Adaptive Hierarchical Clustering of Message Flows in a Multicast Data Dissemination System

Yoav Tock, Nir Naaman, Avi Harpaz, Gidon Gershinsky

IBM Haifa Research Lab, Israel

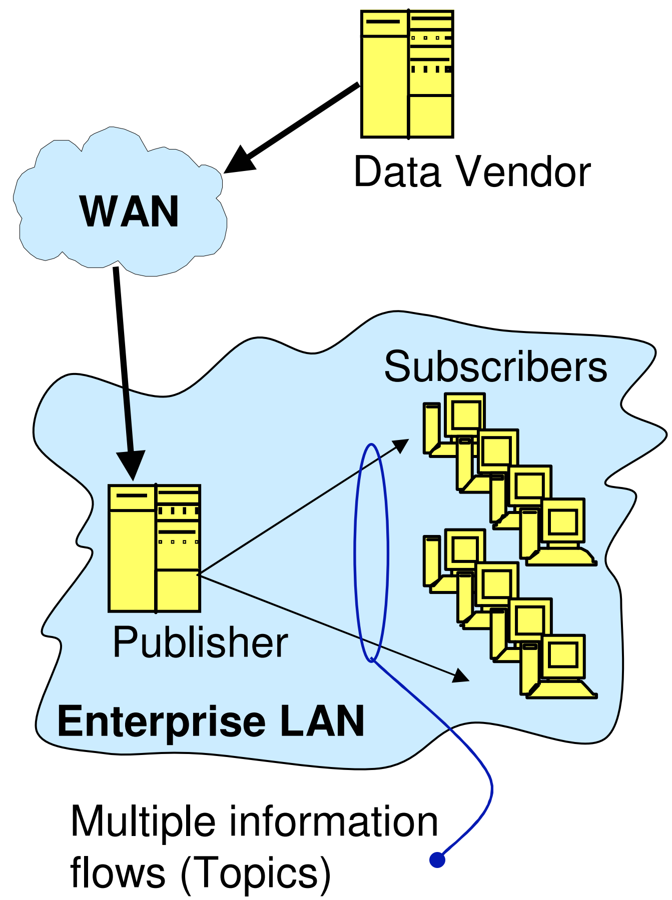
Outline

- Introduction
  - The System – Pub/Sub Messaging for Data Dissemination
  - Multicast Technology
- Multicast Mapping
- Clustering Algorithms
  - Modified K-Means, Hierarchical Clustering
- Real-Life Messaging-Load Model
- Experiments & Results
- An Adaptive System
- Future Directions
- Summary

The Basic Scenario – Pub/Sub for Market Data Dissemination

- Publisher divides data feed into a large number information flows (topics), (~100K) e.g. stock symbols, futures, commodities
- Many stand-alone subscribers (~1K)
- Subscribers display interest heterogeneity - are interested in different yet overlapping subsets of the topics
- Any single topic may be delivered to a large number of subscribers (hot / cold topics)

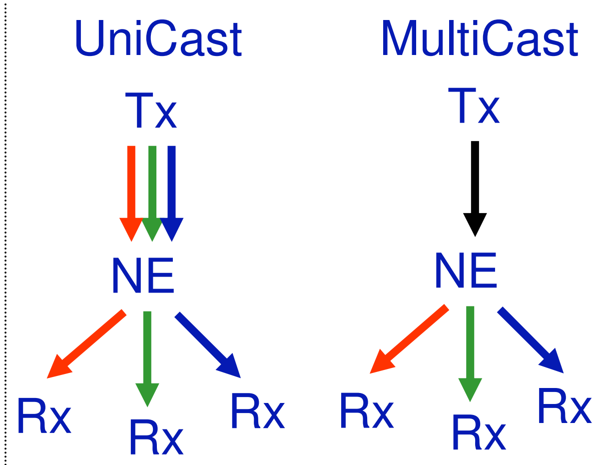
- Unicast – duplicate transmissions
- Flooding (Broadcast) – receivers burdened by unwanted incoming traffic

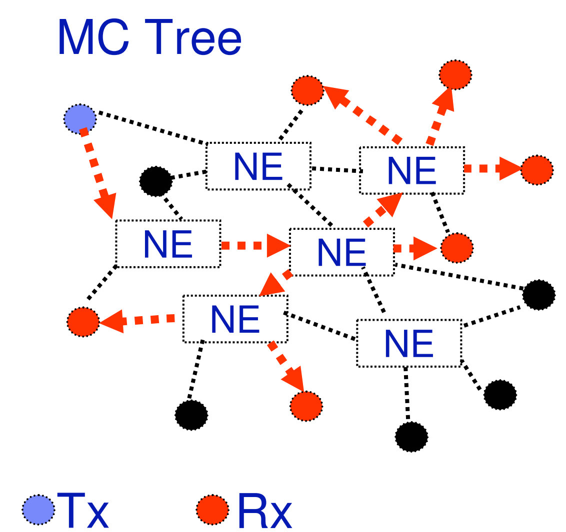


Multicast Technology

- IP multicast, Network layer
  - A single packet sent by a transmitter reaches all the hosts that joined a certain **Multicast Group**
  - Unreliable, no traffic control, no ordering
- Reliable Multicast Transport (RMT) Protocols
  - Reliability, Ordering, Flow & Congestion control
  - "**Session**" or "**Stream**" - transport layer entity
- Cannot allocate a group (or stream) per topic
- Limited number of usable multicast groups (NE state problem, receiver resources)
- Limited number of reliable multicast streams
- # Flows >> # RMT Streams >= # IP MC Groups

