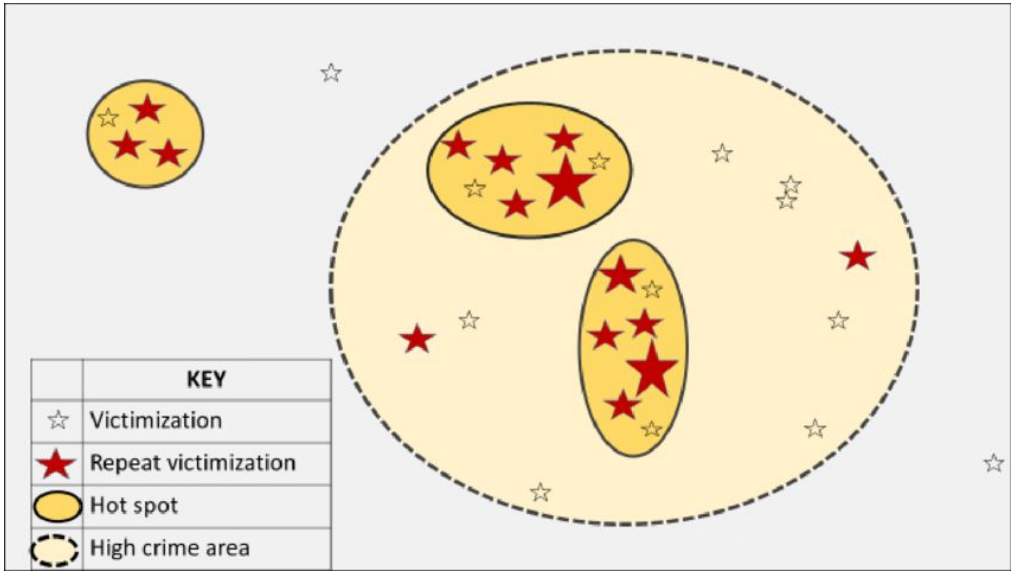
Taming Near Repeat Calculation for Crime Analysis via Cohesive Subgraph Computing

Zhaoming Yin, Xuan Shi
Presenter: Zhaoming Yin, StreamNet Chain LLC,
stplaydog@gmail.com

Self Introduction

Zhaoming Yin had a joyful undergraduate study in Hunan University learning Software Engineering. After that he spent 3 years in Peking University under the supervision of Shiwen Yu for NLP research. Then he headed for Atlanta at GaTech and earned his PhD degree in the area of combinatorial research for bioinformatics and graph analysis advised by Prof David A. Bader. Zhaoming got my first job at Oracle in the Bay area working on Golden Gate, a data base replication software. Two and half years later, he moved back to China and lives in Hangzhou, and worked in Alibaba Cloud on ODPS – a big data platform. After short stint at TRIAS lab as Chief Algorithm Architect and Senior Software Expert in PayTM labs.

Problem Statement



Farrell, Graham, and Ken Pease. "Preventing repeat and near repeat crime concentrations." *Handbook of Crime Prevention and Community Safety* 2 (2017).

Existing Solution

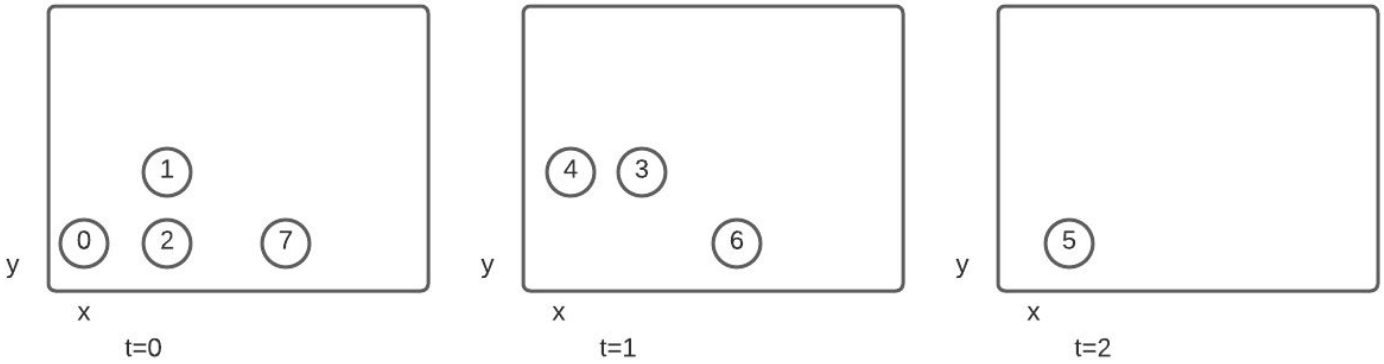
- Knox test method, given there are n crime events
  - compute the pairwise 3D distance
    - 3D means x,y, and t
  - put the events into different bins to cluster the events
  - The complexity of this method would be O(n^2)
  - This method can only handle 2-events cluster

Our Solution

- Build an R-tree using 3-dimensional coordinates x, y, and t.
- Create a graph based on the spatial-temporal coordinates of a specific threshold;
- Based on the graph:
  - compute k-clique, k-core, kDBSCAN, or k-truss for near repeat clusters

