Priority Based Service Placement Strategy in Heterogeneous Mobile Edge Computing

Authors: Meiyan Teng, Xin Li, Xiaolin Qin, and Jie Wu

Reporter: Meiyan Teng **Email:** myteng@nuaa.edu.cn

Nanjing University of Aeronautics and Astronautics, China

Contents

Ⅰ. Introduction

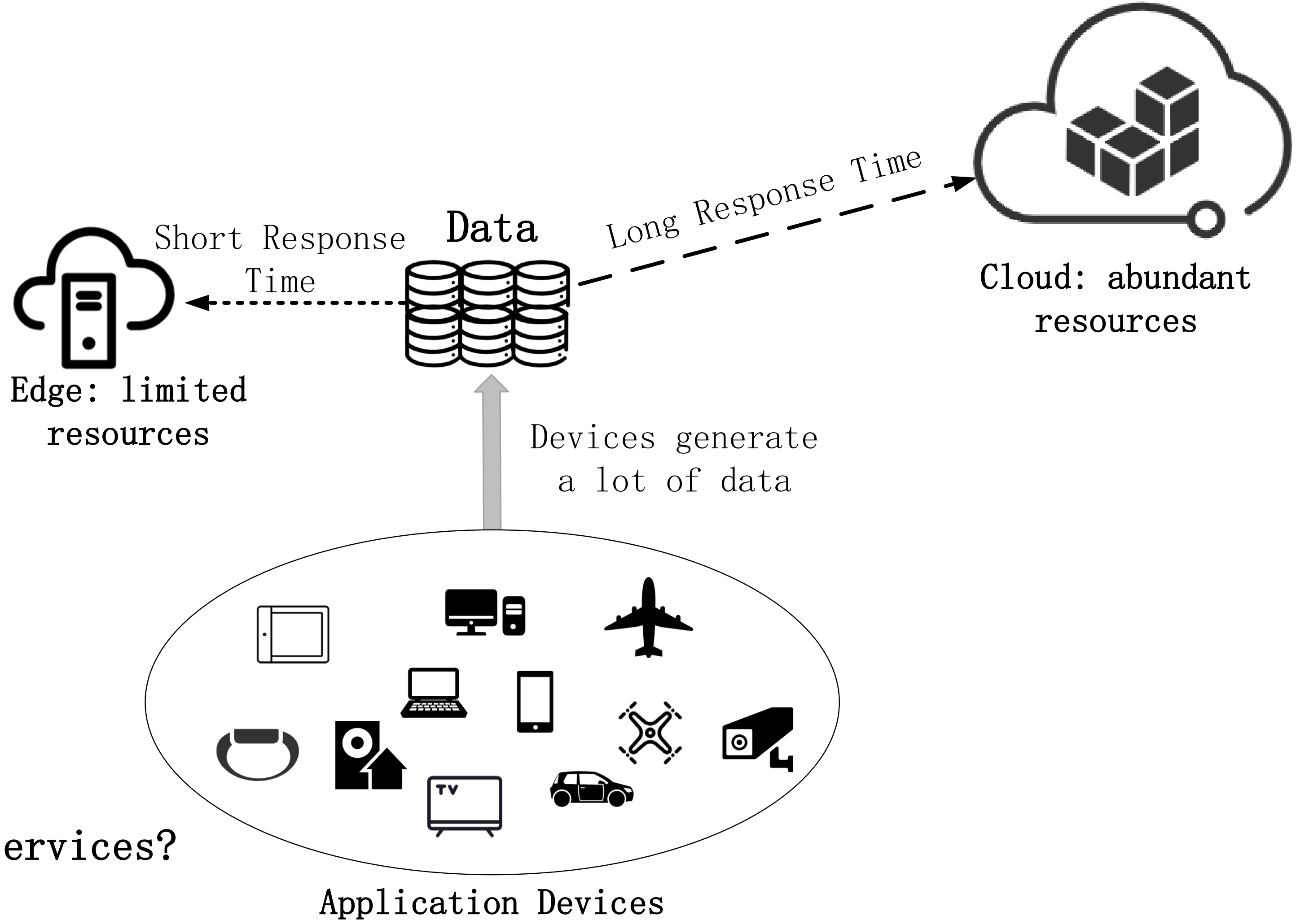
1.1 Motivation

Service:
Delay sensitive;
large amount of data

Cloud:
Advantage: sufficient resources
Defect: long response time

Edge:
Advantage: short response time
Defect: limited resources

How to minimum the average response time of services?



