Wait for Fresh Data? Digital Twin Empowered IoT Services in Edge Computing

Jing Li[†], Song Guo[†], Weifa Liang[‡], **Jie Wu**[¶], Quan Chen[§], Zichuan Xu[$], Wenzheng Xu[%], and Jianping Wang[‡]

[†] The Hong Kong Polytechnic Univ., [‡] City Univ. of Hong Kong, [¶] Temple Univ., [§] Guangdong Univ. of Tech., [$] Dalian Univ. of Tech., and [%] Sichuan Univ.

Outline

Digital twin technique and MEC

Digital Twins monitor physical objects and represent them in virtual worl

Figure: Concept of the digital twin technique.

Physical objects feed digital twins in cloudlets in real-time.

Digital twins provide users with fresh digital twin data.

MEC system model

Figure: An MEC with APs and cloudlets, sensors with digital twins, and users

AoI-aware IoT query services

Age of Information (AoI): measure the freshness of data, which is the time elapsed from the data generation to its usage.

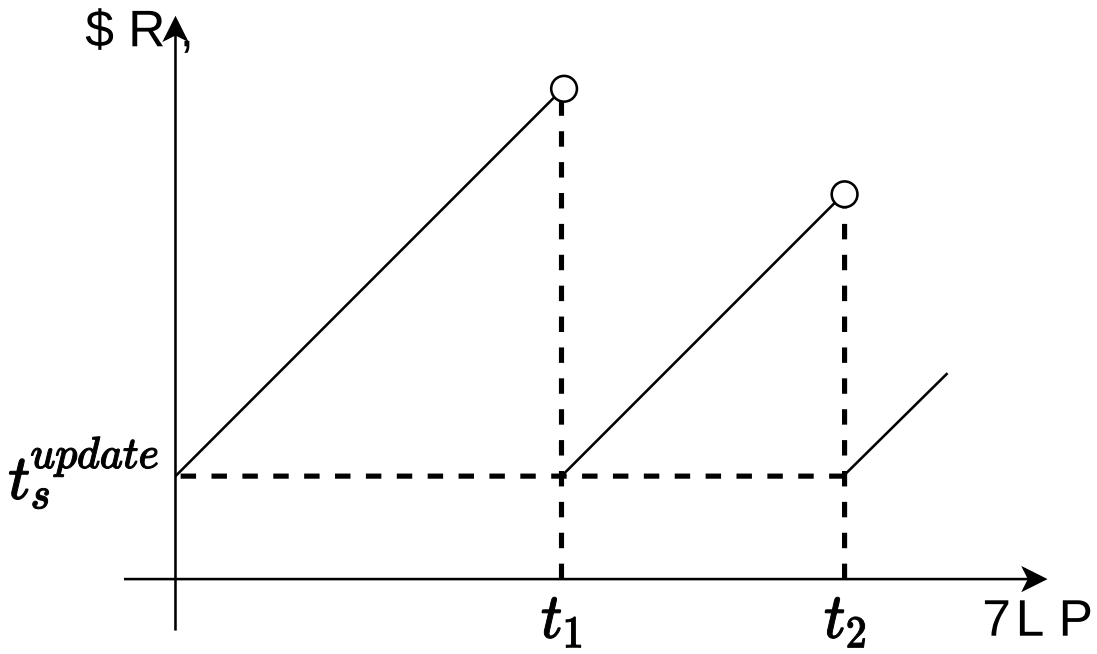


Figure: The AoI evolution of a digital twin when receiving its updates at t_1 and t_2, with t_s^{update} presenting the transmission and processing time from sensor s to its cloudnet.

AoI-aware IoT query services

Query services of IoT applications in MEC, built upon the digital twin data.

The number of data samples by each sensor is constrained because of the energy and cost limitations.

