Tracking the Evolution of Communities in Dynamic Social Networks

Clique

IDIRO TECHNOLOGIES

Derek Greene
Dónal Doyle
Pádraig Cunningham

ASONAM 2010

Clique: Graph & Network Analysis Cluster
School of Computer Science & Informatics
University College Dublin, Ireland

UCD DUBLIN

sfi
science foundation ireland
fondúireacht eolaíochta éireann

Overview

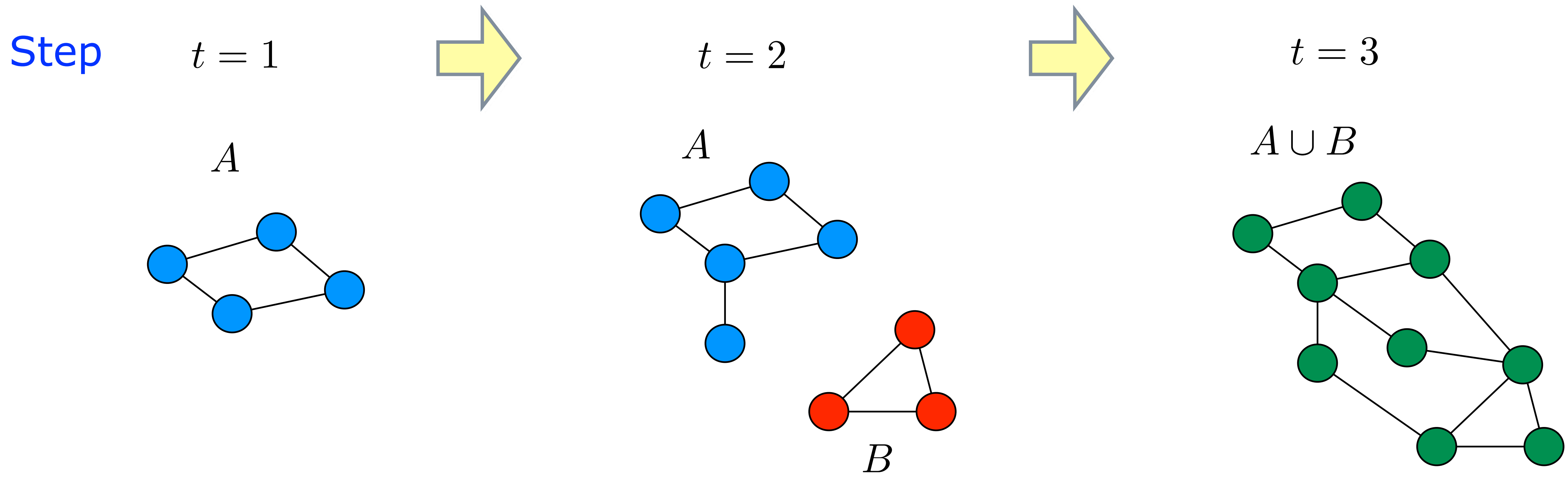
- Community Finding in Dynamic Networks
- Proposed Framework & Algorithm
- Evaluation: Benchmark Graphs
- Evaluation: Mobile Call Graphs
- Conclusions & Future Work

Implementation & Documentation:

https://github.com/derekgreene/dynamic-community

Dynamic Network Analysis

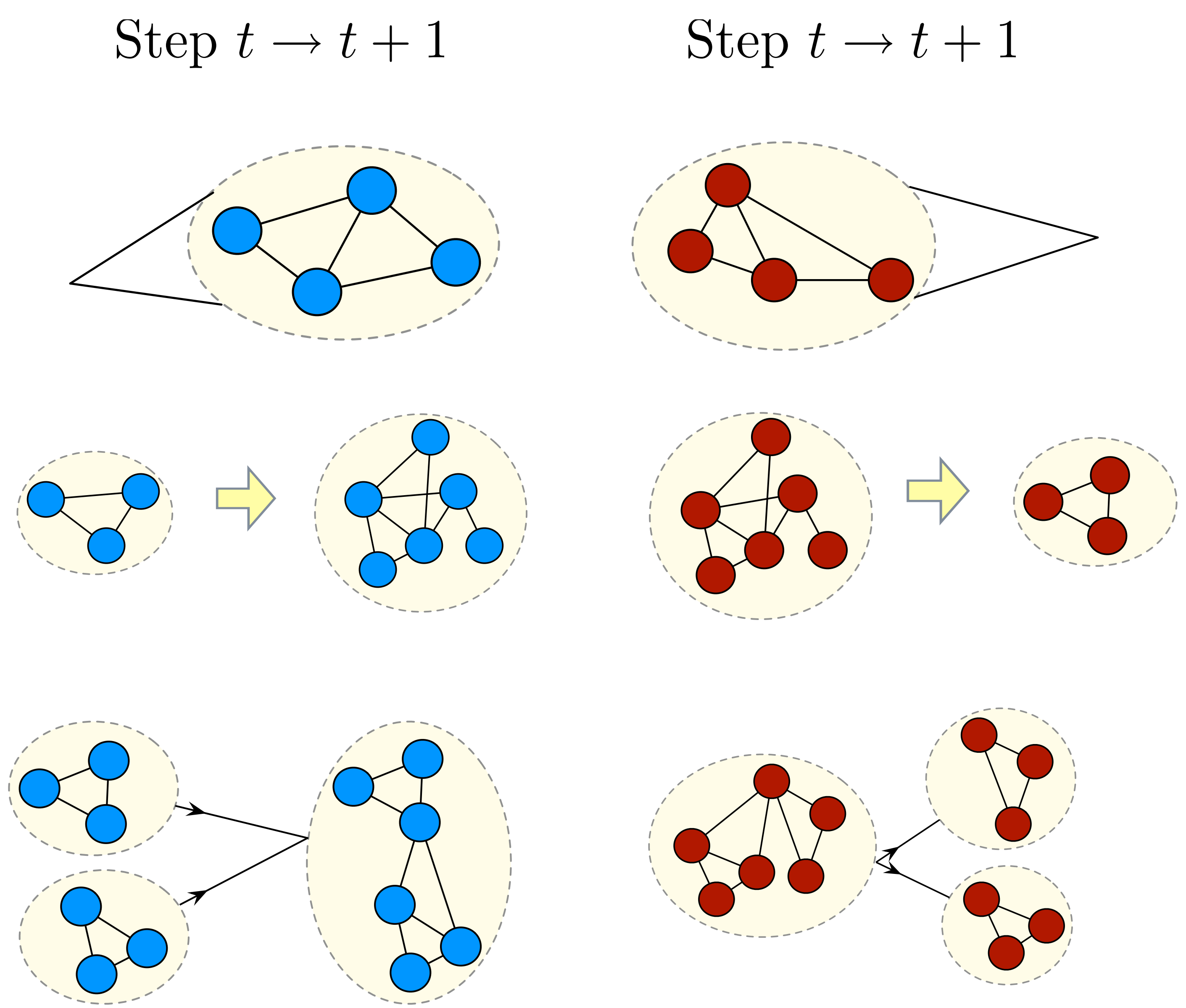
- Want to analyse how communities in a dynamic network form and evolve over time.
- We perform this analysis in an "offline" manner by examining successive snapshots of the network.



Community Evolution

- We can characterise dynamic communities in terms of key life-cycle events (Palla et al, '07; Berger-Wolf et al, '07)

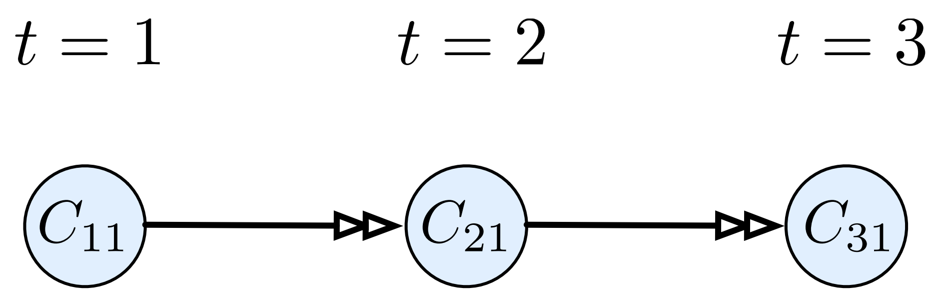
  - Birth & Death of communities
  - Growth & Contraction of communities
  - Merging & Splitting of communities



