HGO: Hierarchical Graph Optimization for Accurate, Efficient, and Robust Network Localization

Haodi Ping, Yongcai Wang*, Deying Li

Intelligent Network and Optimization Laboratory – InLab
the Department of Computer Sciences – CS
Renmin University of China – RUC

August 03, 2020
Contents

1. Background of Network Localization and Motivation
2. Component Partition and Local Realization
3. Close-loop Smoothing for Critical Edges
4. Hierarchical Registration
5. Performance Evaluation
6. Conclusion
1.1 Background of Network Localization

- Network localization services are a fundamental demand of most IoT applications.
  - 5G systems
  - military sensing
  - industrial and automation

- GPS is not practical in many scenarios.
  - high cost
  - power requirements
  - GPS signals

- Network localization technologies calculate node coordinates using measurements collected by lightweight and cheap sensors.
  - distance measurements
  - bearing measurements
  - a combination of them
1.2 Related Work and Weaknesses

Related Work:

- **Theoretical fundamentals**
  1. Localizability conditions [1]
  2. Graph rigidity theories [2]

- **Localization algorithms**
  1. Trilateration-based [3]
  2. Linear Barycentric [4]

Weaknesses:

- **Trilateration and linear barycentric**
  1. Sensitive to measurement sparsity

- **Optimization and component stitching**
  1. Flipping ambiguity and flex ambiguity
  2. Poor performance under sparse measurements
  3. High computation cost in large scale networks
  4. Sensitive to the noises

---

1.3 Motivation

Summarizing and instancing the weakness of existing algorithms:

1. satisfactory accuracy in dense and noiseless networks.
2. poor performance in sparse and noise networks.
3. low efficiency in large scale networks.

**Figure.** $n=30$, noise = 5%, AveDeg=12, Time = 1.72s.

**Figure.** $n=30$, noise = 15%, AveDeg=6, Time = 0.94s.

**Figure.** $n=300$, noise = 5%, AveDeg=12, Time = 23.32s.
1.3 Motivation

However, in the IoT era,

- up to thousands of nodes for a network
- inevitable noisy inter-node measurements impacted by environments
- deployment cost and the limited sensor scope cause inevitable sparsity

- Real-time, accurate, and efficient network localization is a foundational requirement for most IoT applications!
- It’s urgent to tackle the sparsity and noise challenges in network localization problems.
1.4 Main Idea

Key observation:
- The dense sub-components can have reliable local realizations against noises.
- The noises of the sparse edges connecting these components (critical edges) are the main cause of the synchronization error.

To tackle the sparsity and noise challenges,
- decompose $G$ into dense sub-components.
- smooth the noises of the critical edges.
- stitch the sub-results with the smoothed edges.

Figure. Sparse $G$, AvgDeg = 6.

Figure. Den sub-$G$, AvgDeg = 8.
1.5 Problem Model

<table>
<thead>
<tr>
<th>Notation</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \tilde{p}_i(p_i \in \mathbb{R}^d, d = {2, 3}) )</td>
<td>estimation of ( v_i )'s location(( v_i )'s realization)</td>
</tr>
<tr>
<td>( d_{ij}, \theta_{ij} )</td>
<td>true distance and true angle between ( i ) and ( j )</td>
</tr>
<tr>
<td>( \sigma_d, \sigma_{\theta} )</td>
<td>distance and angle measurement noise</td>
</tr>
<tr>
<td>( \hat{d}<em>{ij} \sim N(d</em>{ij}, \sigma_d^2) )</td>
<td>distance measurement between ( i ) and ( j )</td>
</tr>
<tr>
<td>( \hat{\theta}<em>{ij} \sim N(\theta</em>{ij}, \sigma_{\theta}^2) )</td>
<td>angle measurement between ( i ) and ( j )</td>
</tr>
</tbody>
</table>

