20250522 Progress Report

Reporter: Kuo-Wei, Lien 連國煒
Adviser: Prof. Wan-Jiun, Liao
Prof. De-Nian, Yang

iRL
Internet Research Lab

Outline

- Introduction
- System Model
- Problem Formulation
- Algorithm
- Appendix

Introduction

Introduction

- Cooperative caching scheme for multi-view 3D videos in Low-earth-orbit (LEO) satellite networks that optimizes content placement and delivery by leveraging inter-satellite links (ISL) and DIBR synthesis

- **Multi-view 3D Video**

- **Cooperative Caching on LEO satellite network**

- **DIBR Synthesis**

Related Works

[1] M. Yeh, C. -H. Wang, D. -N. Yang, J. -T. Lee and W. Liao, "Mobile Proxy Caching for Multi-View 3D Videos With Adaptive View Selection," in *IEEE Transactions on Mobile Computing*, vol. 21, no. 8, pp. 2909-2921, 1 Aug. 2022

[2] R. Zhao, Y. Ran, J. Luo and S. Chen, "Towards Coverage-Aware Cooperative Video Caching in LEO Satellite Networks," *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*, Rio de Janeiro, Brazil, 2022

[3] S. Liu, X. Hu, Y. Wang, G. Cui and W. Wang, "Distributed Caching Based on Matching Game in LEO Satellite Constellation Networks," in *IEEE Communications Letters*, vol. 22, no. 2, pp. 300-303, Feb. 2018

	[1]	[2]	This Work
Content Type	Multi-view 3D video	2D video content	Multi-view 3D video
Network Type	Mobile proxy networks	LEO satellite networks	LEO satellite networks
DIBR Synthesis	V	X	V (on user)
Coverage Scope	Localized to urban or high-demand areas	Global Coverage	Global Coverage

- Previous works [1] utilize DIBR to boost transmission efficiency
  - Caching on servers and managing caches by centralized algorithm
- [2], [3] use ISL to achieve wide coverage areas and low access latency
- [4] manages cache with the information provided by ground station

[4] J. Li, K. Xue, J. Liu, Y. Zhang and Y. Fang, "An ICN/SDN-Based Network Architecture and Efficient Content Retrieval for Future Satellite-Terrestrial Integrated Networks," in *IEEE Network*, vol. 34, no. 1, pp. 188-195, January/February 2020

Research Challenges

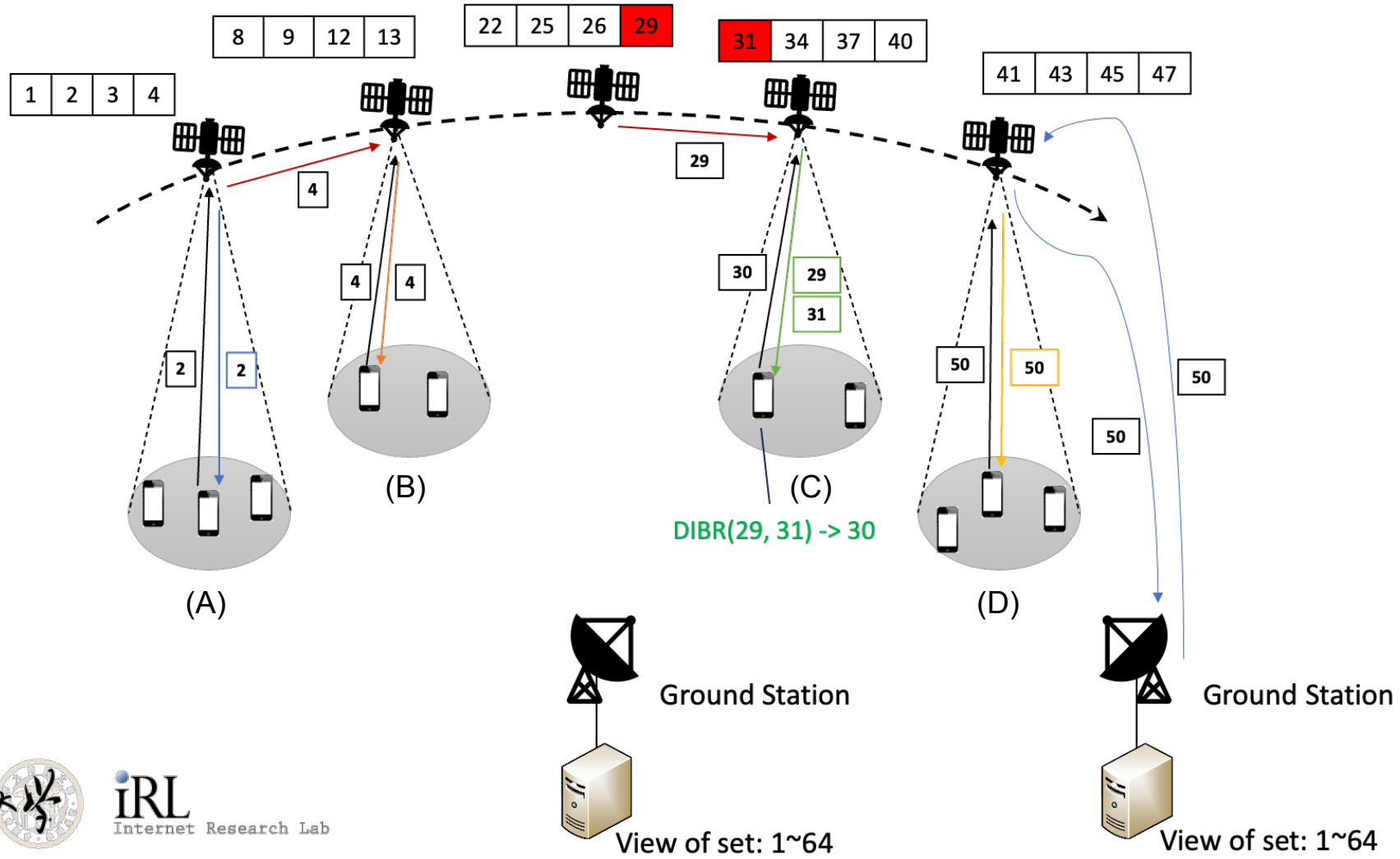
- Moving caching from the terrestrial to LEO satellites introduce some challenges
- **Limited on-board computation and power**
  - Terrestrial servers often have powerful hardwares and enough power
  - LEO satellites have limited computing resources and rely on solar power
- **Unstable and limit bandwidth**
  - Links between satellites and between satellite and user are **wireless** and **long distance**
  - The LEO satellite's number of simultaneous connections has a maximum limit
- **LEO satellite seldom connects with ground station**
- **Each satellite performs independently and manages its cache**
  - Advantage and Disadvantage?
- **View fetching and DIBR synthesis trade-off**