Framework: Key Concepts

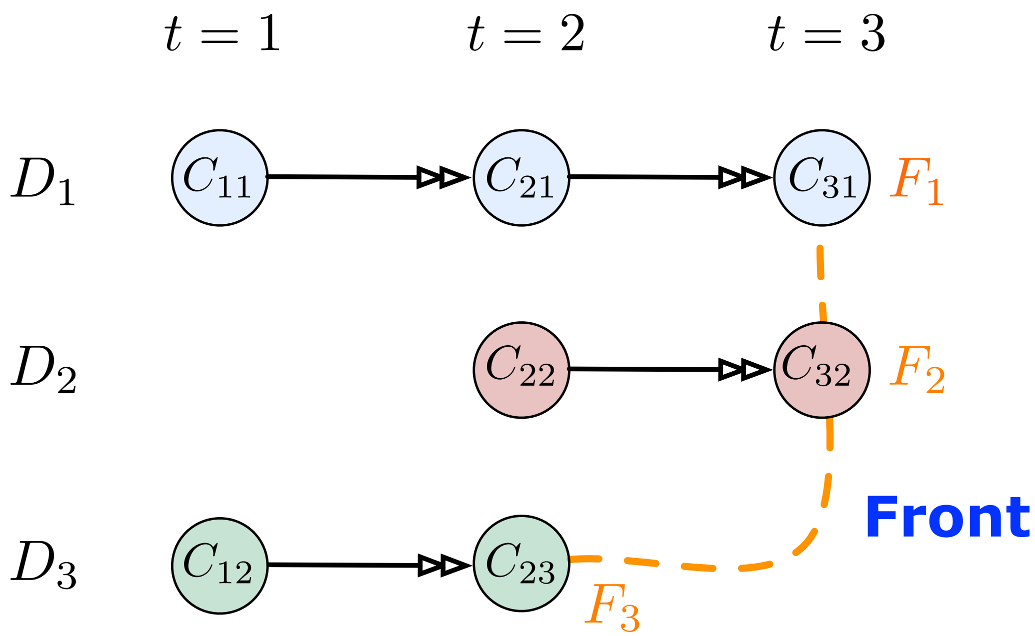
- Time step: Snapshot of the network at a single point in time.
- Step communities: Groups of nodes found by a static community finding algorithm on individual time step graphs.
- Dynamic communities: A chain of related step communities observed over one or more time steps.

➡ Use timeline structures to represent dynamic communities constructed from step communities.



Framework: Examples

Dynamic Communities

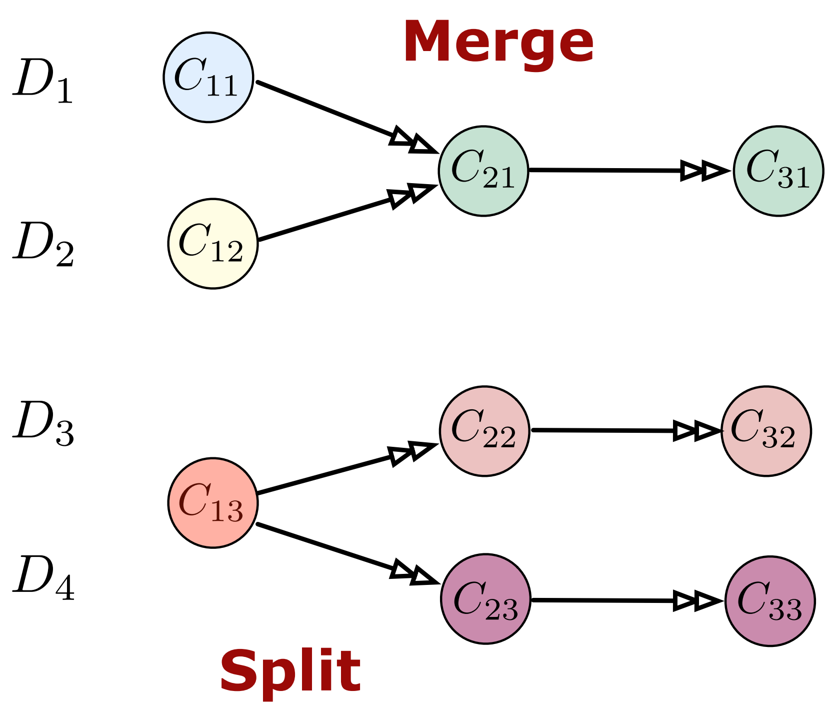


Dynamic Community Timelines

D_1: $\{C_{11}, C_{21}, C_{31}\}$

D_2: $\{C_{22}, C_{32}\}$

D_3: $\{C_{12}, C_{23}\}$



D_1: $\{C_{11}, C_{21}, C_{31}\}$

D_2: $\{C_{12}, C_{21}, C_{31}\}$

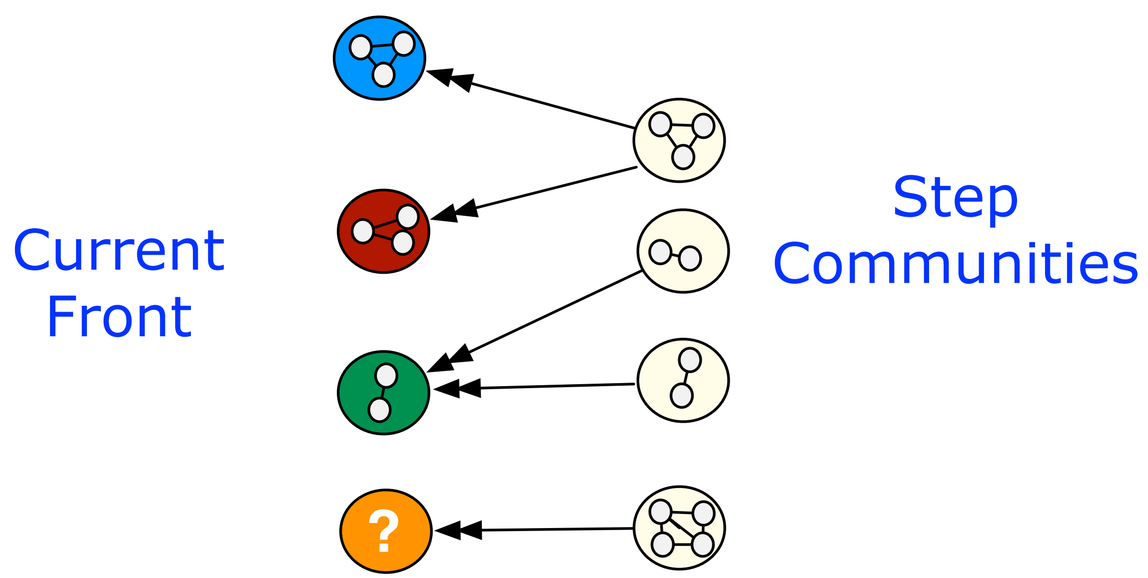
D_3: $\{C_{13}, C_{22}, C_{32}\}$

D_4: $\{C_{13}, C_{23}, C_{33}\}$

Framework: Tracking Communities

Q. How can we match newly arrived step communities with existing dynamic communities?

➡ Many-to-many matching between step communities and dynamic community fronts using Jaccard set similarity.



$$\frac{|C \cap F_i|}{|C \cup F_i|} > \theta$$

Matching Threshold

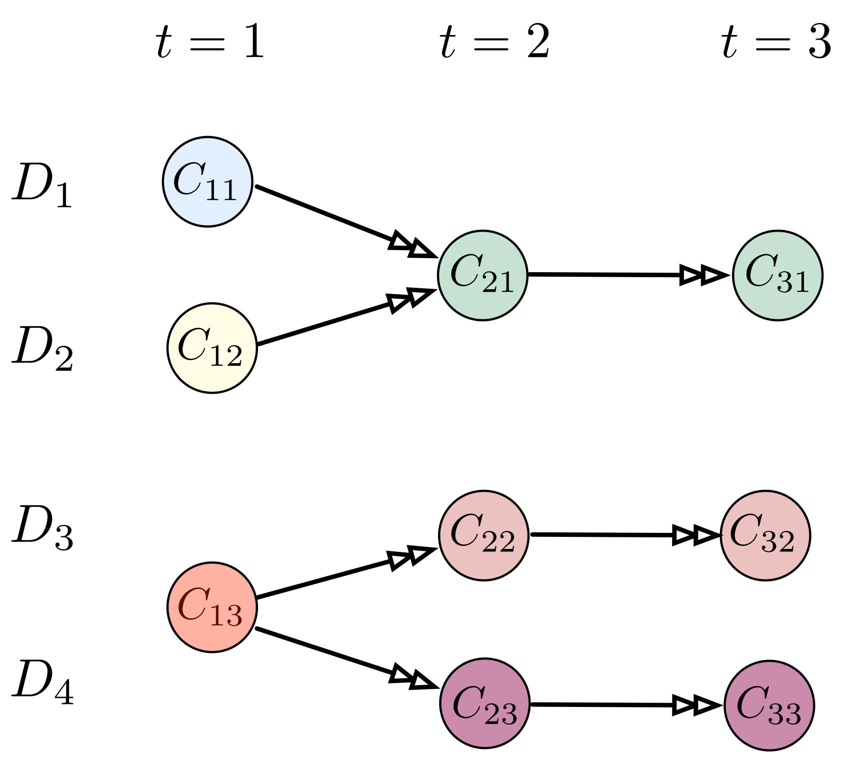
- How do we define a "match"?
  - Small θ : Appropriate for transient communities
  - Large θ : Appropriate for stable communities

Framework: Life-Cycle Events

- Birth: Step community is observed for which there is no matching dynamic community.
- Death: Dynamic community is not observed for at least *d* time steps.
- Merge: Two distinct dynamic communities are matched to a single step community.
- Split: Single dynamic community is matched to two step communities.
- Expansion: New community front is significantly larger than previous one.
- Contraction: New community front is significantly smaller than previous one.

