Multirate Multicast: Algorithms and Implementation

Georgios S. Paschos

Massachusetts Institute of Technology

Joint work with:

Chih-ping Li & Eytan Modiano (MIT)

Kostas Choumas & Thanassis Korakis (CERTH)

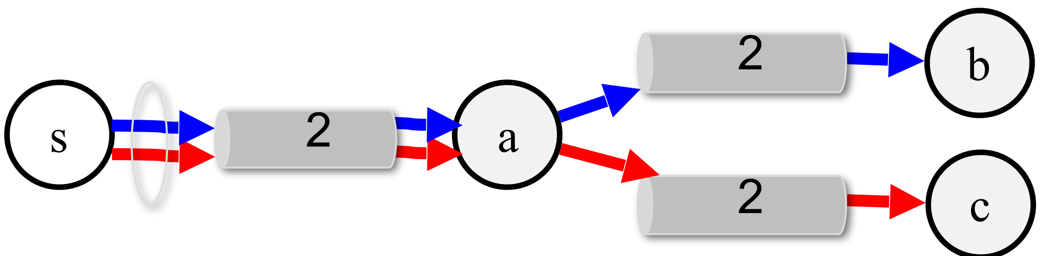
Introduction

- **Resolving congestion** of multicast sessions is complex
  - Several operators involved
  - Large number of receivers
  - Sources may not be trusted
  - Network variability
- Common approach: cooperation source/receivers

In this talk:
Resolve congestion inside the network without source cooperation
- Sources inject packets
- The network adapts by dropping packets

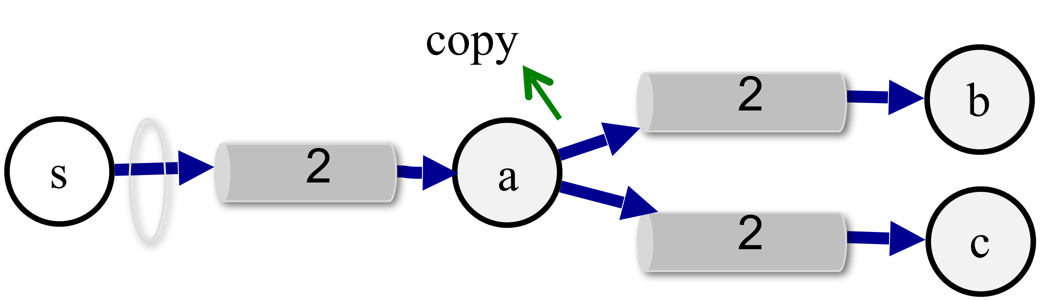
Multicast

- **Multicast** offers efficiency
  - Increases achievable rate
  - Or, reduces aggregated traffic



Multiple unicast

Max achievable rate: 1

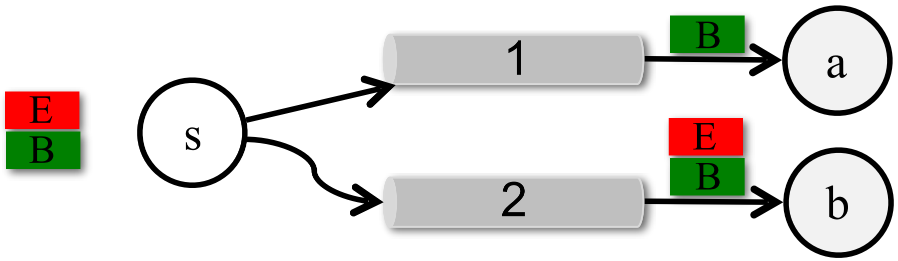


Multicast

Max achievable rate: 2

Multirate multicast

- Transmitting same rate to all receivers: inefficient



- Multirate multicast
  - Same stream at **different rate** per receiver
  - Layered video coding **[Li98]**
    - Basic layer packets: necessary for decoding at lowest quality
    - Enhanced layer packets: improve quality

Congestion control of multicast sessions

- Demand **λ** is outside the throughput region
  - Admit **r**≤**λ**

- Per-receiver NUM:

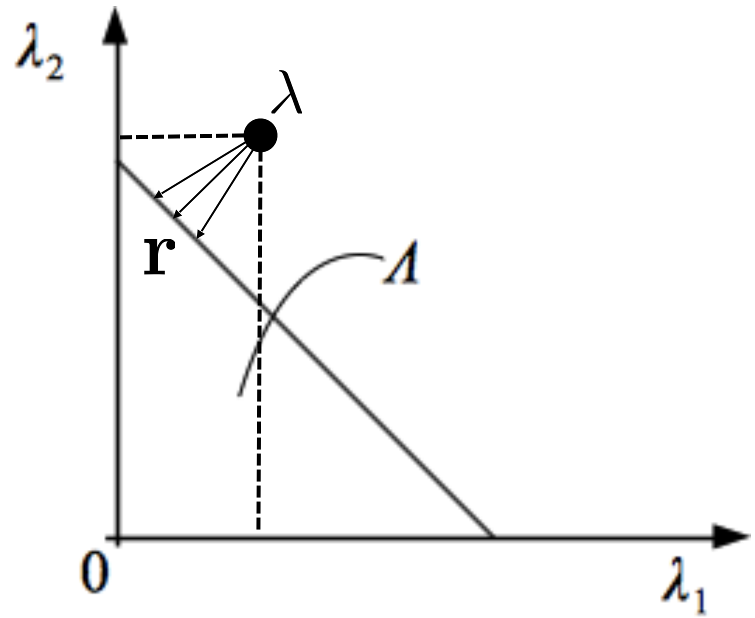
$$\max_{\mathbf{r}} \sum_{c,u} g(r_u^c)$$

throughput of receiver u session c

$$\text{s.t. } \mathbf{r} \in \Lambda$$

$$\mathbf{r} \leq \boldsymbol{\lambda}$$

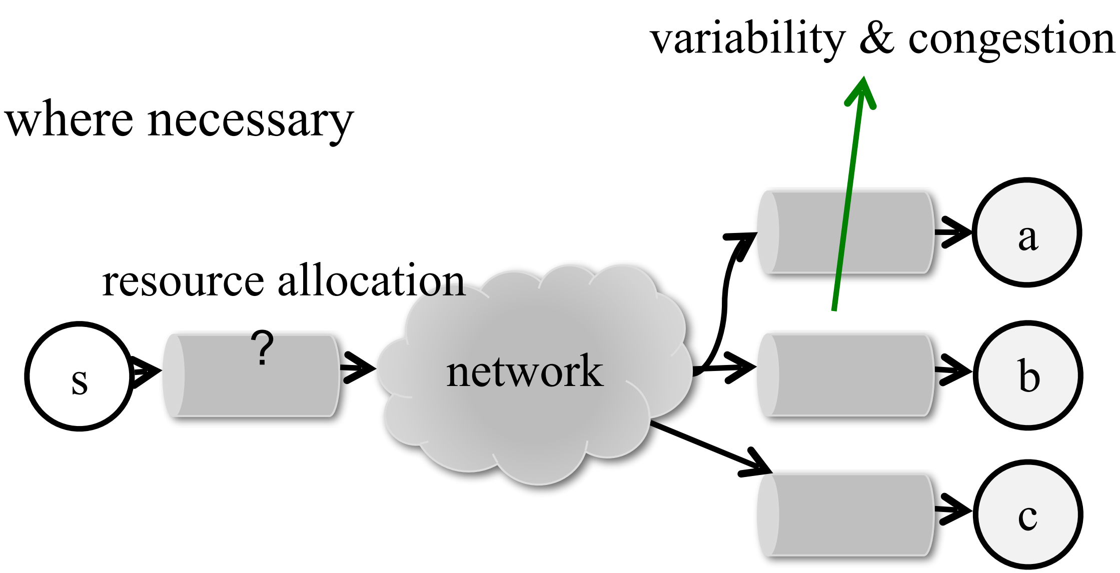
- Example objectives
  - **Max sum throughput** $g(x) = x$
  - **Proportional fairness** $g(x) = \log x$



- Optimal decision affected by network variability
  - Channel quality, network failures, user population, demand, and capabilities

Adaptive solutions

- Previous
  - Primal-Dual algorithms: **[Kar02], [Deb04]**
    - Messaging between sources and receivers
  - Backpressure-based: **[Neely05]**, **[Bui08]**
    - Sources decide how many packets to inject
- Proposed: Adaptation inside the network
  - Sources inject all packets
  - The network drops packets where necessary
- **Goal**: robust solution to the NUM problem



