浙江大學
ZHEJIANG UNIVERSITY

Dependent Function Embedding for Distributed Serverless Edge Computing

Hailiang Zhao @ ZJU.CS.CCNT
http://hliangzhao.me

2021 14th International Conference on Service Science

Motivation

Function-as-a-Service (FaaS) is leading its way to the future service pattern of cloud computing

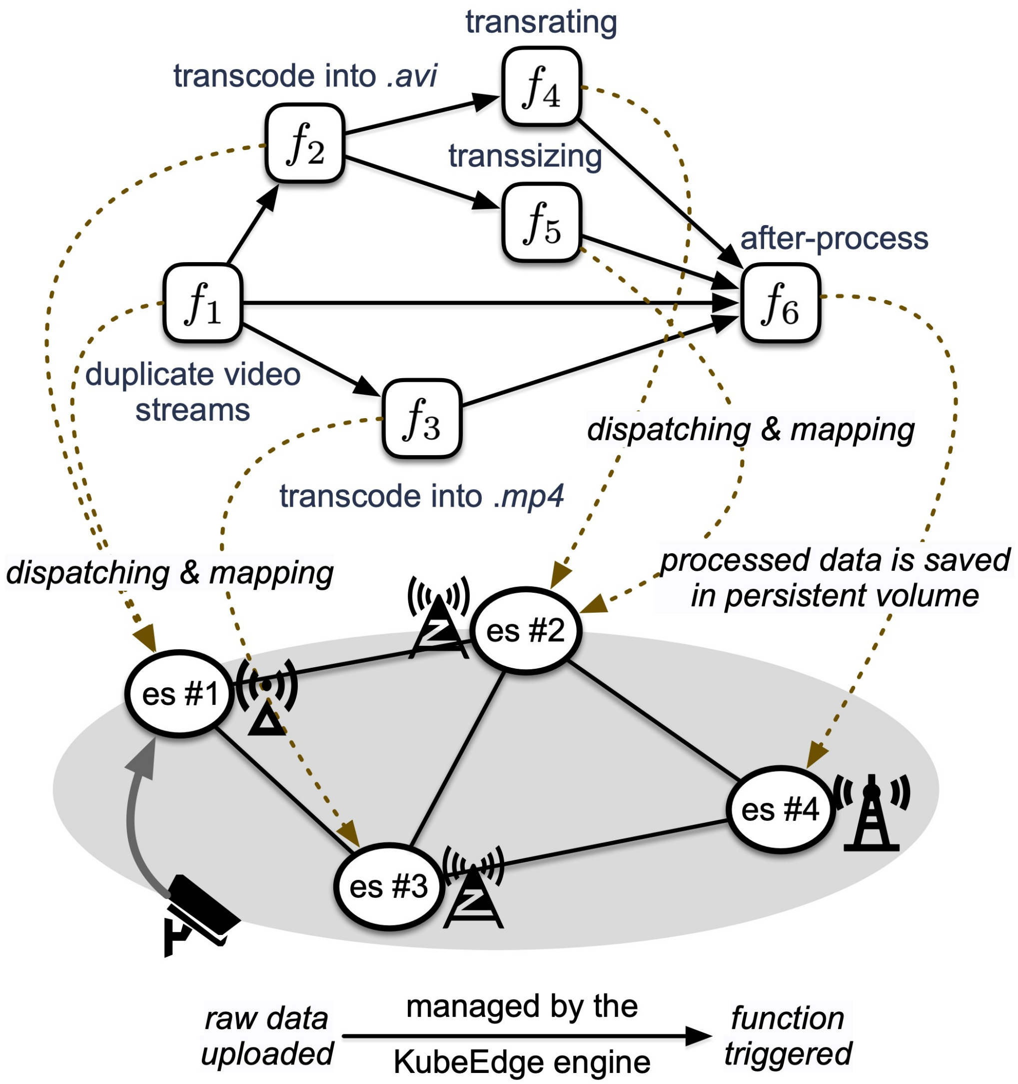
- lightweight containerization with Docker
- service orchestration with Kubernetes

docker kubernetes

Edge Computing is booming as a promising paradigm to push the computation and communication resources from cloud to the network edge

