Joint Resource Allocation and Routing in Wireless Networks via Convex Approximation Techniques

Evangelia Matskani

Advisor: Prof. Nikos Sidiropoulos

Telecommunications Division
Department of Electronic & Computer Engineering
Technical University of Crete

June 15th, 2012
Outline

1. Convex Approximation Algorithms for Back-Pressure Power Control
   - Back-pressure power control problem (BPPC)
   - NP-hardness
   - Successive convex approximation strategy
   - Custom adaptive algorithms
   - Simulation results
   - Concluding remarks

2. Distributed Back-Pressure Power Control for Wireless Multi-hop Networks
   - Distributed implementation of core step of S.A. to BPPC
   - Distributed S.A. Algorithms for BPPC
   - Distributed WMMSE-based algorithms for BPPC
   - Adaptive distributed solutions
   - Simulation results
   - Concluding remarks
Shortest path vs. dynamic back-pressure

**SP**
- DP: BF, FW, ...
- Distributed ✓
- Must know arrival rate
- Quasi-static, very slow to adapt to
  - changing arrivals/load
  - availability/failure
  - fading/interference patterns
- Claim: Low delay (shortest path)
- ... but only at low system loads

**BP [Tassiulas ’92]**
- One-hop differential backlog
- Distributed ✓ Lightweight ✓
- Auto-adapts ✓
- Highly dynamic, agile ✓
- Claim: maximal stable throughput (all paths)
- ... but delay can be large - \( U(\text{load}) \), \( \emptyset \rightarrow \text{rand walk} \)
Favors links with low back-pressure (hence name)
Backtracking / looping possible!
Local communication, trivial computation
Back-pressure routing

- Multiple destinations, commodities?
  - multiple queues per node
  - (max diff backlog) winner-takes-all per link
- Wireline: local communication, trivial computation
- Wireless?
- Broadcast medium: interference
- Link rates depend on transmission scheduling, power of other links
- Through appropriate scheduling, power control ...
- obtain ‘favorable’ topology for throughput maximization
## Back-pressure power control

### SINR

\[ \gamma_\ell = \frac{G_{\ell \ell} p_\ell}{\sum_{k \in \mathcal{L}, k \neq \ell} G_{k \ell} p_k + V_\ell} \]

### Link capacity

\[ c_\ell = \log(1 + \gamma_\ell) \]

### BPPC

\[
\begin{align*}
\max & \quad \left\{ D_\ell(t) c_\ell \right\}_{\ell \in \mathcal{L}} \\
\text{s.t.} & \quad 0 \leq \sum_{\ell: \text{Tx}(\ell)=i} p_\ell \leq P_i, \forall i \in \mathcal{N} \\
& \quad 0 \leq p_\ell \leq P_\ell, \ell \in \mathcal{L}
\end{align*}
\]

### Diff backlog link \( \ell = (i \to j) \) \@ time \( t \)

\[
D_\ell(t) := \begin{cases} 
\max\{0, W_i(t) - W_j(t)\}, & j \neq N \\
W_i(t), & j = N.
\end{cases}
\]

\( \mathcal{F} \) flows: \( D_\ell(t) := \max_{f \in \mathcal{F}} D^{(f)}_\ell(t) \)
Back-pressure power control

**BPPC**

\[
\max_{\{p_\ell\}_{\ell \in \mathcal{L}}} \sum_{\ell \in \mathcal{L}} D_\ell(t) c_\ell \\
\text{s.t. } 0 \leq \sum_{\ell : \text{Tx}(\ell) = i} p_\ell \leq P_i, \forall i \in \mathcal{N} \\
0 \leq p_\ell \leq P_\ell, \ell \in \mathcal{L}
\]

**[Tassiulas et al, ’92 →]**

- Max stable throughput ✓
- Backbone behind modern NUM
- Core problem in wireless networking
- Complexity?
Reminiscent of ...

**DSL: sum-rate maximization**

- Listen-while-talk ✓
- Dedicated (Tx,Rx)
- Free choice of $G_{k,\ell}$’s
- NP-hard [Luo, Zhang]

**BPPC**

**Multi-hop network**

- No listen-while-talk X
- Shared Tx, Rx ⇒
- Restricted $G_{k,\ell}$’s, e.g.: $k, \ell$
  depart from $i \rightarrow G_{k,\ell} = G_{\ell,\ell}$,
  $G_{\ell,k} = G_{k,k}$
- NP-hard?
Peel off

- Backlog reduction $\rightarrow$ BPPC contains DSL $\rightarrow$ also NP-hard
DSL $\rightarrow$ Multi-hop network optimization

- Can reuse tools from DSL
- In particular, lower approximation algorithms:
  - High SINR $\rightarrow (\gamma_\ell \gg 1) \rightarrow \log(\gamma_\ell) \approx \log(1 + \gamma_\ell) \rightarrow$ Geometric Programming
  - Successive approximation from below: SCALE [Papandriopoulos and Evans, 2006]
- Uses

$$\alpha \log(z) + \beta \leq \log(1 + z) \quad \text{for} \quad \begin{cases} 
\alpha = \frac{z_o}{1+z_o} \\
\beta = \log(1 + z_o) - \frac{z_o}{1+z_o} \log(z_o)
\end{cases}$$

tight at $z_o$; $\rightarrow \log(z) \leq \log(1 + z)$ as $z_o \rightarrow \infty$

- Start from high SINR ($\alpha = 1$, $\beta = 0$), tighten bound at interim solution
- **Successive Convex Approximation Approach:**
  - $p(t) \rightarrow$ new $\alpha_\ell(t)$, $\beta_\ell(t)$ at $z_o = \gamma_\ell(p(t))$, $\forall \ell \in \mathcal{L} \rightarrow$ new $p(t)$, $\forall t$
- Majorization
Key difference with DSL