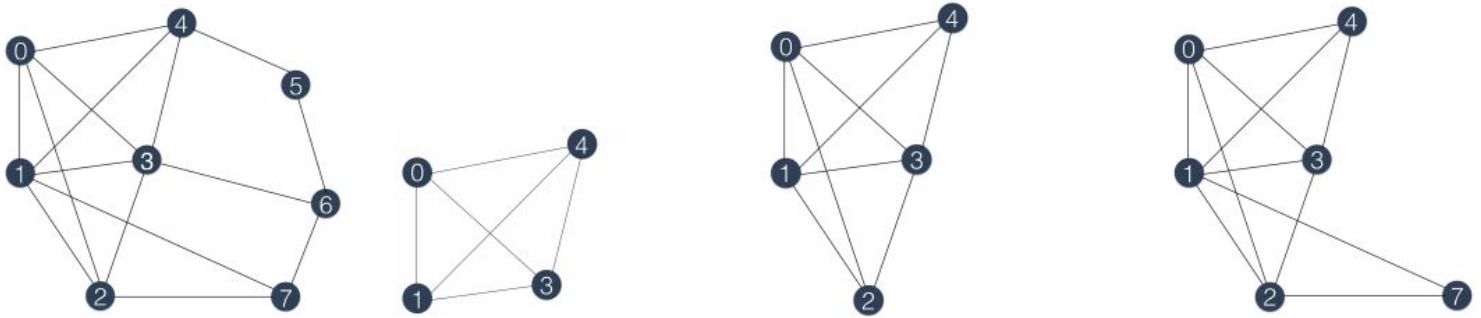
Step 1

based on events happened at time and in the space, we build R-tree index



Step 2

then we get a graph and compute the clusters using different algorithms separately.



(a) Origin (b) 4-clique (c) 3-core (d) 3-truss

Data used for experiments

TABLE I. GENERAL INFORMATION OF GRAPHS.

	#V	#E	#CC	d_avg	d_var	c_avg	c_var
				NY			
BUR	187k	112k	24k	1.25	0.64	0.12	0.064
ROB	198k	152k	27k	1.34	1.38	0.13	0.068
TFT	421k	1.5m	55k	1.77	6.66	0.20	0.089
				DC			
BUR	156k	54k	13k	1.26	0.80	0.10	0.054
ROB	54k	32k	6k	1.40	1.30	0.138	0.069
TFT	344k	1.1m	33k	1.75	7.78	0.17	0.080
				CHI			
BUR	197k	118k	29k	1.29	0.56	0.12	0.060
ROB	124k	68k	14k	1.34	0.96	0.11	0.058
TFT	650k	3.4m	89k	1.84	7.95	0.19	0.081

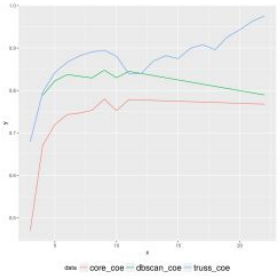
Speed

In seconds

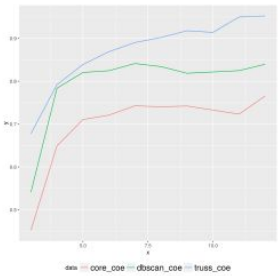
TABLE IV. RESULTS FOR COMPUTATIONAL TIME.

	load	R-tree	edges	CC	truss	core	dbscan	BGL
			NY					
BUR	0.54	0.50	0.15	0.11	6.15	1.53	4.30	2.14
ROB	0.62	0.51	0.21	0.13	7.05	1.97	4.16	2.00
TFT	1.24	1.10	1.06	0.80	10.55	4.05	4.71	302.79
			DC					
BUR	0.47	0.43	0.17	0.05	4.97	0.91	1.59	0.83
ROB	0.15	0.13	0.05	0.03	0.86	0.28	0.32	0.44
TFT	1.07	0.96	0.59	0.44	7.87	1.87	2.37	87.31
			CHI					
BUR	0.62	0.54	0.21	0.12	10.08	1.65	2.6	1.67
ROB	0.39	0.33	0.13	0.07	6.09	1.06	2.11	0.97
TFT	1.10	1.69	4.33	2.76	21.20	7.82	10.93	487.92

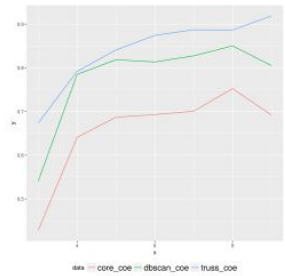
Accuracy



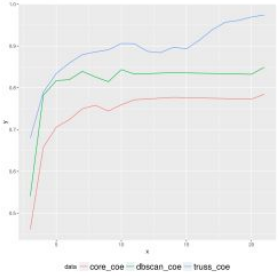
(a) NY BUR



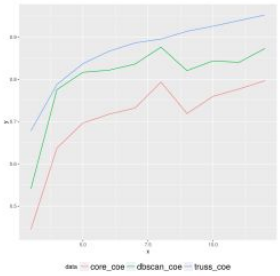
(b) DC BUR



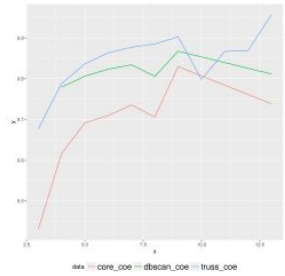
(c) CHI BUR



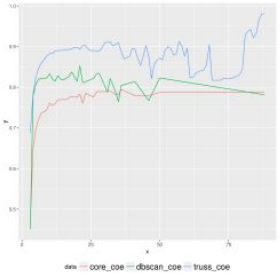
(d) NY ROB



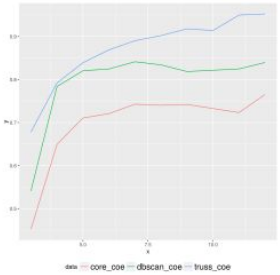
(e) DC ROB



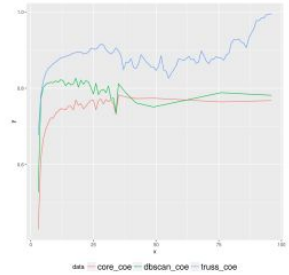
(f) CHI ROB



(g) NY TFT



(h) DC TFT



(i) CHI TFT