Their combination leads to *Serverless Edge Computing*

A Working Example

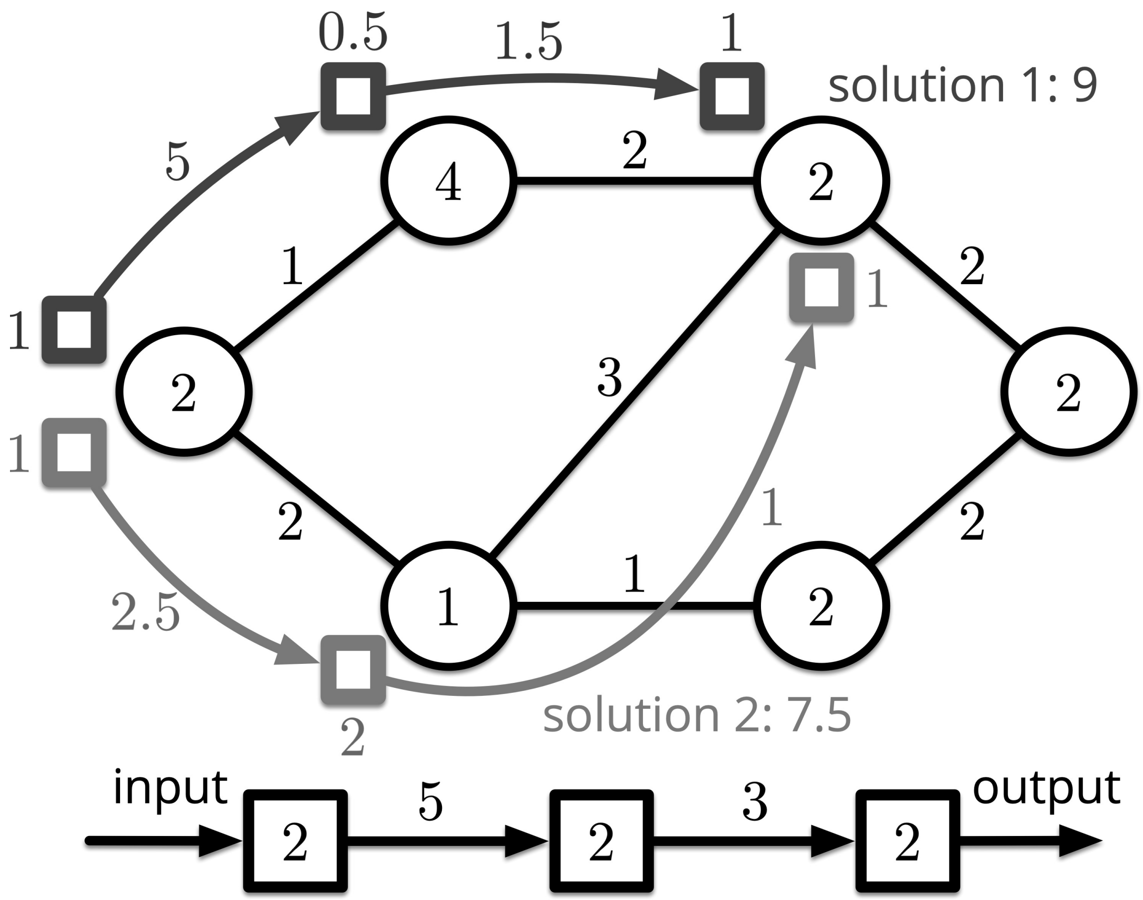


- The Serverless Application is a complex workflow (organized as a DAG in example)
- The Edge servers are modeled as an undirected connected graph

A surveillance camera, uploads the raw video and the actions to take (for example, transcode the raw data into a new container format, such as .mp4, .avi, etc.) to a nearby edge server periodically.