Quality of Services (QoS) is measured by two metrics:

the freshness of query results

the query service delays

AoI-aware IoT query services

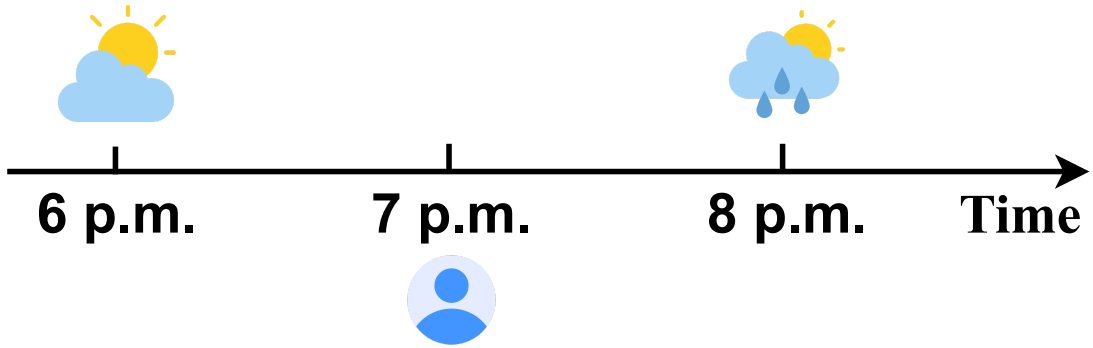


Figure: A weather forecast is updated every 2 hours, e.g., the last update at 6 p.m., the next update at 8 p.m., and now at 7 p.m. Transmission delay is 5 minutes.

Obtain the updated information at 6 p.m. The AoI is 65 minutes, and the service delay is 5 minutes.

Can also wait until 8 p.m. to get the latest information, i.e., the service delay is 65 minutes and the AoI is 5 minutes.

Challenges

Whether to use the **current data of a digital twin with a lower delay** or wait for its **next update with a lower AoI but a longer delay**?

How to **schedule data uploading** of sensors for updating their digital twins, considering limited energy and cost budgets on sensors?

How to **deploy IoT application** instances of users in cloudlets, subject to their computing capacities?

Contributions

Formulate a novel minimization problem of jointly considering the freshness of query results and query service delays.

Show the NP-hardness of the problem.

Develop an approximation algorithm with a provable approximation ratio with moderate resource violation.

Evaluate the algorithm via simulations with promising results.

User queries on digital twin data of sensors

The monitoring slotted-time horizon T *time slots*.

A set U of users with IoT applications for digital twin data of sensors

Each user u U deploys an IoT application in a cloudlet at the beginning of T.

User u issues queries (as his IoT application) for processing data from digital twins of different sensors at the beginning of different time slots.

Each sensor s S has a digital twin $DT(s)$ deployed in a cloudlet v_s, and each s can deliver at most K_s updates within T.

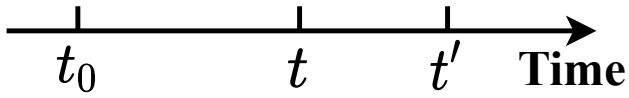
We define *the update delay* t_s^{update} of digital twin $DT(s)$

the data uploading delay from sensor s to cloudlet v_s, and

the processing delay of $DT(s)$ in cloudlet v_s.

QoS model: the weighted sum of **the AoI of query results** and **the query service delays**

A user u issues a query at t. Let t_0 be the updating time of the current data of $DT(s)$. Let t' be the next update time.



For each query, user u needs to determine whether to retrieve the current data at $DT(s)$, or wait for its next update.

Case I. If the user prefers a lower query service delay to a fresh AoI, the user retrieves the data of $DT(s)$ immediately.

Case II. The user waits for a fresher AoI until the next update of $DT(s)$, at the expense of more delays.

QoS model

$Delay(s, u)$ is the service delay consisting of the data transmission delay from $DT(s)$ to user u, and the processing delay.

The AoI of the query result is

$$W_{AoI}(u, t) = \begin{cases} Delay(s, u) + t - t_0, & \text{Case I} \\ Delay(s, u) + t_s^{update}, & \text{Case II} \end{cases} \tag{1}$$

The query service delay is

$$W_{delay}(u, t) = \begin{cases} Delay(s, u), & \text{Case I} \\ Delay(s, u) + t' + t_s^{update} - t, & \text{Case II} \end{cases} \tag{2}$$

The weighted sum is

$$\mathbf{W(u, t)} = \beta \cdot \mathbf{W_{AoI}(u, t)} + (\mathbf{1} - \beta) \cdot \mathbf{W_{delay}(u, t)}, \tag{3}$$

where β is a constant.

