Hierarchical In-Network Compression via Importance Sampling

@IEEE ICDE'15

Arlei Silva[1], Petko Bogdanov[2], Ambuj K. Singh[1]

[1]Computer Science Department – University of California, Santa Barbara, CA.
[2]Computer Science Department – University at Albany, SUNY, NY.

Motivation: Traffic Monitoring

Figure : Real-time traffic in California.

2,000 cameras $\approx$ 0.5T measurements/year

Challenges: Storage, processing, communication

How can we represent traffic conditions in a compact way and still answer queries wrt these conditions with small error?

General Problem: In-Network Attribute Compression

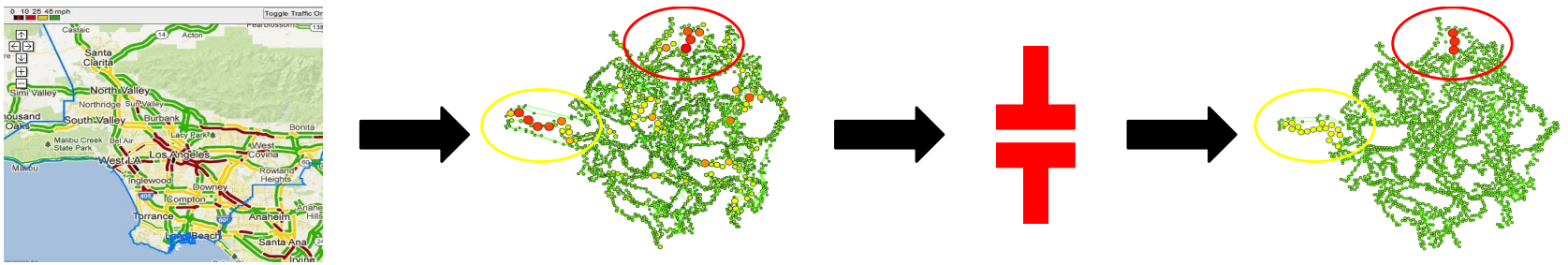


Figure : In-Network attribute compression.

- **Lossy compression of node/edge attribute values**
- **Fixed budget (e.g. bytes)**
- **Network structure is known**

In-Network Attribute Compression

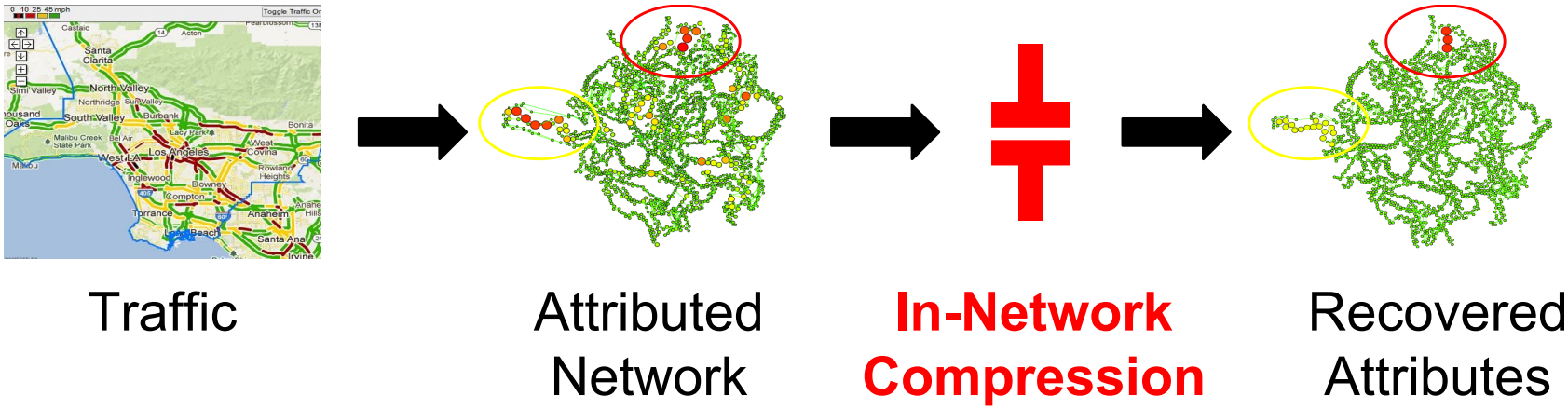


Figure : In-Network attribute compression.

Applications:

- Expertise in professional/business networks
- Gene expression in PPI networks
- User opinions in social networks
- ...

Assumption: Structure carries information about attributes

In-Network Attribute Compression

Input:

1. Attributed graph $G(V, E, W)$ where $W : V \to \mathbb{R}$
2. Fixed budget of b bytes

Output: Lossy compression $\Gamma(V)$

- $|\Gamma|$ is b bytes
- $\Gamma : V \to \mathbb{R}$

Objective function: $\min SSE(\Gamma(V)) = \sum_{v \in V} (W(v) - \Gamma(v))^2$

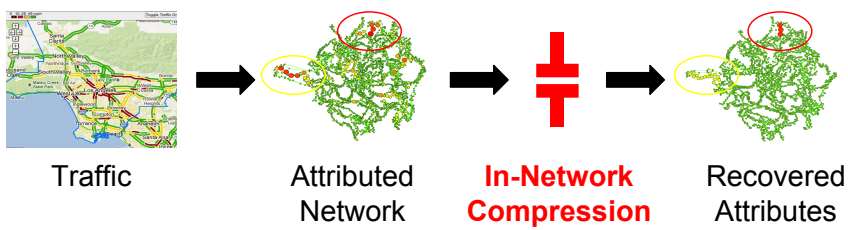


Figure : In-Network attribute compression.