How to minimize the **Makespan**?

Only Function Placement?



- **Solution 1:**

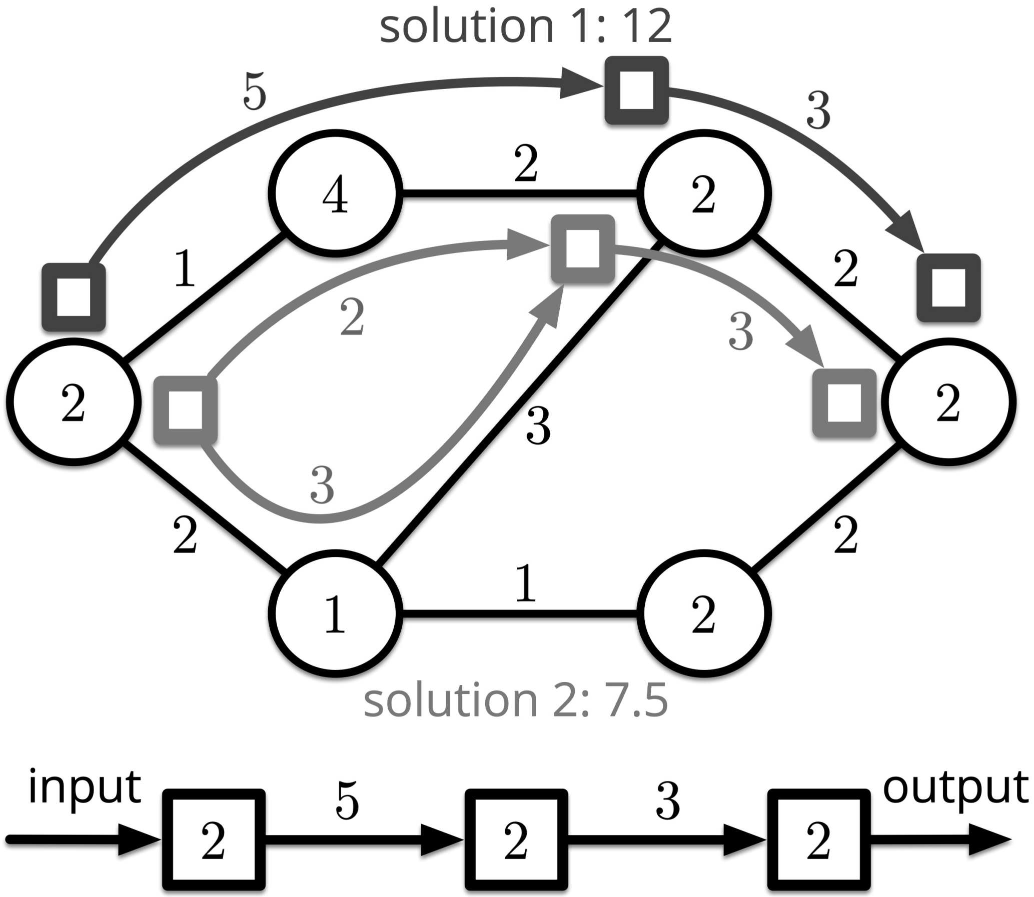
$$\frac{2}{2}+\frac{5}{1}+\frac{2}{4}+\frac{3}{2}+\frac{2}{2}=9$$

- **Solution 2:**

$$\frac{2}{2}+\frac{5}{2}+\frac{2}{1}+\frac{3}{3}+\frac{2}{2}=7.5$$

The traffic routing policy matters!

If Stream Splitting is Allowed



- **Solution 1:**

$$\frac{2}{2}+\left(\frac{5}{1}+\frac{5}{2}\right)+\frac{2}{2}+\frac{3}{2}+\frac{2}{2}=12$$

- **Solution 2:**

$$\frac{2}{2}+\max\left(\frac{2}{1}+\frac{2}{2},\frac{3}{2}+\frac{3}{3}\right)+\frac{2}{2}+\frac{3}{2}+\frac{2}{2}=7.5$$

Data stream splitting may help!

