Computation Placement - Problem Formulation

Hsiao-Yun Tseng

1 INTRODUCTION

placethings allows developers to construct an experimental test bed to evaluate and verify their internet of things (IoT) applications during the development phase. *placethings* leverages the famous Mininet/Containernet emulation environment to span the emulation across virtual machines, containers, and bare metal devices. Developers can configure the emulated network and add virtual or real THINGs. Based on the network and devices configuration, *placethings* also suggests good solutions for deploying IoT applications.

2 INTELLIGENT COMPUTATION PLACEMENT

Given configuration files specifying the needs of offered services, devices' capabilities, and the network statistics, which includes current utilization, estimated computation time, and estimated transmission latency, along with current utilization metrics, *placethings* generates:

- *NetworkTopology*: a directed graph describing how available devices are connected in the network.
- *DeviceCapability*: a data structure describing each device's network capabilities, computational power, and hardware limitations.

Combined with a directed *TaskGraph* generated from a submitted application, the *placethings* formulates computation mapping as a linear programming problem. Placement can be viewed as decisions of whether to assign a task to a specific device. We set the decision variable as:

$$X(t,d) = \begin{cases} 1 & \text{if task } t \text{ is assigned to device } d \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

The objective is to minimize the longest path's overall latency in the task graph; that is, the sum of total execution times and transmission latencies along the path should be the minimum across all possible placements:

$$\begin{aligned} \min_{x} \max_{p \,\in\, \text{path}(G_t)} & \sum_{i=1}^{\text{len}(p)} X(t_i,d_i)L_t(t_i,d_i) \\ & + \sum_{i=1}^{\text{len}(p)-1} X(t_i,d_i)X(t_{i+1},d_{i+1})L_d(d_i,d_{i+1}) \end{aligned} \tag{2}$$

In the objective function, G_t represents the *TaskGraph*, X is the decision variable. Each t_i and d_i represents the mapping of a task to a device of the i_{th} node of the path p in G_t. Finally, L_t is the computation latency, and L_d is the transmission latency between devices. The solver will find the best solution to the objective function given the following constraints:

Constraint 1: Neighbors in the task graph must also be accessible from each other in network graph.

$$\begin{gathered} X(t_i,d_k)X(t_j,d_l)L_d(d_k,d_l) < L_{max} \\ \forall (t_i,t_j) \in E_t, \forall (d_k,d_l) \in V_d, \\ \text{where } G_t = (V_t,E_t), G_d = (V_d,E_d) \end{gathered} \tag{3}$$

G_t is the *TaskGraph*. G_d is the graph describing the relationship between devices, derived from device capability data and network topology. X is the decision variable. Each V and E represents vertices and edges of the graph. A vertex represents a device or a task, while an edge describe the link attributes and relationships between the vertices. The intuition is that if two consecutive tasks t_i and t_j are deployed on two devices d_j and d_l, the transmission latency L_d between them must be less than a threshold L_{max} that is specified by a developer.

Constraint 2: Devices must be capable of providing the assigned service.

$$\begin{gathered} r_d(d_i,k) - \sum_{j=1}^{|V_t|} X(t_j,d)r_t(t_j,k) \leq 0 \\ \forall k \in R, d \in V_d \text{ where } G_d = (V_d,E_d), \end{gathered} \tag{4}$$

Here R is a set of resources required by the tasks, including service availability, CPU utilization, GPU utilization, and bandwidth. The intuition is that a device d_i must be able to support all the tasks t_j assigned to it. The function $r_d(d_i,k)$ returns the amount of available resources k on a device d_i, and $r_t(t_j,k)$ returns the amount of resource requested by a task t_j.

Given the input data, objective function, and constraints, the *placethings* solves this linear programming problem.

Table 1: Notations (katie: just add this to make it easier to read. delete it whenever you want or leave it

Notation	Description
d_i	Device
E_d / E_t	A set of edges in the device graph / task graph
G_d / G_t	Device graph / task graph
$L_d(d_i,d_j)$	Transmission latency from d_i to d_j
$L_t(t_i,d_j)$	Execution latency of task t_i on device d_j
L_{max}	The maximum acceptable transmission latency
R	A set of resources, e.g. RAM, ROM, CPU utilization
$r_d(d_i,k)$	The amount of available resource k on device d_i
$r_t(t_t,k)$	The amount of resource k required by task t_i
t_i	Task
V_d / V_t	A set of vertices in the device graph / task graph
$X(t,d)$	Decision variable