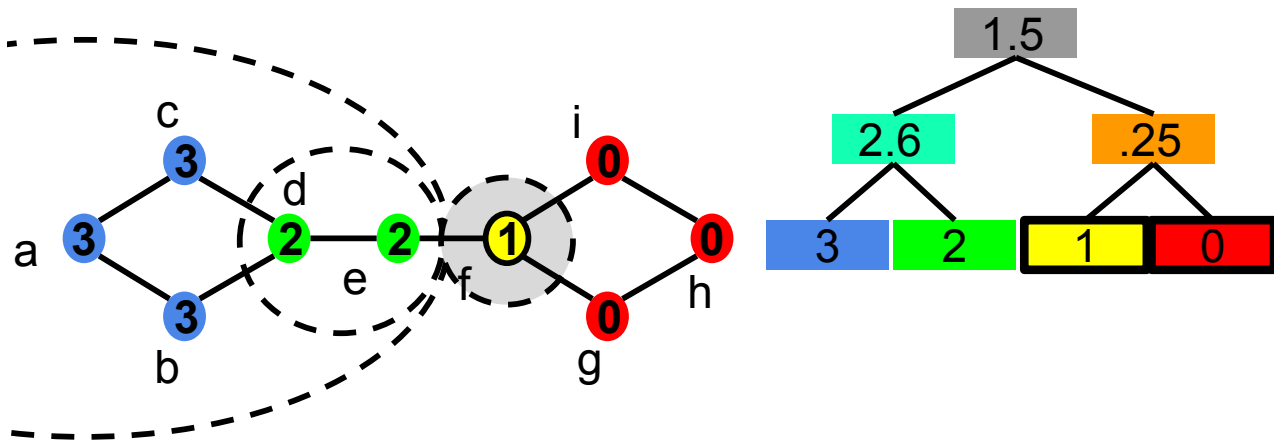
Our Solution: Slice Tree

Slice s(c,r,X):

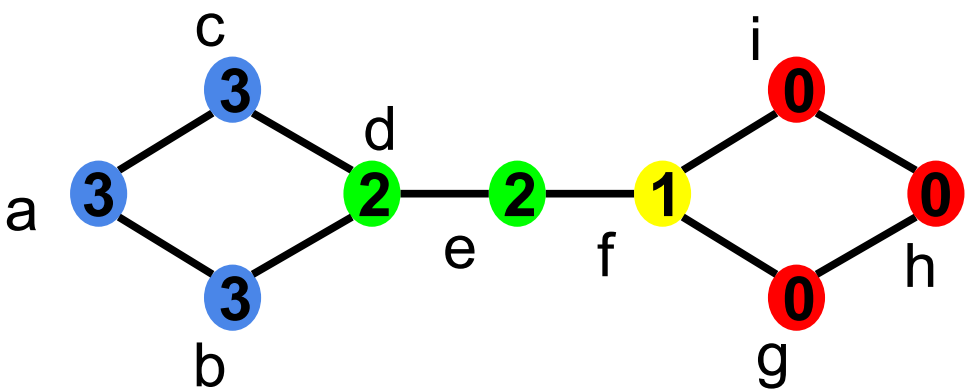
- Partitions X into sets P and $X \setminus P$
- $P = \{u \in X | d(c, u) \leq r\}$

Slice Tree ST(G,k):

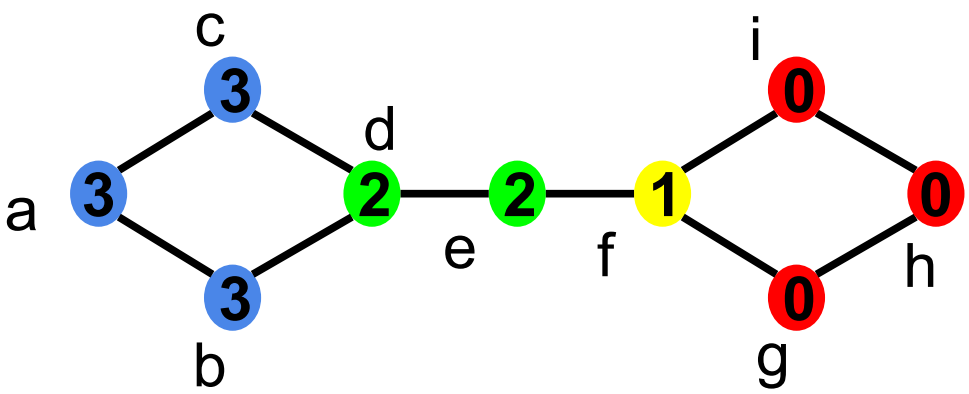
- Binary tree that encodes k recursive slices in G
- $k \Longleftrightarrow$ budget
- $\Gamma(v) =$ average value of its partition
- Finding optimal ST is NP-hard