(in the case the splitting cost can be ignored)

System Model

The heterogenous edge network is formulated as an undirected graph

- ***node:*** edge servers with different processing powers
- **edge:** virtual links (each is constitutive of several physical communication links with different forms)

An application is a **Directed Acyclic Graph** (DAG)

- ***function:*** has dependent relations
- ***data stream:*** measured in GB, function can not be executed before required data are received

We name this as **dependent function embedding** since it is actually embedding one graph into another

Involution of Finish Time

$$T(p(f_j)) = \max\left\{\boxed{\iota_{p(f_j)}}, \max_{\forall i:e_{ij}\in\mathcal{E}}\Big(T(p(f_i)) + t(e_{ij})\Big)\right\} + t(p(f_j))$$

the earliest idle time of $p(f_j)$

- (f_i, f_j) is a function pair with dependent relations
- $p(f_i)$ is the edge server which f_i is dispatched to
- $t(e_{ij})$ is the data transferring time of the stream e_{ij}
- $T\Big(p(f_i)\Big)$ is the finish time of function f_i when it is dispatched to $p(f_i)$

Data Transferring Time

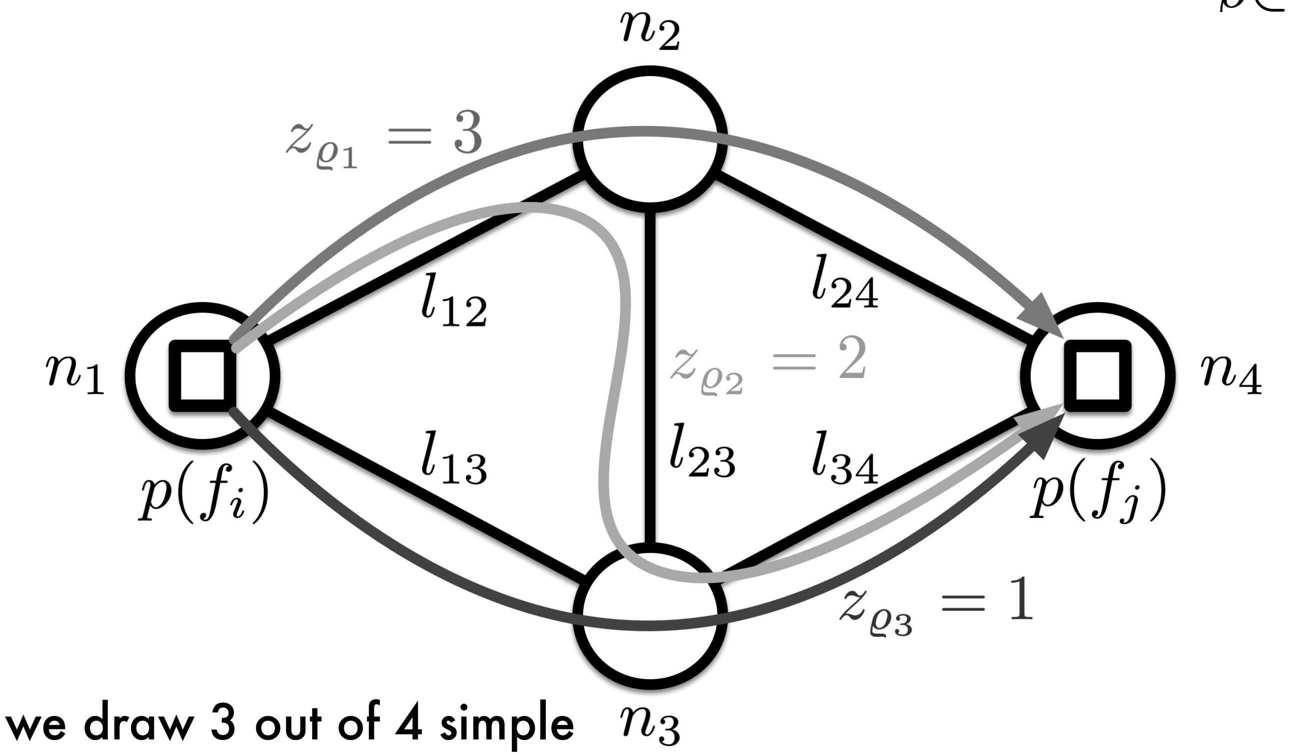
communication start-up cost

$$t(e_{ij}) \triangleq \boxed{\sigma\big(p(f_i),p(f_j)\big)} + \max_{\varrho\in\mathcal{P}\left(p(f_i),p(f_j)\right)} \sum_{l_{mn}\in\varrho} \frac{z_\varrho}{b_{mn}^{\varrho}}$$

- (f_i, f_j) is a function pair with dependent relations
- $p(f_i)$ is the edge server which f_i is dispatched to
- $\mathcal{P}\big(p(f_i),p(f_j)\big)$ is the set of *simple paths* between $p(f_i)$ and $p(f_j)$
- z_ϱ is the data stream size allocated on the simple path ϱ
- b_{mn}^{ϱ} is fixed bandwidth allocated for ϱ and the function pair (f_i, f_j)

Data Transferring Time

$$t(e_{ij}) \triangleq \sigma\big(p(f_i), p(f_j)\big) + \max_{\varrho\in\mathcal{P}\left(p(f_i),p(f_j)\right)} \sum_{l_{mn}\in\varrho} \frac{z_\varrho}{b^{\varrho}_{mn}}$$



we draw 3 out of 4 simple paths between n_1 and n_4