In \( \mathbb{R}^2 \), the problem model can be instanced as:

\[
\tilde{P}^* = \arg \min_{\tilde{P}} \sum_{(v_i, v_j) \in E} \left[ (\tilde{p}_i - \tilde{p}_j) - \begin{bmatrix} \hat{d}_{ij} \cos \hat{\theta}_{ij} \\ \hat{d}_{ij} \sin \hat{\theta}_{ij} \end{bmatrix} \right]^\top \Omega_{ij} \left[ (\tilde{p}_i - \tilde{p}_j) - \begin{bmatrix} \hat{d}_{ij} \cos \hat{\theta}_{ij} \\ \hat{d}_{ij} \sin \hat{\theta}_{ij} \end{bmatrix} \right]_{F_{ij}}
\]

(1)

i.e., we focus on calculating nodes locations using the noisy inter-node distance and angle measurements.
1.6 Overview of Hierarchical Graph Optimization

We adopt a decomposition, local realization, critical edge smoothing and hierarchical registration routine.

1. decomposition using density-based community detection.
2. local realization using typical \textit{GeneralGraphOptimization}(G^2O).
3. detects and smooth critical edges between the components.
4. hierarchical registration using smoothed CEs to generate $\tilde{P}$.

\textbf{Figure.} The overview of HGO.
Component Dividing

- A modularity maximization community detection method: ModulMax is adopted
  - uses only the adjacency matrix of G
  - divide G into $n_c$ density-based disjoint components
    $\mathbf{C} = \{\mathbf{C}_1, \cdots, \mathbf{C}_{n_c}\}$
  - $G^2O$ is applied in sub-graphs to give local realization
    $\tilde{\mathbf{Q}} = \{\tilde{\mathbf{Q}}_1, \cdots, \tilde{\mathbf{Q}}_{n_c}\}$

**Definition (Critical Edge, CE)**

$(v_i, v_j) \in E$ is defined as a Critical Edge if $v_i$ and $v_j$ belong to different components.
Measurement in Realization

- dense components have reliable local realization
- the estimated inter-node distance and angle in a component are more trustworthy than their raw measurements

**Definition (Measurement in Realization (MIR))**

If $v_i$ and $v_j$ are in the same component, Measurement in Realization (MIR) is denoted by $\tilde{e}_{ij} = \tilde{q}_i - \tilde{q}_j$. The corresponding distance and angle of a MIR are calculated by:

\[
\tilde{d}_{ij} = ||\tilde{e}_{ij}||
\]
\[
\tilde{\theta}_{ij} = \text{atan2}(\tilde{e}^1_{ij}, \tilde{e}^2_{ij})
\]

(2)
<table>
<thead>
<tr>
<th></th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Background of Network Localization and Motivation</td>
</tr>
<tr>
<td>2</td>
<td>Component Partition and Local Realization</td>
</tr>
<tr>
<td>3</td>
<td>Close-loop Smoothing for Critical Edges</td>
</tr>
<tr>
<td>4</td>
<td>Hierarchical Registration</td>
</tr>
<tr>
<td>5</td>
<td>Performance Evaluation</td>
</tr>
<tr>
<td>6</td>
<td>Conclusion</td>
</tr>
</tbody>
</table>
Loop Closing Error

- Measurement noise lead to poor localization performance.
- There lacks indicators to quantify noise.
- Denote an edge as a vector $\hat{e}_{ij} = (\hat{d}_{ij} \cdot \cos \hat{\theta}_{ij}, \hat{d}_{ij} \cdot \sin \hat{\theta}_{ij})$.