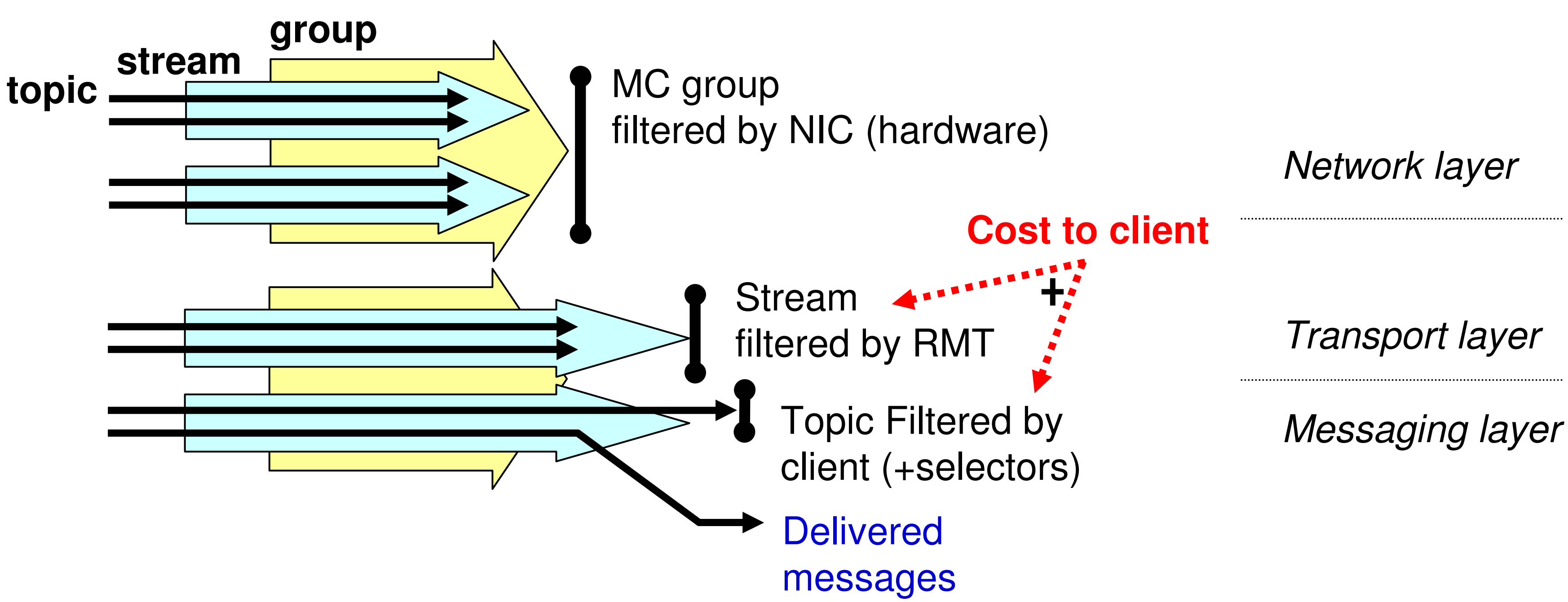
=> Mapping Flows to Streams
=> Mapping Streams to Groups





Map Structure and Filtering Cost

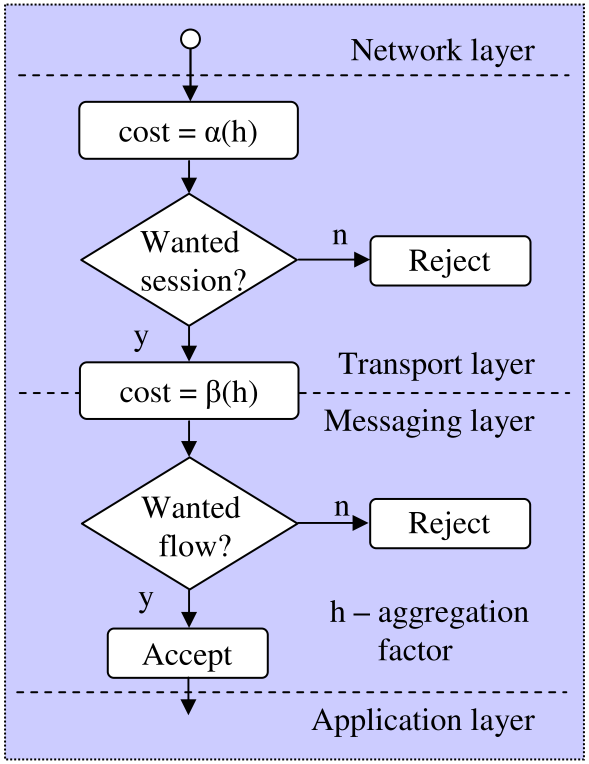
- Each Topic is mapped to a single RMT stream
- Each RMT Stream is mapped to a single multicast group
- Client filtering is a must
- The cost to the client depends on implementation details