Motivation

- Accessibility high bandwidth services in remote and underserved regions
- Repeated transmission of the same views to multiple users introduces redundant data transfer
  - Caching multi-view videos on LEO satellites can effectively reduce video transmission
- Traditional caching replacement policies **(LRU, LFU)** would not yield good results
- Existing LEO caching problems do not consider the **energy** and **bandwidth**

System Model

LEO Caching for Multi-View 3D Videos with Inter-satellite Links



Views Caching Policy

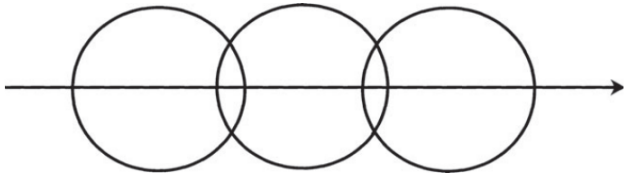
- When LEO satellite receives the user's request
- **Cache Hit**
  A. If requested view cached in access LEO
  B. If the access LEO has not cache, but nearby LEOs have could sharing requested view through inter-satellite link
  C. If requested view is not cached but its access LEO and nearby LEOs can return left and right views to user and be synthesized requested view by user
- **Cache Miss**
  D. if the above cases not satisfied and should fetch the request view from remote server (ground station)

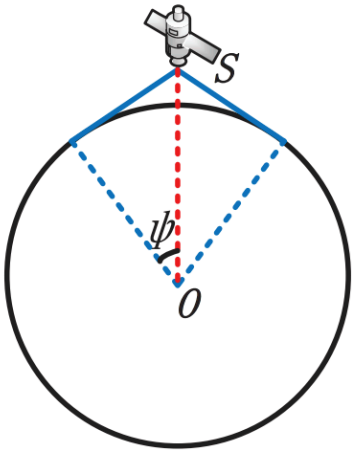
System Model

- K users: $\mathbf{K} = \{1,2,\ldots,k\}$
- N LEO satellites: $\mathbf{N} = \{1,2,\ldots,n\}$
- G Ground stations: $\mathbf{G} = \{1,2,\ldots,g\}$
- V : the universal set of views in a multi-view 3D video
- $V_n(t)$: the state of the cache storing a subset of views in LEO satellite n at time t
- z_i : the size of view i
- $\sum_{i \in V_n(t)} z_i \leq Z$: The total size of cached view in a LEO satellite no more than Z
  - Every LEO satellite have the same cache storage
  - Z: the size of the LEO cache size

System Model: Satellite Model

- Service Area Model: $A_n(t) = \{x(R,\theta,\Phi) \mid \sin\theta \sin\theta_n \cos(\Phi - \Phi_n) + \cos\theta \cos\theta_n \leq \cos\psi\}$
  - R : the radius of earth
  - Φ_n : the longitude of satellite n, $\theta_n = \left|\frac{\pi}{2} - n_{latitude}\right|$
  - ψ : **half cone angle** between covered area and Earth's core
  - The point set of all points in the area covered by satellites: $A_{SOC}(t) = \bigcup_{n=1}^{n} A_n(t)$





- Coverage Model:
  - $\psi = arccos(\frac{R}{R+h} cos\theta_{min}) - \theta_{min}$
  - The coverage area $S = 2\pi R^2(1 - cos\psi)$
  - A larger value of ψ implies the decrease in the coverage

[7] H. Zhou, L. Liu and H. Ma, "Coverage and Capacity Analysis of LEO Satellite Network Supporting Internet of Things," ICC 2019 - 2019 IEEE International Conference on Communications (ICC), Shanghai, China, 2019, pp. 1-6

[8] R. Deng, B. Di, H. Zhang, L. Kuang and L. Song, "Ultra-Dense LEO Satellite Constellations: How Many LEO Satellites Do We Need?," in *IEEE Transactions on Wireless Communications*, vol. 20, no. 8, pp. 4843-4857, Aug. 2021

System Model: Communication Model

- LEO_n to user k transmission:
  - $R_{n,k} = B_{n,k} log_2\left(1 + \frac{P_n \cdot G_{n,k}}{\sigma^2}\right)$
- Inter-satellite link transmission:
  - $R_{n,n+1} = B^{ISL} log_2\left(1 + \frac{P_{ISL} \cdot G_{n,n+1}}{\sigma^2}\right)$
- GS_g to LEO_n transmission:
  - $R_{g,n} = B^{GS} log_2\left(1 + \frac{P_{GS} \cdot G_{g,s}}{\sigma^2}\right)$

- B: the channel bandwidth
- P : the transmit power
- G : the channel gains
- $d_{n,k} = -Rsin\theta_{n,k} + \sqrt{(Rsin\theta_{n,k})^2 + h^2 + 2hR}$
  - the distance between LEO_n and user k
- $\theta_{n,k}$: the elevation angle of user k with respect to LEO_n