Problem definition

Definition

The minimization problem of joint freshness of query results and query service delays is to minimize the average weighted sum of the AoI of query results and query service delays of all queries for the given T:

$$\text{minimize} \sum_{u \in U} \sum_{t \in \mathcal{T}_u} W(u,t) / \sum_{u \in U} |T_u|.$$

Theorem

The minimization problem of joint freshness of query results and query service delays of all queries is NP-hard.

A reduction from the minimum-cost **Generalized Assignment Problem** (GAP).

Approximation algorithm

The minimization problem of joint freshness of query results and query service delays of all queries.

The core idea:

> Decompose the problem into two sub-problems: **the update scheduling problem**, and **the IoT application placement problem**.
>
> Devise an approximation algorithm, by proposing an optimal solution to the first sub-problem and an approximate solution to the second one.

Problem definitions of two sub-problems

Definition

$$W_1(u,t) = \begin{cases} \beta \cdot (t - t_0), & \text{if there is no further update} \\ \min\{\beta \cdot (t - t_0),\ t_s^{update} + (1-\beta)\cdot(t' - t)\}, & \text{if the next update is at } t' \end{cases} \quad (4)$$

The update scheduling problem is to minimize $\sum_{u \in U} \sum_{t \in \mathcal{T}_u} W_1(u,t)$, by scheduling the K_s updates for s S over T.

Definition

$$W_2(u,t) = Delay(s,u). \quad (5)$$

The IoT application placement problem is to minimize $\sum_{u \in U} \sum_{t \in \mathcal{T}_u} W_2(u,t)$, by deploying users on cloudlets, subject to their computing capacities.

Minimizing the optimization objective of the original problem is to minimize the optimization objectives of the two sub-problems independently.

Optimal algorithm for the update scheduling problem

We build an **auxiliary graph** for each sensor.

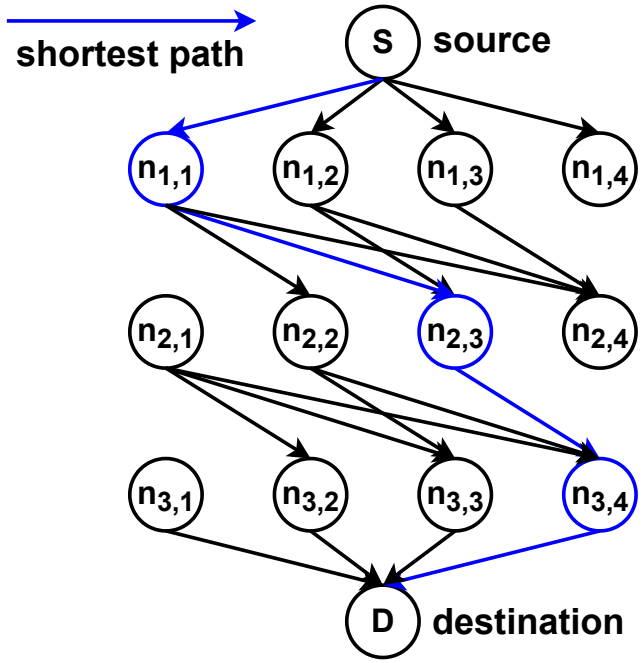


Figure: A graph for s sends K_s (=3) updates at 1, 3, and 4, over $\mathsf{T} = \{1, 2, 3, 4\}$.

Theorem

`Algorithm` *1 delivers an optimal solution to the update scheduling problem, which takes* $O(|U| \cdot |\mathsf{T}| + |\mathsf{S}| \cdot K_{max}^2 \cdot |\mathsf{T}|^2)$ *time, with* $K_{max} = \max\{K_s \mid s \quad \mathsf{S}\}$.

Approximation algorithm for the placement problem

Reduce the problem to a minimum-cost GAP, and an ILP solution for the application placement problem.