- BPPC problem must be solved repeatedly for every slot
- Batch algorithms: prohibitive complexity
- Need adaptive, lightweight solutions (to the extent possible)
- Built custom interior point algorithms
- Normally, one would init using solution of previous slot; take refinement step
- Doesn’t work ...
- Why?
Proper warm-start

- No listen-while-talk, shared Tx/Rx
- Push-pull ‘wave’ propagation
- Solution from previous slot very different from one for present slot
- Even going back a few slots
- Fixed/slowly varying ph. l. propagation conditions
- Deterministic fixed-rate (random but bounded) arrivals
- Quasi-periodic behavior emerges (stable setups) - return to one already visited state
- Idea: hold record of solutions for $W$ previous slots. $W >$ upper bound on period
- $W$ evaluations of present objective function (cheap!)
- Pick the best to warm-start present slot
- Needs few IP steps to converge
Adaptive Successive Convex Approximation

1. $\forall t \rightarrow \{D_\ell(t)\}_{\ell \in \mathcal{L}}$

2. For $t = 1 \rightarrow$ random $\tilde{p}_o(t)$ satisfying log-power constraints;
else for $t \in [2, W]$ set:

$$\tilde{p}_o(t) = \arg \max_{\tilde{p} \in \{\tilde{p}(1), \ldots, \tilde{p}(t-1)\}} f(\tilde{p})$$

$$f(\tilde{p}) := \sum_{\ell=1}^{L} D_\ell(t) \left( \alpha_\ell \left( \tilde{G}_{\ell\ell} + \tilde{p}_\ell - \log \left( \sum_{k=1}^{L} e^{\tilde{G}_{k\ell}} + \tilde{p}_k + e^{\tilde{V}_\ell} \right) \right) + \beta_\ell \right)$$

with $\alpha_\ell, \beta_\ell, \forall \ell \in \mathcal{L}$, updated $\forall \tilde{p} \in \{\tilde{p}(1), \ldots, \tilde{p}(t-1)\}$,
else ($t \geq W + 1$) set:

$$\tilde{p}_o(t) = \arg \max_{\tilde{p} \in \{\tilde{p}(t-W), \ldots, \tilde{p}(t-1)\}} f(\tilde{p})$$

3. Update $\alpha_\ell(t), \beta_\ell(t)$ for $\tilde{p}_o$, i.e., at $z_o = \gamma_\ell(\hat{p}_o(t)), \forall \ell \in \mathcal{L}$

4. repeat: Solve Core problem $\rightarrow$ new $\tilde{p}_o(t)$ $\rightarrow$ new $\alpha_\ell(t), \beta_\ell(t)$ for $\tilde{p}_o(t)$, $\forall \ell \in \mathcal{L}$ $\rightarrow$ new $\tilde{p}_o(t)$ ... until convergence
Prior art for BPPC

- Low-complexity algorithms for BPPC (multihop CDMA wireless networks) [Giannoulis et al 2006]
- high-SINR assumption
- distributed
- run in parallel with system-evolution (real time)

**Back Pressure Best Response**

\[ D_\ell(t) \rightarrow \pi_\ell(t) = \frac{D_\ell(t)}{I_\ell(p(t)) + V_\ell}, \quad \text{where } I_\ell(p(t)) := \frac{1}{G} \sum_{k=1}^{L} G_{k\ell} p_k(t) \]

\[ \pi_\ell(t) \rightarrow \forall k \neq \ell \rightarrow p_\ell(t + 1) = \min \left( \frac{D_\ell(t)}{\sum_{k=1}^{L} \frac{1}{G} G_{k\ell}}, \frac{P_{\ell}^{\max}}{G_{\ell\ell}} \right), \forall \ell \in \mathcal{L}, \forall t \]
Simulation setup

- $N = 6$ nodes, low-left = $s$, top-right = $d$, $L = 25$ links
- $G_{\ell,k} \sim 1/d^4$, $d$ : distance $\text{Tx}(\ell), \text{Rx}(k)$, $G = 128$,
- **no-listen-while-talk**: if $\text{Rx}(\ell) = \text{Tx}(k) \rightarrow G_{k,\ell} = 1/\epsilon$
- $V_\ell = 10^{-12}$, $P_\ell = 5$, $\forall \ell$
- Deterministic fixed-rate arrivals
**Batch high-SINR**

- Maximum stable arrival rate: 9.7 packets per slot (pps)
Adaptive high-SINR

- Maximum stable arrival rate: 9.7 pps
Adaptive high-SINR

- Unstable setup; arrival rate: 9.9 pps
Batch Successive Approximation

- Maximum stable rate: 10.4 pps
Adaptive Successive Approximation

- Maximum stable rate: 10.4 pps
Adaptive Successive Approximation

- Unstable setup; arrival rate: 10.8 pps

Simulation results

E. Matskani (Dept. ECE, TUC)  PhD Defense  June 15th, 2012  21 / 66
**BP Best Response**

- Maximum arrival rate: 5.7 pps

![Graphs showing simulation results for Back Pressure Best Response algorithm with various metrics over time slots.](image)
BP Best Response

- Unstable setup; arrival rate: 5.8 pps

Simulation results

Scenario 1: Back Pressure Best Response algorithm; arrival rate per slot = 5.8
Throughput & average complexity comparison

Table: Attainable stable arrival rates in packets per slot.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>B/A high-SINR</th>
<th>B/A successive</th>
<th>Best Resp</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scenario 1</td>
<td>9.7</td>
<td>10.4</td>
<td>5.7</td>
</tr>
<tr>
<td>Scenario 2</td>
<td>2.4</td>
<td>7.5</td>
<td>2.1</td>
</tr>
<tr>
<td>Scenario 3</td>
<td>12.6</td>
<td>15.7</td>
<td>4.4</td>
</tr>
</tbody>
</table>

BP Best Response
- BP Best Response is very cheap w.r.t. computation
- Indicatively, average run-time: $\approx 0.0085$ sec per slot
- More than 2 orders faster than adaptive version high-SINR / S.A.