[2] R. Zhao, Y. Ran, J. Luo and S. Chen, "Towards Coverage-Aware Cooperative Video Caching in LEO Satellite Networks," *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*, Rio de Janeiro, Brazil, 2022

[7] H. Zhou, L. Liu and H. Ma, "Coverage and Capacity Analysis of LEO Satellite Network Supporting Internet of Things," *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, Shanghai, China, 2019, pp. 1-6

[8] R. Deng, B. Di, H. Zhang, L. Kuang and L. Song, "Ultra-Dense LEO Satellite Constellations: How Many LEO Satellites Do We Need?," in *IEEE Transactions on Wireless Communications*, vol. 20, no. 8, pp. 4843-4857, Aug. 2021

[9] J. -H. Chen, W. -C. Kuo and W. Liao, "SpaceEdge: Optimizing Service Latency and Sustainability for Space-Centric Task Offloading in LEO Satellite Networks," in IEEE Transactions on Wireless Communications, vol. 23, no. 10, pp. 15435-15446, Oct. 2024

System Model: Satellite Energy Model

- The limited resources of LEO satellites pose challenges to maintaining service continuity
- Battery Variables:
  - E_n^H: the energy that reaches satellite n
  - $E_n(t)$: battery level of satellite n at the start of time slot t
  - E_n^{max}: maximum battery capacity
- Energy Consumption:
  - **Transmission**: $E_n^{trans}(t) = P(t) \cdot \frac{z_i}{R}$
    - transmit power $P(t)$ of satellite n by downlink view to user or ISL
  - **Cache Policy Processing (const.)**: E_n^{cache}
- Energy Constraint: $0 \leq E_n^{trans}(t) + E_n^{cache}(t) \leq E_n(t), \quad \forall n, t$
- The battery level of LEO_n for the next time slot: $E_n(t+1) = E_n(t) - E_n^{trans}(t) - E_n^{cache} + E_n^H(t)$

[10] M. M. Gost, I. Leyva-Mayorga, A. Pérez-Neira, M. Á. Vázquez, B. Soret and M. Moretti, "Edge Computing and Communication for Energy-Efficient Earth Surveillance with LEO Satellites," *2022 IEEE International Conference on Communications Workshops (ICC Workshops)*, Seoul, Korea, Republic of, 2022

System Model: View Request Model

- Arriving request: $R_n(t)$ (UTs are uniformly distributed in SOC)
  - A request here is a range of views for users to adaptively change the view angle at different time
  - From a newly joined user
  - From a leaving user (the last request before leaving)
  - From an existing user
  - LEO satellite n receives a request in time t if user is covered by satellite: $k(R, \theta, \Phi) \in A_{SOC}(t)$

- The request range of the users: $\left[R_n(t) - \lfloor \frac{B}{2} \rfloor, R_n(t) + \lfloor \frac{B}{2} \rfloor\right]$
  - B is the size of a request range
  - Fetch a range of views for each user to **support the rapid view change**

[1] M. Yeh, C. -H. Wang, D. -N. Yang, J. -T. Lee and W. Liao, "Mobile Proxy Caching for Multi-View 3D Videos With Adaptive View Selection," in *IEEE Transactions on Mobile Computing*, vol. 21, no. 8, pp. 2909-2921, 1 Aug. 2022

Problem Formulation

Problem Formulation

- An instantaneous cost function $C(t)$ at time t for each user request:

$$C(t) = c_m \sum_i 1_{\{i \notin V_n(t) \cup V_n^{syn}(t)\}} + \sum_{i \in V_n^f(t)} c_{n,f}(z_i) + \sum_{i \in V_n^s(t)} c_{n,s}(z_i, d_i)$$

$$+ c_p(v_l, v_r) \sum_j 1_{j \in \{V_n^{DIBR}(t)\}} + \sum_{i \in V_n^{ISL}(t)} h \cdot c_{ISL}(z_i, d_i^{ISL})$$

1. c_m : the penalty cost for a cache miss (the additional processing latency for cache miss)
2. $c_{n,f}$: the cost to fetch view i with size z_i from remote server(GS)
3. $c_{n,s}$: the cost to return a view with size z_i to the user by data rate d_i
4. c_p : DIBR synthesis cost: $c_p(v_l, v_r) = \alpha \cdot |v_l, v_r| + T_{DIBR}$
5. c_{ISL} : the cost to transmit view i utilizing ISL with size z_i by data rate d_i^{ISL}
6. h: the number of hops

[1] M. Yeh, C. -H. Wang, D. -N. Yang, J. -T. Lee and W. Liao, "Mobile Proxy Caching for Multi-View 3D Videos With Adaptive View Selection," in *IEEE Transactions on Mobile Computing*, vol. 21, no. 8, pp. 2909-2921, 1 Aug. 2022

[11] E. Liotou, K. Samdanis, E. Pateromichelakis, N. Passas and L. Merakos, "QoE-SDN APP: A Rate-guided QoE-aware SDN-APP for HTTP Adaptive Video Streaming," in *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 3, pp. 598-615, March 2018

Problem Formulation

- $V_n^{syn}(t)$: the views that can be synthesized by $V_{n-h}(t)$, $V_n(t)$, $V_{n+h}(t)$ according to the DIBR constraint D
  - $V_n^s(t) = \{i \mid i \notin V_{n-h}(t) \cup V_n(t) \cup V_{n+h}(t)\}$
  - h : the number of hops

- $V_n^f(t) \subseteq V \backslash V_n(t)$: the view set required by LEO_n to be fetched from the GS

- $V_n^{DIBR}(t)$: views are synthesized on the user equipment side
  - LEO satellite only needs to transmit the two reference views (left and right) to the user
  - $V_n^{DIBR}(t) = \{i \mid i \notin V_n(t)$ and both left and right reference views (v_l, v_r) for i exist in $V_n(t)\}$

