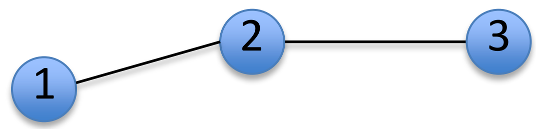


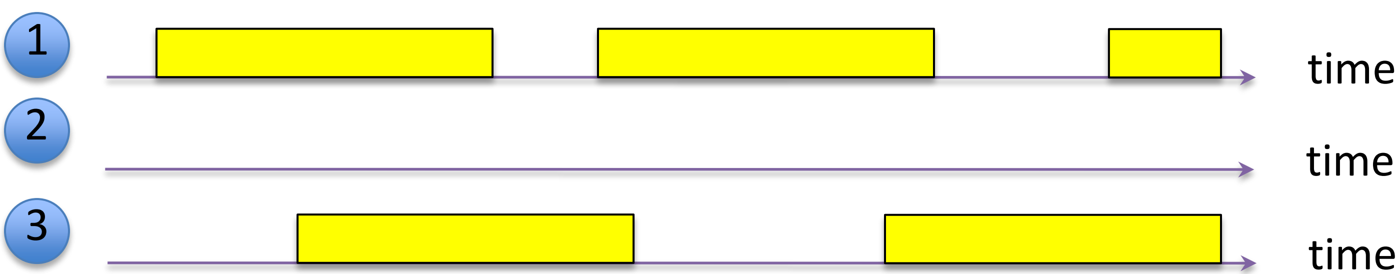
Interference graph

CSMA unfairness

- Partial interference







Objective

- Design fair and efficient CSMA protocols
- More precisely:

$$\max \sum_i U(\gamma_i)$$

γ_i : long term throughput on link i

U : strictly increasing concave utility function

Model

- Interference modeled as a graph:

$$A_{ij} = 1 \iff \text{links } i,\ j \text{ interfere}$$

- Schedule: $m \in \{0,1\}^I$
- Feasible schedule: $m \in \mathcal{M} \iff (m_i m_j = 1 \implies A_{ij} = 0)$
- Rate region:

$$\Gamma = \{\gamma : \exists \pi \in [0,1]^M, \sum_{m \in \mathcal{M}} \pi_m = 1, \forall i, \gamma_i \leq \sum_{m \in \mathcal{M}} \pi_m m_i\}$$

- Goal: solve the following convex program

$$\max \sum_i U(\gamma_i)$$
$$\text{s.t. } \gamma \in \Gamma$$

Rules

- No information are explicitly exchanged among transmitters
- Each transmitter just observes the realized throughput and can sense the channel

- Is it at all possible?
  - Yes, **Jiang-Walrand**, Allerton 2008
  - An other (better) solution, **Hegde-Proutiere**, CISS 2012

Performance of static CSMA

- Transmitter of link *i*
  - Poisson clock of intensity: λ_i
  - Mean channel holding time: σ_i
  - Intensity: ρ_i
- $m(t)$ is a reversible Markov process whose stationary distribution is:

$$\pi_m^\rho = \frac{\prod_i \rho_i^{m_i}}{\sum_{n \in \mathcal{M}} \prod_i \rho_i^{n_i}}$$

- Long term throughput on link *i*: $\gamma_i^\rho = \sum_{m \in \mathcal{M}} \pi_m m_i$

New optimization problem

$$\max V \sum_i U(\gamma_i) - \sum_{m \in \mathcal{M}} \pi_m (\log \pi_m - 1)$$

$$\text{s.t. } \forall i, \gamma_i \leq \sum_{m \in \mathcal{M}} \pi_m m_i$$

$$\sum_{m \in \mathcal{M}} \pi_m = 1$$

- Dual GD for this new problem

Lagrangean

$$L(\gamma, \Pi, q, \mu) = V \sum_i (U(\gamma_i) - q_i(\gamma_i - \sum_{m \in \mathcal{M}} \pi_m m_i)) - \sum_{m \in \mathcal{M}} \pi_m (\log \pi_m - 1) - \mu(\sum_{m \in \mathcal{M}} \pi_m - 1)$$

- Primal solutions:

$$VU'(\gamma_i) = q_i, \quad \forall i$$

$$\log \pi_m = \sum_{i: m_i = 1} q_i - \mu, \quad \forall m \in \mathcal{M}$$

Lagrangean

$$L(\gamma, \Pi, q, \mu) = V \sum_i (U(\gamma_i) - q_i(\gamma_i - \sum_{m \in \mathcal{M}} \pi_m m_i))$$
$$- \sum_{m \in \mathcal{M}} \pi_m (\log \pi_m - 1) - \mu(\sum_{m \in \mathcal{M}} \pi_m - 1)$$