$$\text{Minimize} \quad \sum_{u \in U} \sum_{t \in \mathcal{T}_u} W_2(u,t) \tag{6}$$

subject to:

$$W_2(u,t) = \sum_{v \in V} (W_2'(u,t,v) \cdot x_{u,v}), \quad u \quad U, \quad t \quad T_u \tag{7}$$

$$\sum_{u \in U} c_u \cdot x_{u,v} \leq C_v, \quad v \quad V, \tag{8}$$

$$\sum_{v \in V} x_{u,v} = 1, \quad u \quad U \tag{9}$$

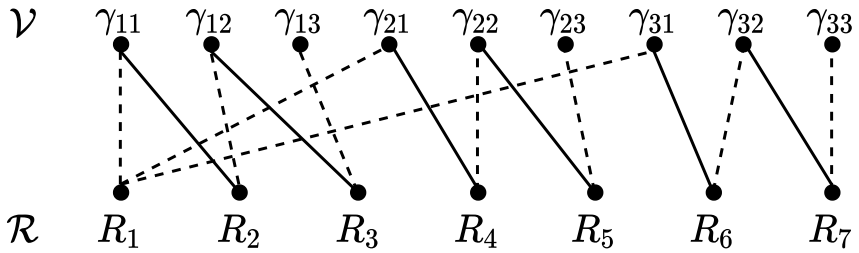
$$x_{u,v} \quad \{0,1\}, \quad u \quad U, \quad v \quad V. \tag{10}$$

By [1], we obtain (1) an optimal fractional solution OPT using LP, and then (2) build a bipartite graph to find its minimum-cost maximum matching.

[1] D. Shomys and E. Tardos. An approximation algorithm for the generalized assignment problem. Mathematical Programming, 1993.

Approximation algorithm for the placement problem

$$\mathbf{X}=\begin{array}{c} \begin{matrix} u_1 & u_2 & u_3 & u_4 & u_5 & u_6 & u_7 \end{matrix} \\ \begin{bmatrix} \mathbf{1/3} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{1/3} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{1/3} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} \end{bmatrix} \end{array} \begin{matrix} v_1 \\ v_2 \\ v_3 \end{matrix}$$



Each solid edge corresponds to 2/3
Each dashed edge corresponds to 1/3

Figure: An illustrative example of a bipartite graph B, with 3 cloudlets and 7 users.

Theorem

The solution value by `Algorithm` *2 for the placement problem is no more than that of the optimal solution, and the amount of computing resource consumed in each cloudlet v V is no more than twice its computing capacity.* `Algorithm` *2 takes* $O(|U|^3 \cdot |V|^3)$ *time.*

Approximation algorithm

`Algorithm` 3, via adopting the update scheduling of sensors by `Algorithm` 1 and the application placement of users by `Algorithm` 2.

Theorem

There is an approximation algorithm, `Algorithm` *3, for the minimization problem of joint freshness of query results and query service delays of all queries.*

The solution value is no more than that of the optimal one, and the amount of computing resource consumed in each cloudlet is no more than twice its computing capacity.

`Algorithm` *3 takes* $O(|U| \cdot |\mathrm{T}| + |\mathrm{S}| \cdot K_{max}^2 \cdot |\mathrm{T}|^2 + |U|^3 \cdot |V|^3)$ *time.*

Benchmarks

Evaluated `Algorithm 3` (`Alg.3`) against the three benchmarks:

Wait: Each application is deployed to minimize the average transmission delay. Each sensor delivers its updates evenly over the time, and each query waits for the next update of the digital twin.

NoWait: Similar to `Wait`, but queries retrieve current data.

Random: Applications are deployed in cloudlets randomly. Sensors randomly send updates to digital twins, while queries randomly retrieve the current data of digital twins or wait for their updates.

Algorithm performance with network sizes from 50 to 25

Alg. 3 outperforms the benchmarks by no less than 9.1%, and the computing capacity of each cloudlet is violated by no more than 1.4%.

Figure: The performance

Figure: The running time

Figure: Utilization ratios of cloudlets by Alg. 3

Investigated the impact of the number of updates on th performance of Alg 3.

Digital twins obtain much fresher data through more updates, and a larg number of updates leads to a larger constructed auxiliary graph, which lea to longer running time.

Figure: The performance

Figure: The running time

Studied the impact of parameter associated with the AoI

A large means we focus on minimizing AoI, and the impact of the valu
of on the running time of Alg :3 is negligible.

Figure: The performance

Figure: The running time

Conclusions

Formulate a novel minimization problem

- of joint considerations of the freshness of query results and query service delays for IoT service queries, and
- to minimize the average weighted sum of AoI of query results and query service delays of all queries.

Devise an approximation algorithm

- with moderate resource violations, and
- evaluate the algorithm performance via simulations.

Future work:

(1) Propose approximation algorithms without any resource violation.

(2) Study system dynamics: uncertain request arrivals and mobility.