Framework: Life-Cycle Events

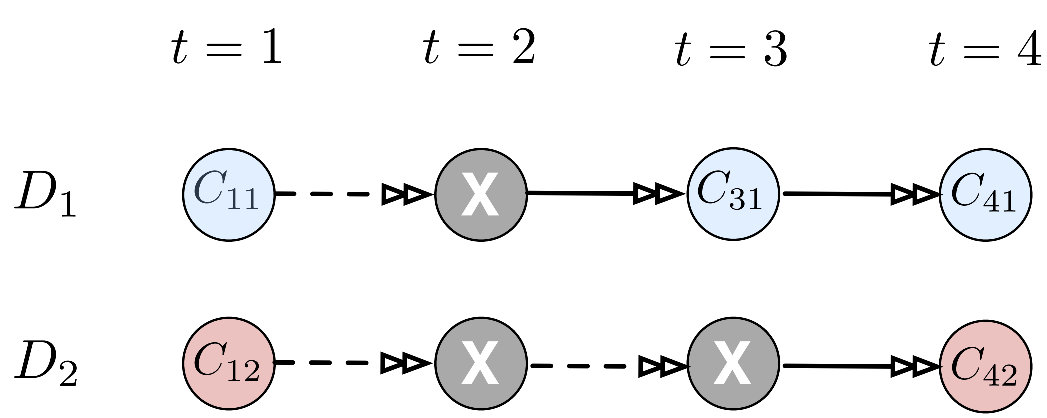
Example Events



Merge $(D_1, D_2) \rightarrow C_{21}$

Split $D_3 \rightarrow (C_{22}, C_{23})$

Intermittent Dynamic Communities



Some dynamic communities will not observed at all time steps after birth. (Palla et al, '07)

Algorithm Summary

1. **Initialise Dynamic Communities:**
   Apply static community finding algorithm for step graph at *t=1* to create initial fronts.

2. **Find Step Communities:**
   For each time *t* > *1* apply static community finding algorithm.

3. **Match Step Communities:**
   For each step community *A* found for step *t*

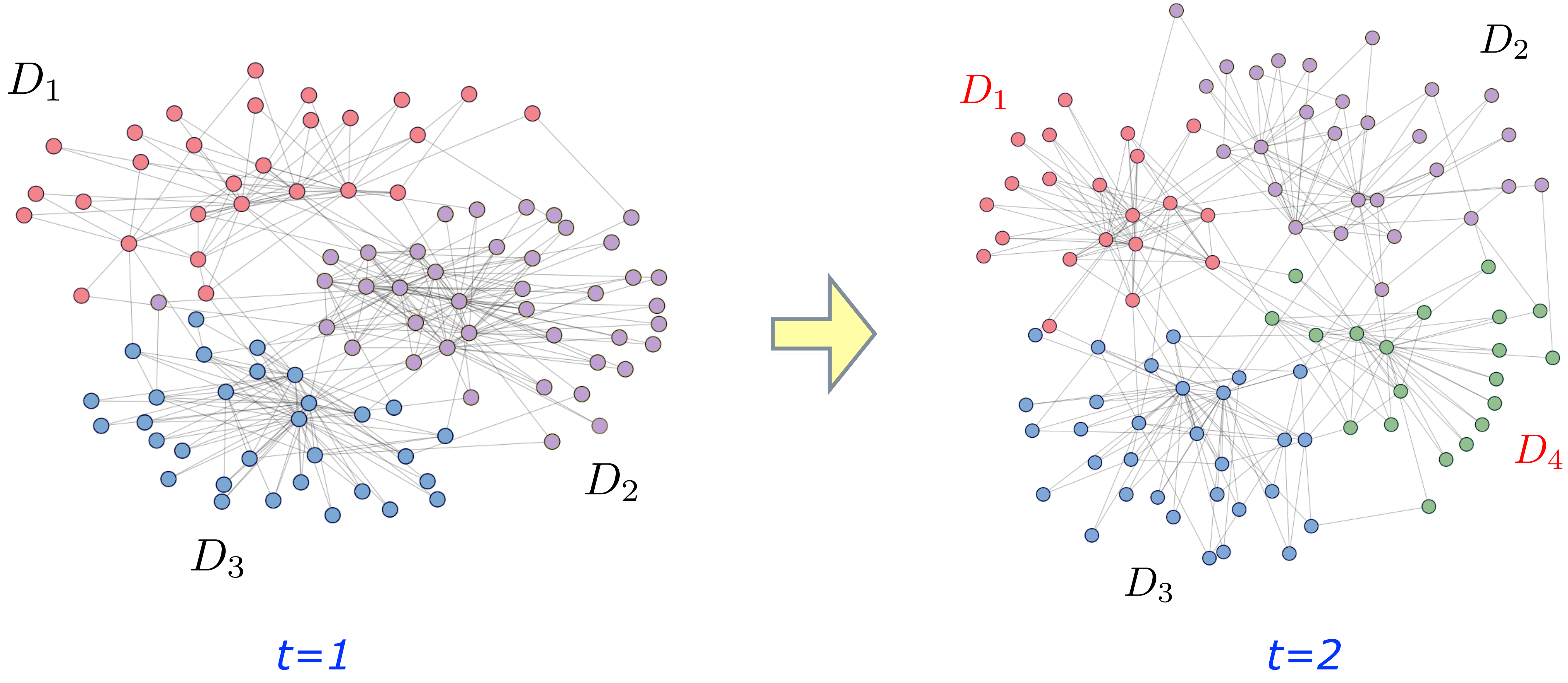
   - For existing front *B*:
     - If $match(A, B) > \theta$ add step community to existing dynamic community.
   - If no match then add as new dynamic community.

4. **Update Dynamic Communities:**
   Replace fronts, and split/merge/remove dynamic communities as necessary.

Evaluation: Benchmark Graphs

Q. How can we verify the accuracy of the proposed algorithm without a ground truth?

- Adapted benchmark software (Lancichinetti & Fortunato, '09) to generate time step graphs with embedded community events.