[1] M. Yeh, C. -H. Wang, D. -N. Yang, J. -T. Lee and W. Liao, "Mobile Proxy Caching for Multi-View 3D Videos With Adaptive View Selection," in *IEEE Transactions on Mobile Computing*, vol. 21, no. 8, pp. 2909-2921, 1 Aug. 2022

Problem Formulation

- $V_n^e(t) \subseteq V_n(t) \cup V_n^f(t)$: the view to set be replaced
  - when the cache space of LEO_n is full and ensuring $\sum_{i \in V_n(t)} z_i \leq Z$

- $V_n^s(t) \subseteq V_n(t) \cup V_n^f(t) \cup V_n^{ISL}(t)$: the set of views returned to user
  - where the range of returned views needs to be in $\left[R_n(t) - \lfloor \frac{B}{2} \rfloor, R_n(t) + \lfloor \frac{B}{2} \rfloor\right]$

- $V_n^{ISL}(t)$: the view set utilizing inter-satellite links (ISL)
  - $V_n^{ISL}(t) = \left\{ i \mid i \in V_{n-1}(t) \cup V_{n+1}(t) \backslash V_n(t) \right\}$

[1] M. Yeh, C. -H. Wang, D. -N. Yang, J. -T. Lee and W. Liao, "Mobile Proxy Caching for Multi-View 3D Videos With Adaptive View Selection," in *IEEE Transactions on Mobile Computing*, vol. 21, no. 8, pp. 2909-2921, 1 Aug. 2022

Problem Formulation

- Objective function: **minimize** the total cost $C(t)$ in T time slots for each user request

$$min \sum_{t=1}^{T} C(t)$$

- **Constraint:**

1. LEO caching storage: $\sum_{i \in V_n(t)} z_i \leq Z$

2. DIBR synthesis constraint: $|v_l - v_r| \leq D,\ \forall v_l, v_r$ used for synthesis

3. Coverage Constraint: $k(R, \theta, \Phi) \in A_{SOC}(t)$

Problem Formulation

- Objective function: **minimize** the total cost $C(t)$ in T time slots for each user request

$$min \sum_{t=1}^{T} C(t)$$

- **Constraint:**

4. Energy Constraint: $0 \leq E_n^{trans}(t) + E_n^{cache}(t) \leq E_n(t), \quad \forall n, t$

5. Bandwidth Constraint: $\sum_k B_{n,k}(t) + \sum_m B_{n,m}(t) + \sum_g B_{g,n}(t) \leq B_n^{max}$

Algorithm

Problem Hardness

Theorem. The LEO Satellite Caching for Multi-View 3D Videos With Inter-Satellite Links Problem (LEO-MV3D-ISL) is NP-hard.

Proof. We will prove this theorem by reducing the Partition Problem to an offline version of the LEO-MV3D-ISL.

Given a multi-set $\mathcal{S} = \{a_1, a_2, \ldots, a_{|\mathcal{S}|}\}$ of positive integers (with sum $2S$), the partition problem is to decide whether $\mathcal{S}$ can be partitioned into two subsets $\mathcal{S}_1$ and $\mathcal{S}_2$ such that the sum of the elements in $\mathcal{S}_1$ equals the sum of the elements in $\mathcal{S}_2$. Given the instance of the partition problem, we map this to an instance of LEO-MV3D-ISL as follows:

1) For each view i, the size z_i corresponds to the partition problem instance's a_i. Specifically, let $|V| = |\mathcal{S}|$ and $z_i = a_i$ for $i = \{1, 2, ..., |V|\}$.
2) Set $|N| = 2$ (representing two LEO satellites) and the cache size Z to S.
3) Let $B = 0$ and $D = 0$, meaning each request corresponds to a single view and cannot be obtained through synthesis.
4) Set $h = 1$ to ensure that if a requested view is not stored on the directly connected LEO satellite, it can still be retrieved via one inter-satellite link transmission.

To complete the proof, it suffices to show that $\mathcal{S}$ is a **YES-instance** if and only if the constructed instance has no cache miss.

If $\mathcal{S}$ is a **YES-instance**, it can be partitioned into two subsets such that the sum of the elements in each subset is $\mathcal{S}$. Thus, the corresponding views can be divided into two groups and stored separately in the caches of the two LEO satellites. As a result, regardless of the incoming request, it can always be retrieved from one of the caches, ensuring that no cache miss occurs.

In an opposite way, we prove this by contradiction. Suppose that $\mathcal{S}$ is not a **YES-instance**, meaning it cannot be divided into two subsets whose sums are both $\mathcal{S}$. This implies that the constructed instance cannot store all the views within the given cache constraints. As a result, there will be at least one request that cannot be served from the caches, leading to a cache miss, which contradicts the assumption that no cache miss occurs.

Since we have demonstrated that solving the offline version of LEO-MV3D-ISL is at least as difficult as solving the Partition Problem, its NP-hardness is established. Given that the original LEO-MV3D-ISL is an online problem, it inherently retains the complexities of the offline version.

Phase 1: Online Algorithm

Algorithm 1 LEO Satellite Cooperative Caching

Require: Local cache state $V_n(t)$, neighbor caches $\{V_{n-1}(t), V_{n+1}(t)\}$, Request range $R_n(t) = [h, l]$, available energy $E_n(t)$ and bandwidth $B_n(t)$.

Ensure: $V_n^f(t)$, $V_n^{DIBR}(t)$, $V_n^{ISL}(t)$, $V_n^s(t)$, $V_n^e(t)$, total cost $C(t)$.