**Definition (Loop Closing Error, LCE)**

Suppose a loop formed by $l$ edges $\{\hat{e}_{1(2)}; \cdots; \hat{e}_{l(1)}\}$, the corresponding LCE is defined as:

$$\delta_L = -(\sum_{i=1}^{l-1} \hat{e}_{i(i+1)} + \hat{e}_{1l})$$  \hspace{1cm} (3)

**Figure.** Demonstration of LCE.

LCE of a loop indicates how the noise measurements make the sum measurements derivate from the optimal case.
Close-loop Smoothing of the Critical Edges

Existing works lack efficient methods to deal with the noises of the critical edges. Exploit the LCE and the better trustworthy of the MIRs to smooth the noises of CEs.

- Suppose $\mathcal{K}$ critical edges denoted by $\{\hat{e}_1, \cdots, \hat{e}_K\}$.
- Assign an adjustment $\Delta_i$ to each critical edge $\hat{e}_i$ such that all loops formed by the critical edges and MIRs have zero LCEs.
- The decision variable is $\Delta = [\Delta_1 \Delta_2 \cdots \Delta_K]^\top$.
- Each loop sets up an equation for $\Delta$. If we find no less than $\mathcal{K}$ loops, $\Delta$ can be solved.

i.e., the target of CE smoothing is setting the LCE of this loop $L$ to zero:

$$\sum_{i \in I(L)} (\hat{e}_i + \Delta_i) + \sum_{j \in J(L)} \tilde{e}_j = 0$$  \hspace{1cm} (4)
There are numerous ways to find $\mathcal{K}$ loops to cover the critical edges.

1. **Shortest Path Loops:**
   - Select any pair of CEs $E_i$ and $E_j$ and construct $C_{2\mathcal{K}}^2 = \mathcal{K}(\mathcal{K} - 1)/2$ loops.
   - To reduce complexity, we can stop setting equations when the number reaches $c\mathcal{K}$ ($c > 1$), while guaranteeing there are no less than $\mathcal{K}$ equations.

2. **Decomposed Pairwise Smoothing**
   - Decompose the critical edge smoothing problem.
   - The critical edges between two components are smoothed by setting up equations using local loops without considering other components.

The critical edges are then adjusted by: $\hat{e}_i' = \hat{e}_i + \Delta_i$.

The adjusted measurements are more reasonable because they lead to less LCE.
Contents

1 Background of Network Localization and Motivation
2 Component Partition and Local Realization
3 Close-loop Smoothing for Critical Edges
4 Hierarchical Registration
5 Performance Evaluation
6 Conclusion
Backbone Graph Construction

Synchronize the locally realized components to form a global realization.

- Construct a backbone graph:
  - project each component to a vertex
  - project the critical edges among the components to multiple edges among the aggregated vertices

- The backbone graph is denoted by $BG = (V^{BG}, E^{BG})$ where $|V^{BG}| = n_c$ and $|E^{BG}| = K$.

**Figure.** (a) Projection; (b) $BG$. 
BG Realization and Final Registration

- **BG** is a multi-graph with $n_c$ nodes and $\mathcal{K}$ edges.
- Localize **BG** by $G^2O$ and treat the realization $\tilde{P}(BG)$ as the global frame.
- Register the local realizations $\{\tilde{Q}_1, \cdots, \tilde{Q}_{n_c}\}$ to $\tilde{P}(BG)$.

1. **backbone graph extension**: Extend $\tilde{P}(BG)$ to infer the global frame coordinates for the critical nodes in each component.

2. **final registration**:
   - CNs have both local and global realization.
   - $\tilde{Q}_i$ can be registered to $\tilde{P}_i$ using ICP.

3. Conduct BG extension and final registration for every $C_i \in C$. The entire G is registered to $\tilde{P}$.

Local realizations and backbone graph realization are carried out in parallel to improve efficiency.
Contents

1. Background of Network Localization and Motivation
2. Component Partition and Local Realization
3. Close-loop Smoothing for Critical Edges
4. Hierarchical Registration
5. Performance Evaluation
6. Conclusion
Simulation Settings

- $n$ nodes are deployed randomly in a field of $100m \times 100m$.
- The average node degree $\bar{\Lambda}$ (reflects network sparse degree) is controlled by varying the maximum ranging radius $R$.
- Ranging and bearing measurements among nodes are generated by $\hat{d}_{ij} \sim N(d_{ij}, \sigma_d^2)$ and $\hat{\theta}_{ij} \sim N(\theta_{ij}, \sigma_\theta^2)$ when $d_{ij} \leq R$.
- The measurement noises are controlled by varying $\sigma$.