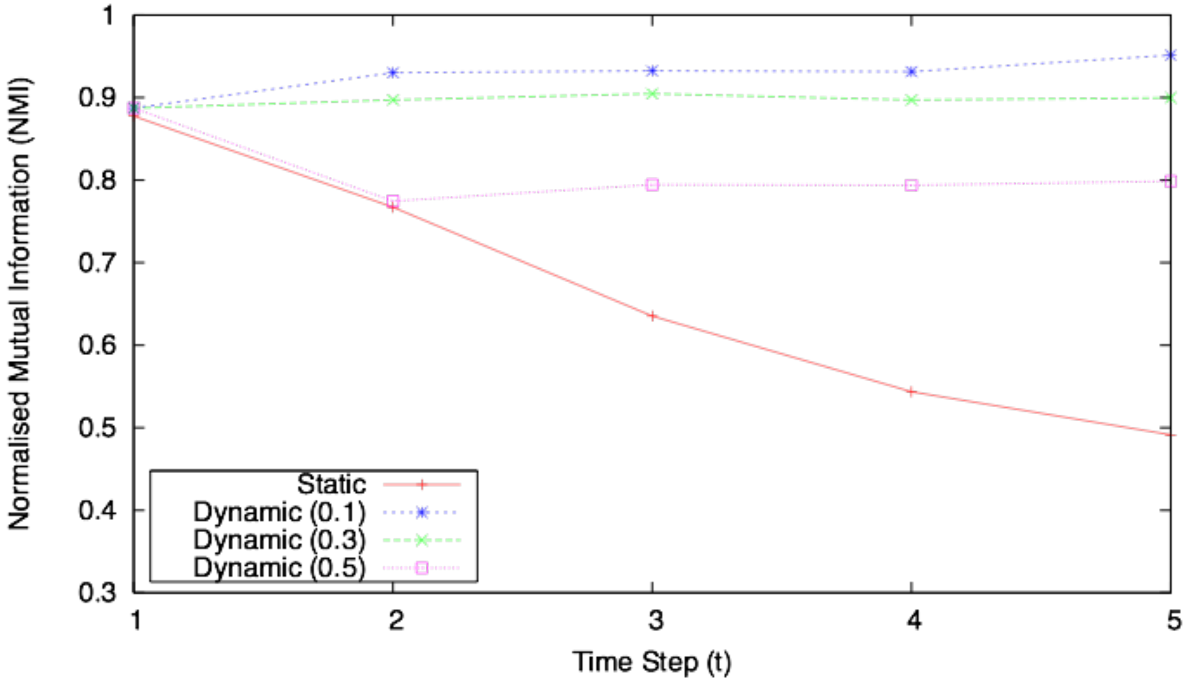
Benchmark Data

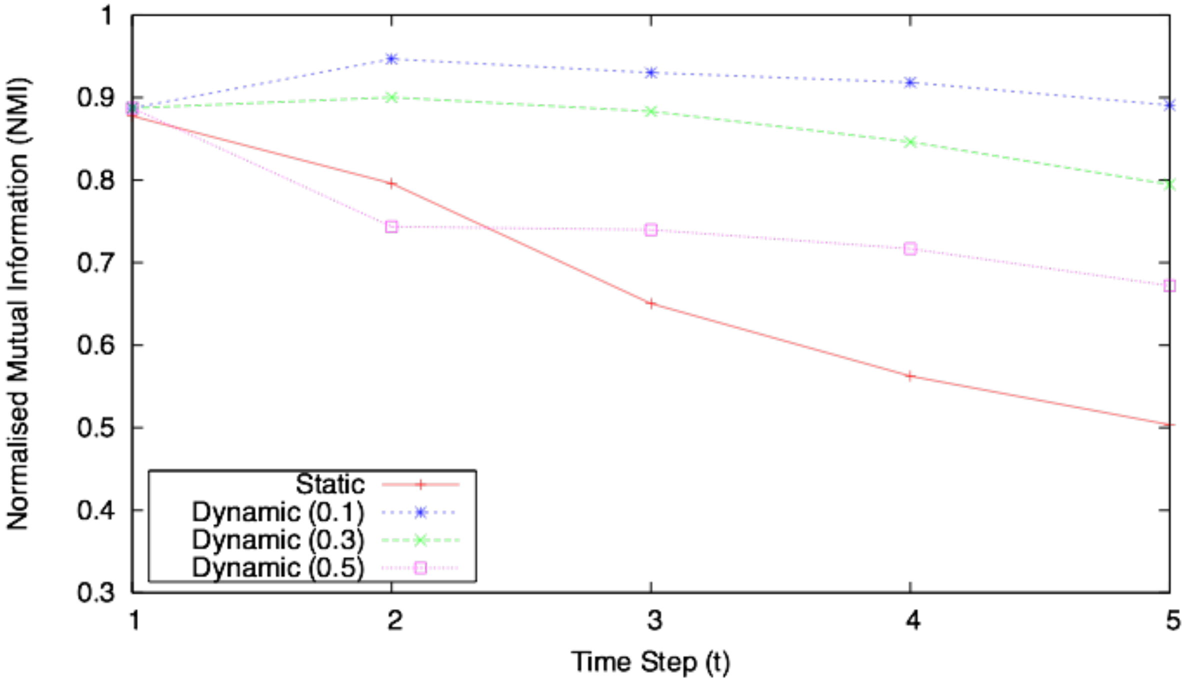
- Generated 4 dynamic benchmark datasets.
  - 5 step graphs, unweighted edges.
  - 15k nodes, degree 20-40.
  - Start with ~400 embedded non-overlapping communities.
- 20% of node memberships switch at each step.
- Additional community events and behaviours are embedded:
  1. Birth and death
  2. Merging and splitting
  3. Expansion and contraction
  4. Intermittent communities

http://mlg.ucd.ie/dynamic

Benchmark Experiments

- Generated disjoint communities on each step using Louvain fast modularity optimization algorithm (Blondel et al, '08).
- Compared to community finding on merged static graphs.
- Measured agreement via Normalized Mutual Information (NMI) with ground truth embedded communities.