Each simple path should be allocated with how much data stream?

Assumption: for each data stream, during its transferring, the bandwidth allocated to it is viewed as constant and $b^{\varrho}_{mn} \ll b^{max}_{mn}$

Problem Formulation

the last function's finish time

$$\mathbf{P}: \min_{\vec{p},\vec{z}} \boxed{\max_{f \in \mathcal{F}_{exit}} T\big(p(f)\big)}$$

$$s.t. \sum_{\varrho \in \mathcal{P}\left(p(f_i),p(f_j)\right)} z_\varrho = s_{ij}, \forall e_{ij} \in \mathcal{E} \quad \Rightarrow$$

the sum of the data stream allocated on each simple path is exactly the data stream with size s_{ij}

$$\vec{z} \geq \vec{0}$$

How to ***place each function*** and ***split each data stream*** to minimize the Makespan?

Finding Optimal Substructure

$$\underset{\text{* here means the earliest}}{\boxed{T^\star(p(f_j))}} = \max\left\{ \max_{\forall i: e_{ij}\in\mathcal{E}} \left[\min_{\underset{\text{the placement decision}}{\boxed{p(f_i)}},\ \underset{\text{the stream mapping decision}}{\boxed{\{z_\varrho\}_{\forall \varrho\in\mathcal{P}_{ij}}}}} \left(T^\star(p(f_i)) + t(e_{ij})\right)\right], \iota_{p(f_j)} \right\} + t(p(f_j))$$

- (f_i, f_j) is a function pair with dependent relations
- $p(f_i)$ is the edge server which f_i is dispatched to
- $t(e_{ij})$ is the data transferring time of the stream e_{ij}
- $T\Big(p(f_i)\Big)$ is the finish time of function f_i when it

  is dispatched to $p(f_i)$

Optimal Data Mapping

When $p(f_i)$ is fixed, we can define the subproblem for stream mapping:

$$\mathbf{P}_{sub}: \min_{p(f_i),\{z_\varrho\}_{\forall \varrho \in \mathcal{P}_{ij}}} \Phi_{ij} \triangleq \left(\boxed{T^\star\left(p(f_i)\right)} + t(e_{ij})\right)$$

decide $p(f_i)$ to get the value

- (f_i, f_j) is a function pair with dependent relations
- $t(e_{ij})$ is the data transferring time of the stream e_{ij}
- $T^\star\left(p(f_i)\right)$ is the earliest finish time of function f_i when it

  is dispatched to $p(f_i)$
- We need to decide where to place f_j and how e_{ij} is mapped