S.A.-BPPC
- Worst case complexity: $O(L^{3.5})$ for batch / $O(L^3)$ for adaptive
- Avg. run-time for batch:
  - high-SINR $\approx 8$ sec / S.A. $\approx 20$ sec per slot
- Avg. run-time for adaptive:
  - high-SINR / S.A. $\approx 1$ to $1.5$ sec per slot
Concluding remarks

- S.A. approach (DSL literature) is meaningful;
- Manifold improvement in end-to-end throughput vs high-SINR-based alg.
- Much shorter backlogs / queueing delays, much faster transient response compared to BPBR
- Push-pull wave propagation, periodic behavior for stable setups ✓
Distributed BPPC

Core (maximization) step of S.A. approach to BPPC

\[
\max_{\{\tilde{p}_\ell \leq \tilde{P}_\ell\}, \ell \in \mathcal{L}} \sum_{\ell=1}^{L} D_\ell(t) c_\ell
\]

\[
c_\ell := \left( \alpha_\ell(t) \left( \tilde{G}_{\ell\ell} + \tilde{p}_\ell - \log \left( \sum_{k=1}^{L} e^{\tilde{G}_{\ell k} + \tilde{p}_k + \tilde{V}_\ell} \right) \right) + \beta_\ell(t) \right)
\]

- **Successive Convex Approximation Approach**: \(\forall t, p(t) \rightarrow \text{new } \alpha_\ell(t), \beta_\ell(t) \) at \(z_o = \gamma_\ell(p(t)), \forall \ell \in \mathcal{L} \rightarrow \text{new } p(t)\)
- Good news: manifold improvement in end-to-end throughput relative to the prior art
- Centralized implementation
- Practical power control ↔ *distributed* across network
Towards distributed implementation of the core step

- **Goal**: decomposition into $L$ parallel subproblems

**Key step**: Shift coupling of $\{\tilde{p}_\ell\}_{\ell \in \mathcal{L}}$ from objective to constraints

- auxiliary variables $\{\tilde{i}_{\ell k}\}_{k \neq \ell} :=$ interference to $\ell$ from $k \neq \ell$, $\forall \ell \in \mathcal{L}$
- add associated constraints: $\tilde{G}_{k\ell} + \tilde{p}_k = \tilde{i}_{\ell k}$, $\forall k \neq \ell$, $\forall \ell \in \mathcal{L}$

$$\begin{align*}
\min_{\tilde{p}, \{\tilde{i}_\ell\}_{\ell \in \mathcal{L}}} & \sum_{\ell=1}^L -D_\ell \alpha_\ell \left( \tilde{G}_{\ell\ell} + \tilde{p}_\ell \right) + D_\ell \alpha_\ell \log \left( \sum_{k=1}^L e^{\tilde{i}_{\ell k}} + e^{\tilde{V}_\ell} \right) - D_\ell \beta_\ell \\
\text{subject to} & \quad \tilde{G}_{k\ell} + \tilde{p}_k = \tilde{i}_{\ell k}, \quad \forall k \neq \ell, \quad \forall \ell \in \mathcal{L}, \\
\text{subject to} & \quad \tilde{p}_\ell \leq \tilde{P}_\ell, \quad \ell \in \mathcal{L}
\end{align*}$$

- $\tilde{i}_\ell := \{\tilde{i}_{\ell k}\}_{k \neq \ell}$, $\tilde{p} := \{\tilde{p}_\ell\}_{\ell \in \mathcal{L}}$
The augmented Lagrange function

- Augmented Lagrange function with penalty parameter $\rho$
- $\{\gamma_{\ell k}\}_{k \neq \ell} := \text{Lagrange multipliers } \forall \ell \in \mathcal{L} \mapsto \tilde{G}_{k \ell} + \tilde{p}_k = \tilde{i}_{\ell k}, \forall k \neq \ell$
- $\lambda_{\ell} := \text{Lagrange multiplier } \mapsto \tilde{p}_\ell \leq \tilde{P}_\ell, \forall \ell \in \mathcal{L}$

Decomposition of augmented Lagrangian

\[
L_{\rho} = \sum_{\ell=1}^{L} \left( -D_\ell \alpha_{\ell} (\tilde{G}_{\ell \ell} + \tilde{p}_\ell) + D_\ell \alpha_{\ell} \log \left( \sum_{\substack{k=1 \\text{to} \ L \\text{to} \ k \neq \ell}} \left. e^{\tilde{i}_{\ell k}} \right) + e^{\tilde{V}_\ell} \right) \right)
\]

\[
-D_\ell \beta_{\ell} + \lambda_{\ell} (\tilde{p}_\ell - \tilde{P}_\ell) + \sum_{\substack{k=1 \\text{to} \ L \\text{to} \ k \neq \ell}} \gamma_{\ell k} \tilde{G}_{\ell k} + \tilde{p}_\ell \left( \sum_{\substack{k=1 \\text{to} \ L \\text{to} \ k \neq \ell}} \gamma_{k \ell} \right)
\]

\[
- \sum_{\substack{k=1 \\text{to} \ L \\text{to} \ k \neq \ell}} \gamma_{\ell k} \tilde{i}_{\ell k} \right) + \frac{\rho}{2} \sum_{\ell=1}^{L} \sum_{\substack{k=1 \\text{to} \ L \\text{to} \ k \neq \ell}} \left( \tilde{G}_{k \ell} + \tilde{p}_k - \tilde{i}_{\ell k} \right)^2
\]
Utilizing Alternating Direction Method of Multipliers