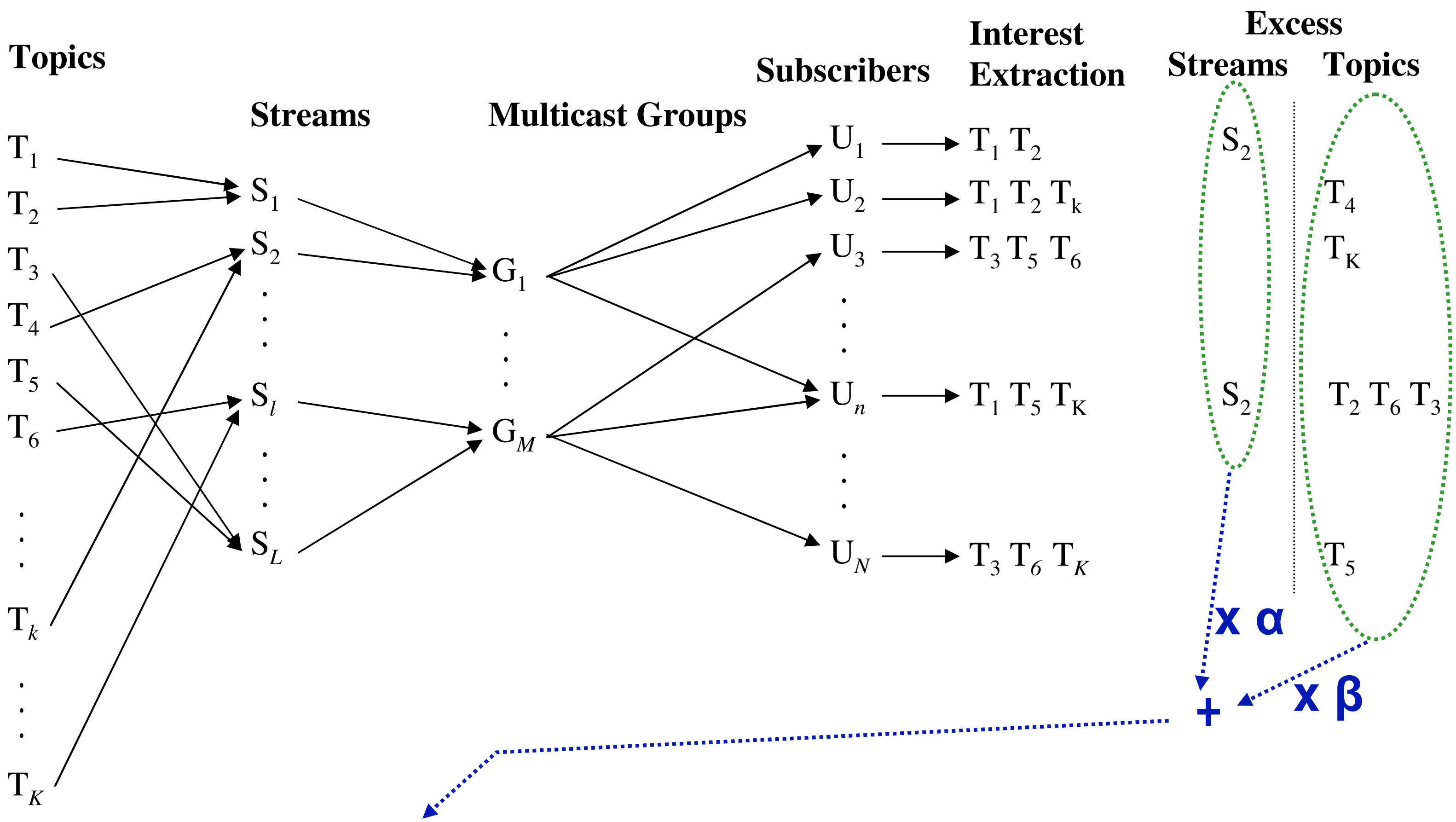
Message Aggregation and Filtering Cost

- Aggregation - multiple messages from the same RMT stream share the same packet
- At transport layer
  - Some processing for each packet
  - Some processing for each message
  - Amortization of packet-level processing across multiple messages, increases performance
- At messaging layer – processing per message
- Depends on implementation

- We estimated the effect of message aggregation and included it in the cost function



Example



- Cost: **α***∑ Excess_Stream(n) + **β***∑ Excess_Topic(n)
- A two level clustering problem

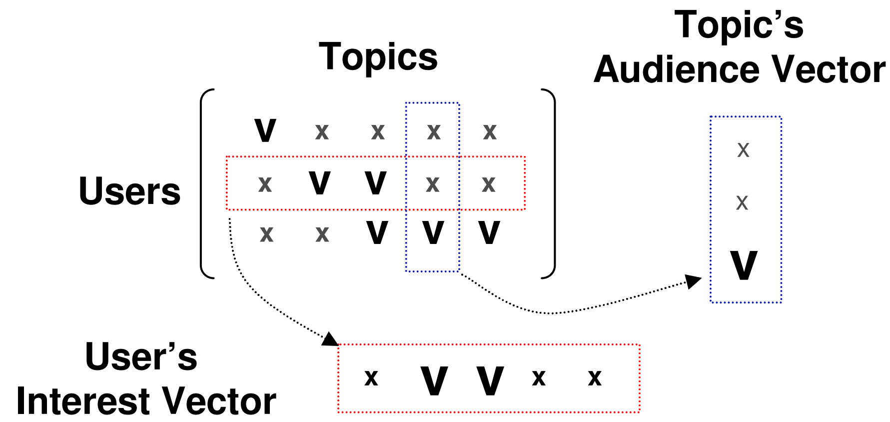
Algorithm Input - Messaging Statistics

- Publication
  - The list of published topics
  - The publication rate of each topic
- Subscription
  - The list of topics each client required
  - Client are anonymous
- Interest Matrix
  - A binary matrix indicating the interest of the clients
- Publication Rate Vector

Topic Rate Vector

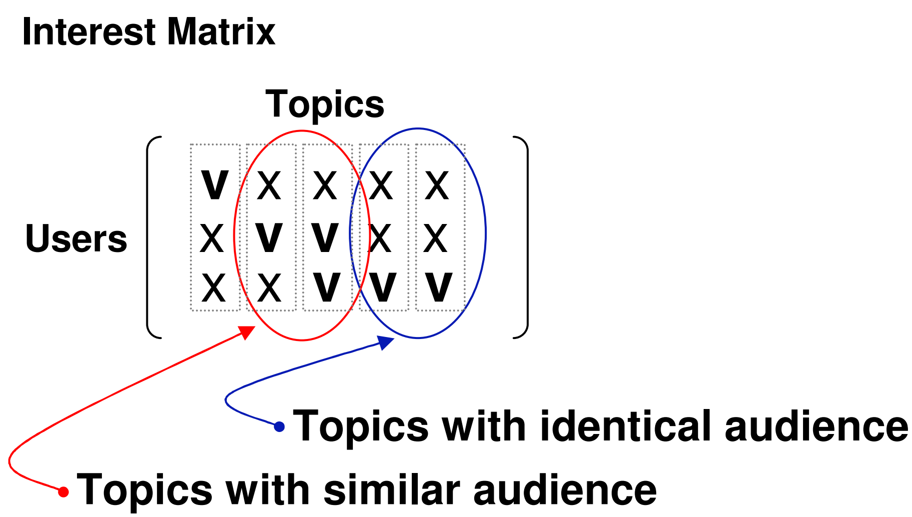
#	Topic [TopicSpace:Topic]	Rate [msg/s]
#1	Cars:Toyota/Hilux	10
#2	Cars:Honda/Civic	20
#3	Comp:IBM/pSeries	30
...etc		

Interest Matrix



Mapping Algorithm

- Input
  - interest matrix, topic rate vector
- Basic insight
  - Put “similar” topics in the same group
  - “Similar” topics have a similar audience
  - A group with a homogenous audience causes less filtering to the audience
- Take the rate into account
  - The cost of putting two topics in the same group
  - The cost of adding a new topic to a group of topics

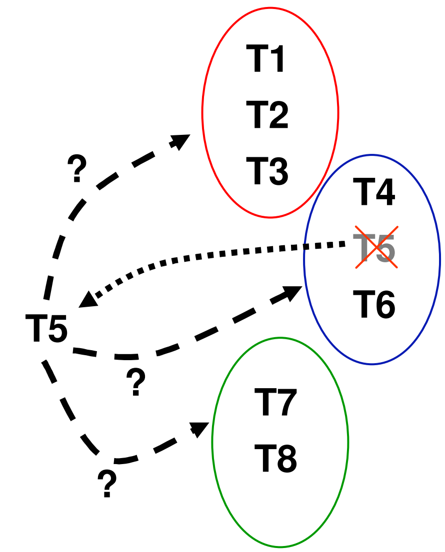


Users	Topics 1	Topics 2	Filtering Cost
1	V	X	R2
2	V	V	0
3	X	V	R1
4	X	X	0
			R1+ R2