Calculate the possibility of view v as $P(v)$.
for all j in range $[h, l]$ **do**
    if $j \in V_n(t)$ **then**
        $\tau_j \leftarrow c_{n,s}(z_j, d_j)$;
    else if $j \in \left(V_{n-1}(t) \cup V_{n+1}(t)\right) \setminus V_n(t)$ **then**
        $\tau_j \leftarrow c_{ISL}(z_j, d_j^{ISL}) + c_{n,s}(z_j, d_j)$;
    else
        $\tau_j \leftarrow c_m + c_{n,f}(z_j) + h \cdot c_{ISL}(z_j, d_j^{ISL}) + c_{n,s}(z_j, d_j)$;
    end if
    Calculate $\mu_{h,j}$;
end for
$(V_n^f(t)^*, V_n^{DIBR}(t)^*, V_n^{ISL}(t)^*, V_n^s(t)^*) = argmin\mu_{k,l}$;
$V_n^e(t)^* \leftarrow argminP(v)$ if the cache space is not sufficient;
return $(V_n^f(t)^*, V_n^{DIBR}(t)^*, V_n^{ISL}(t)^*, V_n^s(t)^*, V_n^e(t)^*)$ and cost C;

Phase 2: Swap Popular View with Ground Station

Algorithm 2 Drop and Fetch from GS for LEO Satellite n

Require: Current local cache state $V_n(t)$, Previous neighbor caches $\{V_{n-1}(t), V_{n+1}(t)\}$; Local popularity ranking $\mathcal{P}_n(v)$; Global popularity ranking $\mathcal{P}_{\mathrm{GS}}(v)$ (available **only** when n is connected to a ground station).

Ensure: Updated cache state $V_n^\star(t)$; Modified total cost $C^\star(t)$.

if isConnectedToGS$(n) =$ **true then**
  $V_{\mathrm{drop}} \leftarrow \arg\min_{v \in V_n(t)} N_n^{(v)}(t)$; ▷ select the view with the smallest subscriber count to drop
  $V_n(t) \leftarrow V_n(t) \setminus V_{\mathrm{drop}}$;
  while $\sum_{i \in V_n(t)} z_i > Z$ **do**
    $V_{fetch} \leftarrow$ most-popular view in GS by $\mathcal{P}_{GS}(v)$;
    if $V_{fetch} \notin V_{n-1}(t) \cup V_n(t) \in V_{n+1}(t)$ **then**
      $V_n(t) \leftarrow V_n(t) \cup V_{\mathrm{fetch}}$;
      $C(t) \leftarrow C(t) + c_{n,f}(z_i),\ i \in V_{fetch}$;
    end if
  end while
end if
return $V_n^\star(t) \leftarrow V_n(t)$ and $C^\star(t) \leftarrow C(t)$;

Example

1,12,13,14	4,5,6,8	3,7,20,21	13,14,17,18
LEO_1	LEO_2	LEO_3	LEO_4

- $T = t$
- $R_2(t) = 7$
- $V_2^s(t) = \{6,8\}$
- $V_2^{DIBR}(t) = \{7\}$
- $V_2^{ISL}(t) = \{\}$
- $V_2^f(t) = \{\}$

$N_1(10)$

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	0	1	0	0	0	1	1	0	0	0	2	3	2	0	0	0	0	1	0	0	0	0	0

$N_2(10)$

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
0	1	0	2	2	1	1	2	0	1	0	1	1	0	0	0	0	0	0	1	0	0	0	0

$N_3(10)$

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
0	0	1	0	0	0	2	0	0	0	0	0	1	1	0	0	0	0	2	2	2	0	0	0

$N_4(10)$

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
0	1	1	0	0	0	0	1	0	0	0	2	3	2	0	1	1	1	0	0	0	0	0	0

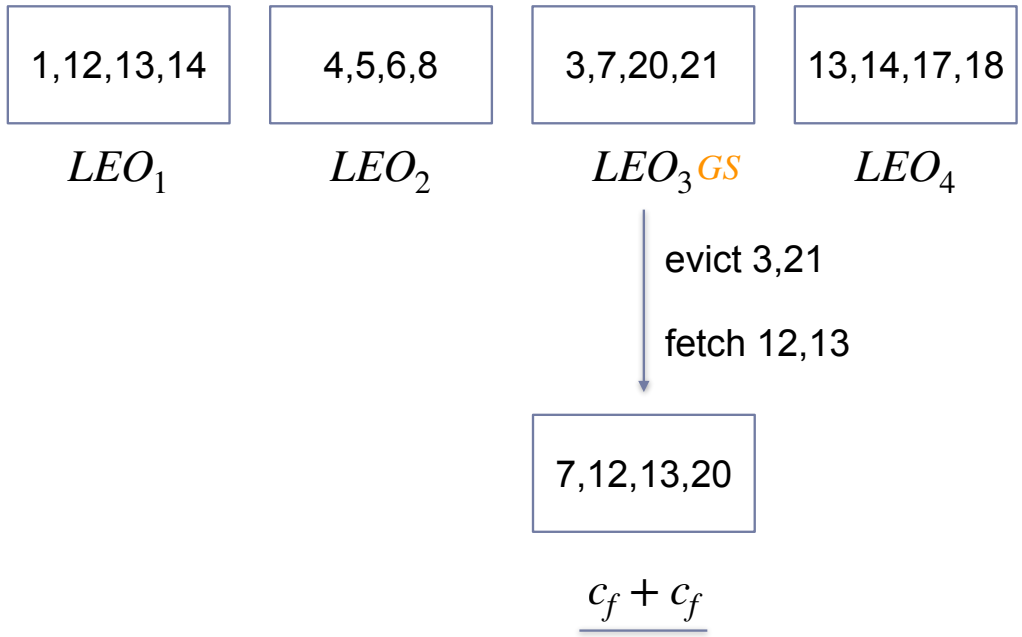
$N_{GS}(10)$

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	2	3	2	2	1	4	4	0	1	0	5	8	5	0	1	1	1	3	3	2	0	0	0