1.2 Related work

1. Part of the service placement strategy is investigated in **the homogeneous environment** (the edge node and service performance are certain);

2. Some studies do not consider **the limitation of node resources**;

3. Most of research do not consider **the uneven distribution of service load**.

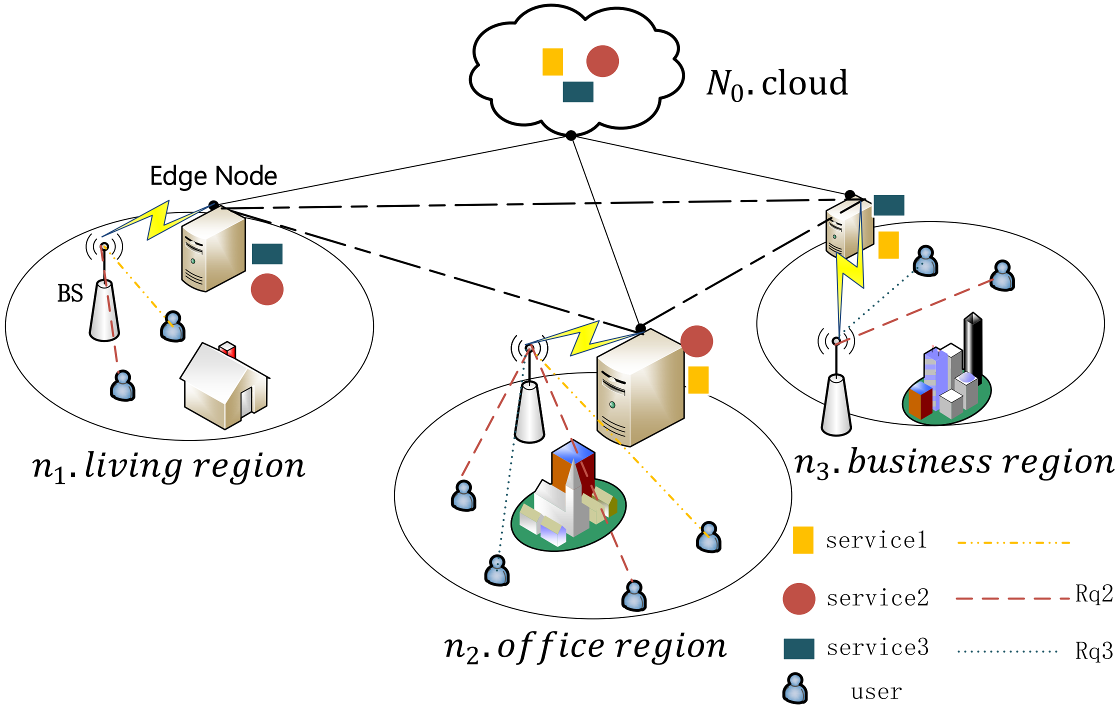
1.3 Contribution

Heterogeneity:

The **node resource**;
The **demand** of service resource;
The **service load distribution**.
The **communication conditions** between cloud and edge node;

↓

a priority placement (2P) algorithm



Ⅱ. Problem Statement

2.1 Formulation

Load distribution: $\Phi_{l,n}$

Schedule delay: $T_{m,n}^{l}$

Placement strategy X:

$$\mathrm{x}_{l,n} = \begin{cases} 1, & service\ l\ placed\ on\ the\ node\ n \\ 0, & otherwise \end{cases}$$

Response time Y:

$$y_n^l = \Theta\left(\sum_{m=0}^{|N|} x_{l,m} = 0\right) \cdot \Gamma_n + \Theta\left(\sum_{m=0}^{|N|} x_{l,m} \neq 0\right) \cdot min\{T_{m,n}^{l} | x_{l,m} = 1\}$$

2.2 Aim

$$min \sum_{l=0}^{|S|} \sum_{n=0}^{|N|} y_{l,n}$$

$$s.t. \sum_{l=0}^{|S|} x_{l,m} \cdot r_l \leq R_m \qquad \forall m \in N$$

$$x_{l,m} \in \{0,1\} \qquad \forall l \in S, \forall m \in N$$

Theorem:

The service placement in a heterogeneous MEC system is NP-hard problem.

南京航空航天大學
NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS

III. Placement Strategy

3.1 Idea

Why we propose 2P algorithm to minimize the response time?

