Formal Problem Statement: Process Map Optimization via Graph Neural Networks

Mathematical Analysis of "Process Is All You Need"

March 8, 2025

1 Preliminaries and Notation

We begin by establishing the formal mathematical notation required to precisely state the problem addressed in the paper.

Definition 1 (Process Map)**.** A process map is defined as a directed graph $G = (V, E)$ where:

- $V = \{v_1, v_2, \ldots, v_n\}$ is a set of nodes representing tasks
- $E \subseteq V \times V$ is a set of directed edges representing dependencies between tasks
- Each node $v_i \in V$ is associated with a feature vector $\mathbf{x}_i \in \mathbb{R}^d$, encoding task attributes
- Each edge $(v_i, v_j) \in E$ is associated with a feature vector $\mathbf{e}_{ij} \in \mathbb{R}^k$, encoding dependency attributes

Definition 2 (Node Features)**.** For each node $v_i \in V$, the feature vector $\mathbf{x}_i \in \mathbb{R}^d$ encodes the following attributes:

$$\mathbf{x}_i = [t_i, \mathbf{r}_i, \tau_i, s_i, \ldots] \tag{1}$$

where:

- $t_i \in \mathbb{R}_+$ represents the task duration
- $\mathbf{r}_i \in \mathbb{R}^m$ represents resource allocation (as an m-dimensional vector)
- $\tau_i \in \{0, 1\}^c$ represents one-hot encoded task type (from c categories)
- $s_i \in \mathbb{R}$ represents task priority score

Definition 3 (Edge Features)**.** For each edge $(v_i, v_j) \in E$, the feature vector $\mathbf{e}_{ij} \in \mathbb{R}^k$ encodes the following attributes:

$$\mathbf{e}_{ij} = [p_{ij}, w_{ij}, c_{ij}, \delta_{ij}, \ldots] \tag{2}$$

where:

- $p_{ij} \in [0, 1]$ represents transition probability from task i to task j
- $w_{ij} \in \mathbb{R}_+$ represents dependency strength or weight
- $c_{ij} \in \mathbb{R}_+$ represents cost associated with the dependency
- $\delta_{ij} \in \mathbb{R}$ represents temporal constraint or delay

Definition 4 (Adjacency Matrix)**.** The connectivity structure of graph G is represented by an adjacency matrix $A \in \{0, 1\}^{n \times n}$ defined as:

$$A_{ij} = \begin{cases} 1, & \text{if } (v_i, v_j) \in E \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

Definition 5 (Event Log)**.** An event log $\mathcal{L} = \{\sigma_1, \sigma_2, \ldots, \sigma_L\}$ consists of L process instances (traces), where each trace $\sigma_l = \langle e_1, e_2, \ldots, e_{m_l} \rangle$ is a sequence of events. Each event e_i is a tuple $(a_i, t_i, \mathbf{r}_i)$ representing activity, timestamp, and resource attributes.

2 Problem Formulation

We can now precisely define the process map optimization problem addressed in the paper.

Problem 1 (Process Map Optimization)**.** Given:

- A process map $G = (V, E)$ with node features $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$ and edge features $\mathbf{E} = \{\mathbf{e}_{ij}\}_{(v_i, v_j) \in E}$
- An event log $\mathcal{L}$ containing historical process executions

The goal is to learn a function $f_\theta : G \to \mathbb{R}^{|V| \times d'}$ that maps the process graph to node embeddings $\mathbf{H} = \{\mathbf{h}_i\}_{i=1}^n$ where $\mathbf{h}_i \in \mathbb{R}^{d'}$, such that these embeddings enable:

1. **Next-Activity Prediction**: Accurately predict the next activity a_{t+1} given the current state s_t of the process instance:

$$P(a_{t+1}|s_t) = g_\phi(\mathbf{h}_{s_t}) \tag{4}$$

   where g_ϕ is a classifier parameterized by ϕ, maximizing accuracy and Matthews Correlation Coefficient (MCC).