Slice Tree

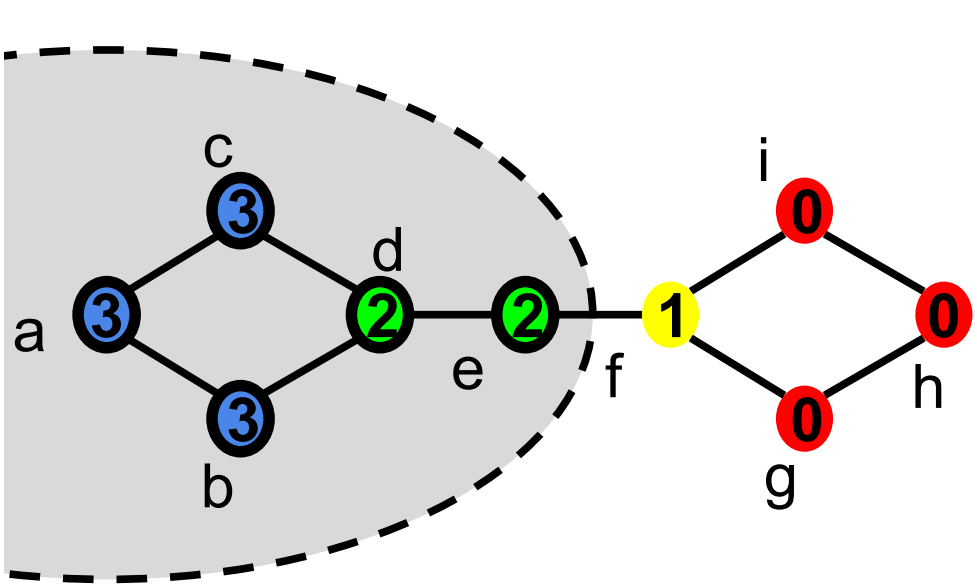


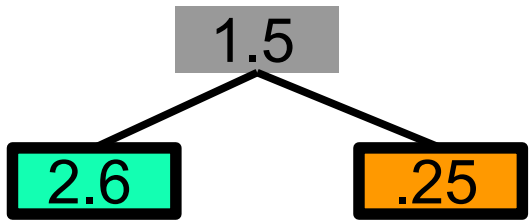
Slice Tree



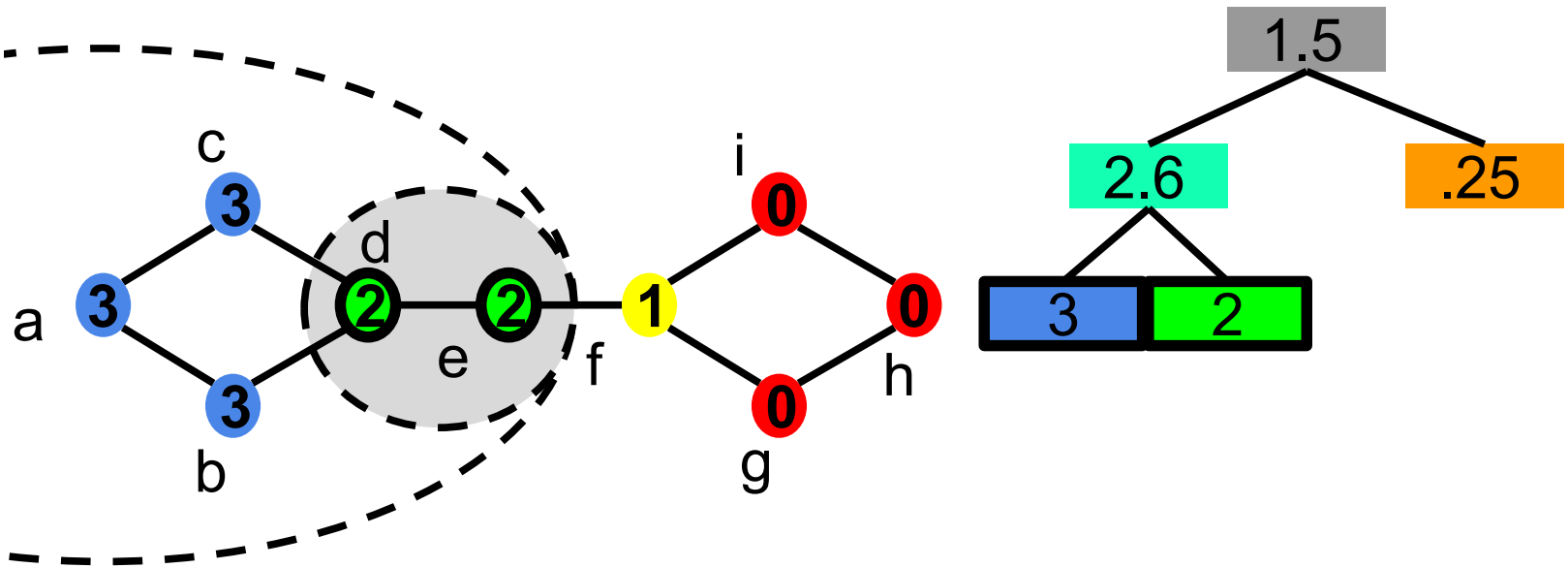
1.5

Slice Tree

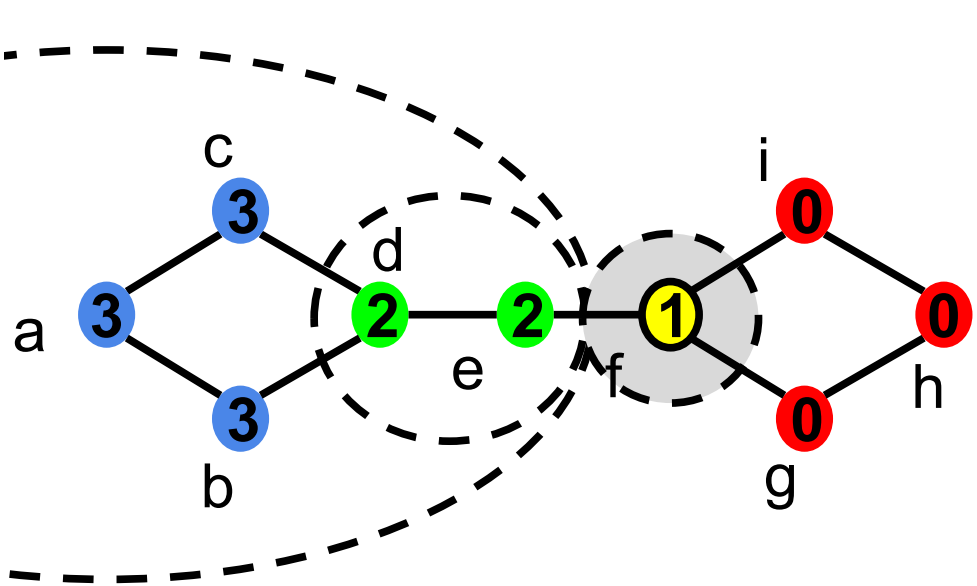


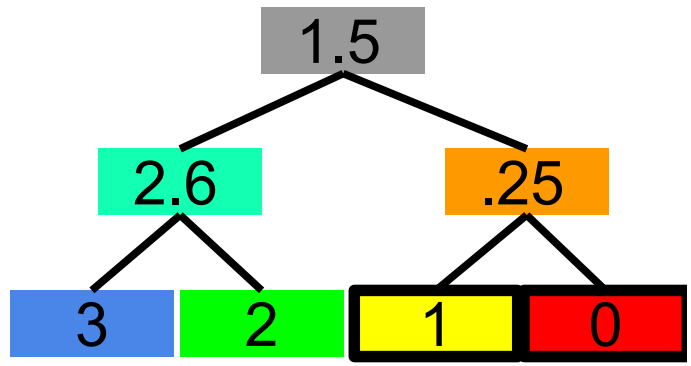


Slice Tree



Slice Tree





Slice Tree vs. Existing Work

Value-sensitive vs. fixed basis vectors

- Motivation: More compression/less error
- Challenge I: Basis vectors become part of the data
  - Alleviated with compact representation
- Challenge II: combinatorial space of basis
  - Alleviated with heuristic + importance sampling

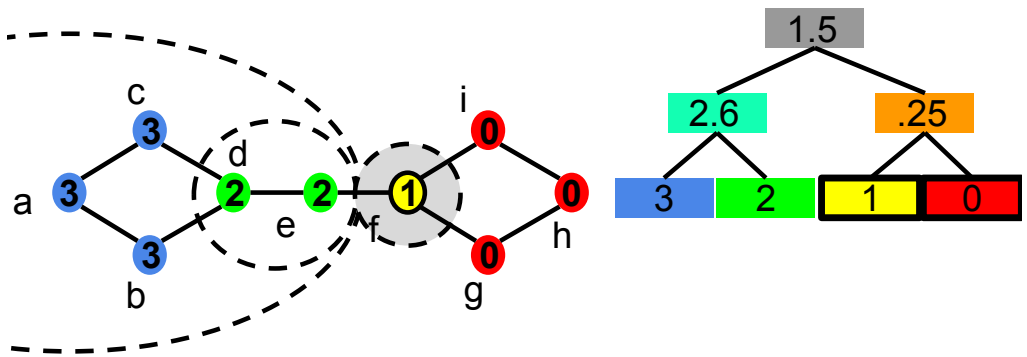


Figure : Slice tree.