Rk – the rate of topic k

Iterative Clustering Algorithm (K-means)

- **Init:** Topics are assigned into a fixed number of groups
- **Move:** In each step, remove a single topic, and move it to the best group – the one producing the lowest cost
- **Cost:** After each epoch, compute total filtering cost
- **Stop:** time elapsed | cost does not improve | exceeded max number of iterations | number of topics moved

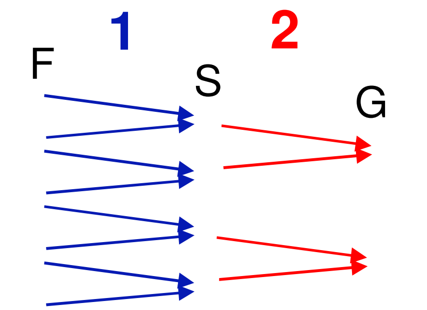


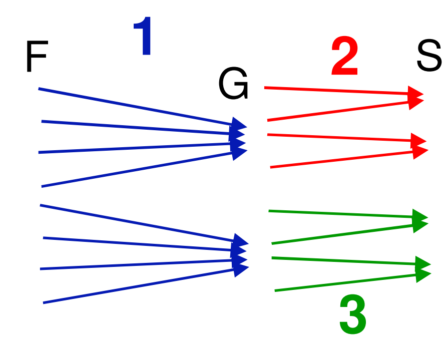
Users	Topic group 1	Topic group 2	Topic group 3	Group audience vector	Candidate topic 5	The cost of adding topic 5 to topic group {1,2,3}
	V	X	V	V	V	0
	V	V	X	V	V	0
	V	V	V	V	V	0
	X	X	V	V	X	R5
	X	X	X	X	V	R1+R2+R3
	X	X	X	X	X	0
						R1+R2+R3+R5

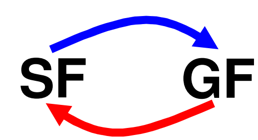
The best group for topic K is the group with the lowest cost

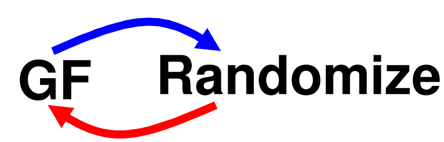
Hierarchical Clustering Algorithms

- Streams First (SF)
  - Cluster flows to streams
  - Cluster the resulting streams into groups
- Group First (GF)
  - Cluster flows into groups
  - Within each group separately, cluster flows into streams.
- An Iterative Approach (IT)
  - Iterative invocation of GF and SF
  - Taking the best map from all the iterations
- Random Restart with Annealing (RRA)
  - Random reassignment of a diminishing percentage of flows to streams,
  - Do a GF step
  - Taking the best map from all the iterations



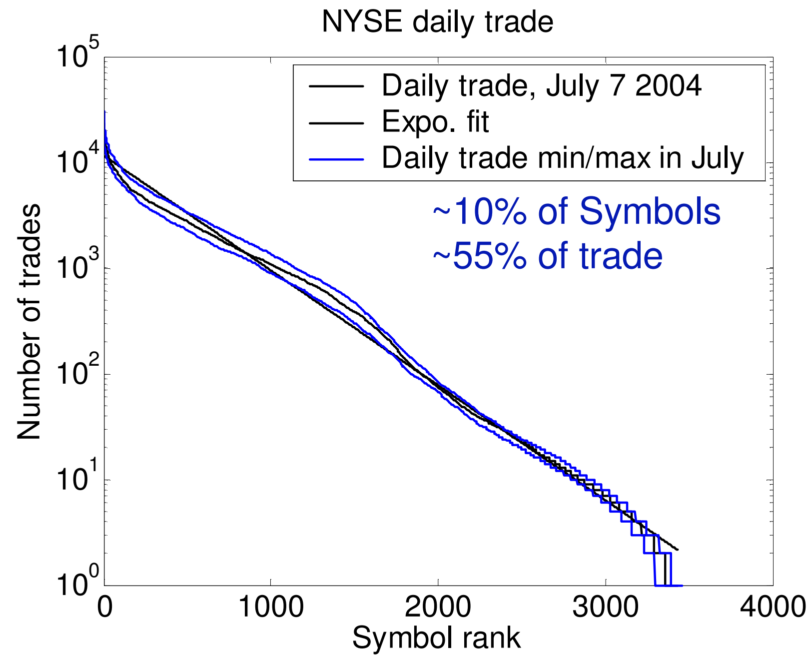


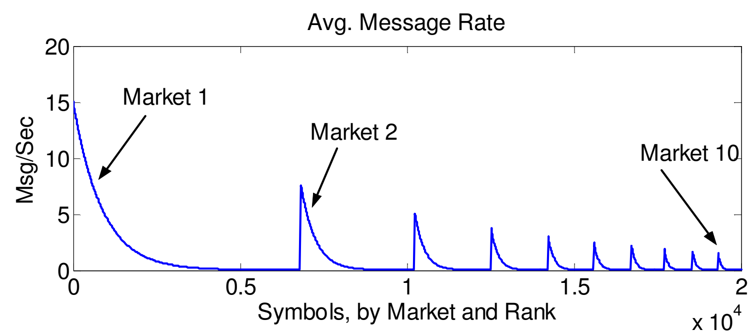




Messaging Load Model – Based on Market Research

- Financial front office
  - Hundreds of users, requiring stock quotes and financial information from several markets
  - Up-stream action (from brokers to market – buy/sell) is reflected in the Down-stream traffic (from market to broker – stock quotes)
- Topic space structure
  - Within each market, symbol popularity and rate are exponentially distributed (NYSE market research)
  - Several different markets, with Avg. popularity and size prop. ~1/m (assumption).

