

Social and Information Networks

Tutorial #1: Maximizing Strong Edges under STC

University of Toronto CSC303
Winter/Spring 2021
Julian Posada

Week 2: January 20 (2021)

Your TA

My name is Julian Posada. I'm a PhD Candidate at the Faculty of Information.

Your TA

My name is Julian Posada. I'm a PhD Candidate at the Faculty of Information.

My background is in Sociology:

- Labour Sociology
- Social Networks
- Computational Social Science

Your TA

My name is Julian Posada. I'm a PhD Candidate at the Faculty of Information.

My background is in Sociology:

- Labour Sociology
- Social Networks
- Computational Social Science

My research studies the experience of Latin American crowdworkers who annotate data for machine learning and verify algorithmic outputs through labour platforms.

Your TA

My name is Julian Posada. I'm a PhD Candidate at the Faculty of Information.

My background is in Sociology:

- Labour Sociology
- Social Networks
- Computational Social Science

My research studies the experience of Latin American crowdworkers who annotate data for machine learning and verify algorithmic outputs through labour platforms.

First time TAing for CS. I have TAed in the past at ICCIT (UTM) and at the Faculty of Information.

Today's agenda

In lecture we've covered Chapter 3 of the textbook on "Strong and Weak Ties".

Today:

- Questions from Lecture
- Review of Graph Terminology
- Maximizing Strong Edges under STC
- Quercus Quiz

Questions?

Some Useful Terminology

- Directed vs. undirected graph

Some Useful Terminology

- Directed vs. undirected graph
- Weighted vs. unweighted graph

Some Useful Terminology

- Directed vs. undirected graph
- Weighted vs. unweighted graph
- Path

Some Useful Terminology

- Directed vs. undirected graph
- Weighted vs. unweighted graph
- Path
- Cycle

Some Useful Terminology

- Directed vs. undirected graph
- Weighted vs. unweighted graph
- Path
- Cycle
- (Strongly) connected component

Some Useful Terminology

- Directed vs. undirected graph
- Weighted vs. unweighted graph
- Path
- Cycle
- (Strongly) connected component
- Giant component

Some Useful Terminology

- Directed vs. undirected graph
- Weighted vs. unweighted graph
- Path
- Cycle
- (Strongly) connected component
- Giant component
- Embeddedness

Some Useful Terminology

- Directed vs. undirected graph
- Weighted vs. unweighted graph
- Path
- Cycle
- (Strongly) connected component
- Giant component
- Embeddedness
- Social capital

Some Useful Terminology

- Directed vs. undirected graph
- Weighted vs. unweighted graph
- Path
- Cycle
- (Strongly) connected component
- Giant component
- Embeddedness
- Social capital
- Triadic closure

Strong Triadic Closure (STC)

We consider a social network where vertices (nodes) are people and edges (links) signify friendship. Edges are weighted, they are either *Strong* (meaning full friendship), or *Weak* (meaning acquaintance).

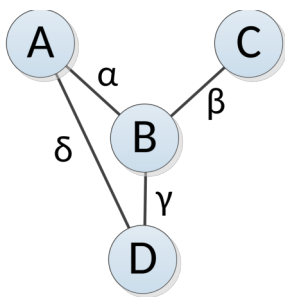
Definition (Strong Triadic Closure)

A graph satisfies Strong Triadic Closure (STC) if whenever (A, B) and (A, C) are strong ties, then there will be a tie (possibly only a weak tie) between B and C .

Strong Triadic Closure (STC)

If we have a graph, and we want to label the edges as strong or weak so that it obeys STC, is it difficult to minimize the number of strong edges?

What about the number of weak edges?



$$G=(V,E)$$

Solving MinSTC

Definition (MinSTC)

MinSTC is the problem of labeling the edges of a graph so that STC holds and the number of weak edges in the labeling is minimized.