Noteworthy points

- $\tilde{p}_\ell$ and $\{\tilde{i}_{\ell k}\}_{k \neq \ell}$ are local primal variables for link $\ell$
- $\lambda_\ell$ and $\{\gamma_{\ell k}\}_{k \neq \ell}$ are local dual variables for link $\ell$
- $L_\rho = \sum_{\ell=1}^{L} L_\ell \left( \tilde{p}_\ell, \tilde{i}_\ell, \lambda_\ell, \{\gamma_{\ell k}\}_{k \neq \ell} \right) + \text{regularization term}$
- Quadratic regularization term $\rightarrow$ strict convexity w.r.t. $\tilde{p}$

- **ADMoM** with dual ascent method $\rightarrow$ decentralized solution possible
- ADMoM $\rightarrow$ favorable convergence properties in our context, unlike dual decomposition method
- ADMoM’s steps $\mapsto$ update $\tilde{p}$, $\{\tilde{i}_\ell\}_{\ell \in \mathcal{L}}$, and $\{\gamma_{\ell k}\}_{\ell \in \mathcal{L}, k \neq \ell}$
- Projected gradient step $\mapsto$ update $\lambda$
- $L$ parallel subproblems: $\forall \ell \in \mathcal{L}$ the iterates boil down to ...
ADMoM and dual ascent method

\( \tilde{p}_\ell \)-optimization step, \( \forall \ell \in \mathcal{L} \) (\( s := \text{iteration index} \))

\[
\tilde{p}_\ell(s) := \arg \min_{\tilde{p}_\ell} -D_\ell \alpha_\ell \tilde{p}_\ell + \lambda_\ell (s - 1) \tilde{p}_\ell + \tilde{p}_\ell \left( \sum_{k=1}^{L} \gamma_{k\ell} (s - 1) \right)
\]

\[
+ \frac{\rho}{2} \sum_{\substack{k=1 \\ k \neq \ell}}^{L} (\tilde{G}_{\ell k} + \tilde{p}_\ell - \tilde{i}_{k\ell}(s - 1))^2
\]

- convex quadratic in \( \tilde{p}_\ell \) → closed form update
- Available control channels among nodes → information exchanges
- Local link gain information: Tx(\( \ell \)) knows \( G_{\ell k} \) to Rx(\( k \))

**Feedback requirements** from interfering links:
- dual variables \( \{\gamma_{k\ell}(s - 1)\}_{k \neq \ell} \)
- auxiliary variables \( \{\tilde{i}_{k\ell}(s - 1)\}_{k \neq \ell} \)
\[ \{ \tilde{i}_{\ell k} \}_{k \neq \ell} (s) := \arg \min_{\{ \tilde{i}_{\ell k} \}_{k \neq \ell}} D_\ell \alpha_{\ell} \log \left( \sum_{k=1, k \neq \ell}^{L} e^{\tilde{i}_{\ell k}} + e^{\tilde{V}_{\ell}} \right) - \sum_{k=1, k \neq \ell}^{L} \gamma_{\ell k}(s-1)\tilde{i}_{\ell k} \]

\[ + \frac{\rho}{2} \sum_{k=1, k \neq \ell}^{L} \left( (\tilde{G}_{k\ell} + \tilde{p}_k)(s) - \tilde{i}_{\ell k} \right)^2 \]

- solved e.g., via damped Newton’s method
- **Requires**: interference \((\tilde{G}_{k\ell} + \tilde{p}_k)(s)\) from \(\forall k \neq \ell\) to \(\ell\) at iteration \(s \to \) estimated by \(\ell\), or communicated to \(\ell\)
ADMoM and dual ascent method: optimization steps

**γ_{lk}- update step, ∀k \neq ℓ, ℓ ∈ ℋ**

\[ γ_{lk}(s) := γ_{lk}(s - 1) + ρ \left( (\tilde{G}_{kl} + \tilde{p}_k)(s) - \tilde{v}_{lk}(s) \right) \]

- **step-size:** strictly equal to ρ, and ρ > 0, according to ADMoM

**λ_{ℓ}- projected gradient step, ∀ℓ ∈ ℋ**

\[ λ_{ℓ}(s) := \left[ λ_{ℓ}(s - 1) + δ_s (\tilde{p}_ℓ(s) - P_ℓ) \right]_0^+ \]

- **step-size:** e.g., \( δ_s = δ_1/s, \ δ_1 > 0, \) or a sufficiently small constant \( δ > 0 \)
Distributed convex approximation of BPPC

Core step 1 algorithm (global constant $\rho$)

Given $D_\ell$, $\alpha_\ell$, $\beta_\ell$, $\forall \ell \in \mathcal{L}$, and $s :=$ iteration counter

- **Initialization:** For $s = 0$, set: $\rho > 0$, $\delta_1 > 0$, $\{\lambda_\ell(0)\}_{\ell=1}^L > 0$, $\{\gamma_{\ell k}(0)\}_{\ell \in \mathcal{L}, k \neq \ell} > 0$, and $\{\tilde{i}_{\ell k}(0)\}_{\ell \in \mathcal{L}, k \neq \ell}$ random

- $\forall \ell \in \mathcal{L}$: transmit initial $\gamma_{\ell k}(0)$ and $\tilde{i}_{\ell k}(0)$ to link $k$, $\forall k \neq \ell$