Outline

- Reformulate per-receiver NUM using **dropping rates**

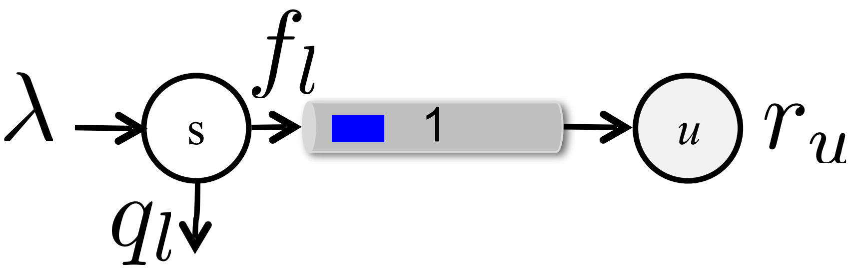
- In-network optimal control policy
  - Threshold-based dropping
  - Backpressure routing
  - Receiver-based congestion control

Maximum throughput

Maximum utility

Per-receiver NUM (dropping rate formulation)

- **Throughput maximization** $\max_{\boldsymbol{f},\boldsymbol{q}} \sum_u r_u$



$$\begin{aligned}\lambda &= q_l + f_l \\ &= q_l + r_u\end{aligned}$$

- Equivalent to **minimization of weighted dropping rates**

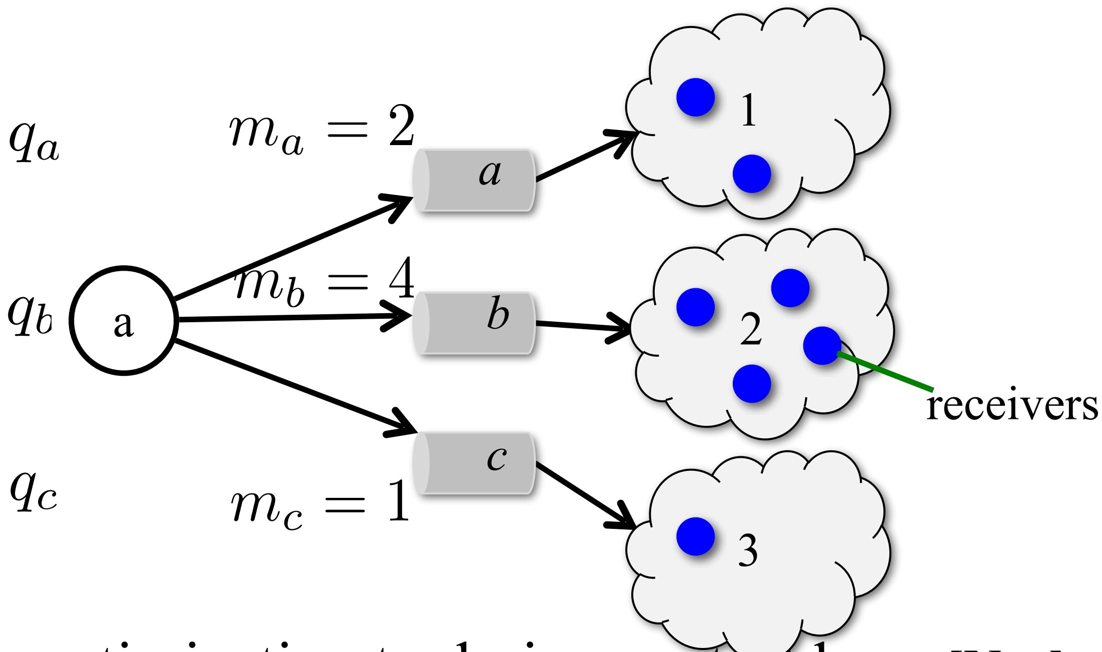
$$\min_{\boldsymbol{f},\boldsymbol{q}} \sum_l m_l q_l$$