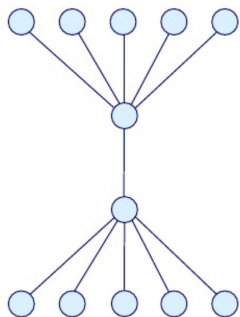
MinSTC is an NP-Hard problem, so it requires approximation algorithms (the definition is beyond the scope of the course, for this course it means we don't have a polynomial time solution).

We can solve MinSTC by reducing it to the Minimum Vertex Cover Problem, and using an approximation algorithm to solve the vertex cover problem.

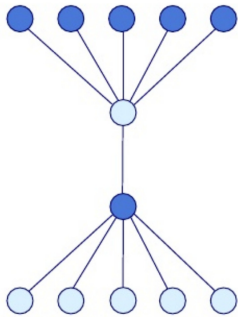
Minimum Vertex Cover

- Given a graph $G = (V, E)$, a **vertex cover** is a subset of vertices (nodes) that contains at least one vertex (node) of each edge
- A **minimum vertex cover** is a vertex cover of minimum size

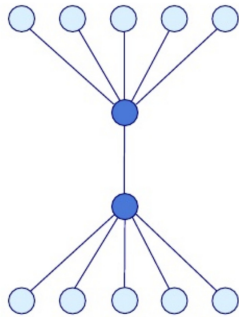
Input graph



Vertex cover



Minimum vertex cover



Solving MinSTC: “Covering” Open Triangles

For a set of edges E , we can define STC as:

$$\forall (A, B), (B, C) \in E : \text{strong}((A, B)) \wedge \text{strong}((B, C)) \implies (A, C) \in E$$

Solving MinSTC: “Covering” Open Triangles

For a set of edges E , we can define STC as:

$$\forall (A, B), (B, C) \in E : \text{strong}((A, B)) \wedge \text{strong}((B, C)) \implies (A, C) \in E$$

By the contrapositive, we have an equivalent definition:

$$\forall (A, B), (B, C) \in E : (A, C) \notin E \implies \text{weak}((A, B)) \vee \text{weak}((B, C))$$

Solving MinSTC: “Covering” Open Triangles

For a set of edges E , we can define STC as:

$$\forall (A, B), (B, C) \in E : \text{strong}((A, B)) \wedge \text{strong}((B, C)) \implies (A, C) \in E$$

By the contrapositive, we have an equivalent definition:

$$\forall (A, B), (B, C) \in E : (A, C) \notin E \implies \text{weak}((A, B)) \vee \text{weak}((B, C))$$

i.e. For the “open triangle” A, B, C (i.e. a triangle missing exactly one side), we must cover it (i.e. make one of the present sides weak).

Solving MinSTC: Reduction to Vertex Cover

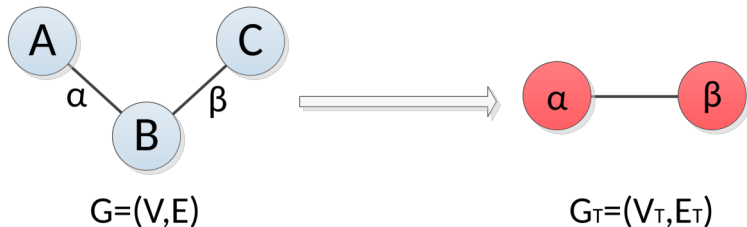
Therefore consider the graph $G = (V, E)$ that we want to perform MinSTC on. Let us define the corresponding graph $G_T = (V_T, E_T)$ such that:

- $V_T = E$
- $(\alpha, \beta) \in E_T \iff$ the edges α and β are an open triangle in G

Solving MinSTC: Reduction to Vertex Cover

Therefore consider the graph $G = (V, E)$ that we want to perform MinSTC on. Let us define the corresponding graph $G_T = (V_T, E_T)$ such that:

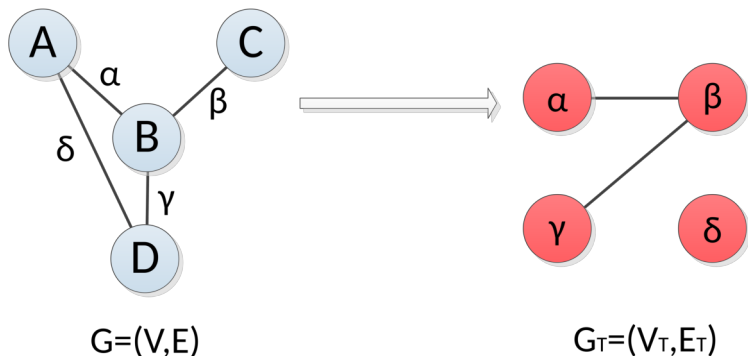
- $V_T = E$
- $(\alpha, \beta) \in E_T \iff$ the edges α and β are an open triangle in G



Solving MinSTC: Reduction to Vertex Cover

Therefore consider the graph $G = (V, E)$ that we want to perform MinSTC on. Let us define the corresponding graph $G_T = (V_T, E_T)$ such that:

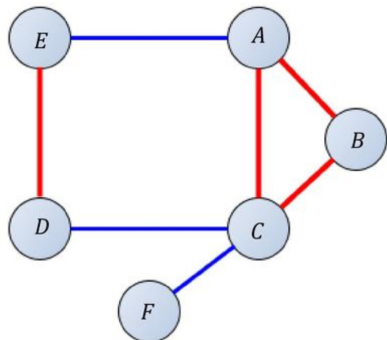
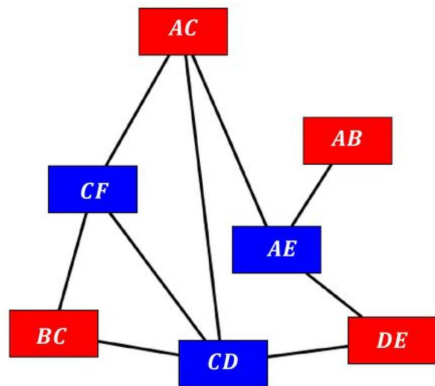
- $V_T = E$
- $(\alpha, \beta) \in E_T \iff$ the edges α and β are an open triangle in G



Solving MinSTC: Reduction to Vertex Cover

Therefore consider the graph $G = (V, E)$ that we want to perform MinSTC on. Let us define the corresponding graph $G_T = (V_T, E_T)$ such that:

- $V_T = E$
- $(\alpha, \beta) \in E_T \iff$ the edges α and β are an open triangle in G



Solving MinSTC: Reduction to Vertex Cover

Therefore consider the graph $G = (V, E)$ that we want to perform MinSTC on. Let us define the complementary graph $G_T = (V_T, E_T)$ such that:

- $V_T = E$
- $(\alpha, \beta) \in E_T \iff$ the edges α and β are an open triangle in G

Therefore, TFAE:

- MinSTC on G
- Finding the minimal set of weak edges that covers all open triangles in G
- Finding the minimal set of vertices of G_T such that all edges have at least one endpoint in the set
- Min Vertex Cover on G_T

Therefore we can solve (or approximate) MinSTC by solving (resp. approximating) Min Vertex Cover

Solving Min Vertex Cover: Greedy Algorithm

A greedy approximation to solve the min vertex cover is to repeatedly add the vertex that covers the **most uncovered edges** until all edges are covered.

This approach often works well in practice, but it can be shown that if run on a graph $G' = (V', E')$ then it may return a solution that is $\Theta(\log |V'|)$ times larger than the optimal solution. So, this is a $\Theta(\log n)$ -approximation.

Solving Min Vertex Cover: Maximal Matching Algorithm

Definition (Matching)

A matching of a graph $G' = (V', E')$ is a subset $e \subseteq E'$ of edges with no common endpoints

The maximal matching approach iterates over all edges in E' . Whenever it finds an edge $e \in E'$ that is not covered by the current cover, then both endpoints of the edge are added to the current cover.

In other words, a collection of **non-adjacent edges** of the graph where no additional edges can be added.

Note that as the true optimal cover must contain one endpoint for every edge, then the solution returned is at most twice the size of the optimal solution. Thus it is called a 2-approximation.

Quercus Quiz