2. **Cycle Time Minimization**: Reduce the overall cycle time T_{cycle} of the process:

$$T_{\text{cycle}} = \sum_{(v_i, v_j) \in P_{\text{critical}}} (T_j - T_i) \tag{5}$$

   where P_{critical} represents the critical path in the process map.

3. **Bottleneck Detection**: Identify bottleneck tasks $V_{\text{bottleneck}} \subset V$ that significantly impact overall process performance:

$$V_{\text{bottleneck}} = \{v_i \in V \mid \text{BottleneckScore}(v_i) > \tau\} \tag{6}$$

where τ is a threshold parameter.

4. **Resource Optimization**: Determine optimal resource allocation $\mathbf{R}^* = \{\mathbf{r}_i^*\}_{i=1}^n$ that minimizes costs while maintaining process efficiency:

$$\mathbf{R}^* = \arg\min_{\mathbf{R}} \sum_{i=1}^{n} c(\mathbf{r}_i) \quad \text{subject to} \quad T_{\text{cycle}}(\mathbf{R}) \leq T_{\text{target}} \tag{7}$$

where $c(\mathbf{r}_i)$ is the cost function for resource allocation $\mathbf{r}_i$.

3 Subproblems

The paper decomposes the main problem into several interconnected subproblems:

3.1 Graph Representation Learning

Problem 2 (Graph Representation Learning)**.** Learn node embeddings $\mathbf{H} = \{\mathbf{h}_i\}_{i=1}^n$ that effectively capture both local task characteristics and global workflow dependencies. This requires:

$$\mathbf{H} = f_\theta(G) = f_\theta(V, E, \mathbf{X}, \mathbf{E}) \tag{8}$$

where f_θ is a Graph Neural Network parameterized by θ.

3.2 Robust Feature Encoding

Problem 3 (Robust Feature Encoding)**.** Develop a normalization strategy that handles noisy or incomplete task and dependency data:

$$\hat{\mathbf{x}}_i = \text{Normalize}(\mathbf{x}_i) \tag{9}$$

$$\hat{\mathbf{e}}_{ij} = \text{Normalize}(\mathbf{e}_{ij}) \tag{10}$$

3.3 Multi-objective Optimization

Problem 4 (Multi-objective Optimization)**.** Design a loss function $\mathcal{L}(\theta, \phi)$ that balances multiple optimization objectives:

$$\mathcal{L}(\theta, \phi) = \lambda_1 \mathcal{L}_{\text{task}} + \lambda_2 \mathcal{L}_{\text{workflow}} + \lambda_3 \mathcal{L}_{\text{regularization}} \tag{11}$$

where $\lambda_1, \lambda_2, \lambda_3$ are weighting hyperparameters.

3.4 Scalability

Problem 5 (Scalability)**.** Ensure the computational approach scales to large process maps with thousands of tasks:

$$\text{Time Complexity} = O(f(|V|, |E|)) \tag{12}$$

$$\text{Space Complexity} = O(g(|V|, |E|)) \tag{13}$$

where f and g are sublinear or at most linear functions of the graph size.

4 Challenges

The formulated problem includes several intrinsic challenges:

1. **Complexity**: Process maps can contain complex dependencies, both sequential and concurrent, requiring expressive model architectures.

2. **Incompleteness**: Real-world process data often contains missing values or incomplete traces, necessitating robust feature representation.

3. **Dynamicity**: Processes evolve over time, with changing resource availability, priorities, and constraints, requiring adaptive modeling approaches.

4. **Heterogeneity**: Tasks and dependencies have diverse attributes (categorical, numerical, temporal), requiring flexible feature encoding.

5. **Multi-objective Nature**: Balancing potentially competing objectives (e.g., minimizing cycle time vs. minimizing resource costs) requires sophisticated optimization strategies.