Optimal Data Mapping

Theorem. The optimal objective value of $\mathbf{P}_{sub}$ is

$$\min_{\vec{z}_{ij}} \|\mathbf{A}\vec{z}_{ij}\|_\infty = \frac{s_{ij}}{\sum_{k=1}^{|\mathcal{P}_{ij}|} 1/A_{k,k}}$$

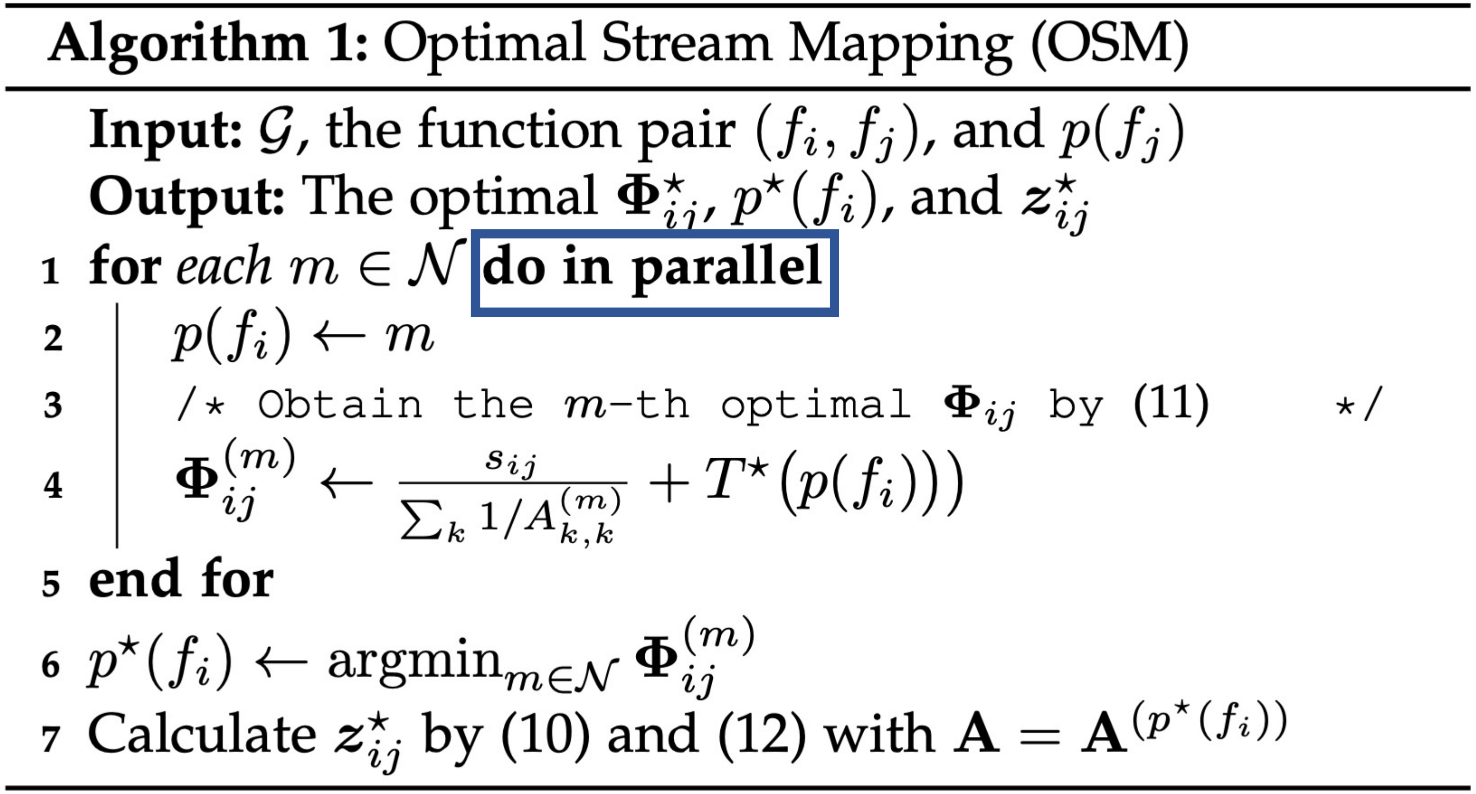
iff

$$A_{u,u}\vec{z}_{ij}^{(u)} = A_{v,v}\vec{z}_{ij}^{(v)}, 1 \leq u \neq v \leq |\mathcal{P}_{ij}|$$

The proof is omitted here.

Optimal Data Mapping

Based on the theorem, we propose the algorithm OSM for solving $\mathbf{P}_{sub}$



Algorithm 1: Optimal Stream Mapping (OSM)

```
Input: $\mathcal{G}$, the function pair $(f_i, f_j)$, and $p(f_j)$
Output: The optimal $\mathbf{\Phi}^\star_{ij}$, $p^\star(f_i)$, and $\boldsymbol{z}^\star_{ij}$
for each $m \in \mathcal{N}$ do in parallel
    $p(f_i) \leftarrow m$
    /* Obtain the $m$-th optimal $\mathbf{\Phi}_{ij}$ by (11) */
    $\mathbf{\Phi}^{(m)}_{ij} \leftarrow \frac{s_{ij}}{\sum_k 1/A^{(m)}_{k,k}} + T^\star(p(f_i)))$
end for
$p^\star(f_i) \leftarrow \text{argmin}_{m \in \mathcal{N}} \mathbf{\Phi}^{(m)}_{ij}$
Calculate $\boldsymbol{z}^\star_{ij}$ by (10) and (12) with $\mathbf{A} = \mathbf{A}^{(p^\star(f_i))}$
```

DP-based Embedding

Based on the theorem, we propose the algorithm DPE for solving $\mathbf{P}$:

```
Algorithm 2: DP-based Embedding (DPE)
Input: $\mathcal{G}$ and $(\mathcal{F}, \mathcal{E})$