Error Reduction of a Slice

$$SSE(Y) = \sum_{v \in Y} (w(v) - \mu(Y))^2$$

$$\begin{aligned} \phi(s) &= SSE(X) - SSE(P) - SSE(X \setminus P) \\ &= (\mu(P) - \mu(X))^2 \frac{||P||||X||}{||X \setminus P||} \end{aligned}$$

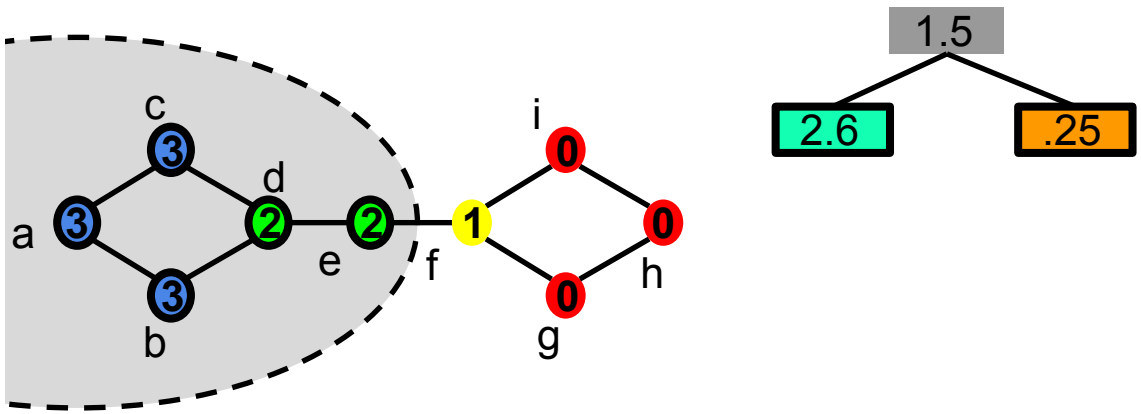


Figure : $\phi(s) = (1.5 - 2.6)^2 6 \times 9/4 = 13.6$

Greedy Algorithm

Require: Network G, budget k
Ensure: ST
1: Slice $ST \leftarrow$ empty Slice Tree
2: **while** Number of slices less than k **do**
3: Add best slice $s(c, r)$ to ST
4: **end while**

Time complexity: $O(k.V.E)$

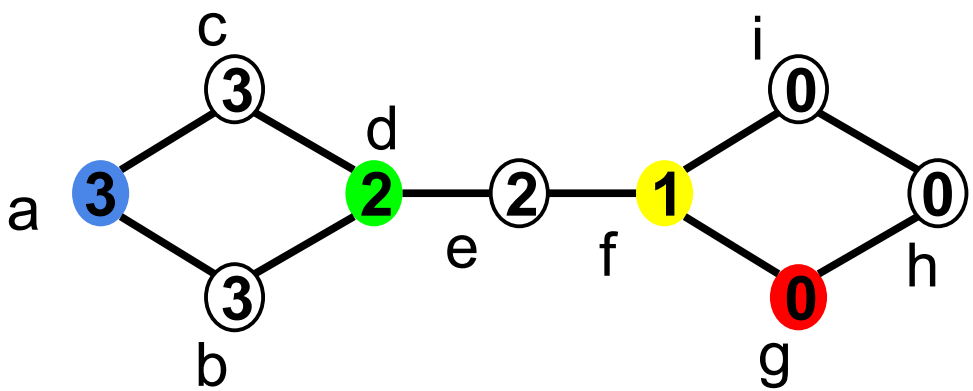
- V possible centers
- $O(E)$ cost to check all slices with given center
- k slices

Do we need to check all values to find a good slice?

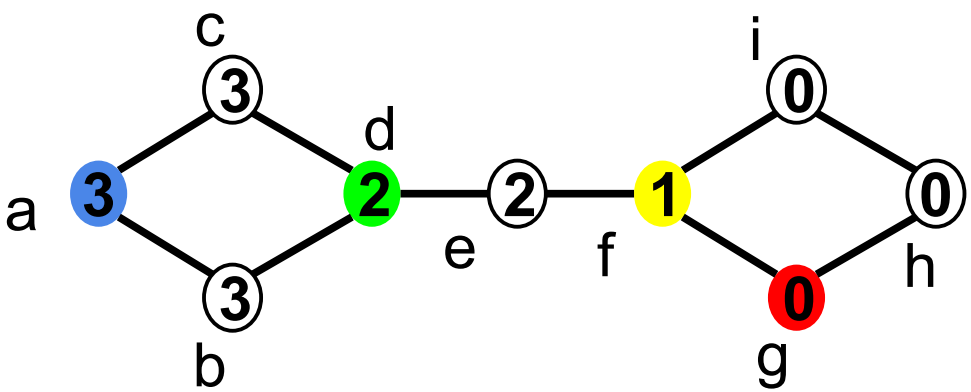
How can we sample values/vertices?

- Uniform: $p(v) = \frac{1}{||X||}$
- Importance: $p(v) = \frac{|w(v) - \mu(X)|}{\lambda}$

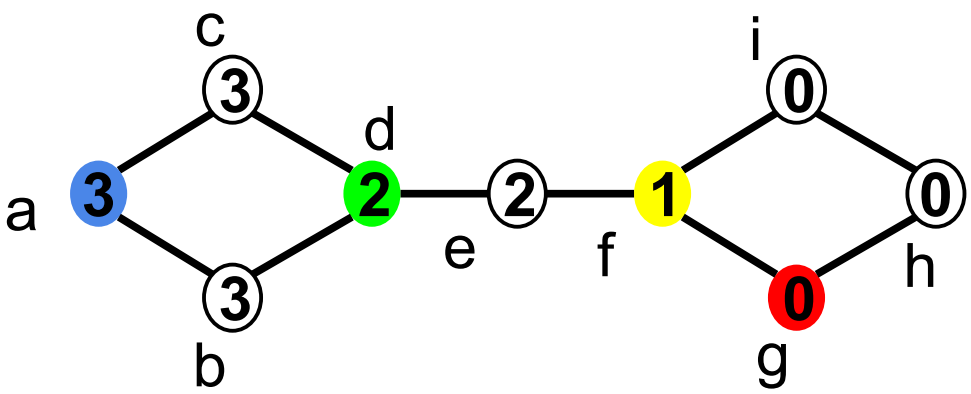
Importance sampling: Sampling process biased towards values that are more relevant for computing an estimate