  - Where m_l is the number of receivers fed through link *l*

Dropping rate minimization for max throughput

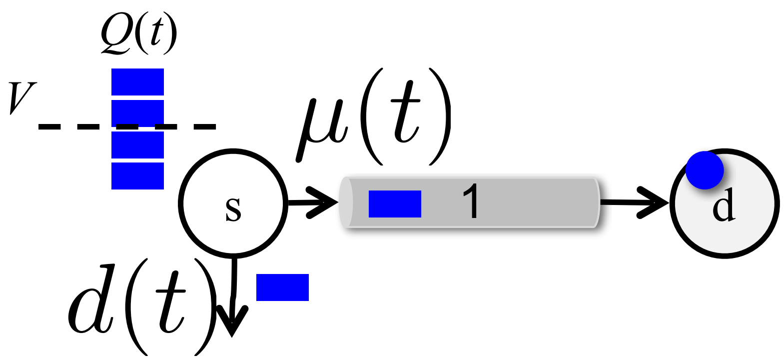
$$\min_{\boldsymbol{f},\boldsymbol{q}} \sum_l m_l q_l$$

- Where m_l is the number of receivers fed through link l
- The more the fed receivers, the more costly to drop packets!



- Use Lyapunov optimization to derive control laws **[Neely10]**

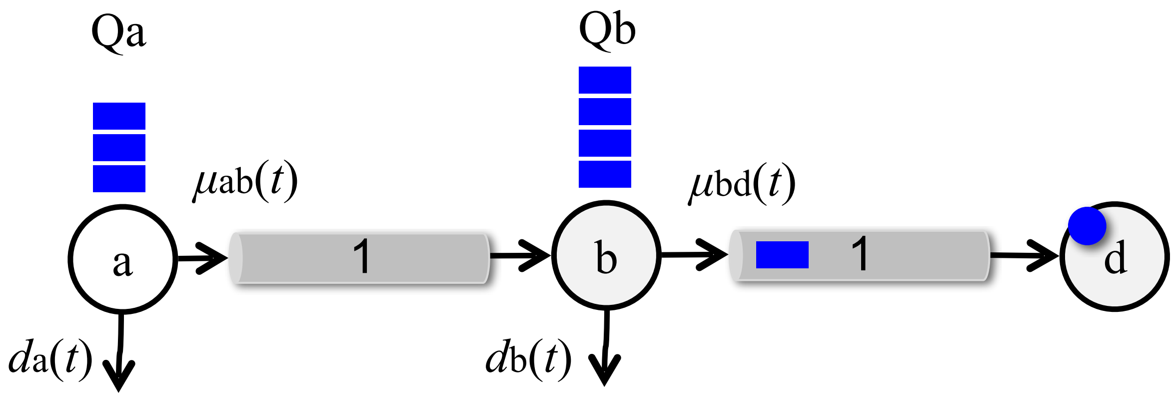
Real-time control – proposed dropping



- At every slot
  - Choose $\mu(t)$ to route packets
  - Choose $d(t)$ to drop packets

- **Threshold-based dropping**: If $Q(t)>V$ drop d_{max} packets, else zero

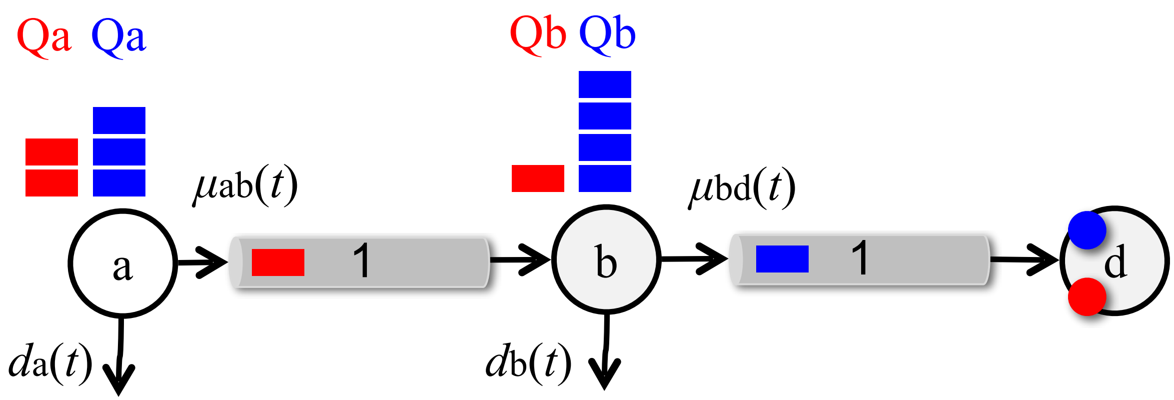
Proposed routing (path)