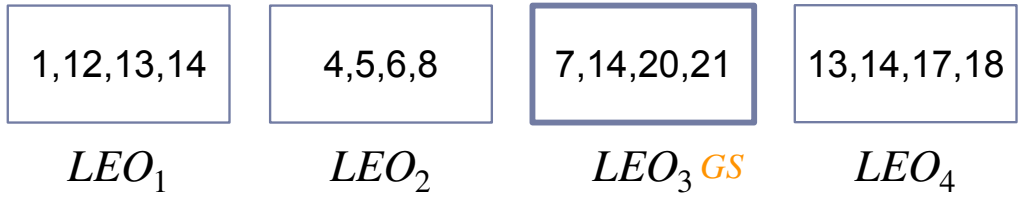
- $C(t) = (c_s) + (c_p) + (c_s)$
- $V_2^e(t) = \{\} \rightarrow V_2(t) = \{4,5,6,8\}$

Ex 縮小一點，這些表格可以共用同一個 column

Example ($T = t + 1$ Phase 2 Update)



Example (No phase 2)



- $T = t + 1$
- $R_3(t+1) = 13$
- $V_3^s(t+1) = \{12,14\}$
- $V_3^{DIBR}(t+1) = \{13\}$
- $V_3^{ISL}(t+1) = \{12\}$
- $V_3^f(t+1) = \{12\}$

$N_1(10)$

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	0	1	0	0	0	1	1	0	0	0	2	3	2	0	0	0	0	1	0	0	0	0	0

$N_2(10)$

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
0	1	0	2	2	1	1	2	0	1	0	1	1	0	0	0	0	0	0	1	0	0	0	0

$N_3(10)$

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
0	0	1	0	0	0	2	0	0	0	0	0	2	1	0	0	0	0	2	2	2	0	0	0

$N_4(10)$

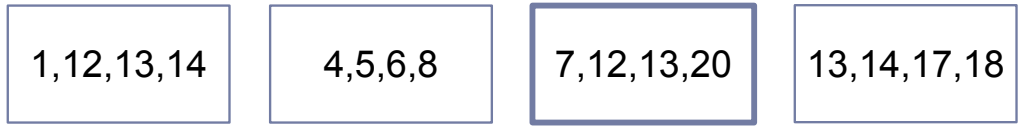
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
0	1	1	0	0	0	0	1	0	0	0	2	3	2	0	1	1	1	0	0	0	0	0	0

$N_{GS}(10)$

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	2	3	2	2	1	4	4	0	1	0	5	9	5	0	1	1	1	3	3	2	0	0	0

- $C(t+1) = (c_m + c_f + c_s) + (c_p) + (c_s)$
- $V_3(t+1) = \{7,12,14,20,21\} \rightarrow V_3^e(t+1) = \{12\} \rightarrow V_3(t+1) = \{7,14,20,21\}$

Example (With phase 2)



- $T = t + 1$
- $R_3(t+1) = 13$
- $V_3^s(t+1) = \{12,14\}$
- $V_3^{DIBR}(t+1) = \{13\}$
- $V_3^{ISL}(t+1) = \{14\}$
- $V_3^f(t+1) = \{\}$

$N_1(10)$

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	0	1	0	0	0	1	1	0	0	0	2	3	2	0	0	0	0	1	0	0	0	0	0

$N_2(10)$

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
0	1	0	2	2	1	1	2	0	1	0	1	1	0	0	0	0	0	0	1	0	0	0	0

$N_3(10)$

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
0	0	1	0	0	0	2	0	0	0	0	0	1	1	0	0	0	0	2	2	2	0	0	0

$N_4(10)$

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
0	1	1	0	0	0	0	1	0	0	0	2	4	2	0	1	1	1	0	0	0	0	0	0

$N_{GS}(10)$

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	2	3	2	2	1	4	4	0	1	0	5	9	5	0	1	1	1	3	3	2	0	0	0

- $C^{\star}(t+1) = (c_s) + (c_p) + (c_{ISL} + c_s)$
- $V_3(t+1) = \{7,12,13,14,20\} \rightarrow V_3^e(t+1) = \{12\} \rightarrow V_3(t+1) = \{7,13,14,20\}$

Example

	No phase 2	With phase 2
Cache miss	1	0
Fetch	1	2
ISL	0	1
DIBR	2	2
Transmission	4	4

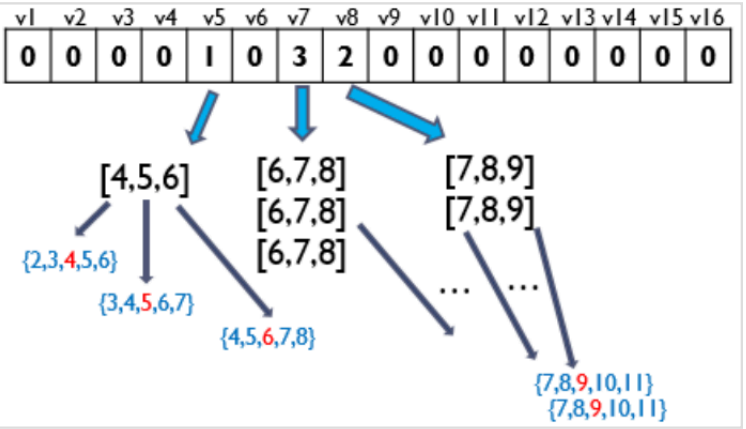
Time Complexity Analysis

- Phase 1: $O\left(T \cdot (|V| log |V| + B\mathcal{Z}D)\right)$
  - Calculate $O(|V| log |V|)$ views to find $P(i)$ for each view i
  - To minimize the total cost, DP spends at most $O(\mathcal{Z})$ and $O(2\mathcal{Z})$ to search if a view j exists in the connecting LEO cache or in the neighboring LEO cache and derive $\mu_{h,j}$ with $O(D-1)$
    - One loop time complexity is $O(\mathcal{Z}D)$
    - $\mathcal{Z} = \frac{Z}{z_i}$ is the maximum number of cached views
    - Since the request range is at most B, the total DP time complexity is $O(B\mathcal{Z}D)$
  - Spend at most $O(|V|)$ to decide the evicted views with the lowest possibility $P(i)$
- Phase 2: $O(\mathcal{Z}^2)$
  - Select views with the fewest subscriber count to drop spend at most $O(\mathcal{Z})$
  - Fetch views from GS at most $O(\mathcal{Z})$ loop and search $O(3\mathcal{Z})$, the total time complexity is $O(\mathcal{Z}^2)$