Sampling

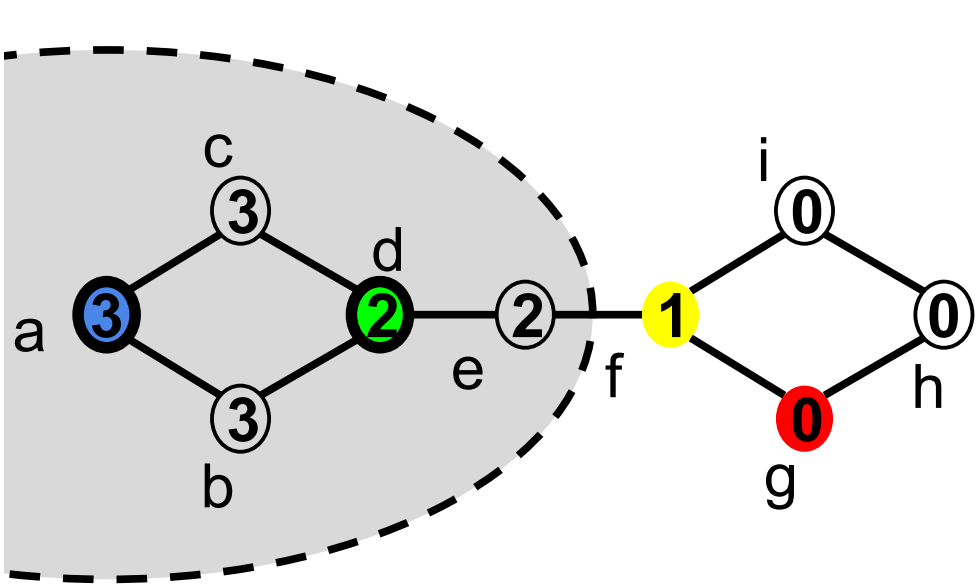


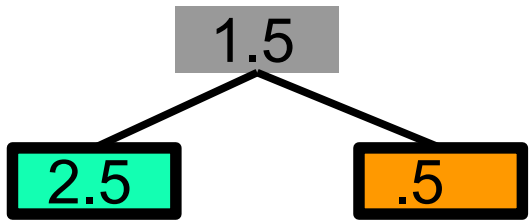
Sampling



1.5

Sampling





Importance Sampling

Importance sample B:

$$p(v) = \frac{1}{\lambda} \boxed{|w(v) - \mu(X)|}$$

where $\lambda = \sum_v |w(v) - \mu(X)|$

Error reduction:

$$\phi(s) = (\boxed{\mu(P) - \mu(X)})^2 \frac{||P||||X||}{||X \setminus P||}$$

Importance Sampling

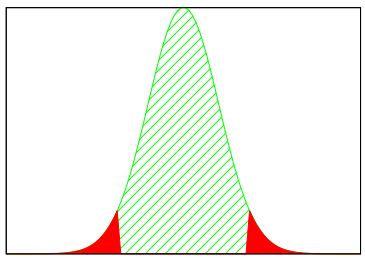
Importance sample B:

$$p(v) = \frac{1}{\lambda} \boxed{|w(v) - \mu(X)|}$$

where $\lambda = \sum_v |w(v) - \mu(X)|$

Error reduction:

$$\phi(s) = (\boxed{\mu(P) - \mu(X)})^2 \frac{||P||||X||}{||X \setminus P||}$$



Importance Sampling

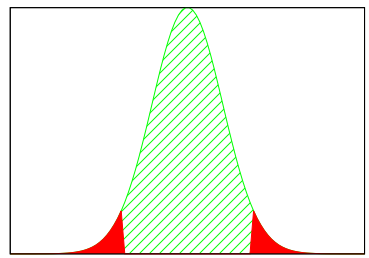
Importance sample B:

$$p(v) = \frac{1}{\lambda}\boxed{|w(v) - \mu(X)|}$$

where $\lambda = \sum_v |w(v) - \mu(X)|$

Error reduction:

$$\phi(s) = (\boxed{\mu(P) - \mu(X)})^2 \frac{||P||||X||}{||X \setminus P||}$$



Unbiasing ($\mathbb{E}[\mu(B_P)] = \mu(P)$):

$$\mu(B_P) = \frac{1}{\Psi(B_P)} \sum_{v \in B_P} \frac{\lambda}{|w(v) - \mu(X)|} w(v)$$

where $B_P = B \cap P$ and $\Psi(B_P) = \sum_{v \in B_P} \frac{\lambda}{|w(v) - \mu(X)|}$

Pruning Candidate Slices with Sampling

$\phi(s)$ **can be bounded by** **counting samples in** P **:**

$$Pr[\phi(s) < \frac{(p' + \epsilon)^2 \lambda^2}{||P||} \frac{||X||}{||X \setminus P||}] > 1 - \delta$$

where $p' = \frac{||P \cap B||}{||B||}$ and $\epsilon = \sqrt{\frac{-1}{2||B||} \log(\delta)}$.

Time complexity $\approx O(k.B.E)$

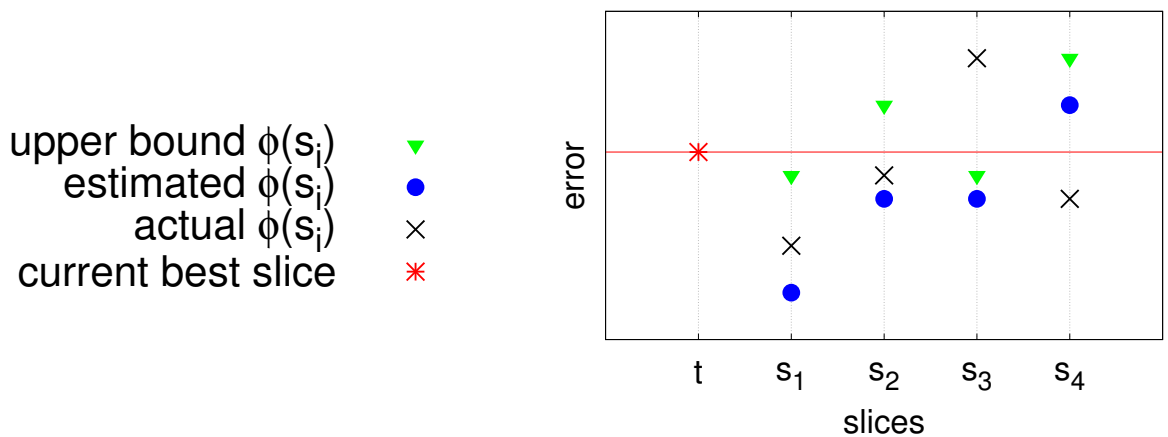


Figure : s_1 and s_3 are be pruned.