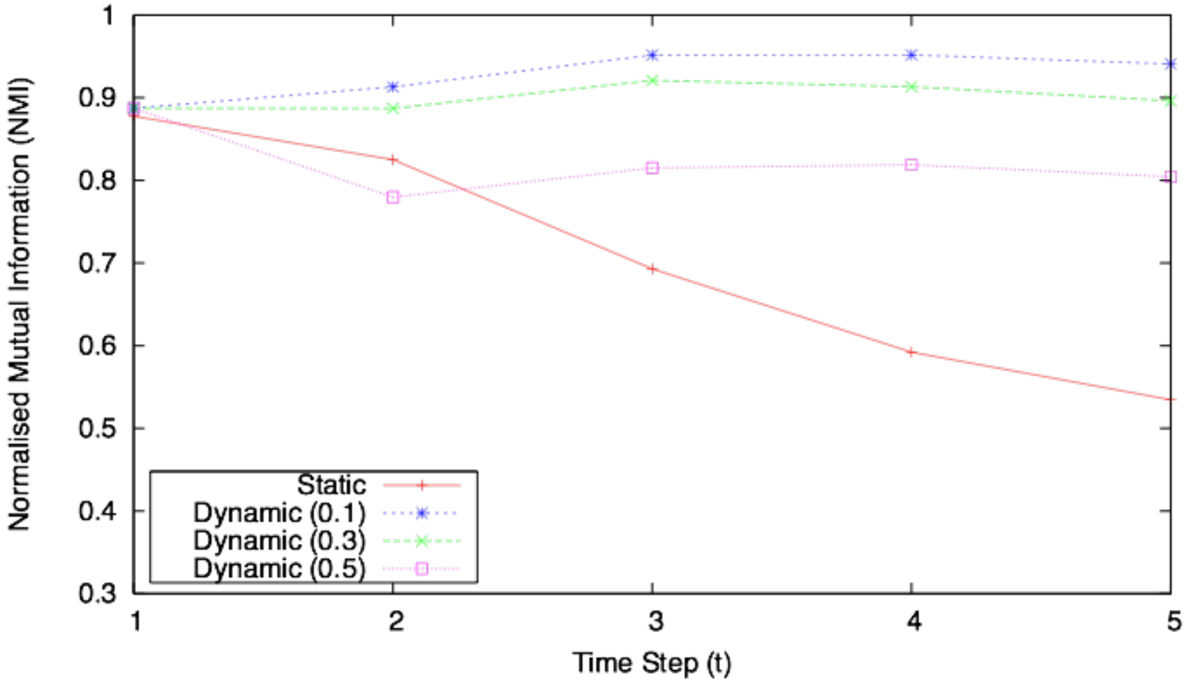




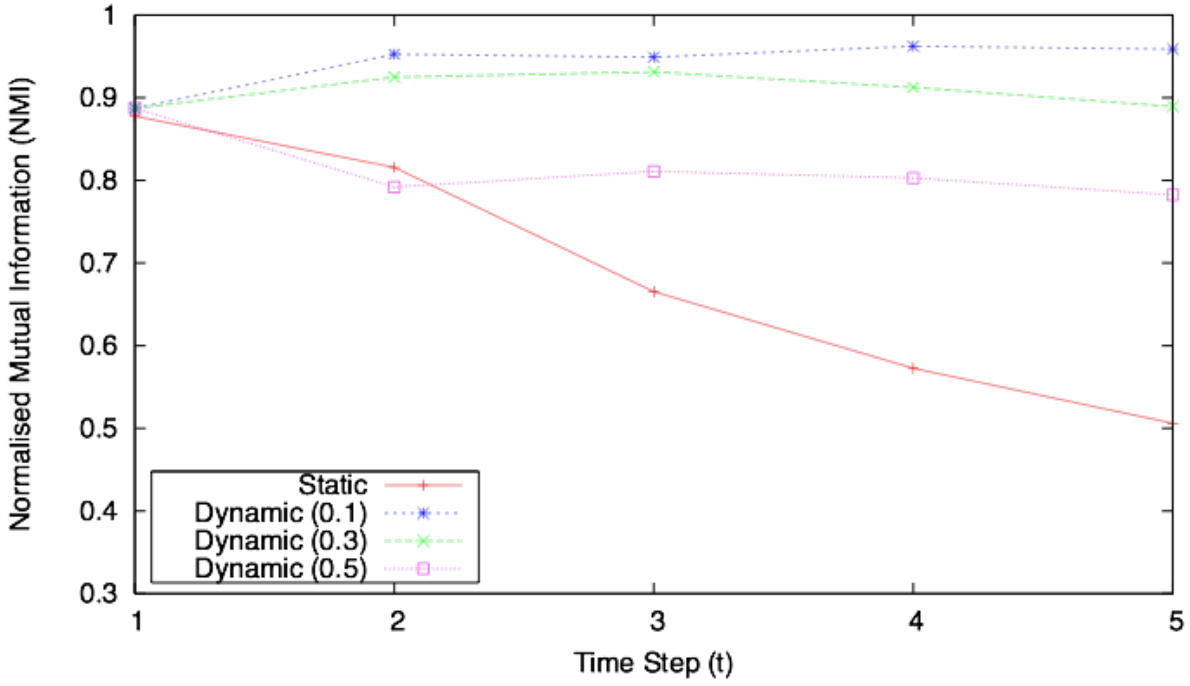
Birth/Death ***Merge/Split***

Benchmark Experiments

- Generated disjoint communities on each step using Louvain fast modularity optimization algorithm (Blondel et al, '08).
- Compared to community finding on merged static graphs.
- Measured agreement via Normalized Mutual Information (NMI) with ground truth embedded communities.



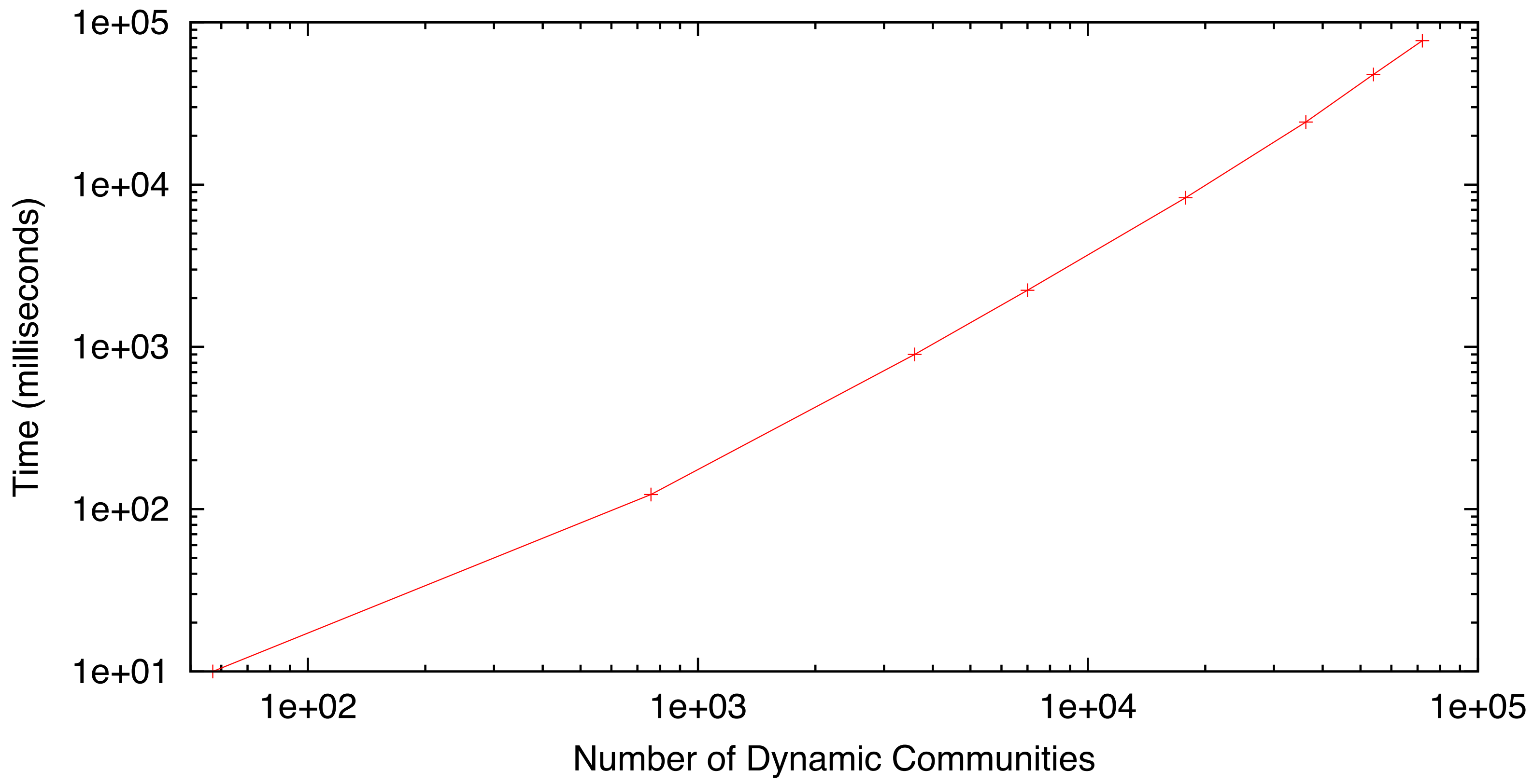
Intermittent



Expand/Contract

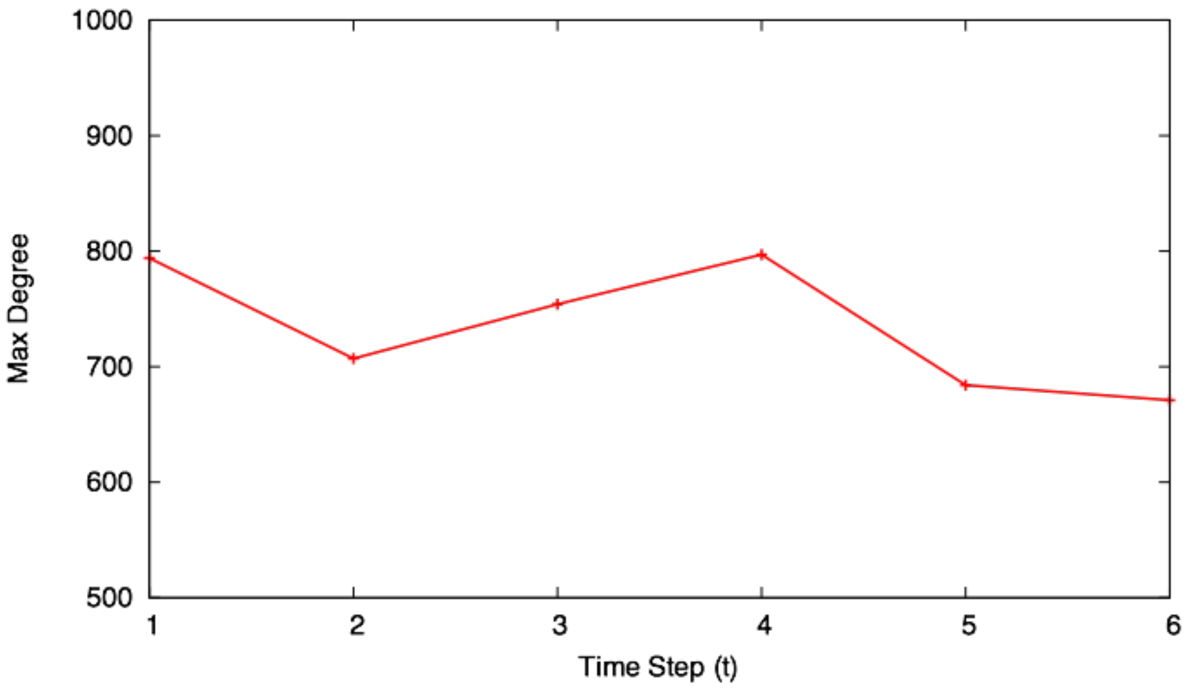
Benchmark Scalability

- Processed 1 million node graph in 85 seconds to identify 70k dynamic communities.
- Experimental bottleneck was graph generation process.