Reason1:

Uneven load distribution → More loaded nodes with higher values → prioritize the high-value nodes

Reason2:

the high-value nodes' resources are tight → can not meet all requests → Prioritize services according performance

Algorithm 1 Total Delay Algorithm

Input: Node set $N(n \in N)$, attributes: $\langle R_n, \Gamma_n \rangle$; Service set $S(l \in S)$. attributes: $\langle r_l, P_l \rangle$; Schedule delay $\mathbf{T}$; Service load distribution $\boldsymbol{\Phi}$; Upper limit of replicas θ.

$Q = averageDelay(T, \Phi)$;
$G = nodeCandidateSet(Q)$;
$L = initServiceCandidateSet(G, 1)$;
$X = servicePlacement(L, G, Q, r, R, \Phi, \theta)$;
for each $l \in S$ **do**
  $P_l = getSum(X)$;
  if $P_l == 0$ **then**
    for each $n \in N$ **do**
      $y_{l,n} = \Gamma_n$;
    end for
  else
    for each $n \in N$ **do**
      $y_{l,n} = minResponseTime(X, T)$;
    end for
  end if
end for
for each $l \in S$ **do**
  for each $n \in N$ **do**
    $t = t + y_{l,n} \cdot \Phi_{l,n}$;
  end for
end for

Output: Total delay time is t.

Node value (priority) ?

Service placement strategy ?

Response time

Total delay time

3.2 Priority

1. Average delay time:

$$Q_m^l = \frac{\sum_{n=0}^{|N|} T_{m,n}^l \cdot \Phi_{l,n}}{\sum_{n=0}^{|N|} \Phi_{l,n}}$$

2. The ideal node set:

$$G_{l,k} = \left\{ n_{p_1}, \cdots n_{p_i} \cdots \middle| Q_{n_{p_1}}^l \le \cdots \le Q_{n_{p_i}}^l \le \cdots, n_{p_i} \in N \right\}$$

3. The initial service candidates set:

$$L_n = \left\{ l_{p_1}, \cdots l_{p_i} \cdots \middle| G_{l_{p_i},1} = n, l_{p_i} \in S \right\}$$

The node priority: $|L_n|$

3.3 strategy

Algorithm 2 Service Priority Placement Algorithm

Input: Average delay **Q**; node candidate set **G**; service candidate set **L**; attributes of nodes $\langle R_n, \Gamma_n \rangle$; attributes of services $\langle r_l, P_l \rangle$; service load distribution $\boldsymbol{\Phi}$; Upper limit of replicas θ .

while (! isEmpty (L)) **do**
  $e \leftarrow \text{argmax}_{n \in N} |\boldsymbol{L_n}|$;
  Order $L_e = \{l_{p_1}, l_{p_2}, \cdots, l_{p_k}\}$,so that $\Omega_{l_{p_i}} \geq \Omega_{l_{p_{i+1}}}, \forall i < k$;
  for each $l_{p_i} \in L_e$ **do**
    if $r_{l_{p_i}} \leq R_e$ **then**
      $x_{l_{p_i},e} = 1$;
      $R_e \leftarrow R_e - r_{l_{p_i}}$;
      $P_{l_{p_i}} ++$;
    else
      $x_{l_{p_i},e} = 0$;
    end if
    if $P_{l_{p_i}} < \theta$ **then**
      $e' \leftarrow findNextNode(G, l_{p_i}, e)$;
      update$(\Omega_{l_{p_i}})$;
      $L_{e'} \leftarrow L_{e'} \cup l_{p_i}$;
    end if
  end for
  $Clear(L_e)$;
end while

Output: Service placement strategy is **X**.

Service priority:

$$\Omega_{l,e} = \frac{\Delta Q_l + k_1 \cdot \sum_{n=0}^{|N|} \Phi_{l,n}}{Q_e^l + k_2 \cdot P_l}$$

Ⅳ. Performance Evaluation

4.1 Comparison

Greedy algorithm:

place the service with the shortest latency on each node.

