Boosting Topic-Based Publish-Subscribe Systems with Dynamic Clustering

Tova Milo, Tal Zur, Elad Verbin

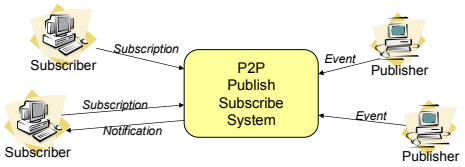
Presented by: Herodotos Herodotou
February 19, 2008

Outline

- Motivation
- Background on Pub-Sub Systems
- Topic Clusters
- Cost Model
- Dynamic Clustering Algorithm
- Subscription Algorithm
- Implementation
- Experimental Results

Motivation

- **Challenges:**
  - Dynamic Nature of P2P Pub-Sub Environment
  - Decentralized Nature of P2P Pub-Sub Environment
- **Goal:**
  - Maximize performance



Background: Pub-Sub Systems

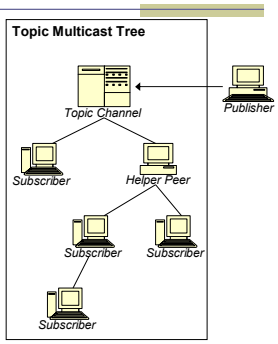
- **Three Components:**
  - Subscribers
    - Express interest in events
  - Publishers
    - Create topics and generate events
  - Infrastructure
    - Match events with interests and deliver events

Background: Pub-Sub Systems

- **Categories:**
  - Topic-Based
    - Subscribers specify interests as “topics”
    - Publishers label events with topics
  - Content-Based
    - Subscribers specify interests through event filters
    - Published event content matched against filters

Topic-Based Pub-Sub Systems

- **Operations:**
  - Create
  - Publish
  - Subscribe
  - Unsubscribe
- **Costs:**
  - Transmission Cost
  - Maintenance Cost



Topic Clusters

- **How to decrease maintenance cost?**
  - Group topics with similar sets of subscribers into a "virtual" topic
    => Unify underlying supporting structures
    => Eliminate redundancy in structures of individual topics
- **Side effects:**
  - Users can subscribe to topics or topic-clusters
  - Need local filters to eliminate irrelevant events

Cost Model

- **Maintenance Cost:**

$$MC = \sum_{t \in T} cost_m(size(t)) + \sum_{c \in C} (cost_m(size(c)) + cost_a(|c|))$$

- **Dissemination Cost:**

$$DC = \sum_{t \in T} freq(t) \times [(cost_n() + cost_d(size(t))) + \sum_{\{c | t \in c\}} (cost_n() + cost_d(size(c)))]$$

- **Overall Cost:**

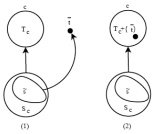
$$OC = MC + DC$$

Dynamic Clustering Algorithm

- Cluster Update Operations:
  - Topic 1 + Topic 2 → New Cluster
  - Topic + Cluster → New Cluster
  - Cluster 1 + Cluster 2 → New Cluster
  - Old Cluster → New Cluster + Topic
  - Old Cluster → Topic 1 + Topic 2
- Algorithm:
  - Perform Update if and only if it is beneficial

Adding Topic to Existing Cluster 1/2

- Benefit Estimation:



Update is beneficial when:
$|\tilde{s}| > \frac{1+freq(\tilde{t})(|S_c|+1)}{1+freq(\tilde{t})}$

- Triggering Benefit Estimation:
  - Frequency of events in topic
  - Number of subscribers to cluster
  - Size of subscriber subset registered to topic

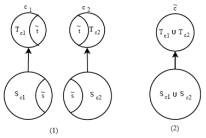
Adding Topic to Existing Cluster 2/2

- Trigger Benefit Estimation Probabilistically:
  - When user subscribes to topic *t*, perform benefit estimation with probability *ω*.
  - Conduct survey among *c* cluster's subscribers.
  - $|\tilde{s}| \approx \left(\begin{array}{c} \text{Fraction of common} \\ \text{subscribers to } t \,\&\, c \end{array} \right) * (\text{cluster size})$
- Notes:
  - $\omega > 1 - (1 - \phi)^{\frac{1}{|\tilde{s}|}}$

    where *φ* = configuration parameter

Merging two Clusters 1/3

- Benefit Estimation:

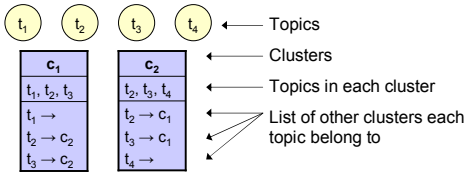


Update is beneficial when:
$|\tilde{s}| > \frac{\alpha|T_{c1} \setminus \tilde{t}| \cdot |S_{c2} \setminus \tilde{s}| + \alpha|T_{c2} \setminus \tilde{t}| \cdot |S_{c1} \setminus \tilde{s}| - |\tilde{t}|(\alpha+1)}{\alpha|\tilde{t}| + 1}$

- Triggering Benefit Estimation:
  - Frequency of events in topics
  - Number of topics in each cluster
  - Number of subscribers in each cluster
  - Number of common topics and subscribers

Merging two Clusters 2/3

- Estimate number of common topics:



  - Number of common topics = number of times c_2 appears in the lists of c_1
  - *Lists maintenance through cluster update notifications*

Merging two Clusters 3/3

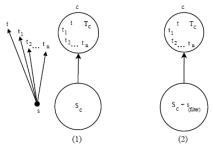
- Estimate number of common subscribers:
  - Maintain overall number of subscribers
  - Sample subscribers periodically to learn which other clusters they are subscribed to
  - $|\tilde{s}| \approx \left(\text{Fraction of common subscribers to } c_1 \,\&\, c_2\right) * (\text{number of subscribers})$

Subscriptions

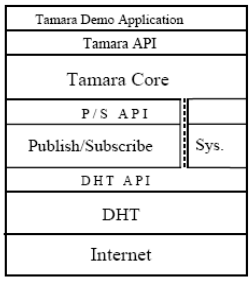
- Optimal Subscription Problem:
  - Find set of topics and clusters to subscribe, that cover all interests and minimizes cost
  - NP complete
- Greedy Subscription Algorithm
  - Treat each individual request for adding/removing a topic heuristically

Greedy Subscription Algorithm

- User *s* wishes to subscribe to topic *t*
  - If *t* belongs to a cluster *s* is subscribed to, update the filter
  - Otherwise two choices:
    - Subscribe *s* to *t* directly
    - Subscribe *s* to new cluster that includes *t*



Implementation



Experiments on Synthetic Data 1/4

- Events Frequency Models

Frequency	% of topics
0.6-0.6	15
0.2-0.4	5
0.1-0.2	25
0.02 - 0.1	45
0.002 - 0.02	10

(a) Average

Frequency	% of topics
1-3	5
0.5-1	20
0.1-0.5	50
0.01-0.1	20
0.001-0.01	5

(b) Frequent

Frequency	% of topics
0.1-0.2	10
0.01-0.1	40
0.001-0.01	50

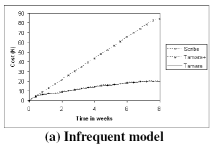
(c) Infrequent

- Quality of Clustering

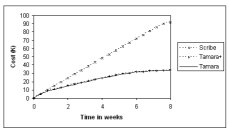
Events	Algo.	Structure maint.	Event diss.	OC
Average	Tamara	-78.75%	+45.45%	-63.73 %
	K-means	-53.75%	+9.1%	-46.15%
Frequent	Tamara	-73.75%	+15.62%	-48.21%
	K-means	-65%	+9.3%	-43.75%
Infrequent	Tamara	-88.75%	+150%	-77.38%
	K-means	-65%	0	-61.9%

Experiments on Synthetic Data 2/4

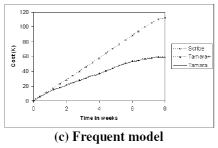
- Operation Cost Over Time



(a) Infrequent model



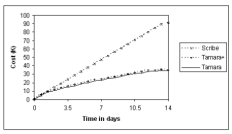
(b) Average model



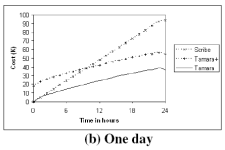
(c) Frequent model

Experiments on Synthetic Data 3/4

- Effect of User Subscription Rate



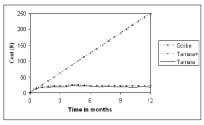
(a) Two weeks



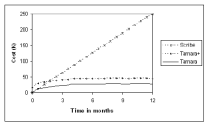
(b) One day

Experiments on Synthetic Data 4/4

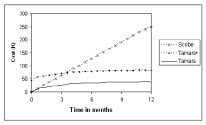
- Effect of Maintenance Interval



(a) Linux updates for $M = 2$ hours



(b) $M = 12$ hours



(c) $M = 24$ hours

Discussion

- Stability of Dynamic Algorithm is not addressed
- Functions used in Cost Model are too generic
  - Do not take network distance into consideration
- Adjustment cost is ignored
- Excessive use of heuristics
  - Triggering benefit estimation process
  - Greedy subscription algorithm