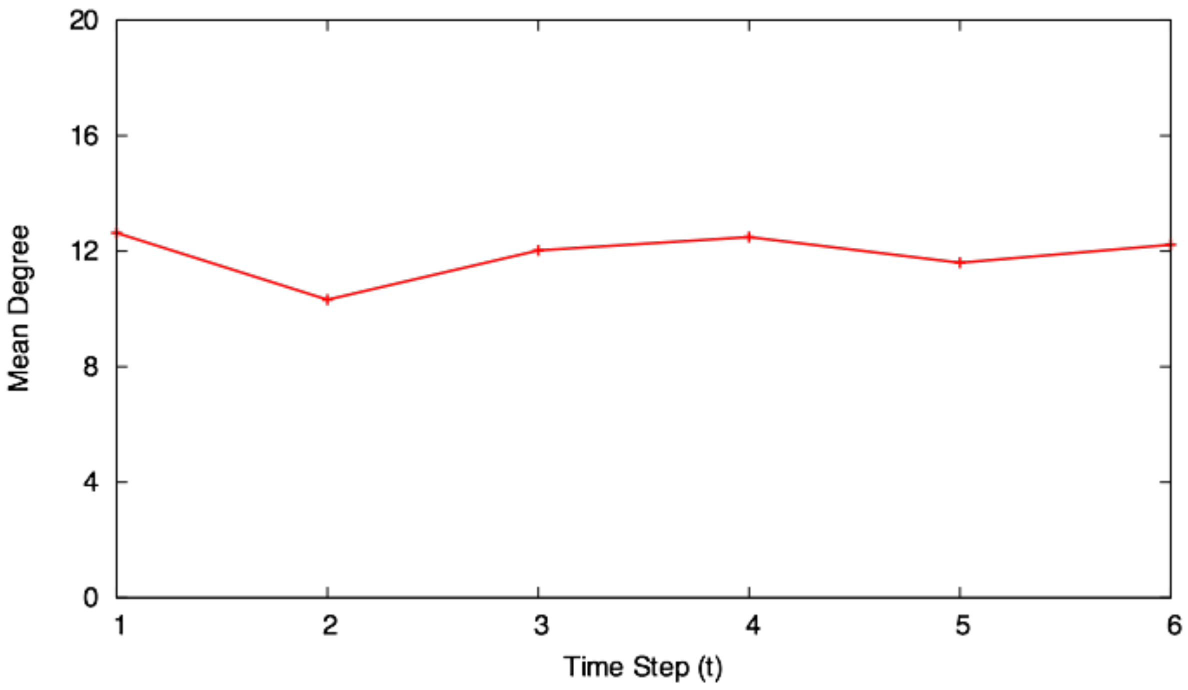


Evaluation: Mobile Call Graphs

- Analyzed weekly batch call graphs covering 6 months.
- Constructed 6 x 4 week time step graphs from union of sets of weekly nodes and edges.
  - 3.0-4.2 million nodes, 20-26 million edges per step.



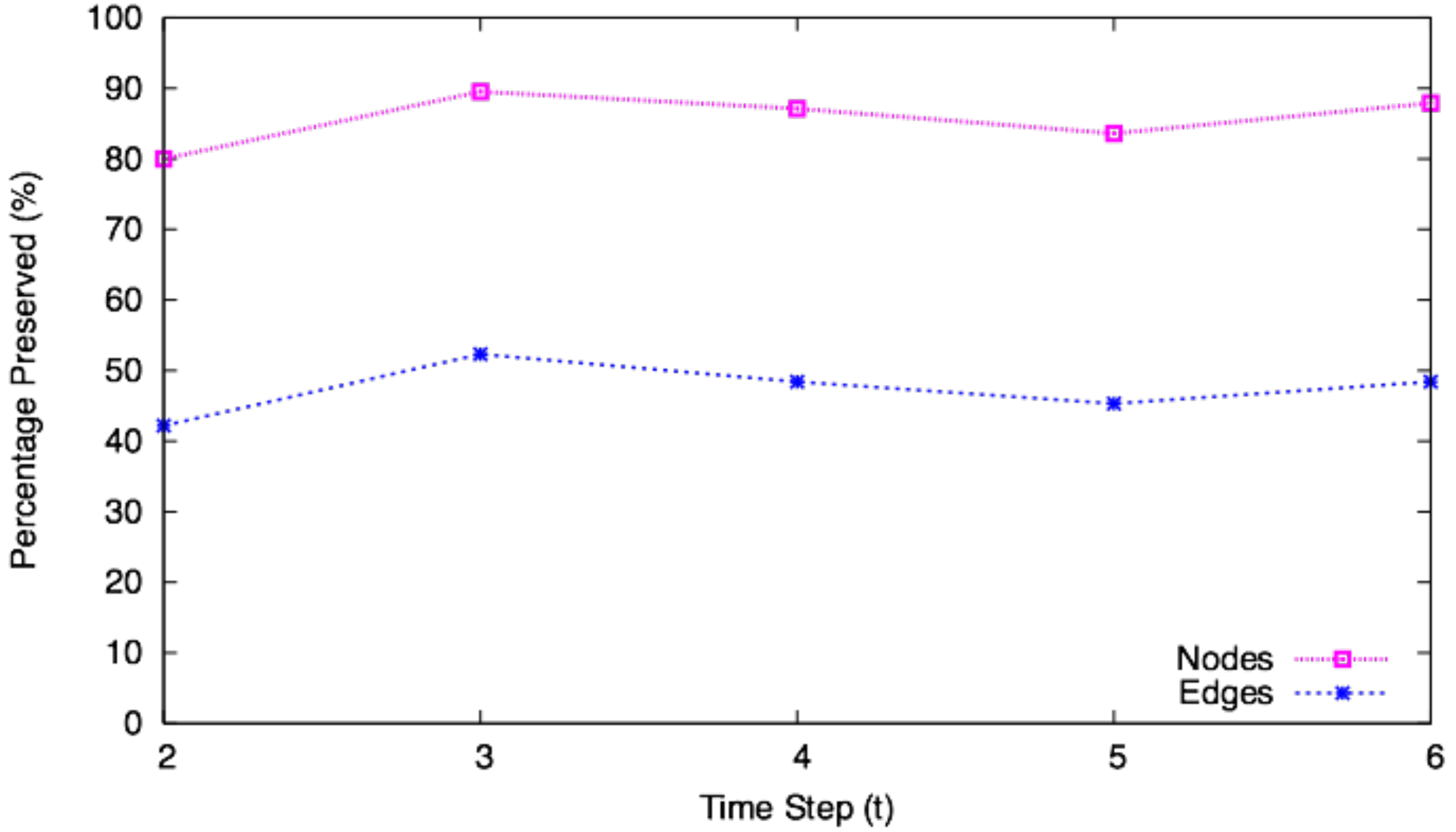
Maximum Node Degree



Mean Node Degree

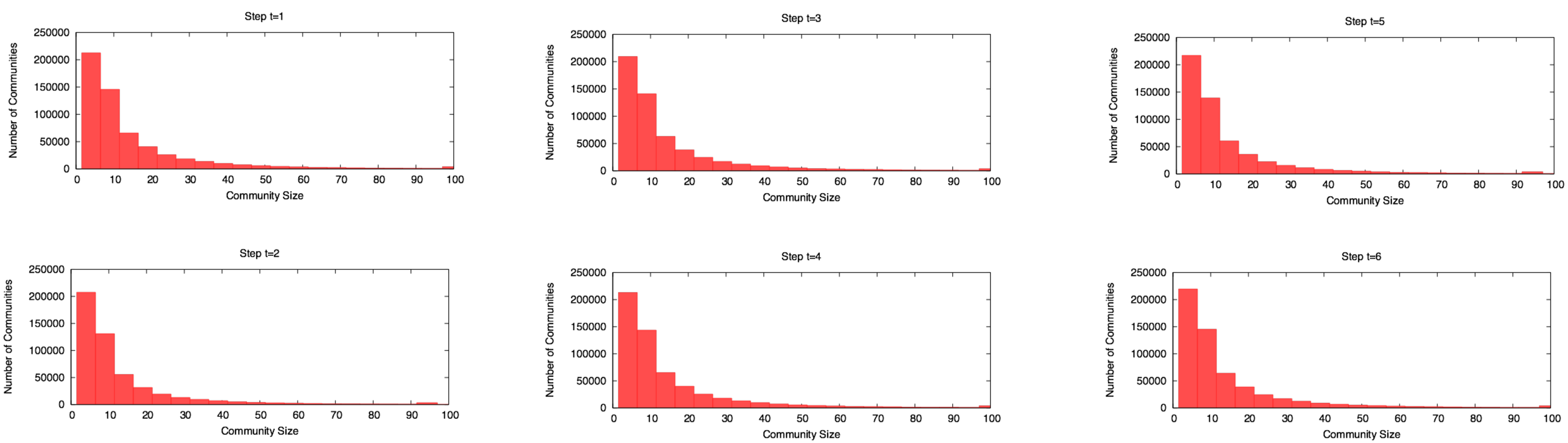
Step Graphs

- On average 86% of nodes present in consecutive time monthly step graphs, with 47% of edges preserved.