Pruning Candidate Slices with Sampling

$\phi(s)$ **can be bounded by** **counting samples in** P **:**

$$Pr[\phi(s) < \frac{(p' + \epsilon)^2 \lambda^2}{||P||} \frac{||X||}{||X \setminus P||}] > 1 - \delta$$

where $p' = \frac{||P \cap B||}{||B||}$ and $\epsilon = \sqrt{\frac{-1}{2||B||} \log(\delta)}$.

Time complexity $\approx O(k.B.E)$

From sample

- upper bound $\phi(s_i)$
- estimated $\phi(s_i)$
- actual $\phi(s_i)$
- current best slice

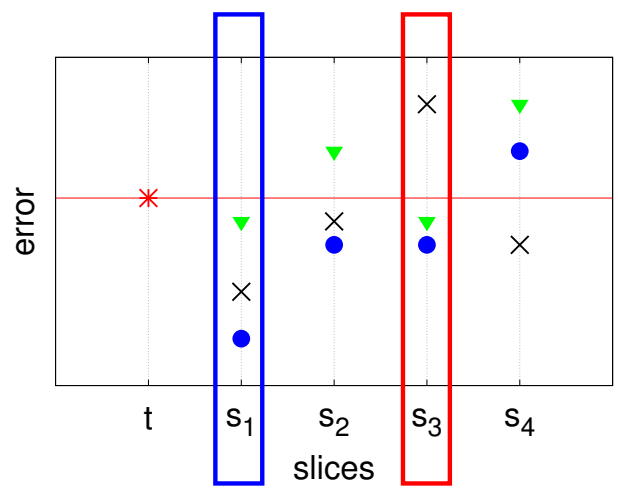


Figure : s_1 and s_3 are be pruned.

Experimental Results

name	vertex	edge	value	$\|\|V\|\|$	$\|\|E\|\|$
Traffic	sensor	highway	speed	2k	6k
Human	gene	interaction	expression	4k	9k
DBLP	author	collaboration	#papers	1.3m	5.2m
Twitter	user	following	sentiment	.7m	19.2m

Table : Real datasets.

Error reduction: SSE(V) - SSE($\Gamma(V)$)

Evaluation:

1. Importance sampling
2. Compression results
3. Case studies

Importance Sampling: DBLP

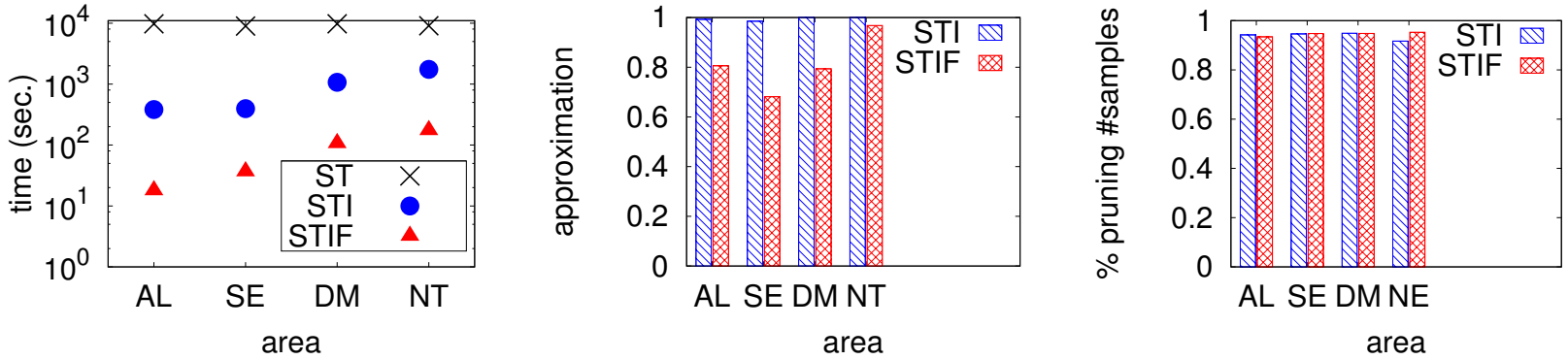


Figure : Algorithms, Soft. Engineering, Data Management, Networks

Importance sampling enables the efficient discovery of slices

- Up to 500x faster ST computation
- Small error with guarantees

Compression Results: Traffic and DBLP

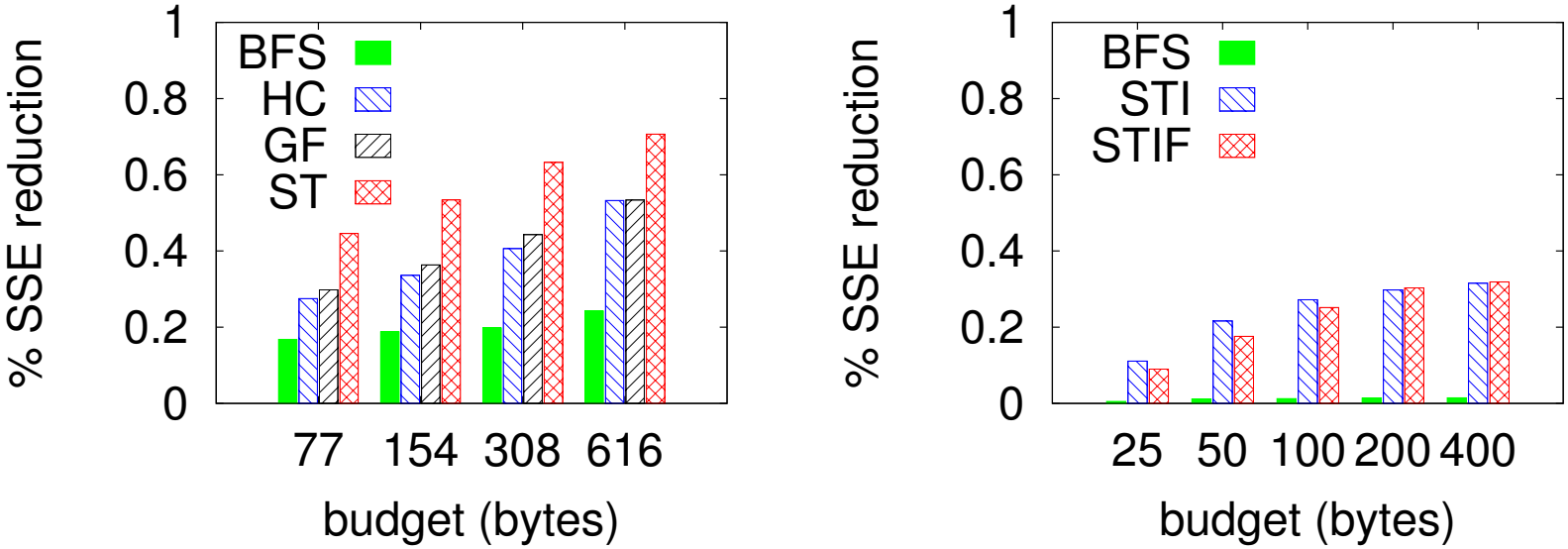


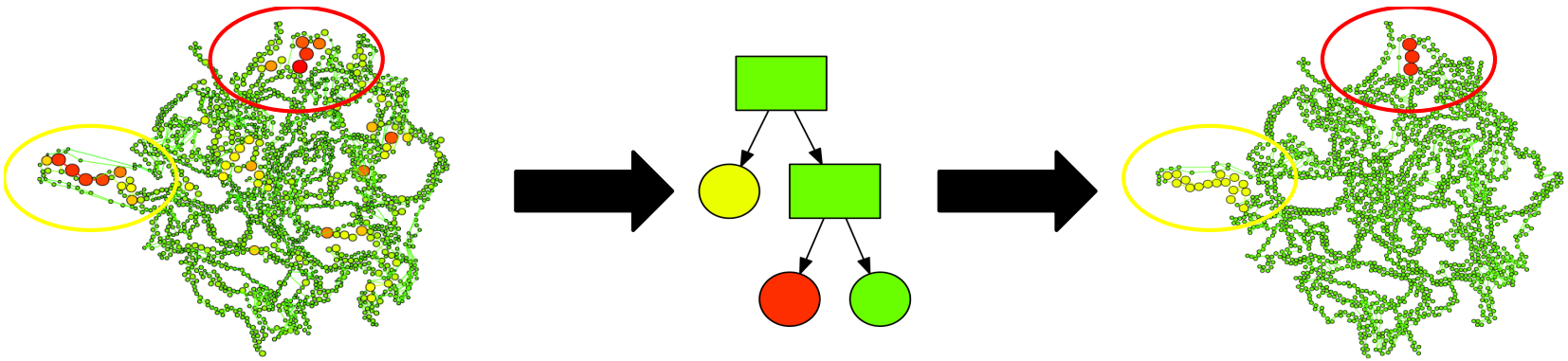
Figure : Traffic (1) and DBLP (2)

ST achieves up to a 2-fold gains over the best baseline (GF)

Baselines:

- GF: Sandryhaila and J. Moura. Discrete signal processing on graphs. ICASSP'13.
- HC: Hammond et al. Wavelets on graphs via spectral graph theory. ICML'11.
- BFS: BFS+Wavelets.

Case Studies: Traffic



Traffic **Slice Tree** Recovered Traffic

Figure : Speeds: very-low=10mph, low=45mph, normal=85mph

First three ST slices capture the main congested areas

Concluding Remarks

Slice Tree is an efficient attribute compression strategy that decomposes a network into slices via importance sampling.

Main contributions:

- We introduce ST and show that its computation is NP-hard
- We propose a heuristic and sampling scheme for ST
- We show that ST is effective and captures relevant phenomena (e.g. traffic jams)

Hierarchical In-Network Compression via Importance Sampling

@IEEE ICDE'15

Arlei Silva[1], Petko Bogdanov[2], Ambuj K. Singh[1]

[1]Computer Science Department – University of California, Santa Barbara, CA.
[2]Computer Science Department – University at Albany, SUNY, NY.

Existing Work

Haar trees: Hierarchical clustering + Haar wavelets

- Gavish et al.'10

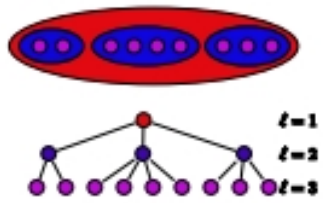


Figure : Haar trees.

Graph Fourier: Projecting values in the eigenspace

- Zarni and Gotsman'00, Sandriyhaila et al.'11, Shuman et al.'13

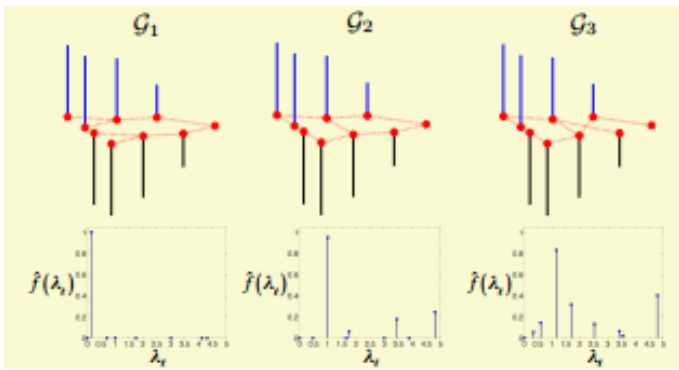


Figure : Graph Fourier.