- **Repeat:** Set $s := s + 1$

  1. $\forall \ell \in \mathcal{L}$: $\tilde{p}_\ell$-optimization step to obtain $\tilde{p}_\ell(s)$
  2. $\forall \ell \in \mathcal{L}$: $\tilde{i}_\ell$-optimization step to obtain $\tilde{i}_\ell(s)$
  3. $\forall \ell \in \mathcal{L}$: $\gamma_{\ell k}$-update step $\forall k \neq \ell$ to obtain $\{\gamma_{\ell k}(s)\}_{k \neq \ell}$
  4. $\forall \ell \in \mathcal{L}$: $\lambda_\ell$-update step to obtain $\lambda_\ell(s)$
  5. $\forall \ell \in \mathcal{L}$: transmit $\gamma_{\ell k}(s)$ and $\tilde{i}_{\ell k}(s)$ $\rightarrow$ link $k$, $\forall k \neq \ell$

- **Until:** convergence (within $\epsilon$-accuracy); then $\tilde{p}_\ell^{opt} := \tilde{p}_\ell(s)$, $\forall \ell \in \mathcal{L}$
Distributed core step algorithm

- **Convergence** *should* be based on *local* computation and communication
- Each link may keep track of a local metric, e.g.,
  - successive differences of its local augmented Lagrange function
  - norm of its residual local equality constraint violation vector $r_{\ell}(s)$, with elements $r_{\ell k}(s) := \tilde{G}_{k \ell} + \tilde{p}_k(s) - \tilde{i}_{\ell k}(s)$, $\forall k \neq \ell$
- Local metric under $\epsilon \to$ local convergence
- **Termination** of the distributed protocol:
  - each link maintains *binary flag*: $1 \leftrightarrow$ convergence w.r.t. its local metric (within given $\epsilon$) is *achieved*
  - *distributed consensus-on-the-min* algorithm among links $\rightarrow$ iterates terminate once *all* links reach convergence
- Variations of ADMoM [Bertsekas ’96, Boyd 2011]
  - *adaptive local penalty parameters* $\rightarrow$ accelerate convergence (see thesis)
  - core step 2: less dependent on initial choice of penalty parameters
Distributed adaptive S.A. algorithm for BPPC (warm re-start)

- \( \forall t \rightarrow \{D_\ell(t)\}_{\ell \in \mathcal{L}} \quad W > \text{expected period} \)
- **Initialization**: For \( t = 1 \):
  \( \alpha_\ell(t) = 1, \beta_\ell(t) = 0, \forall \ell \in \mathcal{L} \lambda_{\ell,o} > 0, \{\gamma_{\ell,k,o}\}_{k \neq \ell} > 0, \tilde{i}_{\ell,o} \) random
  
  For \( t \in [2, W] \) set \( \forall \ell \in \mathcal{L} \):

  \[ \tau_o = \arg \min_{\tau \in \{t-1, \ldots, 1\}} |D_\ell(t) - D_\ell(\tau)| \]

  For \( t \geq W + 1 \) set \( \forall \ell \in \mathcal{L} \):

  \[ \tau_o = \arg \min_{\tau \in \{t-1, \ldots, t-W\}} |D_\ell(t) - D_\ell(\tau)| \]

  \( \alpha_\ell(t) := \alpha_\ell^{\text{opt}}(t - \tau_o), \quad \beta_\ell(t) := \beta_\ell^{\text{opt}}(t - \tau_o) \)

  \( \lambda_\ell := \lambda_\ell^{\text{opt}}(t - \tau_o), \quad \{\gamma_{\ell,k}(t)\}_{k \neq \ell} := \{\gamma_{\ell,k}^{\text{opt}}(t - \tau_o)\}_{k \neq \ell}, \quad \tilde{i}_\ell := \tilde{i}_\ell^{\text{opt}}(t - \tau_o) \)
Weighted sum-MSE Minimization for WSR Maximization

- Distributed Sum-Utility Maximization for MIMO IBC [Christensen et al 2008], [Shi et al, 2011]
- In [Shi et al, 2011]:
  - **Proof**: WSR Maximization is *equivalent* to properly weighted sum-MSE Minimization
  - **Proof**: Iterative WMMSE → local optimal (st. point) of WSRM
- Use iterative WMMSE to approx. solve BPPC at physical layer
- MIMO IBC → SISO IC (in our context):
  - $h_{k\ell} :=$ channel Tx($k$) → Rx($\ell$), $\sim \mathcal{CN}(0, 1)$, scaling: $|h_{k\ell}|^2 = G_{k\ell}$, $\forall \ell, k \in \mathcal{L}$
  - Alternatively: $h_{k\ell} \in \mathbb{R}$, equal to $\sqrt{G_{\ell k}}$, $\forall k, \ell \in \mathcal{L}$
Distributed BPPC for Wireless Multi-hop Networks

The Iteratively Weighted MMSE Approach to BPPC

Weighted MMSE approach to dif. backlog WSRM

Weighted sum-MSE Minimization

\[
\min_{\{w_\ell,u_\ell,v_\ell\}_{\ell \in \mathcal{L}}} \sum_{\ell=1}^{L} D_\ell(t) \left( w_\ell e_\ell - \log(w_\ell) \right)
\]

\text{s. t. } |v_\ell|^2 \leq P_\ell , \quad \ell \in \mathcal{L}

- \(w_\ell^{opt} = e_\ell^{-1}, \quad \forall \ell \in \mathcal{L}\)
- \(u_\ell^{opt} \equiv u_\ell^{mmse} = \frac{h_\ell v_\ell}{\sum_{k=1}^{L} |h_{k\ell}|^2 |v_k|^2 + V_\ell}, \quad \forall \ell \in \mathcal{L}\)

- \(e_\ell \) := m.sq. error, \(w_\ell > 0 \) := weight var, \(v_\ell/u_\ell \in \mathbb{C}^{1 \times 1} \) := gain Tx(\ell)/Rx(\ell)