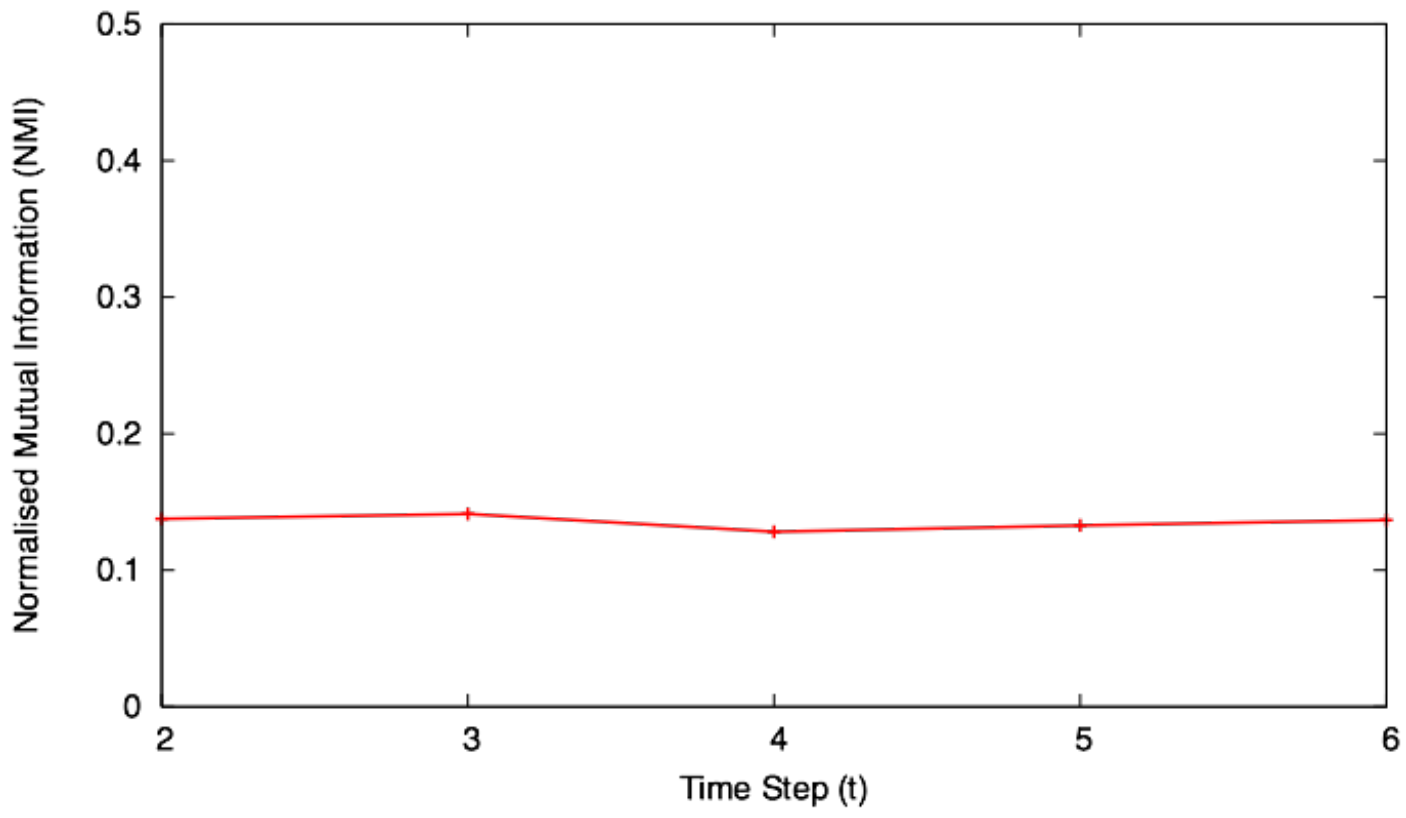
Step Communities

- Applied MOSES algorithm (McDaid & Hurley, '10) to each time step graph to produce overlapping communities.
- Number of distinct *step communities* found in each monthly graph roughly similar (502k-574k).
- Over ~99.9% of step communities have size < 100.
  - Highly similar community size distributions.



Step Communities

- Comparatively low level of agreement (NMI) between node memberships of step communities found in consecutive time steps (~10%).



Q. Is there sufficient signal across the time steps to perform dynamic community finding?

Dynamic Communities

- Applied dynamic community finding to step communities for a range of matching threshold values: $\theta \in (0.2, 0.5)$
- Process took 7-8 hours for a full 6 month set on single core.
  - Running time could be considerably reduced via parallelization.
  - Dynamic analysis can be performed incrementally.

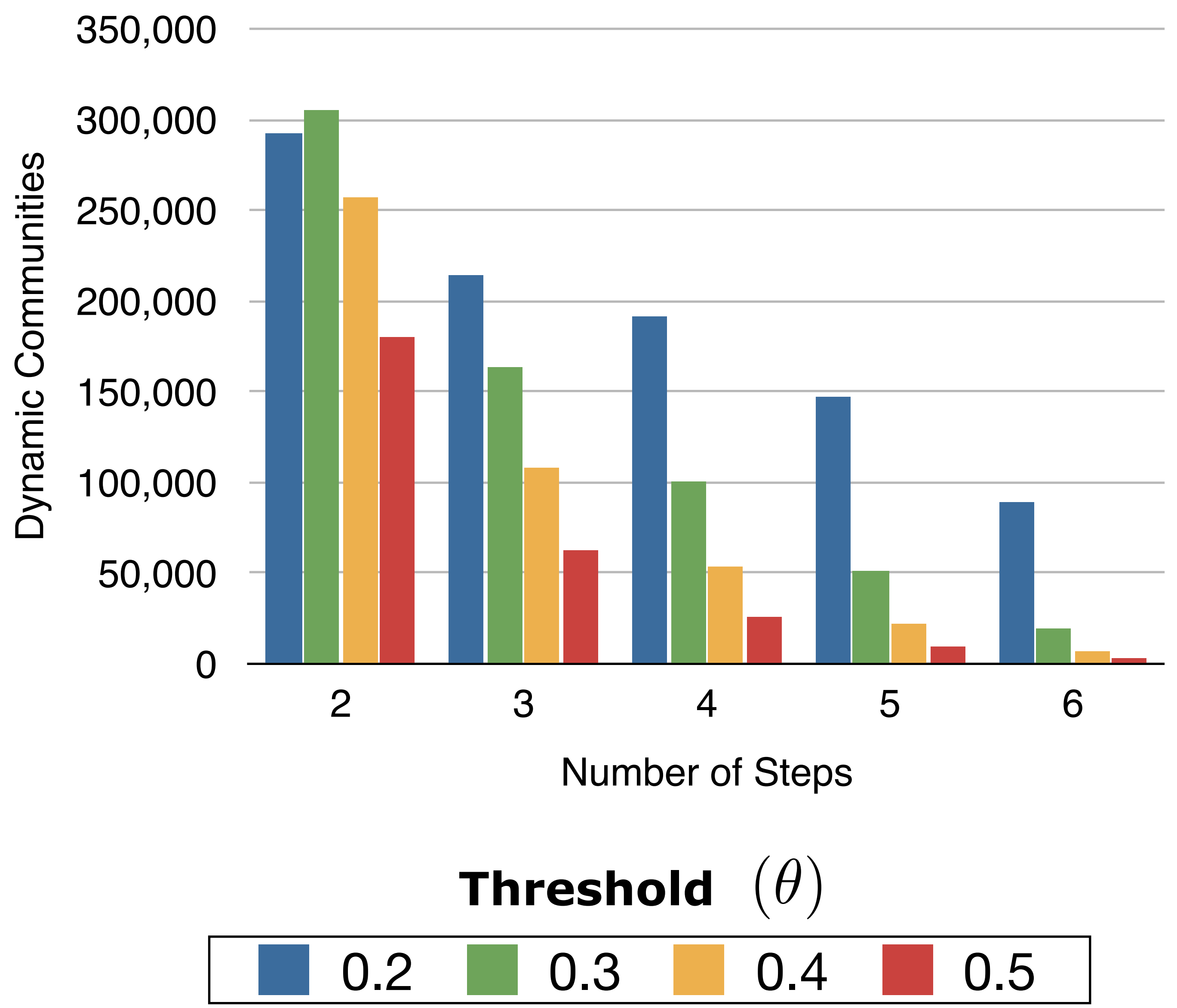
Stricter Matching

Threshold (θ)	Total Dynamic Communities	Long-Lived (%)	Intermittent (%)
0.2	2,014,651	46.3767669933899%	33.8441744997024%
0.3	2,306,976	27.7237387818512%	19.8014630407945%
0.4	2,626,672	17.0640643369252%	17.4%
0.5	2,900,921	9.7%	15.7%