- Primal solutions:

$$V U'(\gamma_i) = q_i, \quad \forall i$$

$$\pi_m = \exp(\sum_{i : m_i = 1} q_i - \mu), \quad \forall m \in \mathcal{M}$$

Lagrangean

$$L(\gamma, \Pi, q, \mu) = V \sum_i (U(\gamma_i) - q_i(\gamma_i - \sum_{m \in \mathcal{M}} \pi_m m_i))$$
$$- \sum_{m \in \mathcal{M}} \pi_m (\log \pi_m - 1) - \mu(\sum_{m \in \mathcal{M}} \pi_m - 1)$$

- Primal solutions:

$$\gamma_i = U'^{-1}(q_i/V), \quad \forall i$$

$$\pi_m = \frac{\prod_i e^{q_i^{m_i}}}{\sum_{n \in \mathcal{M}} \prod_i e^{q_i^{n_i}}}, \quad \forall m \in \mathcal{M}$$

Dual GD algorithm

$$q_i[k+1] = q_i[k] + \alpha(U'^{-1}(q_i/V) - \sum_{m \in \mathcal{M}} \pi_m m_i)$$

- Implementation: via CSMA!
- The transmitter of link i chooses its CSMA parameters such that:

$$\rho_i[k] = \exp(q_i[k])$$

Dual GD algorithm

$$q_i[k+1] = q_i[k] + \alpha(U'^{-1}(q_i/V) - \sum_{m \in \mathcal{M}} \pi_m m_i)$$

- Implementation: via CSMA!
- The transmitter of link i chooses its CSMA parameters such that:

$$\rho_i[k] = \exp(q_i[k])$$

- Issue: the long-term throughput on link *i* cannot be observed

Practical implementation

- Time divided into frames
- During frame *k*, the transmitter of link *i* runs CSMA with parameter $\rho_i[k]$, and observes the throughput $S_i[k]$ obtained in this frame
- At the end of frame *k*:

$$q_i[k+1] = q_i[k] + \alpha(U'^{-1}(q_i[k]/V) - S_i[k])$$

$$\rho_i[k+1] = \exp(q_i[k+1])$$

- Note that if the frame is very long:

$$S_i[k] \approx \sum_m \pi_m m_i, \quad \pi_m = \frac{\prod_i e^{q_i[k]^{m_i}}}{\sum_n \prod_i e^{q_i[k]^{n_i}}}$$

Convergence

- Run the previous algorithm with the following step-sizes:

$$\sum_k \alpha[k] = \infty \qquad \sum_k \alpha[k]^2 < \infty$$

Theorem We have: $\lim_{k\to\infty} q[k] = q_\star, \quad \lim_{k\to\infty} \gamma[k] = \gamma_\star, \quad \text{a.s.}$

where $\gamma[k] = \dfrac{1}{k}\sum_{s=1}^{k} S[s]$, and

where $\gamma_\star$ and $q_\star$ solve the modified optimization problem.

Convergence

- Modified vs. initial optimization problems
- Let $\gamma^\star$ be the solution of

$$\max \sum_i U(\gamma_i)$$
$$\text{s.t. } \gamma \in \Gamma$$

- Then: $|\sum_i (U(\gamma_i^\star) - U(\gamma_{i,\star}))| \leq \frac{K}{V}$

An alternative approach

Objective:

$$\max W(\pi) = \sum_i U(\sum_{m \in \mathcal{M}} \pi_m m_i)$$

$$\text{s.t.} \quad \sum_{m \in \mathcal{M}} \pi_m = 1$$

Steepest Coordinate Ascent

$$\left[\frac{\partial W}{\partial \pi_m}\right]_\pi = \sum_{i:m_i=1} U'(\gamma_i(\pi))$$

$$\gamma_i(\pi) = \sum_{i:m_i=1} \pi_m$$

Steepest coordinate ascent algorithm: select schedule such that

$$m^\star \in \arg\max_m \sum_{i:m_i=1} U'(\gamma_i(\pi))$$

Simulated Steepest Ascent

Steepest ascent algorithm can be approximately implemented by sampling allocations according to the distribution:

$$\xi^{\lambda,\pi}(m) = \frac{1}{Z(\lambda,\pi)} \lambda^{\sum_{i:m_i=1} U'(\gamma_i(\pi))}$$

Decentralized implementation: agents observe their realized throughputs and then adapt their channel access rate accordingly.