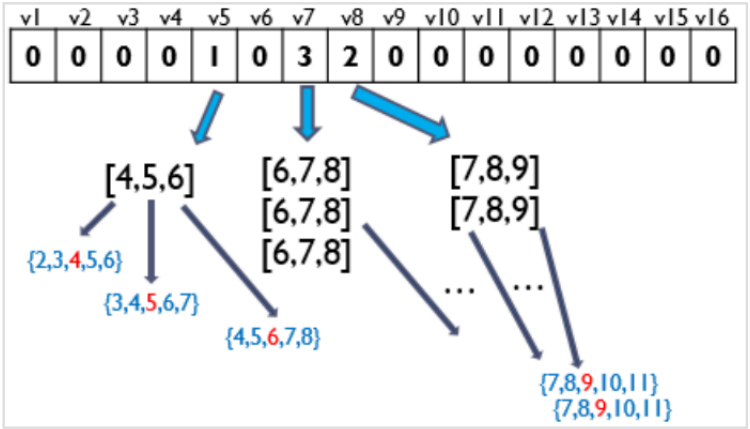
Appendix

Online Algorithm

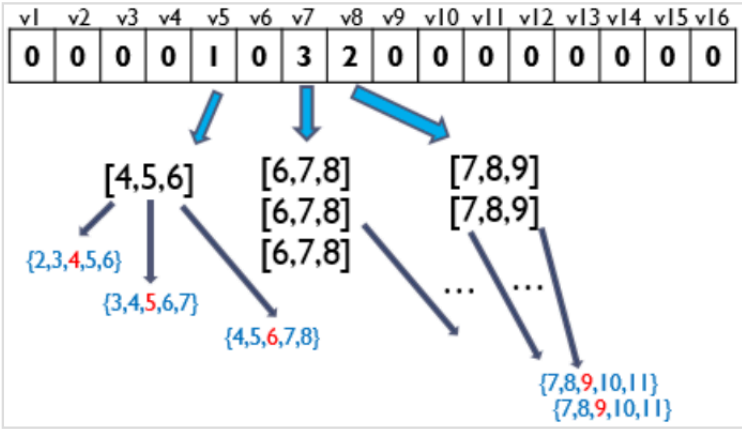
- τ_j is the total cost to fetch view j
- $\mu_{i,j}$ be the minimum cost of view set $[i, j]$
- $\mu_{h,j} = min_{max\{j-D,h\} \le i < j} \left(\tau_j + \mu_{h,i} + (\alpha(j-i) + T_{DIBR})(j-i-1) \right)$
- $P_n(v)$ be the possibility that view v at LEO_n is subscribed in the next time slot



$P_{n-1}(v)$



$P_n(v)$

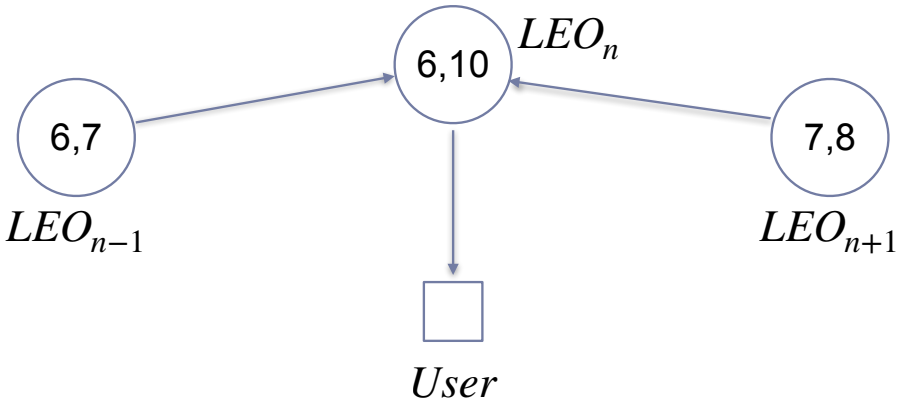


$P_{n+1}(v)$

Algorithm Example

$$\mu_{h,j} = min_{max\{j-D,h\} \le i < j} \left(\tau_j + \mu_{h,i} + (\alpha(j-i) + T_{DIBR})(j-i-1) \right)$$

- $R_n(t) = [h, l] = [5,10]$

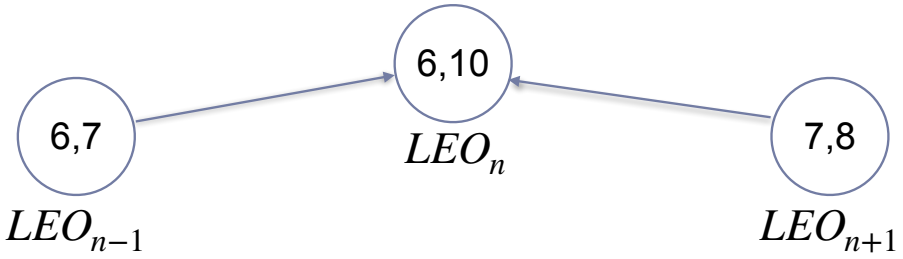


1. $\tau_5 = c_m + c_{n,f}(z_5) + c_{n,s}(z_5, d_5)$
   - $\mu_{5,5} = \tau_5$ (cache miss and transfer 5)
2. $\tau_6 = c_{n,s}(z_6, d_6)$
   - $\mu_{5,6} = \tau_6 + \mu_{5,5}$ (transfer 5,6)
3. $\tau_7 = c_{ISL}(z_7, d_7^{ISL}) + c_{n,s}(z_7, d_7)$
   - $\mu_{5,7} = \tau_7 + \mu_{5,5} + [2\alpha + T_{DIBR}]$ (transfer 5,7 and synthesis 6 at user)
   - $\mu_{5,7} = \tau_7 + \mu_{5,6}$ (transfer 5,6,7)