- **Backpressure routing**: Transmit at capacity if Qa(t)>Qb(t) (positive differential backlog)

Backpressure+threshold-based dropping
= maximum sum throughput for single session unicast

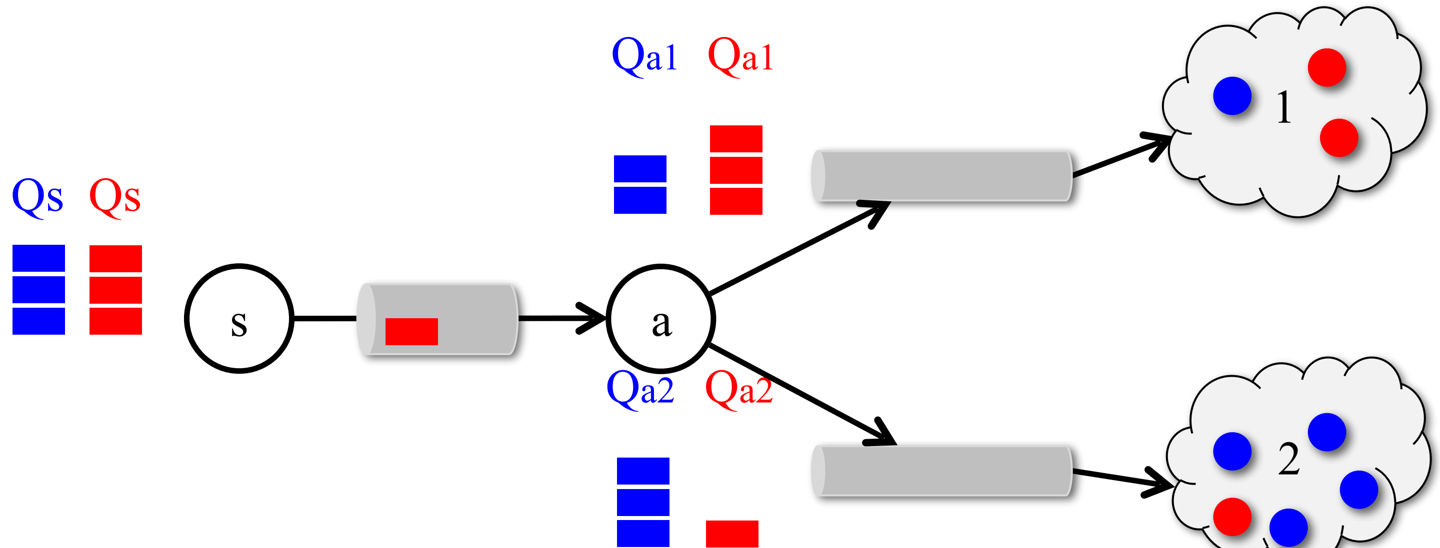
Proposed routing (multiple sessions)



- **Backpressure weight**: $W_{ab}^c = Q_a^c - Q_b^c$
  - Transmit the session with the maximum weight

Backpressure+threshold-based dropping = maximum sum throughput for multiple unicast

Proposed routing (tree)



- Backpressure weight: $$W_l^c = m_l Q_l^c - \sum_{k:l=p(k)} m_k Q_k^c$$

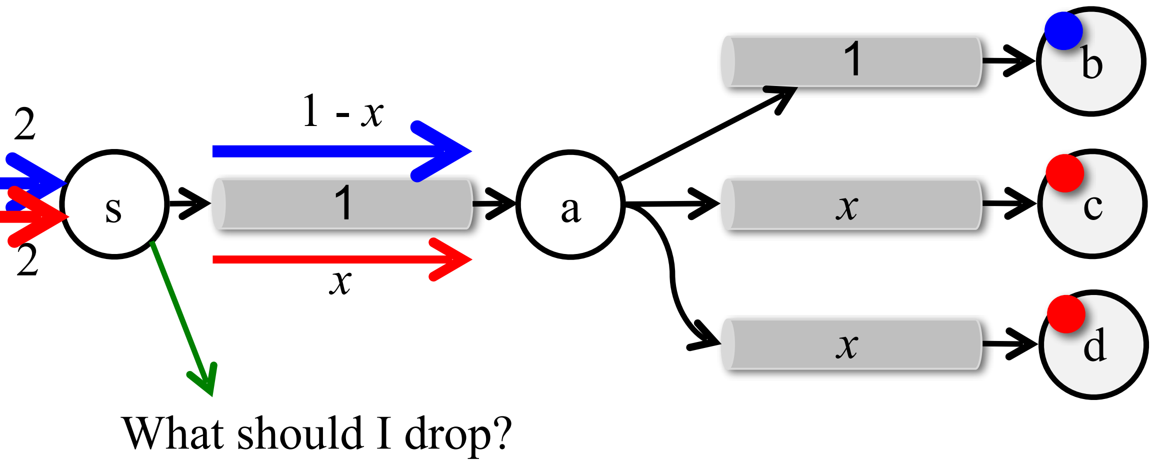
link (s,a) $\begin{cases} W_{sa} = 5Q_s - Q_{a1} - 4Q_{a2} \\ W_{sa} = 3Q_s - 2Q_{a1} - Q_{a2} \end{cases}$