Slotted-CSMA implementation

Time is slotted.

At the beginning of slot k:

1. Select a link uniformly at random, say link *i*
2. The transmitter of link i observes the throughput obtained so far: $\gamma_i[k-1]$
3. With probability $\dfrac{\lambda^{U'(\gamma_i[k-1])}}{\lambda^{U'(\gamma_i[k-1])}+1}$, it accesses the channel if idle

   Otherwise it releases the channel

Convergence

Note that if $\gamma_i[k-1]$ was fixed, equal to $\gamma_i(\pi)$, we would sample the schedule with distribution $\xi^{\lambda,\pi}$. This is not the case. Nevertheless we have:

Theorem We have: $\forall \epsilon > 0, \exists \lambda : \quad \lim_{k \to \infty} \gamma[k] = \gamma^{\lambda}, \quad \text{a.s.}$

where $|\sum_i (U(\gamma_i^{\lambda}) - U(\gamma_i^{\star}))| \leq \epsilon$

Extensions

Set of schedules Ω

Objective: maximize $W(p) = \sum_i U(\sum_{s\in\Omega} p_s \mu_{i,s_i})$

over p: $\sum_s p_s = 1$

Example: Multi-channel wireless networks

Ω: set of proper channel allocations

Channel 0: the link remains inactive

μ_{i,s_i}: rate at which link i transmits on channel s_i

Steepest Ascent

$$\frac{\partial W}{\partial p_s}(p) = \sum\nolimits_i \mu_{is_i} U'(\gamma_i(p))$$

$$\gamma_i(p) = \sum\nolimits_s p_s \mu_{is_i}$$

Steepest ascent algorithm selects the distribution concentrating at $s_\star$ with:

$$s_\star \in \arg\max_s \sum\nolimits_i \mu_{i,s_i} U'(\gamma_i(p))$$

Simulated Steepest Ascent

Steepest ascent algorithm can be approximately implemented by sampling allocations according to the distribution:

$$\xi^{\lambda,p}(s) = \frac{1}{Z(\lambda,p)} \times \lambda^{\sum_i \mu_{i,s_i} U'(\gamma_i(p))}$$

Decentralized implementation: agents observe their realize throughputs and then randomly select a channel.

Slotted-CSMA implementation

At the beginning of time period k,

1. A link is chosen uniformly at random for possible update, say link i;
2. Link i measures the average rate $\gamma_i(k-1)$ received so far;
3. It selects a channel from the set of possible channels $A_i(k-1)$ (no interference with neighbors) according to the distribution:

$$\alpha(c) \sim \lambda^{\mu_{i,c}\, U'(\gamma_i(k-1))}, \text{ for } c \in A_i(k-1)$$

Convergence

Note that if $\gamma_i(k-1)$ was fixed, equal to $\gamma_i(p)$, we would sample according to $\xi^{\lambda,p}$. This is not true, and we sample from a time-varying distribution.

Theorem For any ϵ, there exists λ (large enough) such that for any initial condition, the SSA algorithm converges in the following sense:

$$\lim_{k\to\infty} \gamma(k) = \gamma^{\lambda}, \qquad \text{almost surely}$$

$$\left| \sum_i [U(\gamma_i^{\lambda}) - U(\sum_s p_s^{\star} \mu_{i,s_i})] \right| < \epsilon.$$

Asynchronous CSMA implementation

- Static multi-channel CSMA: link-*i* transmitter accesses channel *c* at rate ρ_{ic}
  Stationary distribution: $\zeta^{\rho}(s) = \frac{1_{\{s \in \Omega\}}}{Z(\rho)} \prod_i \rho_{i s_i}$
- CSMA samples from distribution $\xi^{\lambda,p}(s)$ if:

$$\rho_{ic} = \lambda^{\mu_{ic} U'(\gamma_i(p))}$$

Asynchronous CSMA implementation

At the beginning of time period k,

Each link sets its CSMA parameters on the various channel so that:

$$\rho_{ic}(k) = \lambda^{\mu_{i,c}\, U'(\gamma_i(k-1))}$$

Comparison with JW algorithm

- Updates in SSA algorithm:

$$\rho_{ic}(k) = \exp\left[\mu_{ic} T U' \left(U'^{-1}\left(\frac{\log \rho_{ic}(k-1)}{T\mu_{ic}}\right)\frac{k-1}{k} - \frac{S_i(k-1)}{k}\right)\right]$$