Non-loaded algorithm:

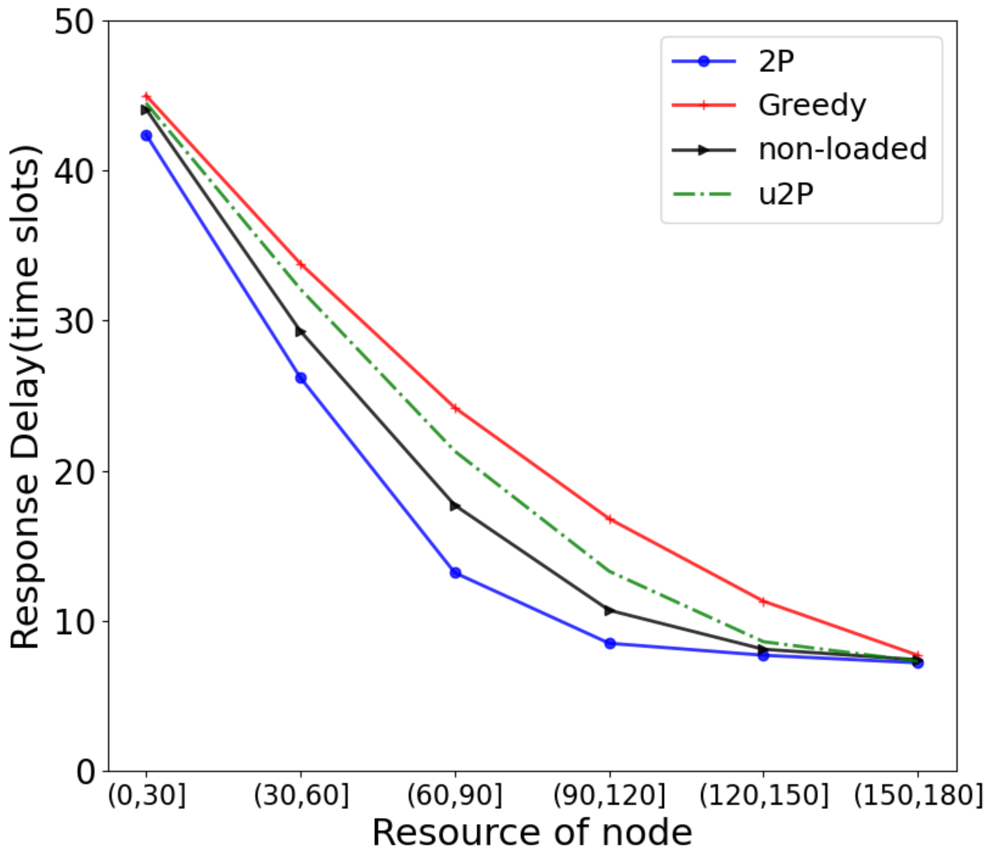
average delay: $Q_e'^l = \frac{\sum_{n=0}^{|N|} T_{e,n}^l}{|N|}$, service priority: $\Omega'_{l,e} = \frac{\Delta Q_l}{Q_e^l + k \cdot P_l}$

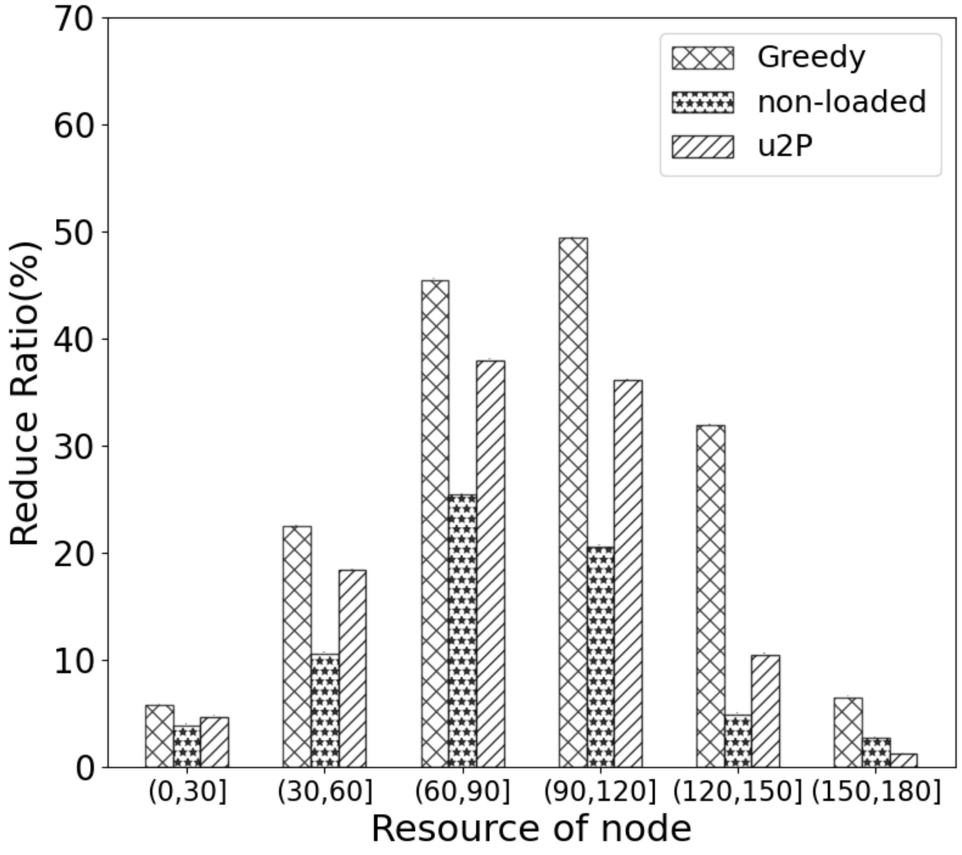
Unitary priority placement (u2p) algorithm:

$$\Omega''_{l,e} = Q_e^l$$

4.2 Results

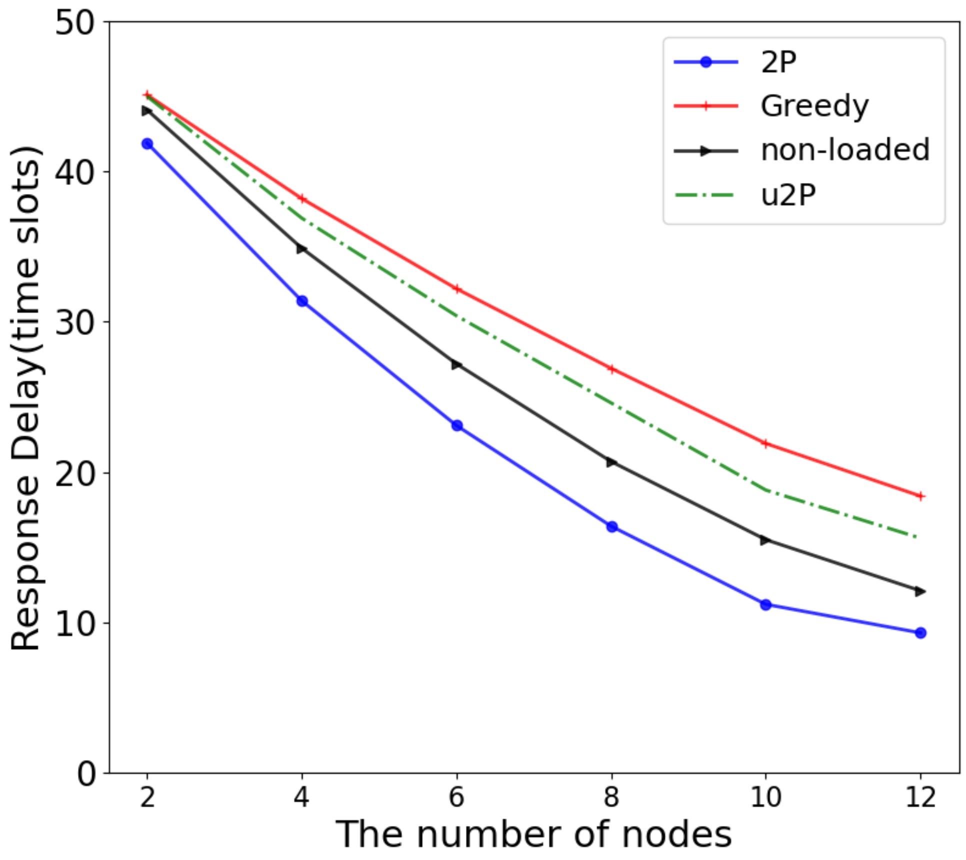
Test1:Change the resource of nodes R_n, where $\theta = 2$ and $N = 9$