Following algorithms are compared:

1. **HGO**: HGO using Shortest Path Loops, when $c = (K - 1)/2$.
2. **HGO-DPS**: HGO using Decomposed Pairwise Smoothing.
3. **G$^2$O**: the state-of-the-art centralized graph optimization method.
4. **ARAP**: patch decomposition and hierarchical optimization method.
5. **WCS**: density based component partition and weighted stitching method.
Visualizing the Localization Accuracy

Visualized accuracy of $G^2O$ and HGO when $\sigma = 30\%$, $n = 400$ and $\bar{\Lambda} = 10$.

**Figure.** $SSE(\tilde{\mathbf{P}}) = \frac{1}{n} \sum_{i=1}^{n} ||\tilde{\mathbf{p}}_i - \mathbf{p}_i||$

- $\tilde{\mathbf{p}}_i$ is the ground truth
- $\mathbf{p}_i$ is the estimated location
- Blue lines show the difference to the ground truth.

1. HGO provides more reliable result than $G^2O$.
2. HGO outperforms $35.14\%$ in average.
Accuracy in Different Noise Levels

SSE cumulative distribution function (CDF) when $\sigma = 30\%$, $n = 400$ and $\bar{\Lambda} = 10$.

![Cumulative Distribution Function](image)

1. HGO outperforms other algorithms obviously in different noise levels.
2. HGO-DPS is slightly better than previous algorithms.

Figure. The cumulative distribution of SSE.
Accuracy Vs. Sparsity and Noise Levels

$n = 100$, $\bar{\Lambda}$ varies in $\{6, 8, 10, 12\}$, $\sigma$ varies in $\{10\%, 20\%, 30\\%\}$.

**Figure.** SSE of different algorithms in different $\bar{\Lambda}$ and $\sigma$.

1. All the five algorithms return rather accurate results showing their good performances in dense networks.
2. HGO and HGO-DPS improves accuracy than G2O, ARAP and WCS under all parameter settings.
3. The improvement are more obvious in highly noisy and sparse networks.
$n$ varies from 200 to 2000. $\bar{\lambda} = 10$.

In HGO and HGO-DPS, the local component realizations are run in parallel threads.

1. HGO stably outperforms $G^2O$ in time efficiency by using constant $c$ and by parallel implementation.
2. HGO-DPS is faster than HGO-SPL since the decomposed critical edges smoothing are more efficient.
3. Both these two algorithms show smaller time growth rate than $G^2O$ and ARAP with the increase of $n$.

**Figure.** Comparing time consumption in different graph scales.
Contents

1 Background of Network Localization and Motivation
2 Component Partition and Local Realization
3 Close-loop Smoothing for Critical Edges
4 Hierarchical Registration
5 Performance Evaluation
6 Conclusion
Conclusion

To address the challenge of sparseness and noise in network localization,

- HGO exploits the structure unevenness and noise smoothing to improve the accuracy, robustness and scalability.
- HGO outperforms state-of-the-art algorithms in efficiency and accuracy, especially in spares and noise graphs.
- Extending structure-aware, hierarchical approaches to multi-layers and to 3D networks is an important direction.
Thanks
Haodi Ping, Yongcai Wang*, Deying Li
Email: {haodi.ping, ycw, deyingli}@ruc.edu.cn

The full paper and slides are available at http://in.ruc.edu.cn/haodiping/

Intelligent Network and Optimization Laboratory – InLab
the Department of Computer Sciences – CS
Renmin University of China – RUC