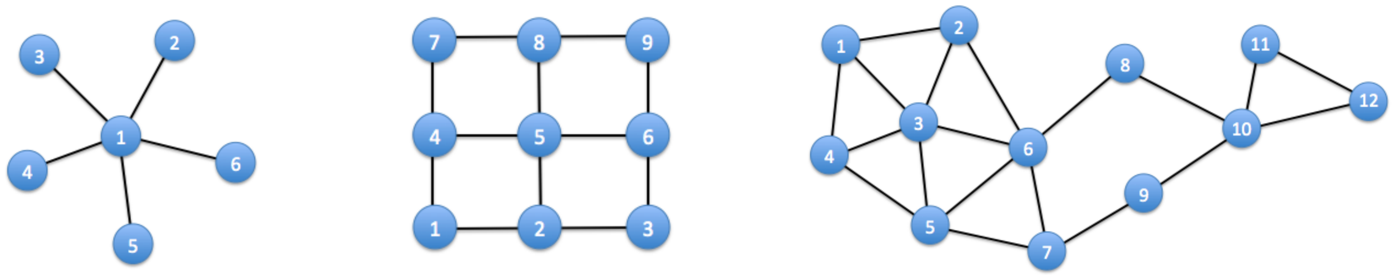
$$T = \log(\lambda)$$

- Updates in Jiang-Walrand algorithm:

$$\frac{\rho_{ic}(k)}{\rho_{ic}(k-1)} = \exp\left[\frac{\mu_{ic}}{k}\left(U'^{-1}\left(\frac{\log \rho_{ic}(k-1)}{T\mu_{ic}}\right) - S_i(k-1)\right)\right]$$