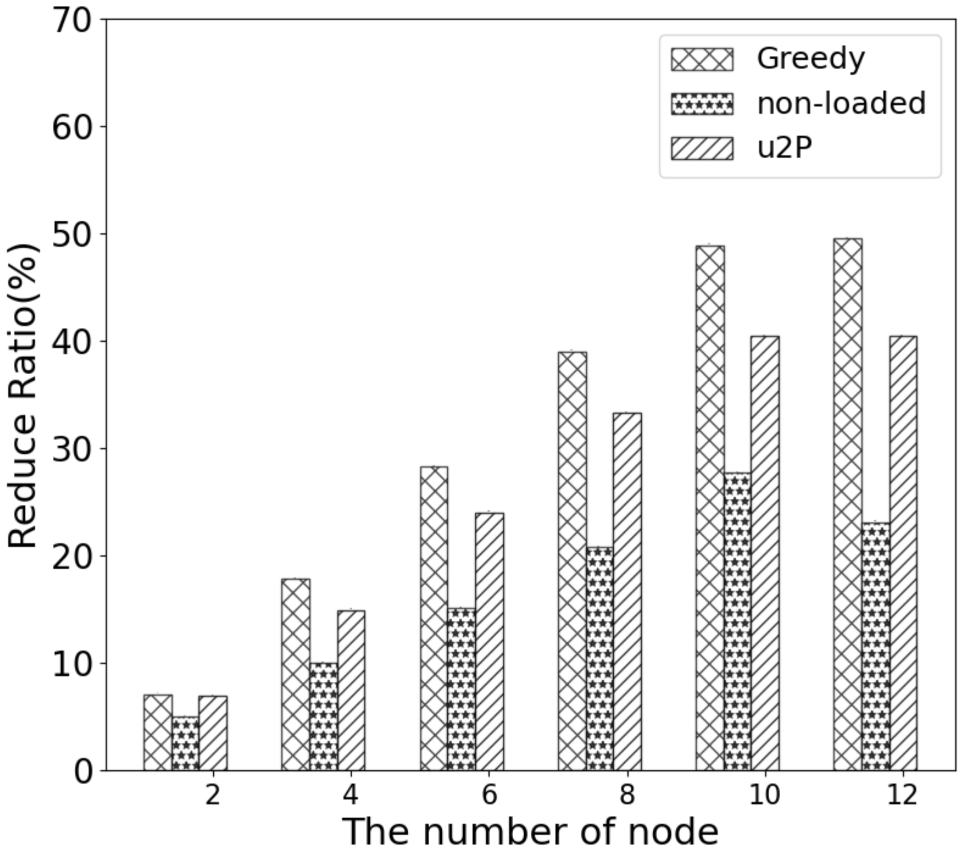




4.2 Results

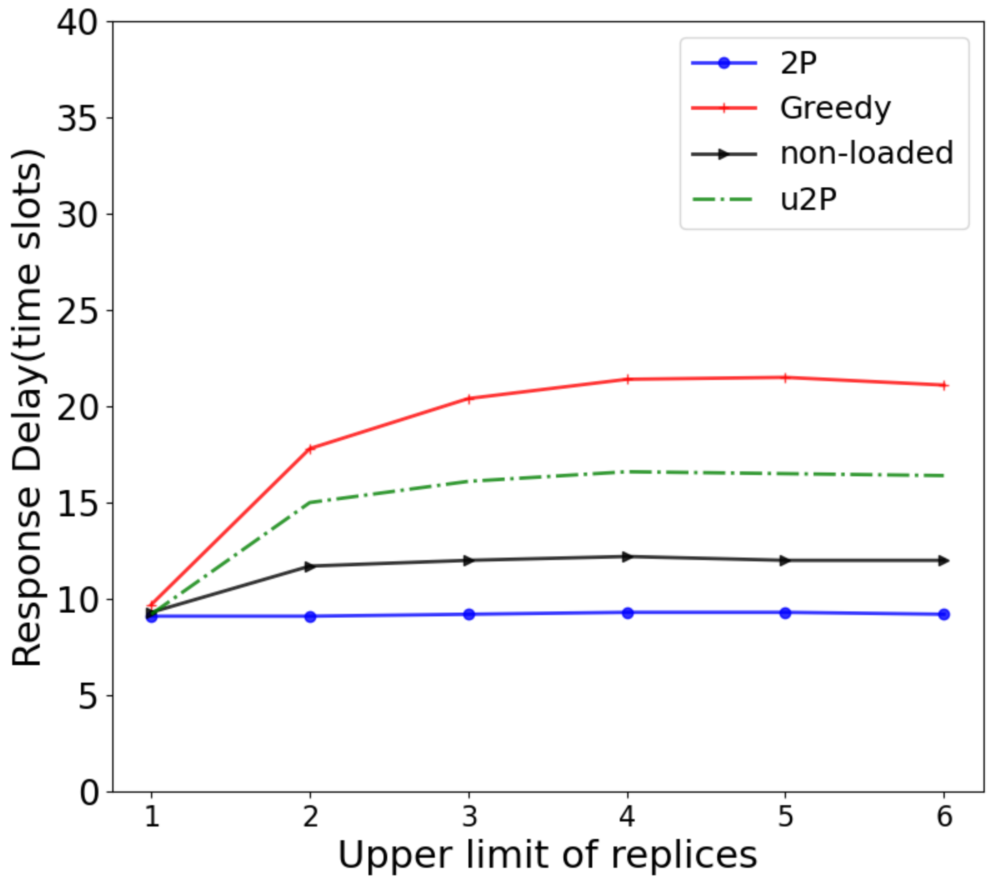
Test2:Change the number of nodes N, where $\theta = 2$ and $R_n \in [50,100]$