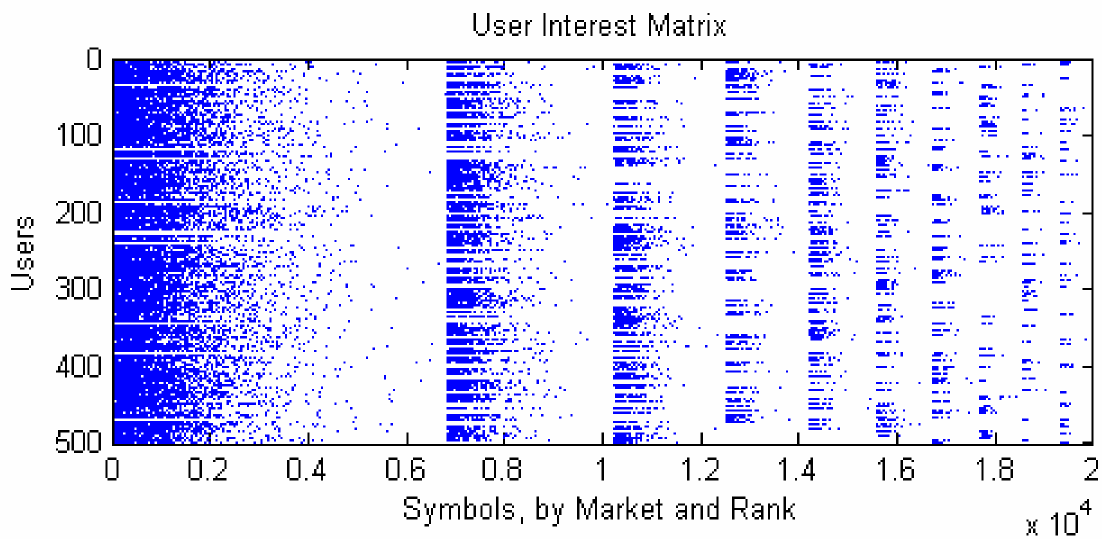


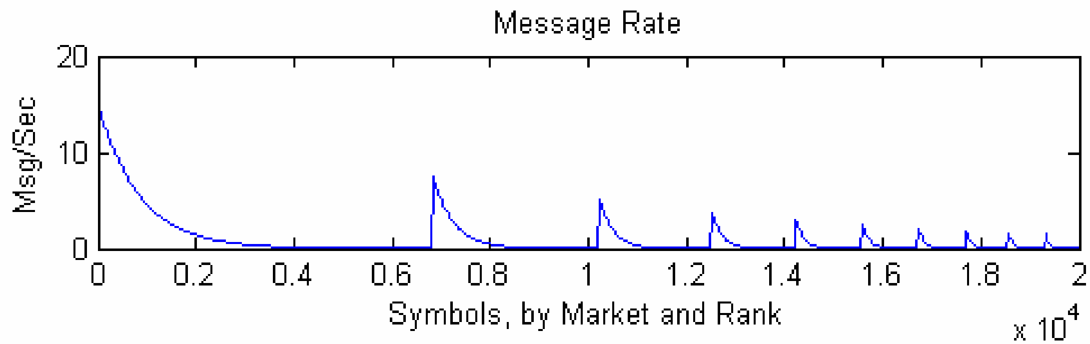


Real Life Messaging Load Model

- Based on statistical analysis of NYSE daily trade data
- 20K Topics
- 500 Subscribers
- Avg. ~70 topics / user
- Min 15 topics / user
- Max 115 topics / user
- Avg. message fan out ~**10.1** clients

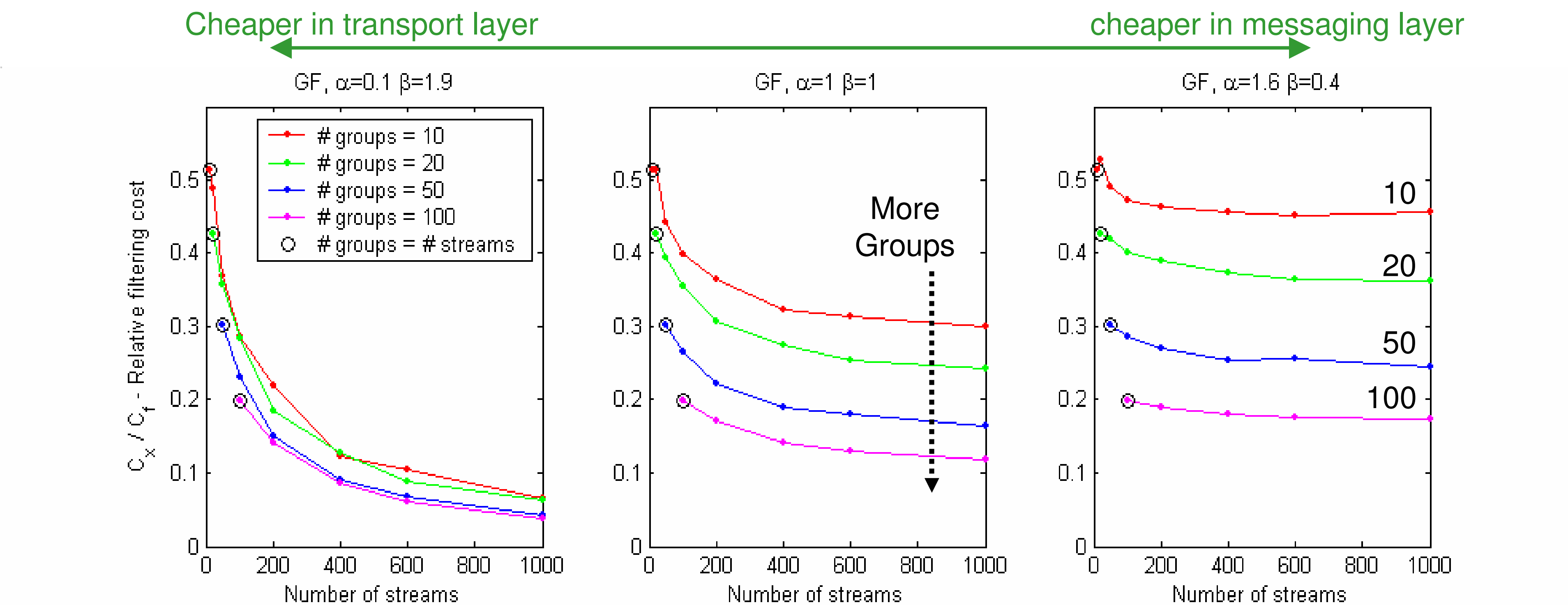
- Multicast - message is transmitted once
- Unicast transmitter data rate is **x10** of multicast !