Simplifying WMSE minimization yields BPPC at physical layer

\[
\max_{\{v_\ell\}_{\ell \in \mathcal{L}}} \sum_{\ell=1}^{L} D_\ell(t) \log \left( 1 + \frac{|h_{\ell\ell}|^2 |v_\ell|^2}{\sum_{k=1}^{L} |h_{k\ell}|^2 |v_k|^2 + V_\ell} \right)
\]

\text{s. t. } |v_\ell|^2 \leq P_\ell , \quad \ell \in \mathcal{L}

Upon change of variables:

- \(p_\ell = |v_\ell|^2, \quad \forall \ell \in \mathcal{L}\)
- \(G_{k\ell} = |h_{k\ell}|^2, \forall k, \ell \in \mathcal{L}\)
Iterative WMMSE algorithm for BPPC

1. Initialize: $v_\ell = \sqrt{P_\ell}$, $\forall \ell \in L$
2. Repeat
   3. $(w_\ell)' \leftarrow w_\ell$, $\forall \ell \in L$
   4. $u_\ell \leftarrow \frac{h_\ell v_\ell}{\sum_{k=1}^{L} G_{k\ell} v_k^2 + V_\ell}$, $\forall \ell \in L$
   5. $w_\ell \leftarrow (1 - u_\ell h_\ell v_\ell)^{-1}$, $\forall \ell \in L$
   6. $v_\ell \leftarrow \left[ \frac{D_\ell(t) w_\ell u_\ell h_\ell}{\sum_{k=1}^{L} D_k(t) w_k u_k^2 G_{k\ell}} \right]^{\sqrt{P_\ell}}$, $\forall \ell \in L$
8. Until $|\log(w_\ell) - \log ((w_\ell)')| \leq \epsilon$, $\forall \ell \in L$
9. Set $p_\ell(t) := v_\ell^2$, $\forall \ell \in L$

- $(u, v, w)$ — space
- Convex w.r.t. to each $u, v, w$ holding others fixed
- Block coordinate descent technique → suboptimal solution
- Feedback: $\{D_k u_k^2 w_k\}_{k \neq \ell} \rightarrow \ell$
- Consensus-on-the-min algorithm for termination of iterates
- One shot approximation to BPPC $\forall t$
- Low complexity
Distributed Adaptive WMMSE algorithm for BPPC

- Same stable setup considered; exploit expected periodicity due to push-pull nature of solution
- Same strategy for warm re-start with the S.A. algorithm
- speed up convergence of WMMSE

Distributed Adaptive WMMSE for BPPC (warm re-start)

- **Power - initialization:**
  
  For $t = 1$ set: $v_\ell = \sqrt{P_\ell}$, $\forall \ell \in \mathcal{L}$
  
  For $t \in [2, W]$ set:
  
  $$
  \tau_o = \arg \min_{\tau \in \{t-1, \ldots, 1\}} |D_\ell(t) - D_\ell(\tau)|, \quad v_\ell = \sqrt{p_\ell(t - \tau_o)}, \forall \ell \in \mathcal{L}
  $$

  For $t \geq W + 1$ set:
  
  $$
  \tau_o = \arg \min_{\tau \in \{t-1, \ldots, t-W\}} |D_\ell(t) - D_\ell(\tau)|, \quad v_\ell = \sqrt{p_\ell(t - \tau_o)}, \forall \ell \in \mathcal{L}
  $$

- $W >$ expected period
Scheduling heuristic

- Consider $A \neq B$ subsets of links: $A \rightarrow i$ and $i \rightarrow B$
- BPPC (under our specific setups / assumptions): node $i$ favors subset with maximum sum of differential backlogs
- Each node $i \in \{N\} \setminus N$ knows backlogs of all other nodes $j \neq i, N$

\[ \ell = (i, j), j \neq N \]
\[ D_\ell(t): \text{node } j, W_j \rightarrow i \]
\[ \forall \text{ links departing from } i \]
\[ \forall \text{ nodes except } N \text{ (sink)} \]
\[ \text{possible only in our specific setups} \]
\[ \downarrow \# \text{ of variables} \]
\[ \downarrow \text{computational complexity} \]
\[ \text{mode 2 implementation} \]

Scheduling heuristic (network layer)

For each node $i \in \{N\} \setminus N$:
- Calculate $D_\ell(t)$, $\forall \ell: \text{Tx}(\ell) = i$,
  \[ D_\ell(t) := \begin{cases} \max\{0, W_i(t) - W_j(t)\}, & j \neq N \\ W_i(t), & j = N. \end{cases} \]
- Calculate $D_k(t)$, $\forall k: \text{Rx}(k) = i$,
  \[ D_k(t) := \max\{0, W_j(t) - W_i(t)\}, j \neq N. \]
- If $\sum_{\ell: \text{Tx}(\ell) = i} D_\ell(t) > \sum_{k: \text{Rx}(k) = i} D_k(t)$,
  set $D_k(t) = 0, \forall k: \text{Rx}(k) = i$,
- else
  set $D_\ell(t) = 0, \forall \ell: \text{Tx}(\ell) = i$
Simulation setup

- $N = 6$ nodes, low-left = s, top-right = d, $L = 25$ links
- $G_{\ell,k} \sim 1/d^4$, $d$: distance Tx($\ell$), Rx($k$), $G = 128$,
- no-listen-while-talk: if Rx($\ell$) = Tx($k$) $\rightarrow G_{k,\ell} = 1/\epsilon$
- $V_{\ell} = 10^{-12}$, $P_{\ell} = 5$, $\forall \ell$
- Deterministic fixed-rate arrivals
Performance evaluation of core step algorithms