Output: Optimal value and corresponding solution
for $j = |\mathcal{F}_{etry}| + 1$ to $|\mathcal{F}|$ do
    for each $n \in \mathcal{N}$ do
        $p(f_j) \leftarrow n$          // Fix the placement of $f_j$
        for each $f_i \in \{f_i | e_{ij}$ exists$\}$ do
            if $p^\star(f_i)$ has been decided then
                continue
            end if
            if $f_i \in \mathcal{F}_{etry}$ then
                $\forall p(f_i) \in \mathcal{N}$, update $T^\star(p(f_i))$ by (2)
            end if
            Obtain the optimal $\mathbf{\Phi}^\star_{ij}$, $p^\star(f_i)$, and $\boldsymbol{z}^\star_{ij}$ by
              calling OSM
        end for
        Update $T^\star(p(f_j))$ by (7)
    end for
end for
return $\max_{f \in \mathcal{F}_{exit}} \text{argmin}_{p^*} T^\star(p^*(f))$, $\boldsymbol{z}^\star$, and $\mathbf{p}^\star$
```

Theorem. In worst case, the complexity of DPE is

$$O\Big(|\mathcal{N}| \times |\mathcal{E}| \times \max\big\{|\mathcal{N}|, \max_{n_i, n_j} |\mathcal{P}_{ij}|\big\}\Big)$$

a function may be a predecessor of multiple functions

update of the optimal substructure

Simulation Results

Compared with a well-known heuristic HEFT, and a related state-of-the-art algorithm FixDoc, DPE achieves the smallest makespan over all the 2119 DAGs with absolute superiority under arbitrary parameters.