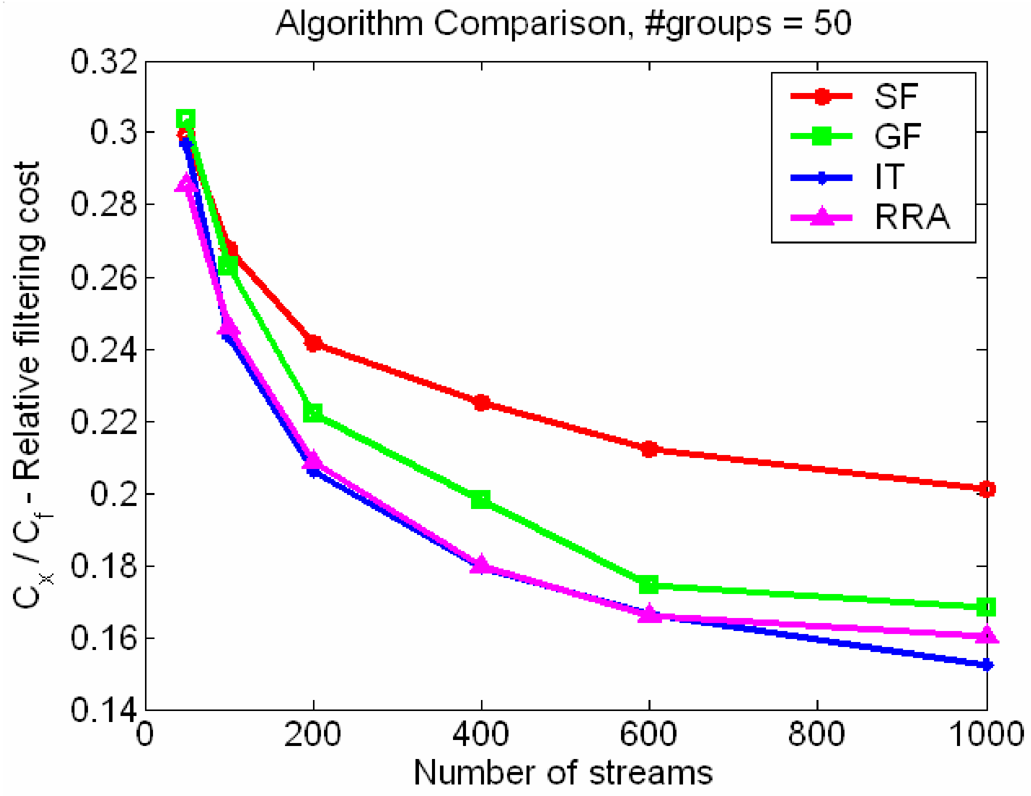
The Effect of the Number of Groups and Streams

- Increasing the number of streams and groups always improves performance
- Hierarchical filtering is more efficient than non-hierarchical
- Relative effectiveness depends on the amount of work in each filtering layer



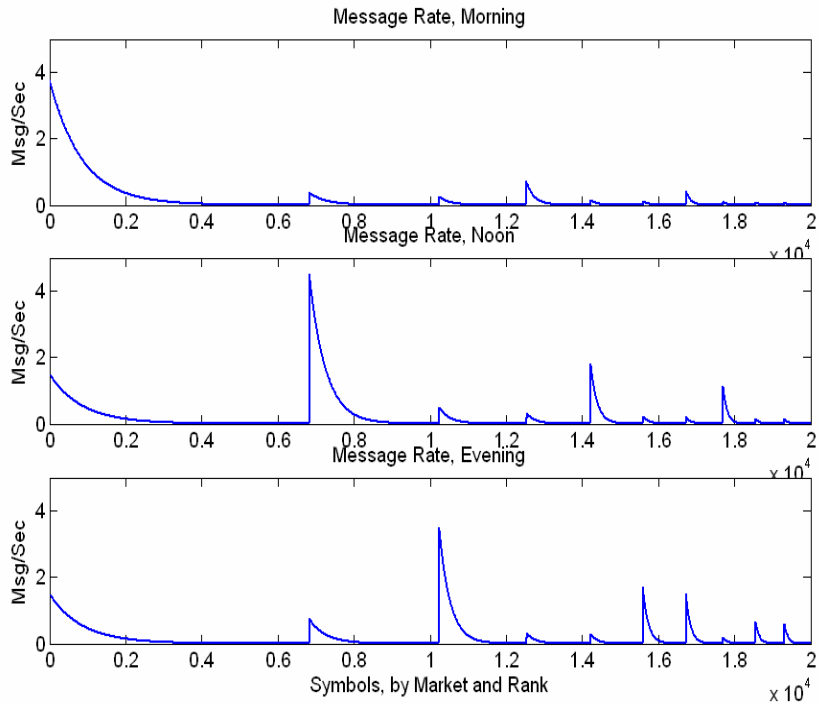
Algorithm Comparison

- GF is better than SF
- GF is fastest (not shown)
- Iterative algorithms
  - produce better results
  - take longer to execute (not shown)
- GF / Random = 0.4 - 0.6



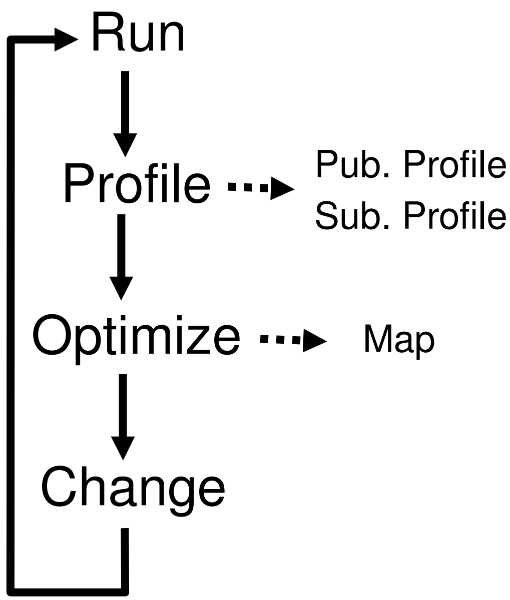
The Case For Adaptive Mapping

- User interest & message rate change during the day
  - Across markets
  - Within a market
  - In response to world events
  - Trading hours
- Manual management
  - Expensive, intractable
  - Error prone
- The “average” map
  - Of yesterday or a few days back
- Dynamic, Adaptive
  - Adapts to interests and rate
  - Runtime migration mechanism



Adaptive Multicast Infrastructure

- Run: running a messaging load in a given configuration.
- Profile: profiling publications and subscription.
- Optimization: the profiling results are fed into the optimization algorithm. The result is a map.
- Change: change publisher and subscriber configuration to the new map.

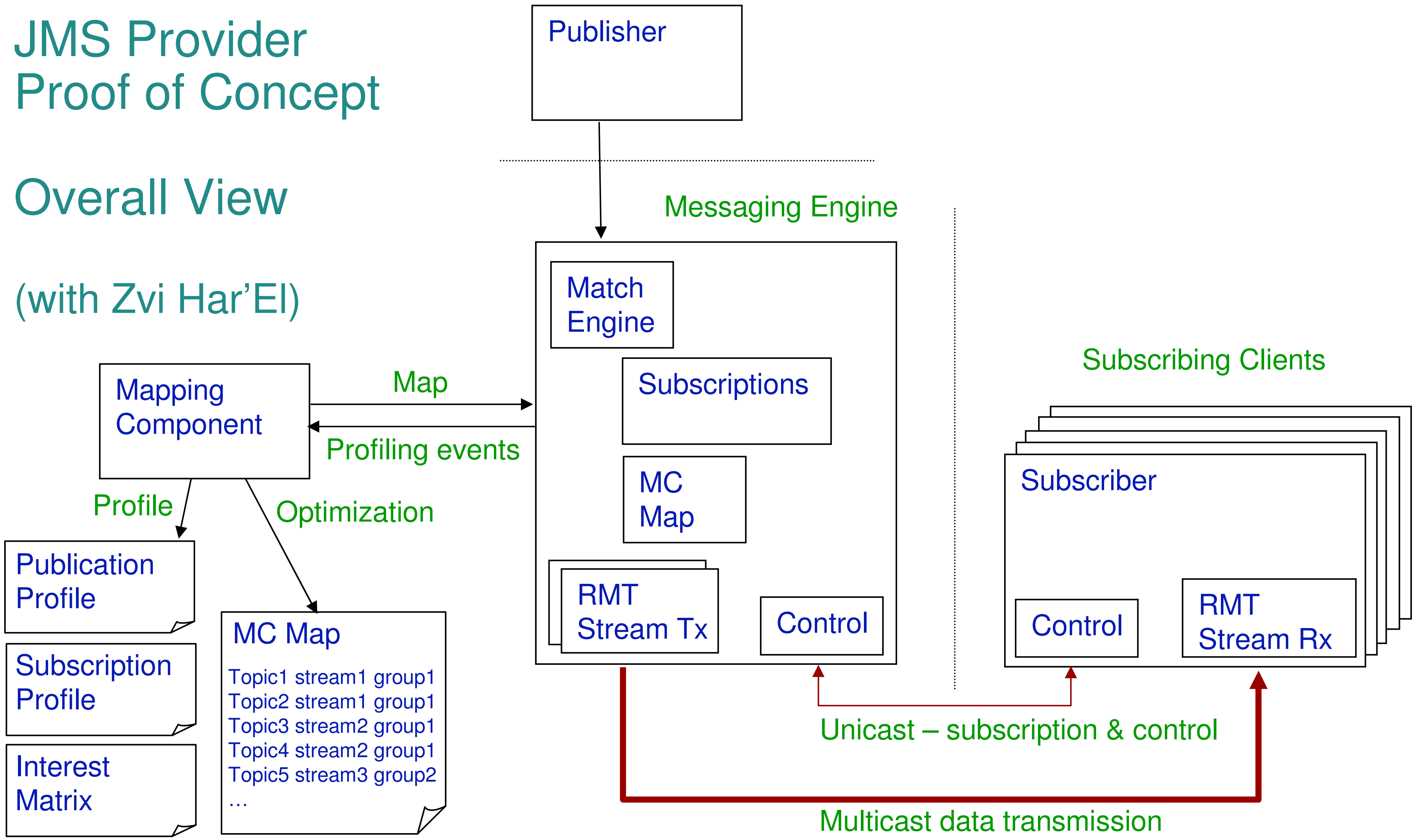


- The optimization starts from a previous map (fast)
- The adaptation time scale can be days, hours, minutes
- Change process is automatic, subject to QoS requirements
- Manual override – process control, map editing, pub/sub profiles

JMS Provider
Proof of Concept

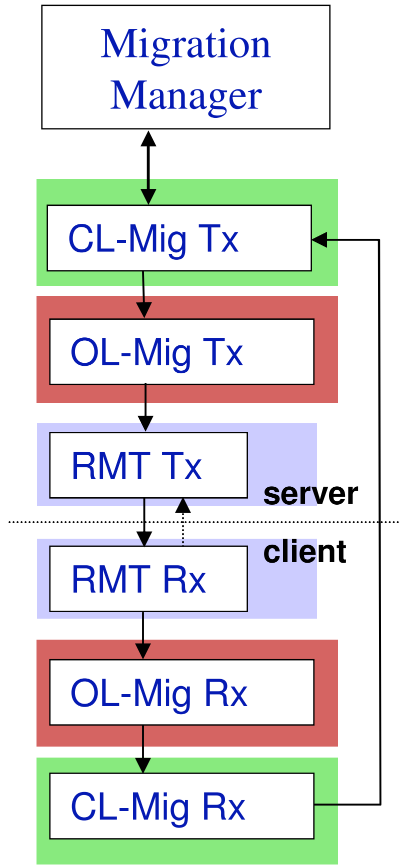
Overall View

(with Zvi Har'El)



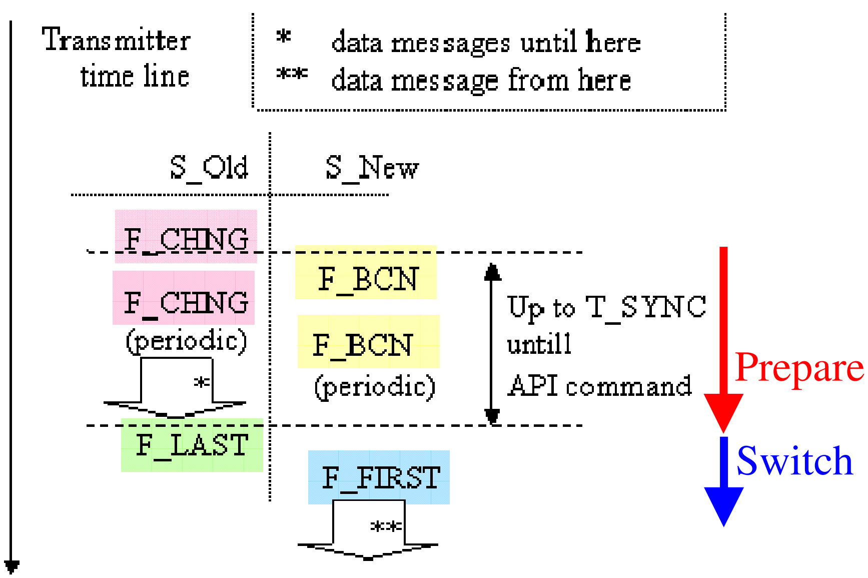
Migration Protocol

- General requirements
  - Preserve flow message sequencing
  - Avoid duplicate transmissions
  - Conform with the multicast reliability guarantees
  - Fast - reasonable time from start to finish
  - Scalable – number of clients / subscriptions
- Efficient protocol
  - Not "stop the world", no pipeline drainage
  - Messaging activity and throughput is hardly affected
- Use existing RMT API with minimal changes
- A layered approach
  - Isolation of lower level protocols
  - Allow for two levels of quality of service

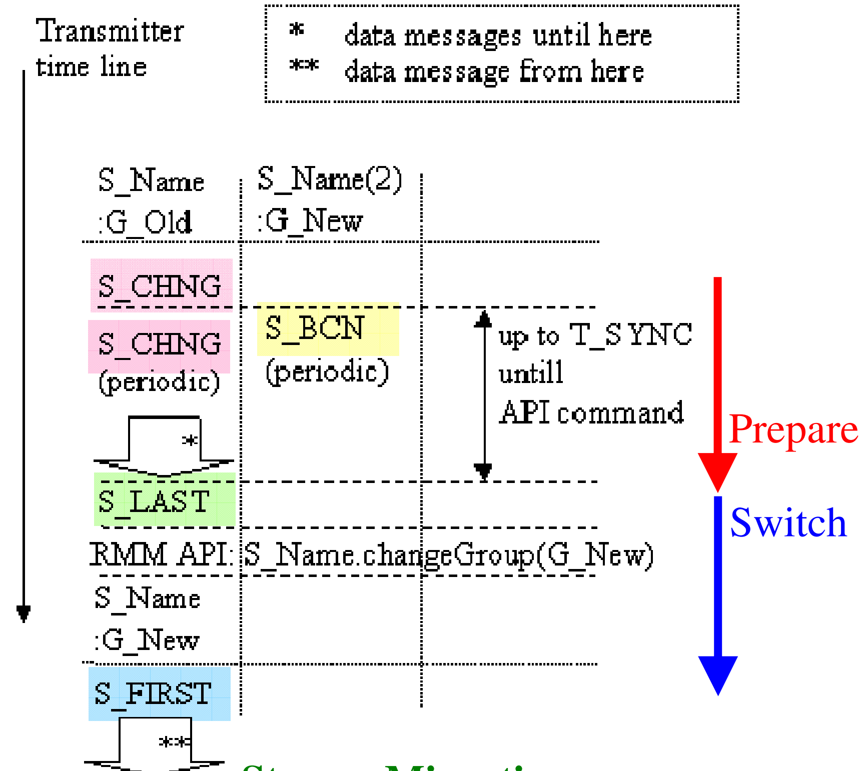


Flow & Stream Migration - Open Loop

- Based on standard reliable multicast transport protocol / infrastructure (e.g. PGM)
- No feedback from receivers, reliability based on timing, receiver detects failures
- Prepare phase – two signals (Change + Beacon)
- Switch phase – two signals (Last + First)



Flow Migration



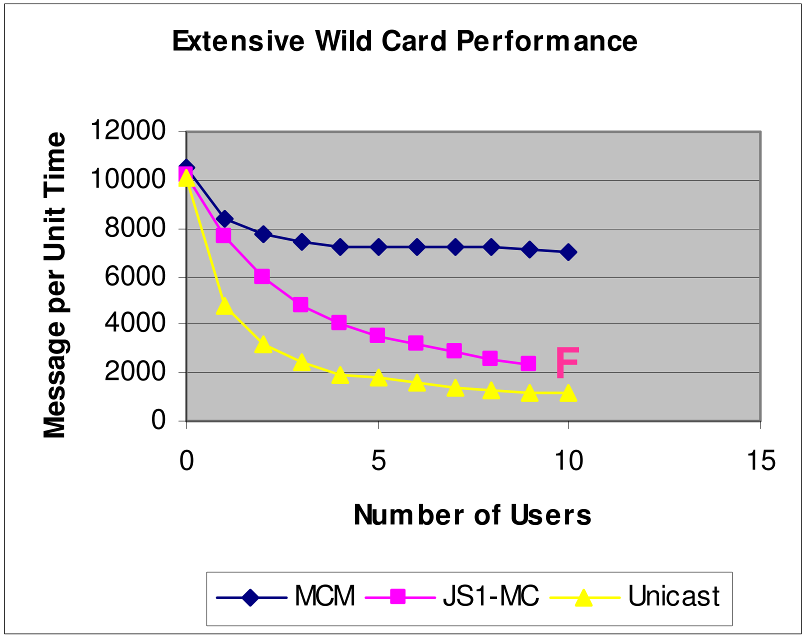
Stream Migration

Overall Performance – Extensive Wild Card

- JMS messaging provider POC
- Hierarchic topic-based
- Subscriptions are unique and overlapping, e.g.
  - /*/b/*/d
  - /a/*/*/d
- ~7 topics/user @ 10 users

- Unicast – approx. 1/n
- JS1-MC – stream per unique subscription:
  causes data duplication – performance degradation is almost like unicast

- Multicast mapping is scalable, and applicable



Current & Future Work (With Gregory Chockler, Roie Melamed)

- Distributed Large Scale Pub/Sub
  - A large number of topics (x1000)
  - A large number of users (x10000)
  - Correlated user interests (x100 / User)
  - High churn
  - No IP multicast
- Based on overlay network, P2P
  - That takes into account the user interest
- How do we
  - Define abstract dissemination channels
  - Map topics to abstract dissemination channels
  - Migrate topics between channels

Summary

- Large scale multicast Pub/Sub
  - A huge number of topics
  - A limited number of RMT streams, IP multicast groups
  - Hierarchic approach
- Cost function – hierarchic filtering, message aggregation
- Estimated the relative cost of transport vs. messaging layer filtering
- Iterative clustering algorithm based on K-means
- Several hierarchic clustering algorithms
- Real-life messaging load based on NYSE market research
- Hierarchic filtering is better then flat
- Advantage for efficient filtering at transport layer
- The challenges of an adaptive fully distributed system

The End