Simulation experiment:
- 100 packets/control slots, deterministic arrival rate: 9 packets/slot
- Centralized Batch high-SINR $\rightarrow$ resulting $D_\ell(t)$, $\forall \ell$, $\forall t \in \{1, 100\}$
- High-SINR $\leftrightarrow$ $\alpha_\ell(t) = 1$, $\beta_\ell(t) = 0$, $\forall \ell$ and $\forall t$
- Solve above 100 instances via distributed core step algorithms:
  - Initialization: $\{\lambda_\ell\}_{\ell \in \mathcal{L}}$, $\{\gamma_{\ell k}\}_{\ell \neq k \in \mathcal{L}} \rightarrow 1$,
    $\{\tilde{i}_\ell\}_{\ell \in \mathcal{L}}$ random, $\delta_s = \delta = 0.01$, $\forall s$
  - Termination criterion: local metrics must drop under $\epsilon = 10^{-2}$, $\forall \ell \in \mathcal{L}$
  - variation of local augmented Lagrange function, $\forall \ell \in \mathcal{L}$
  - norm of residual consensus constraints vector $r_\ell(s)$, $\forall \ell \in \mathcal{L}$
### The role of the penalty parameter

**Table:** Results for problem instance at 30\textsuperscript{th} slot, for various $\rho$ and initial $\{\rho_{\ell,o}\}_{\ell \in \mathcal{L}}$. Tolerance $\epsilon := 10^{-2}$. Objective value for Batch high-SINR: 97.6388.

<table>
<thead>
<tr>
<th>$\rho$ / ${\rho_{\ell,o}}_{\ell \in \mathcal{L}}$</th>
<th>0.002</th>
<th>0.003</th>
<th>0.01</th>
<th>0.1</th>
<th>0.5</th>
<th>1</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>core step 1: # iterations</td>
<td>1058</td>
<td>732</td>
<td>318</td>
<td>106</td>
<td>311</td>
<td>463</td>
<td>1212</td>
</tr>
<tr>
<td>core step 1: objective value</td>
<td>97.71</td>
<td>97.7</td>
<td>97.67</td>
<td>97.52</td>
<td>97.49</td>
<td>97.42</td>
<td>97.07</td>
</tr>
<tr>
<td>core step 2: # iterations</td>
<td>127</td>
<td>103</td>
<td>103</td>
<td>97</td>
<td>125</td>
<td>127</td>
<td>256</td>
</tr>
<tr>
<td>core step 2: objective value</td>
<td>97.58</td>
<td>97.56</td>
<td>97.58</td>
<td>97.57</td>
<td>97.57</td>
<td>97.57</td>
<td>97.65</td>
</tr>
</tbody>
</table>

- ‘sweet spot’ at $\rho \sim 0.1$: **minimum** # of iterations required
- Very low or high values of $\rho \rightarrow$ slow down convergence.
- Adaptive $\{\rho_{\ell}\}_{\ell \in \mathcal{L}}$: core step 2 less dependent on the initial choice
Simulation results

- Distributed core step algorithms vs centralized high-SINR, for 100 problem instances

Core step 1 (left): Objective value (top), abs. difference (bottom), $\rho = 0.01$.
Core step 2 (right): Objective value (top), abs. difference (bottom), $\rho = 0.1$. 
Performance evaluation of distributed S.A. algorithms for BPPC

- Performance of distributed high-SINR / S.A. examined through simulations
  - Scenario 1: **Small network** \((N = 6, L = 25)\) nodes, **moderate interference** \((G = 128)\)
  - Scenario 2: **Small network** \((N = 6, L = 25)\) nodes, **stronger interference** \((G = 8)\)
  - Scenario 3: **Larger network** \((N = 12, L = 121)\) nodes, **moderate interference** \((G = 128)\)

- Fixed physical layer propagation conditions
- Deterministic fixed-rate arrivals
- no-listen-while-talk: if \(\text{Rx}(\ell) = \text{Tx}(k)\) \(\rightarrow G_{k,\ell} = 1/\text{eps}\)
- \(V_\ell = 10^{-12}, P_\ell = 5, \forall \ell\)
Distributed adaptive high-SINR

- Scenario 3- larger network; $N = 12$, $G = 128$, max stable rate: 12.6 pps
Distributed adaptive high-SINR

- Scenario 3: unstable setup; arrival rate: 12.7 pps
Distributed adaptive Successive Approximation

- Scenario 3: larger network; $N = 12$, $G = 8$, max stable rate: 15.7 pps
Distributed adaptive Successive Approximation

- Scenario 3- unstable setup; arrival rate: 16.1 pps
Simulation results

**Table:** Maximum stable throughput and average weighted sum-rate attained by all algorithms in all scenarios. (1):= core step 1, (2):= core step 2, w.s.r. := weighted sum-rate.

<table>
<thead>
<tr>
<th>Scen. 1</th>
<th>Batch high-SINR (1) / (2)</th>
<th>Adapt. high-SINR (1) / (2)</th>
<th>Batch S.A. (1) / (2)</th>
<th>Adapt. S.A. (1) / (2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>rate</td>
<td>9.7 / 9.7</td>
<td>9.7 / 9.7</td>
<td>10.4</td>
<td>10.4</td>
</tr>
<tr>
<td>w.s.r.</td>
<td>151.32 / 151.32</td>
<td>151.31 / 151.31</td>
<td>184.46</td>
<td>184.45</td>
</tr>
<tr>
<td>Scen. 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rate</td>
<td>2.4 / 2.4</td>
<td>2.4 / 2.4</td>
<td>7.8</td>
<td>7.8</td>
</tr>
<tr>
<td>w.s.r.</td>
<td>0.95 / 0.96</td>
<td>0.95 / 0.96</td>
<td>107.48</td>
<td>107.47</td>
</tr>
<tr>
<td>Scen. 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rate</td>
<td>12.6 / 12.6</td>
<td>12.6 / 12.6</td>
<td>15.7</td>
<td>15.7</td>
</tr>
<tr>
<td>w.s.r.</td>
<td>276.19 / 276.21</td>
<td>276.18 / 276.2</td>
<td>454.31</td>
<td>454.3</td>
</tr>
</tbody>
</table>
Adaptive WMMSE

