Induced 6-Cycle Counting in Bipartite Networks

February 16, 2021

Abstract

Finding subgraphs in bipartite networks is integral to understanding its underlying structure. The smallest cycle in a bipartite network is a 4-cycle, which is also known as a butterfly. Previous works have developed efficient butterfly counting algorithms. In this paper, we propose a parallel algorithm for efficiently counting induced 6-cycles.

1 Introduction

In unipartite graphs, the smallest cycle is a 3-cycle, which is also known as a triangle. In bipartite graphs, triangles do not exist. The smallest non-trivial subgraph in a bipartite graph is a 4-cycle, which is also known as a butterfly.

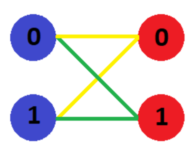

Figure 1: This graph depicts a butterfly. Nodes in u are in blue while nodes in v are in red. The butterfly is made of two wedges, which are highlighted in yellow and green.

Recent algorithms for butterfly counting use the concept of combining wedges (2-paths) to count butterflies (see Figure 1).

Given the relevance of bipartite graphs in real-world relationships, it is desirable to find larger motifs within these graphs. This paper presents a framework for counting induced 6-cycles (see Figure 2) in bipartite graphs which uses the affordances of parallization to maximize efficiency.

2 Notation

We work on a simple bipartite graph $G = (U,\ V,\ E)$ where U is the set of nodes in the left set, V is the set of nodes in right set, and E is the set of edges. An

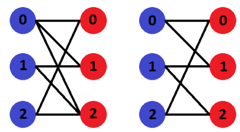

Figure 2: The graph on the left depicts a non-induced 6-cycle. The graph on the right depicts an induced 6-cycle. Nodes in u are in blue while nodes in v are in red. In the non-induced graph, the removal of the edge from u0 to v2 won't affect the 6-cycle.

induced 6-cycle is a set of six nodes *u1, u2, u3* $\in U$ and *v1, v2, v3* $\in V$ such that its internal edges exactly form a cycle.

Algorithm 1 Par6CycleCount(G)

Input: G: graph
Output: c: count of induced 6-cycles

$W \leftarrow$list of wedges $(u1 \rightarrow v1 \rightarrow u2)$
$c \leftarrow 0$
parallel for each $w \in W$ **do**
  BFS on $u2$ until depth 4 $(u2 \rightarrow v2 \rightarrow u3 \rightarrow v3 \rightarrow u4)$ and if $u1 = u3$, $v1 = v3$ or $u2 = u4$, skip w
  parallel for each $u4 \in$ depth 4 **do**
    if $u4 = u1$ **then**
      $c \leftarrow c + 1$
    end if
  end for
end for