- Transmit the session with the maximum weight

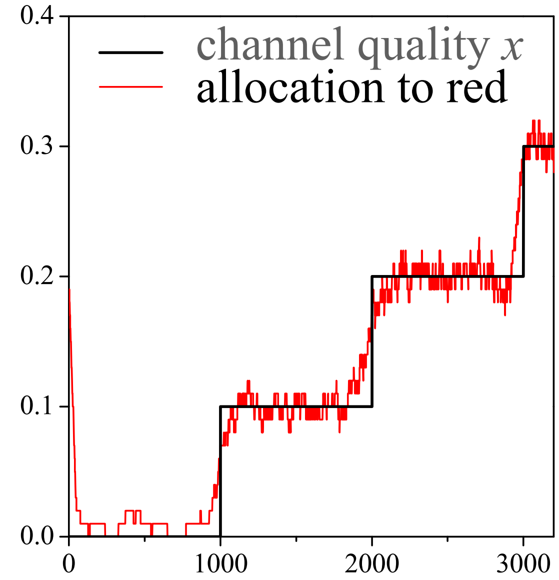
Max throughput for multirate multicasting on trees

Our policy adapts to channel changes

- Variable capacities x

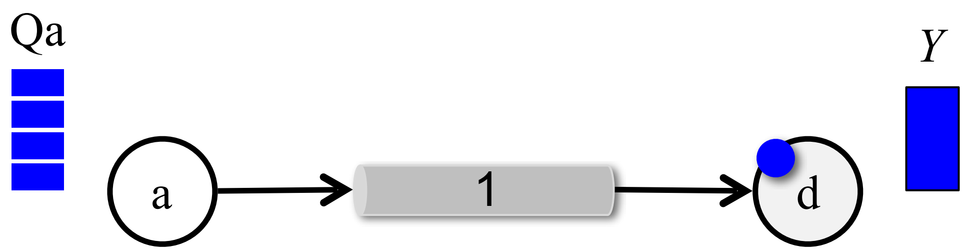


Threshold-based dropping+BP



- Goal: maximize total throughput
  - Optimal allocation on link (s,a): red = x, blue =1-x

Utility maximization



Add a "Virtual pressure"

Low: send more packets
High: stop sending

- Update Backpressure weight calculation: $W_{ad} = Q_a - Y$
- How Y evolves:
  - **Arrivals**: #of packets arriving at d
  - **Service**:

$$\max_x V g(x) + Y x$$
$$\text{s.t.} \quad x \in [0, \max]$$

parameter

utility function

virtual pressure

Theorem: As $V \to \infty$, the average received rate approaches the **optimal** utility.

Testbed experimentation

- Testbed experimentation
  - NITOS testbed (Volos, Greece)
- Implementation of the policy
  - Exchange backlog information
  - Virtual slot mechanism
  - Scheduling on wireless links
- Results
  - Full throughput ✔
  - Maximum Utility: selected nodes get more video layers ✔
  - Negligible messaging overhead ✔
  - 8-10% CPU occupancy ✔

Conclusions

- Proposed a multi-rate multicast congestion control scheme
  - Resource allocation **in the network**
  - Adaptive and distributed solution of the per-receiver NUM

- Future work:
  - Energy efficiency
  - Multi-hop wireless

- Visit our demo!
  - Wednesday 12:30-3:30pm
  - Harbour B