- Scenario 1- maximum stable rate: 12.4 pps > S.A.-BPPC (10.4 pps)
Adaptive WMMSE

- Scenario 1 - unstable setup; arrival rate: 12.5 pps
Adaptive WMMSE

- Scenario 3- larger network; max stable rate: 12.1 pps < S.A. (15.7 pps)
Distributed adaptive high-SINR

- Scenario 3: unstable setup; arrival rate: 12.2 pps
Adaptive WMMSE (mode 2)

- Scenario 3 - maximum stable rate: 15.7 pps (same for S.A.)
WMMSE versus S.A. under identical problems

- Fair comparison w.r.t. weighted sum-rate
- → under identical problem instances
- Experiment:
  - Let batch S.A. drive the network (schedule all links with $D_\ell(t) > 0, \forall \ell, \forall t$) (mode 1)
  - Solve problem with dif. backlogs resulting at each time slot from S.A. via batch WMMSE
  - Test WMMSE under mode 1, mode 2
WMMSE versus S.A. in scenario 1

- Rate: 10.4 pps, w.s.r. batch S.A. vs w.s.r. batch WMMSE (mode 1)
Simulation results

WMMSE versus S.A. in scenario 1

- Rate: 10.4 pps, w.s.r. batch S.A. vs w.s.r. batch WMMSE (mode 2)
Distributed BPPC for Wireless Multi-hop Networks

Simulation results

WMMSE versus S.A. in scenario 2

- Rate: 7.8 pps, w.s.r. batch S.A. vs w.s.r. batch WMMSE (mode 1)
WMMSE versus S.A. in scenario 2

- Rate: 7.8 pps, w.s.r. batch S.A. vs w.s.r. batch WMMSE (mode 2)
WMMSE versus S.A. in scenario 3

- Rate: 15.7 pps, w.s.r. batch S.A. vs w.s.r. batch WMMSE (mode 1)
WMMSE versus S.A. in scenario 3

- Rate: 15.7 pps, w.s.r. batch S.A. vs w.s.r. batch WMMSE (mode 2)
Summarizing simulation results

Table: Average throughput and average weighted sum-rate attained by WMMSE and S.A. algorithms in all scenarios. w.s.r. := weighted sum-rate.

<table>
<thead>
<tr>
<th>Scen. 1</th>
<th>WMMSE (Batch / Ad.) mode 1</th>
<th>WMMSE (Batch / Ad.) mode 2</th>
<th>S.A. (Batch / Ad.) mode 1</th>
<th>S.A. (Batch / Ad.) mode 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max rate</td>
<td>12.4</td>
<td>10.4</td>
<td>10.4</td>
<td>10.4</td>
</tr>
<tr>
<td>Avg. w.s.r</td>
<td>155.4</td>
<td>184.4</td>
<td>184.4</td>
<td>184.4</td>
</tr>
<tr>
<td>Scen. 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Max rate</td>
<td>12.3</td>
<td>7.8</td>
<td>7.8</td>
<td>7.8</td>
</tr>
<tr>
<td>Avg. w.s.r</td>
<td>168.6</td>
<td>107.3</td>
<td>107.4</td>
<td>107.4</td>
</tr>
<tr>
<td>Scen. 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Max rate</td>
<td>12.1</td>
<td>15.7</td>
<td>15.7</td>
<td>15.7</td>
</tr>
<tr>
<td>Avg. w.s.r</td>
<td>149</td>
<td>454.4</td>
<td>454.3</td>
<td>454.3</td>
</tr>
</tbody>
</table>
### Average complexity of WMMSE

**Table:** Best and worst cases of average run-time and average number of iterations per slot, of all WMMSE-based algorithms, concerning all scenarios considered.

<table>
<thead>
<tr>
<th></th>
<th>Batch WMMSE mode 1</th>
<th>Adaptive WMMSE mode 1</th>
<th>Batch WMMSE mode 2</th>
<th>Adaptive WMMSE mode 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max avg. run-time</td>
<td>0.06</td>
<td>0.0125</td>
<td>0.114</td>
<td>0.008</td>
</tr>
<tr>
<td>Min. avg. run-time</td>
<td>0.015</td>
<td>0.0061</td>
<td>0.055</td>
<td>0.0045</td>
</tr>
<tr>
<td>Max avg. # iter.</td>
<td>118</td>
<td>1</td>
<td>376</td>
<td>3</td>
</tr>
<tr>
<td>Min avg. # iter.</td>
<td>20</td>
<td>1</td>
<td>242</td>
<td>1</td>
</tr>
</tbody>
</table>

- Batch S.A. is not comparable (orders of magnitude slower)
- Adaptive S.A.: min avg. run-time 0.0065 seconds, max: 0.2 seconds (steady state)
Concluding remarks

- WMMSE clearly better than S.A.-BPPC complexity-wise
- Inconclusive results regarding performance:
  - WMMSE outperforms in strong interference scenario
  - S.A.-BPPC holds the edge in larger networks
- Comparing *suboptimal* BPPC solutions w.r.t. average weighted sum-rate may be a *fallacy* → which algorithm supports higher stable throughput?
- Scheduling heuristic *improves throughput performance* of WMMSE in larger networks
- Improvement to original WMMSE is simple but worthwhile, effective power initialization
• This research was co-financed by the European Union (European Social Fund - ESF) and Greek national funds through the Operational Program “Education and Lifelong Learning” of the National Strategic Reference Framework (NSRF) - Research Funding Program: Heracleitus II. Investing in knowledge society through the European Social Fund.

• Parts of this work appear in:

