Efficient Multicast on a Terabit Router

Punit Bhargava punit@cisco.com
Sriram C. Krishnan srikrish@cisco.com
Rina Panigrahy rinap@cisco.com

Distributed Router
128 linecard: 5 Terabit system

Need 7 bits to specify address unicast
128 bits (16 bytes) to address multicast (exactly)

64 Byte Cell

Address to Destinations Table:
2\(^{20}\) 128-bit entries

128 bits

Need to supercast efficiently

Number of multicast connections

Run out of space in address-to-destinations table

Distinct destination sets have to “share” an entry in table: program the super set (union)

\[
\begin{align*}
1011 & \ldots \ldots \\
0110 & \ldots \ldots \\
\end{align*}
\] \quad \{ 1111 \ldots \ldots \}

Supercast: send to more linecards and discard at non-subscribing linecards
Minimum Cumulative Supercast (MCS) Problem

Given $N$ destinations sets (DS), positive integers $K$, $M$, such that $M < N$, does there exist an $M$-clustering (assignment of each DS to a cluster, $f: N \rightarrow M$) such that “total supercast amongst all clusters” $< K$?
Example: N=4, M=2

Amount of supercast: Number of dominated zeros

| 1 1 0 0 1 0 1 1 | 1 1 1 0 1 1 1 1 |
| 0 1 1 0 0 1 0 1 | 1 1 0 0 1 0 1 1 |
| 1 0 0 1 0 0 1 0 | 0 1 1 0 0 1 0 1 |
| 0 1 0 1 0 1 0 1 | 1 0 0 1 0 0 1 0 |

Assigned set:
Destination set assigned for cluster: OR/union of DSs in cluster

| 1 1 0 0 1 0 1 1 | 1 1 0 0 1 0 1 1 |
| 0 1 1 0 0 1 0 1 | 0 1 1 0 0 1 0 1 |
| 1 0 0 1 0 0 1 0 | 1 0 0 1 0 0 1 0 |
| 0 1 0 1 0 1 0 1 | 0 1 0 1 0 1 0 1 |

One cluster cost & cost savings of a cluster

| 1 1 0 0 1 0 1 1 |
| 0 1 1 0 0 1 0 1 |
| 1 0 0 1 0 0 1 0 |
| 0 1 0 1 0 1 0 1 |

One cluster cost: total number of zeros in all destination sets (otherwise eliminate column of zeros)
Additional (more than one) clusters should yield cost savings
### One cluster cost & cost savings of a cluster

<table>
<thead>
<tr>
<th>1 1 0 0 1 0 1 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 1 0 0 1 0 1</td>
</tr>
<tr>
<td>1 0 0 1 0 0 1 0</td>
</tr>
<tr>
<td>0 1 0 1 0 1 0 1</td>
</tr>
</tbody>
</table>

No cost savings

One cluster cost: total number of zeros in all destination sets

### One cluster cost & cost savings of a cluster

<table>
<thead>
<tr>
<th>1 1 0 0 1 0 1 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 1 0 0 1 0 1</td>
</tr>
<tr>
<td>1 0 0 1 0 0 1 0</td>
</tr>
<tr>
<td>0 1 0 1 0 1 0 1</td>
</tr>
</tbody>
</table>

One cluster cost: number of zeros in Destinations Matrix (otherwise eliminate column of zeros)

Cluster with column of zeros yields savings
Cost savings = columns of zeros
Talk Outline

- Introduction/Motivation
- Problem Formulation
- Computational Complexity
- Heuristic Algorithms
- Experimental Results
- Conclusions

Minimum Cumulative Supercast is NP-Complete

Maximum Edge Biclique [Peeters 2003]:

Given a bipartite graph, find complete subgraph with maximum edge count

Complete (sub)-graph has all ones adjacency matrix
Reduction for NP-hardness of MCS

Given bipartite graph
\( G = (V_1, V_2, E \subseteq (V_1 \times V_2) ) \),
positive integer \( K' \);
Is there a biclique with \( \geq K' \) Edges?

\[
\begin{align*}
|V_1| & \quad |V_2| \\
\text{Adjacency matrix} & \\
\end{align*}
\]

\[
\begin{align*}
N &= |V_1| + 1 \\
n &= |V_2| \\
M &= 2 \text{ (want two clustering)} \\
K &= |E| + 1 - K' \\
\end{align*}
\]

Cluster without extra row identifies required biclique

Yes-instance of MEB \( \Rightarrow \) Yes-instance of MCS

Given bipartite graph
\( G = (V_1, V_2, E \subseteq (V_1 \times V_2) ) \),
positive integer \( K' \);
Is there a biclique with \( \geq K' \) Edges?

\[
\begin{align*}
|V_1| & \quad |V_2| \\
\text{Adjacency matrix} & \\
\end{align*}
\]

\[
\begin{align*}
N &= |V_1| + 1 \\
n &= |V_2| \\
M &= 2 \text{ (want two clustering)} \\
K &= |E| + 1 - K' \\
\end{align*}
\]

Cluster without extra row identifies required biclique
Hardness of Approximation

Theorem: It is RSAT-hard to approximate MCS to a factor better than 31/28.

Talk Outline

- Introduction/Motivation
- Problem Formulation
- Computational Complexity
- Heuristic Algorithms
- Experimental Results
- Conclusions
Algorithm types: online and offline

**Online algorithm is necessary**

Switch Fabric provides extra copy of address to destinations table:
Split up label to destinations table: online and offline section

![Diagram showing online and offline sections with spawn offline algorithm](image)

**Online greedy row clustering**

Pick the cluster that results in least cost increase
Two components to cost increase
O(Mn) time to decide cluster choice

![Online greedy row clustering diagram](image)
Offline greedy row clustering: two greedy alg.

Can we discover well clustered inputs? Random seeding of M clusters will miss 1/e clusters
Choose target of 4M clusters:
  Distribute N destination sets among 4M clusters
  Greedily pick best pair-wise merge to reduce from 4M to M clusters
  $O(NMn)$

Column clustering

| 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |

Tradeoff quality for run-time
Split columns in two clusters: n/2 columns in each
Target number of clusters per instance: $M^{1/2}$
Online complexity: $O(M^{1/2}n)$
Offline complexity: $O(NM^{1/2}n)$
### constant time algorithms

\[ m = \log M \text{ ; number of bits to represent cluster number} \]

\[ O(m): \text{practically constant time} \]

**Compute m bit signature (cluster number)**

**Signature: a proximity measure**
- Destination sets that don't differ by much
  will have same signature

---

### Subset Intersection Signature (SIS)

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

**Pick m random subsets of \{1, 2, ..., n\}**

Each of m bits of signature is determined based on intersection with corresponding subset

Choice of subset can exploit knowledge of input distribution
Column classification: special case of SIS

If each of the subsets is distinct and a singleton, amounts to picking m columns and classifying based on these

Easy method to cluster

Random Permutation Signature (RPS)

Pick m random permutations of \{1, 2, ..., n\}
Subject the DS to the m permutations
Let \(i_1, i_2, ..., i_m\) be the index of the first zero (or one) location in the permuted DS
Form m bit signature:
Can choose more bits from one index and use fewer indices or fewer bits from each and more indices
Example: if \(n=256\) and \(m=10\); Each min-index is 8 bits wide.
## Which algorithm to choose?

Choose the best algorithm (online) the connection rate can afford

Resort to more expensive algorithm offline

## Talk Outline

- Introduction/Motivation
- Problem Formulation
- Computational Complexity
- Heuristic Algorithms
- Experimental Results
- Conclusions
### Classes of inputs

1. **Universe of inputs**: $N = 2^n$

2. **Bernoulli IID inputs**: each of $n$ destinations in a destination set turned on independently with probability $p$

3. **Pre-clustered inputs**

### What we measure

**Average Bandwidth Waste**: $\frac{\text{total-cost}}{Nn}$

What fraction of linecards ($n$) receive supercast, i.e., packets to drop

One-cluster average bandwidth waste is fraction of zeros in a destination set:
- **Bernoulli IID inputs**: $1-p$
- **Universe of inputs**: $0.5$
For the universe of inputs, on average a destination set has \(\frac{n}{2}\) zeros.

Column classification by \(m\) columns achieves:
- Average BW waste: \(\frac{(n-m)}{2n}\)
- Average BW savings: \(\frac{m}{2n}\)

Theorem: No \(2^m\)-clustering for universe of inputs can achieve average BW savings greater than \(\frac{m}{n}\).

### Universe of Inputs

<table>
<thead>
<tr>
<th>(n)</th>
<th>(m)</th>
<th>greedy</th>
<th>Column-classification: (\frac{(n-m)}{2n})</th>
<th>RPS</th>
<th>2-Column Clusters</th>
<th>SIS</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>4</td>
<td>0.342</td>
<td>0.357</td>
<td>0.464</td>
<td>0.353</td>
<td>0.426</td>
</tr>
<tr>
<td>16</td>
<td>8</td>
<td>0.206</td>
<td>0.250</td>
<td>0.453</td>
<td>0.230</td>
<td>0.334</td>
</tr>
<tr>
<td>20</td>
<td>10</td>
<td>0.200</td>
<td>0.250</td>
<td>0.456</td>
<td>0.224</td>
<td>0.372</td>
</tr>
</tbody>
</table>

SIS: Pick subsets with \(1/p\) (here \(p = \frac{1}{2}\)) ones.
Random (Bernoulli IID) inputs

Theorem: Given N destination sets over n destinations with each destination chosen independently and with probability p and M ( = 2^m )possible clusters, then:

Any clustering can achieve no more cost savings over the one cluster cost than
O(1/p(N.logM + Mn))

There is a clustering which achieves a cost savings of
Ω( (1-p)(N.logM + Mn))

Random inputs: bounds on BW savings

<table>
<thead>
<tr>
<th>Total BW savings</th>
<th>Average BW savings</th>
</tr>
</thead>
<tbody>
<tr>
<td>O(1/p(N.logM + Mn))</td>
<td>O(1/p(logM/n + M/N))</td>
</tr>
<tr>
<td>Ω( (1-p)(N.logM + Mn))</td>
<td>Ω( (1-p)(logM/n + M/N))</td>
</tr>
</tbody>
</table>

When p is a constant close to 0 or 1 (e.g., 0.5): bounds match (upto constant factors)

When number of destination-sets per cluster (N/M) is high, column-classification (first term) dominates
Random Inputs: Table of results (Avg BW waste)

<table>
<thead>
<tr>
<th>Prob. p</th>
<th>1 - p</th>
<th>greedyy</th>
<th>2 column clusters</th>
<th>RPS</th>
<th>SIS</th>
<th>Col. Classif:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.9</td>
<td>0.241</td>
<td>0.325</td>
<td>0.846</td>
<td>0.441</td>
<td>0.713</td>
</tr>
<tr>
<td>0.3</td>
<td>0.7</td>
<td>0.427</td>
<td>0.471</td>
<td>0.688</td>
<td>0.563</td>
<td>0.569</td>
</tr>
<tr>
<td>0.5</td>
<td>0.5</td>
<td>0.354</td>
<td>0.379</td>
<td>0.499</td>
<td>0.427</td>
<td>0.406</td>
</tr>
<tr>
<td>0.7</td>
<td>0.3</td>
<td>0.220</td>
<td>0.247</td>
<td>0.300</td>
<td>0.269</td>
<td>0.244</td>
</tr>
<tr>
<td>0.9</td>
<td>0.1</td>
<td>0.061</td>
<td>0.070</td>
<td>0.100</td>
<td>0.095</td>
<td>0.081</td>
</tr>
</tbody>
</table>

N=100,000 (DSs); n=64 (linecards); m=12 (4096 clusters)
Column classification: 1 - p – (1-p)m/n

Random inputs: Plot of Avg BW waste with greedy algorithm vs prob p

N/M at least 300,000/16000; savings proportional to m/n (waste to n/m)
Random Inputs: plot of ratio of avg BW waste of greedy and col. Classific. Vs probability p

![Plot of ratio of avg BW waste of greedy and col. Classific. Vs probability p]

Preclustered inputs: comparison of Avg BW waste from different methods

<table>
<thead>
<tr>
<th>p₁</th>
<th>p₂</th>
<th>Expt ABW</th>
<th>Grdy M-cl</th>
<th>G-4M</th>
<th>G-4M-M</th>
<th>2 Col Clusters</th>
<th>RPS</th>
<th>SIS</th>
<th>1-p₁*p₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>30</td>
<td>0.206</td>
<td>0.395</td>
<td>0.232</td>
<td>0.273</td>
<td>0.546</td>
<td>0.828</td>
<td>0.615</td>
<td>0.91</td>
</tr>
<tr>
<td>30</td>
<td>70</td>
<td>0.088</td>
<td>0.171</td>
<td>0.082</td>
<td>0.085</td>
<td>0.601</td>
<td>0.743</td>
<td>0.656</td>
<td>0.79</td>
</tr>
<tr>
<td>50</td>
<td>50</td>
<td>0.248</td>
<td>0.433</td>
<td>0.248</td>
<td>0.251</td>
<td>0.604</td>
<td>0.711</td>
<td>0.655</td>
<td>0.75</td>
</tr>
<tr>
<td>50</td>
<td>70</td>
<td>0.149</td>
<td>0.262</td>
<td>0.144</td>
<td>0.145</td>
<td>0.556</td>
<td>0.627</td>
<td>0.603</td>
<td>0.65</td>
</tr>
<tr>
<td>70</td>
<td>70</td>
<td>0.210</td>
<td>0.329</td>
<td>0.207</td>
<td>0.208</td>
<td>0.451</td>
<td>0.493</td>
<td>0.488</td>
<td>0.51</td>
</tr>
</tbody>
</table>

N=100,000 (Destination sets); n=64 (linecards); m=6 (64 clusters)
## Summary

Cluster DSs to fit into address to destinations table in switch fabric to minimize supercast

Proved hard to solve exactly and approximately
Proposed algorithms of varying run-time/quality:
  - greedy row clustering
  - column clustering
  - constant time methods: RPS & SIS (column classification)
Derived “tight” bounds for savings for random inputs

## Previous Work

Extended Version of paper:

With proof of tight bound on bandwidth savings for random inputs &

Greedy reduction of 4M to M clusters to discover pre-clustered inputs, etc,

Email authors or visit URL in paper