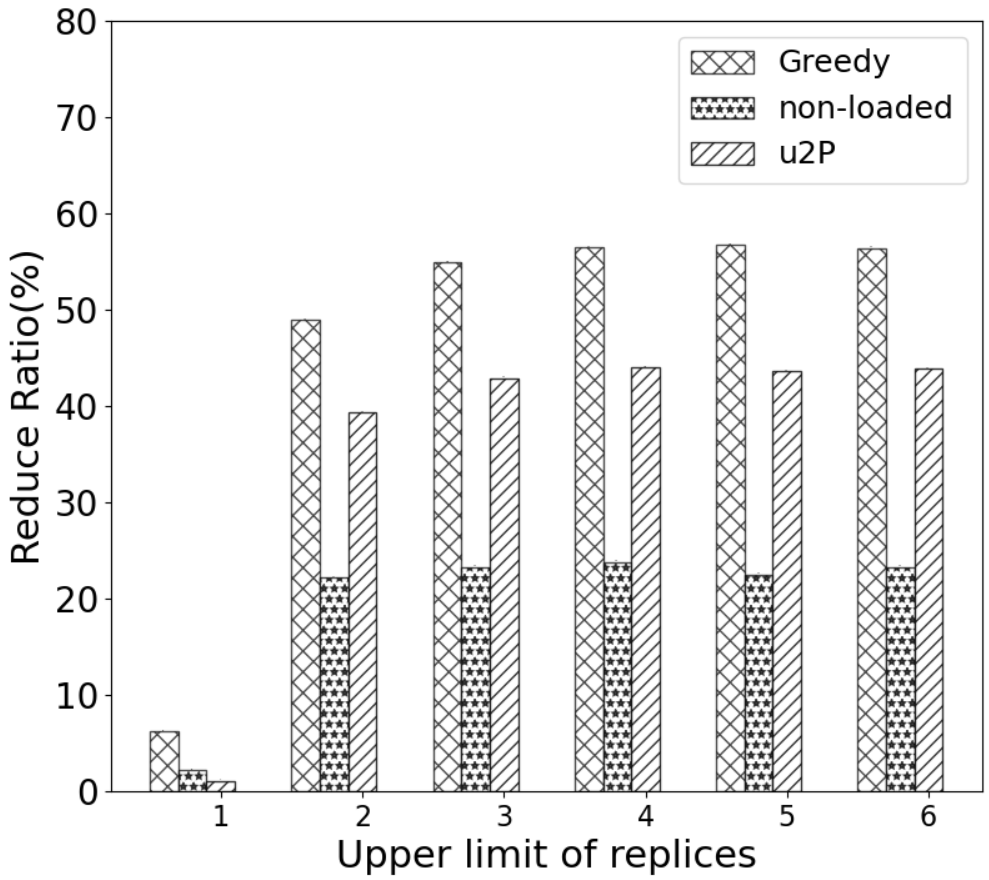




4.2 Results

Test3:Change the upper limit of service replicas θ, where $\mathrm{N} = 12$ and $R_n \in [50,100]$





V. Conclusion

5. Conclusion

（1）Defined the priority of the nodes and services according to their contribution;

（2）Propose a priority placement (2P) algorithm;

（3）Conduct simulations and the results show that the 2P algorithm has better performance.

南京航空航天大學
NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS

Thank you for listening!