Slice Tree: Complexity

Optimal ST(G,k) minimizes the error $\sum_{v \in V}(W(v) - \Gamma(v))^2$

Finding an optimal ST is NP-hard

- Reduction from **Vertex Cover**

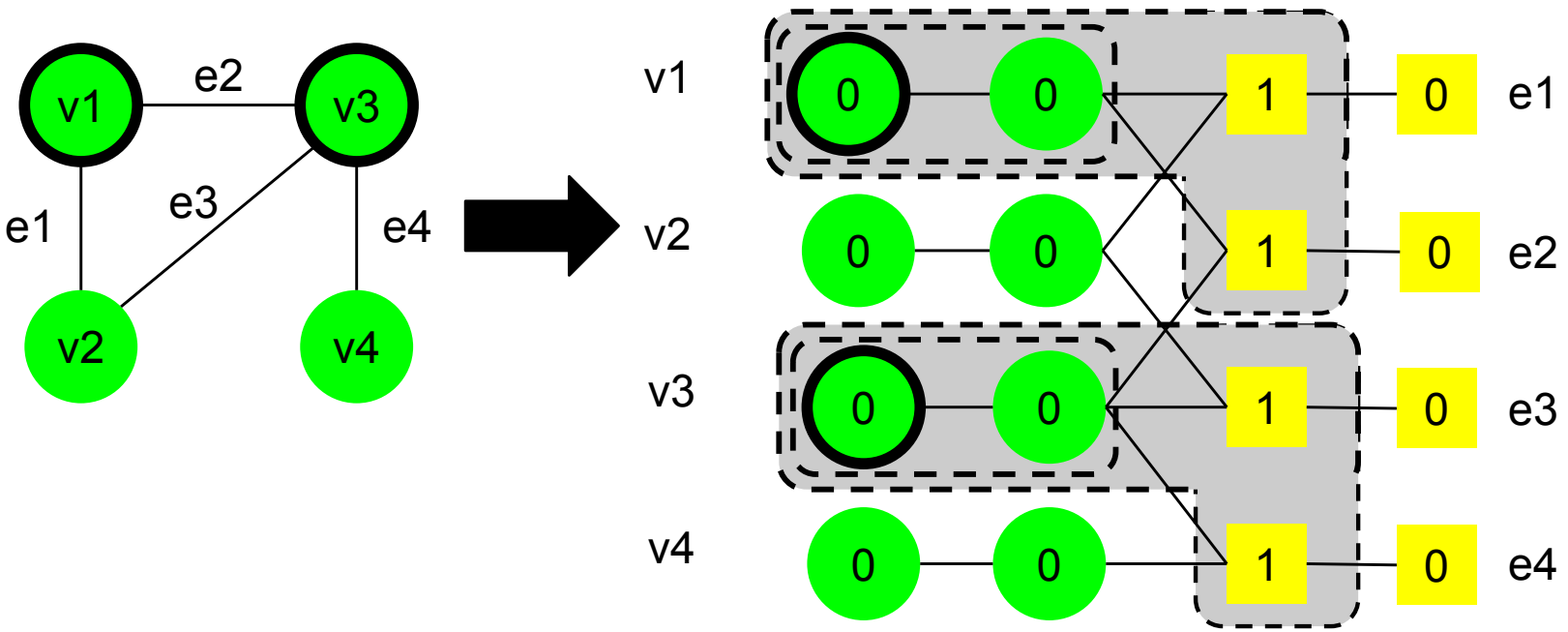
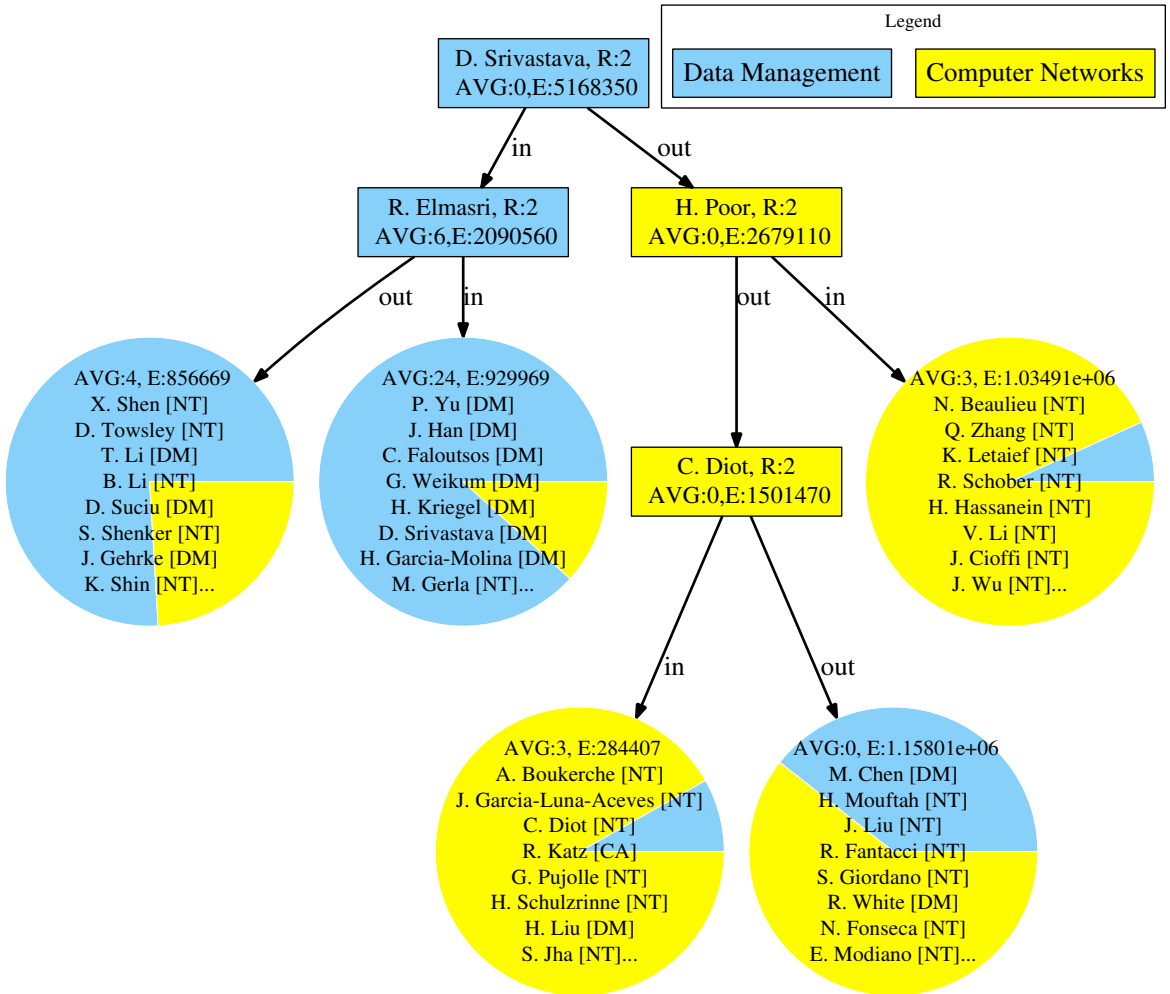


Figure : VC to ST.

Case Studies: DBLP



For mixed research interests, ST separates the 2 communities