Numerical experiments

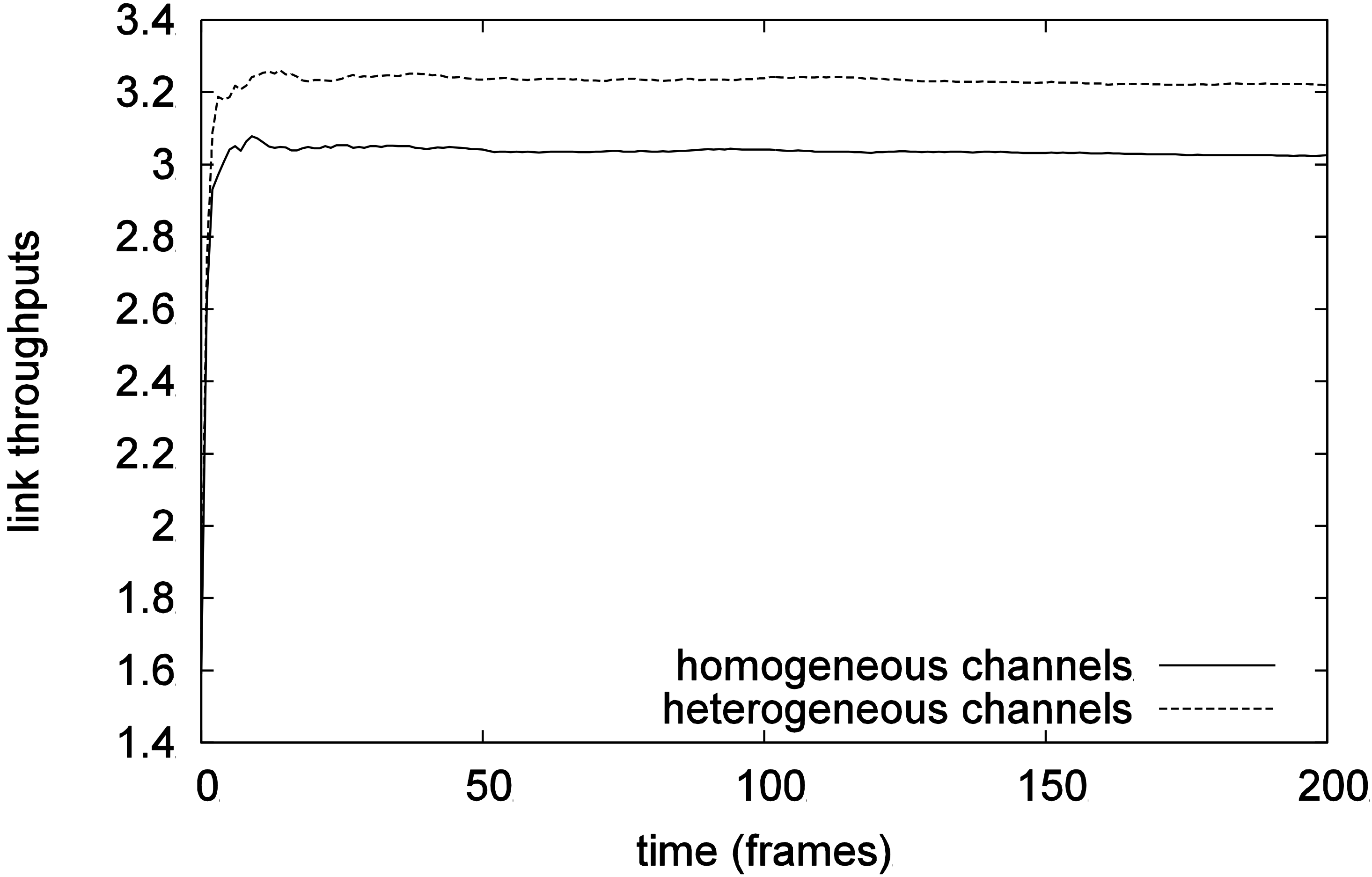
- Networks



- Homogeneous vs heterogeneous (unif. on [1,10]) channels
- Logarithmic utility
- Performance metrics
  - Cumulative average throughput per link
  - Convergence time: the time it takes so that all cumulative throughputs to be within 5% of their limits

    Notice: different than mixing time.

Random network – 2 channels



Complete interference graph

- JW algorithm does not follow steepest ascent direction
- Convergence time

Simulated Steepest Ascent

Number of links	**3**	**6**	**10**
Homogeneous channels	9.2	23.5	48.1
Heterogeneous channels	25.0	67.5	84.6

Jiang-Walrand

Number of links	**3**	**6**	**10**
Homogeneous channels	75.2	120.4	213.3
Heterogeneous channels	114.7	209.1	439.4

An element of the proof

In the convergence theorem of adaptive CSMA, everything works as if there was a separation of time-scales (updates of CSMA parameters – CSMA dynamics). Why?

- Stochastic approximation
- Stochastic approximation under Markovian noise

See **V. Borkar**, Stochastic Approximation: A dynamical system view point, 2008

Stochastic Approximation

- Algorithm: $x_{n+1} = x_n + a_n \times (h(x_n) + \xi_{n+1}), \quad \forall n.$

- Assumptions: $E[\xi_{n+1} | \mathcal{F}_n] = 0, \quad a.s., \forall n$

  h *L*-Lipschitz

  $$\sum_n a_n = \infty, \quad \sum_n a_n^2 < \infty,$$

  $$E[\|\xi_{n+1}\|^2 | \mathcal{F}_n] \leq K(1 + \|x_n\|^2), \quad a.s., \forall n$$

  $$\sup_n \|x_n\| < \infty, a.s.$$

ODE method

- Time: $t(0) = 0, \quad t(n) = \sum_{k=0}^{n-1} a_k, \forall n \geq 1$

$$\lim_{n \to \infty} t(n) = \infty$$

- Continuous piece-wise linear interpolation: $\bar{x}(t)$

$$\bar{x}(0) = 0$$

$$\bar{x}(t) = x_n + (x_{n+1} - x_n) \times \frac{t - t(n)}{t(n+1) - t(n)},$$

$$\forall t \in [t(n), t(n+1))$$

ODE method

- Approximate ODE: $$x^s(s) = \bar{x}(s)$$ $$\dot{x}^s(t) = h(x^s(t)), \quad \forall t \geq s$$

- The interpolated algorithm trajectory is well approximated by the ODE:

Theorem For any $T > 0$,

$$\lim_{s \to \infty} \sup_{t \in [s, s+T]} \|\bar{x}(t) - x^s(t)\| = 0, a.s.$$

ODE method

Corollary If *h* has a unique globally asymptotically stable point $x^\star$ then $\lim_{n\to\infty} x_n = x^\star$.

Stochastic Approximation with Markovian noise

- Example:
  - Control parameter: x_k
  - Observation: $Y_k = \int_k^{k+1} f(m(t))dt$
  - System (or "noise") dynamics: $m(t)$ non-homogenous Markov process whose transitions depend on the control parameter
  - If the control parameter is fixed to x, the process is irreducible, ergodic with stationary distribution η^x
  - Updates: $x_{k+1} = x_k + \alpha_k h(x_k, Y_k)$

Averaging principle

- Decoupling time-scales: the systems dynamics are as if the Markov noise was averaged over its evolving stationary distribution.

$$x_{k+1} = x_k + \alpha \sum_y \eta^{x(k)}(y) h(x_k, y)$$

Theorem The dynamics of x_k converge weakly (u.o.c.) towards those of the solution of:

$$\dot{x} = \sum_y \eta^x(y) h(x, y)$$