281,950 long-lived communities

Long-Lived Communities

Number of steps in which dynamic communities appear

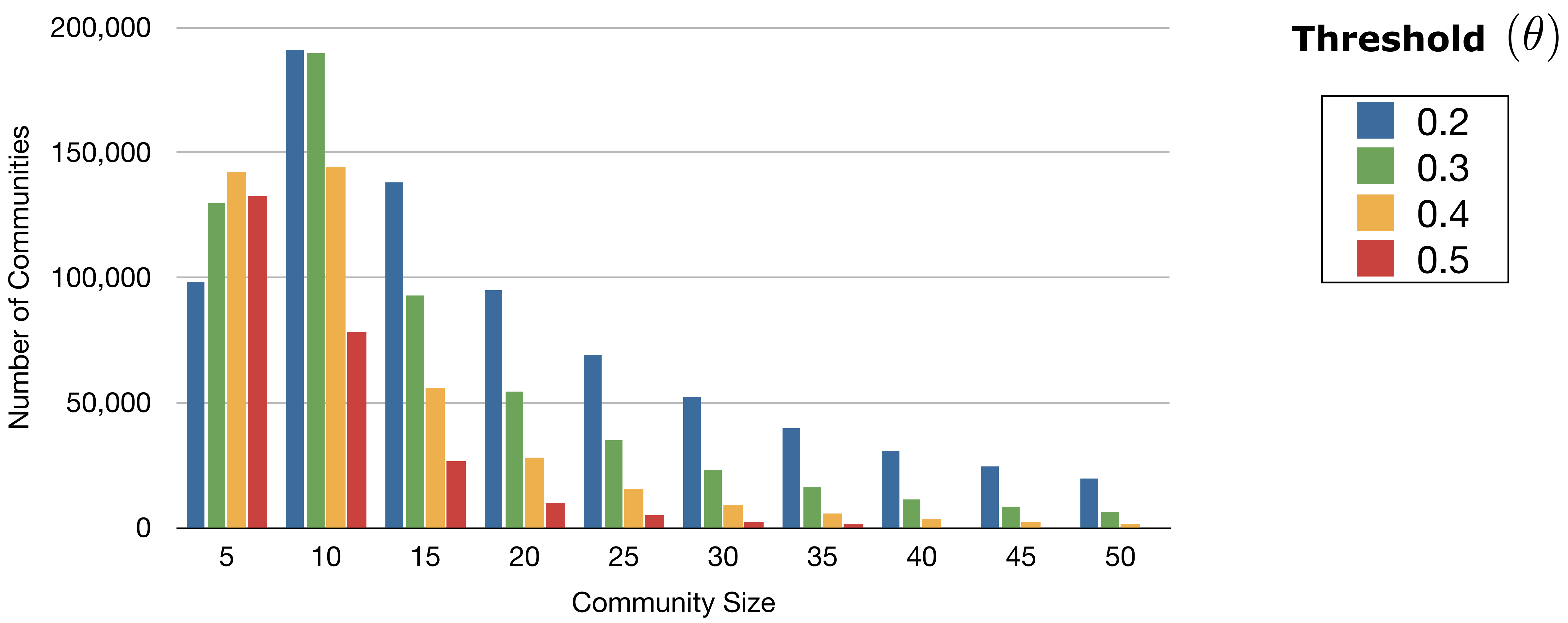


➡ Even in strictest case, algorithm identifies ~190k dynamic communities observed in at least 50% of the time steps.

Dynamic Community Sizes

- Define overall dynamic community membership as union of step community memberships in its timeline.

Long-Lived Dynamic Community Size Distributions



Dynamic Community Sizes

- Define overall dynamic community membership as union of step community memberships in its timeline.

Long-Lived Dynamic Community Size Distributions

Stricter Matching

Threshold (θ)	**Size <= 50** (%)	**Mean Community Size**
0.2	86.7%	29
0.3	95.2%	16.8
0.4	98.4%	11.2
0.5	99.5%	7.9

➡ Small core groups of users are present in dynamic communities across the entire timeline.

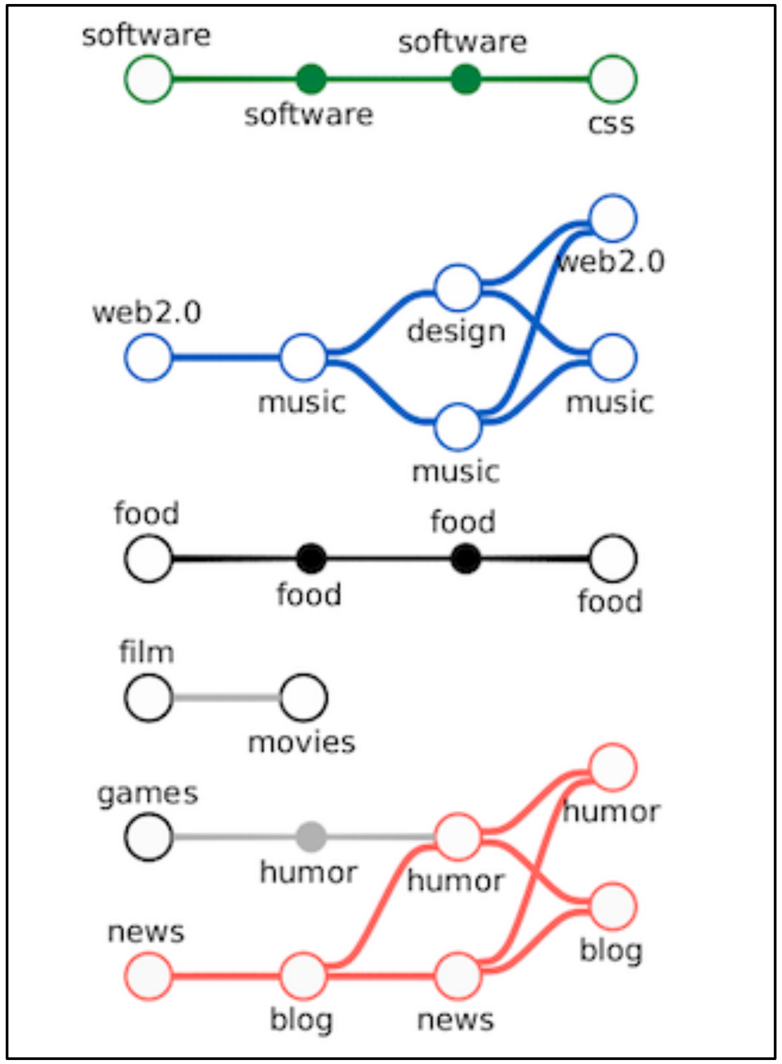
Conclusions

- We have proposed a simple, scalable approach for identifying long-lived communities in dynamic networks.
- Approach is robust to volatile changes in community memberships on synthetic and mobile call data.

Q. How to interpret the large volume of output of dynamic community finding process?

➡ We have implemented a "metro map" visualization metaphor for illustrating dynamic group evolution...

➡ Need a scalable solution for large networks.



References

D. Greene, D. Doyle, and P. Cunningham, “Tracking the evolution of communities in dynamic social networks,” in *Proc. International Conference on Advances in Social Networks Analysis and Mining (ASONAM’10)*, 2010.

- C.Tantipathananandh,T. Berger-Wolf, and D. Kempe, “A framework for community identification in dynamic social networks”. In *Proceedings of KDD 2007.*
- G. Palla, A. Barabasi, and T. Vicsek, “Quantifying social group evolution,” *Nature*, vol. 446, no. 7136, 2007.
- A. Lancichinetti and S. Fortunato. “Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities”. *eprint arXiv*: 0904.3940, 2009.
- V. Blondel, J. Guillaume, R. Lambiotte, and E. Lefebvre. “Fast unfolding of communities in large networks”. *J. Stat. Mech*, 10008, 2008.
- D. Chakrabarti, R. Kumar, and A. Tomkins. “Evolutionary clustering”. In *Proceedings of KDD 2006*.
- A. McDaid and N. Hurley. "Detecting highly overlapping communities with Model-based Overlapping Seed Expansion". *ASONAM 2010*