Efficient counting algorithms

Given a temporal network and a motif, we want to efficiently count the number of instances of the motif in the temporal network.

**General algorithm for any motif**
1. Ignore timestamps to get a static graph and a static motif.
2. Find instances of the static motif in the static graph (using known algorithms).
3. For each static motif instance, fetch the motif instances in each temporal edge list using a dynamic programming algorithm that maintains subsequence counts. Runs in linear time in the number of timestamps.

**Special case analysis.** Runs in $O(nm)$ time for 2-node, k-edge motifs, where $m$ is the total number of temporal edges. This is optimum, i.e., linear in the size of the data for constant $k$.

**Faster algorithms for special cases**

- **3-node, 3-edge stars**
  - **Problem**: have to enumerate pairs of neighbors of high-degree nodes
  - **Improvement**: count for all neighbors simultaneously
  - **Runtime complexity**: $O(m)$, where $m$ is the total number of edges (optimal)

- **3-node, 3-edge triangles**
  - **Problem**: a static edge with many timestamps may appear in several triangles
  - **Runtime complexity**: $O(m^2)$ complexity, where $T$ is the number of static triangles

**Triangle speedups**

<table>
<thead>
<tr>
<th>Data source</th>
<th>Wiki</th>
<th>Stack Overflow</th>
<th>Bitcoin</th>
<th>Texts</th>
<th>Phone calls</th>
</tr>
</thead>
<tbody>
<tr>
<td># temp. edges</td>
<td>10M</td>
<td>63M</td>
<td>123M</td>
<td>800M</td>
<td>2.04B</td>
</tr>
<tr>
<td>Speedup</td>
<td>1.92x</td>
<td>1.29x</td>
<td>56.5x</td>
<td>2.28x</td>
<td>1.42x</td>
</tr>
</tbody>
</table>