Algorithm Example

$$\mu_{h,j} = min_{max\{j-D,h\} \le i < j} \left(\tau_j + \mu_{h,i} + (\alpha(j-i) + T_{DIBR})(j-i-1) \right)$$

- $R_n(t) = [h, l] = [5,10]$

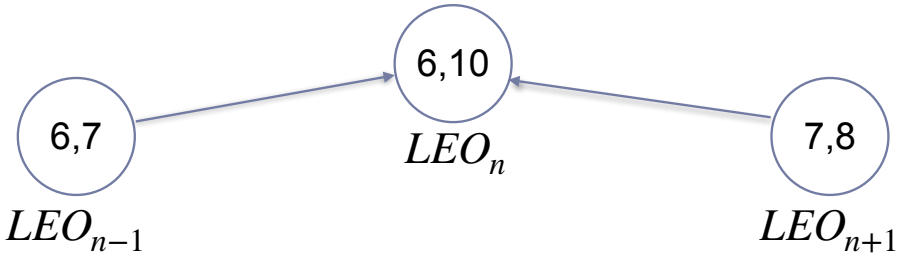


4. $\tau_8 = c_{ISL}(z_8, d_8^{ISL}) + c_{n,s}(z_8, d_8)$
   - $\mu_{5,8} = \tau_8 + \mu_{5,5} + [3\alpha + T_{DIBR}] \cdot 2$ (transfer 5,8 and synthesis 6,7 at user)
   - $\mu_{5,8} = \tau_8 + \mu_{5,6} + [2\alpha + T_{DIBR}]$ (transfer 5,6,8 and synthesis 7 at user)
   - $\mu_{5,8} = \tau_8 + \mu_{5,7}$ (transfer 5~7,8) -> (5,7,8 or 5,6,7,8)

Algorithm Example

$$\mu_{h,j} = min_{max\{j-D,h\} \le i < j} \left(\tau_j + \mu_{h,i} + (\alpha(j-i) + T_{DIBR})(j-i-1) \right)$$

- $R_n(t) = [h, l] = [5,10]$

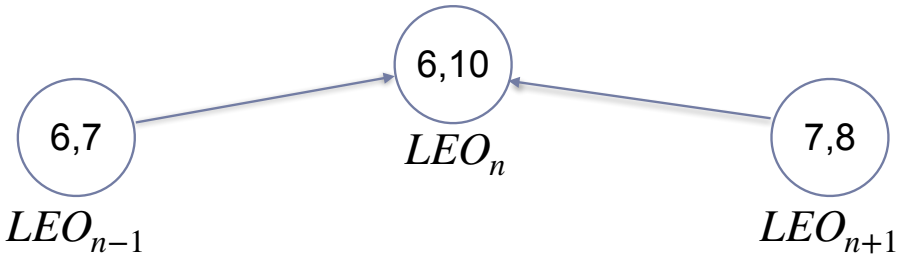


5. $\tau_9 = c_m + c_{n,f}(z_9) + c_{n,s}(z_9, d_9)$
   - $\mu_{5,9} = \tau_9 + \mu_{5,6} + [3\alpha + T_{DIBR}] \cdot 2$ (transfer 5,6,9)
   - $\mu_{5,9} = \tau_9 + \mu_{5,7} + [2\alpha + T_{DIBR}]$ (transfer 5~7,9)
   - $\mu_{5,9} = \tau_9 + \mu_{5,8}$ (transfer 5~8,9)

Algorithm Example

$$\mu_{h,j} = min_{max\{j-D,h\} \le i < j} \left(\tau_j + \mu_{h,i} + (\alpha(j-i) + T_{DIBR})(j-i-1) \right)$$

- $R_n(t) = [h, l] = [5,10]$



6. $\tau_{10} = c_{n,s}(z_{10}, d_{10})$
   - $\mu_{5,10} = \tau_{10} + \mu_{5,7} + [3\alpha + T_{DIBR}] \cdot 2$ (transfer 5~7,10)
   - $\mu_{5,10} = \tau_{10} + \mu_{5,8} + [2\alpha + T_{DIBR}]$ (transfer 5~8,10)
   - $\mu_{5,10} = \tau_{10} + \mu_{5,9} +$ (transfer 5~9,10)
- If the final $\mu_{5,10}$ decide to transfer 5,7,10
  - $V_n^f(t) * = \{5\}, V_n^{DIBR}(t) = \{6,8,9\}, V_n^{ISL}(t) * = \{7\}, V_n^s(t) * = \{5,7,10\}$
  - $V_n^e(t) *$ would choose the lowest popularity view to evict (5,6,7,10) if cache storage is full

Competitive Ratio Analysis

- For view $i, j \in V$, denoting $M_n(i,j)$ the **minimum total size of views** in $V_n(t) \subseteq V$ such that $i, j \in V_n(t)$ and $V_n(t) \cup V_n^s(t) = \{i \ (mod \ |V|), \ i+1 \ (mod \ |V|), \ldots, j \ (mod \ |V|)\}$
  - $V_n^s(t)$ denotes the set of views that can be synthesized by $V_{n-1}(t), V_n(t), V_{n+1}(t)$ based on D
  - $M_n(i,j) = \min_{j-D \le k \le j-1} \{M_n(i,k) + z_j\}$, where $M_n(i,i) = z_i$ for $i \in V$
- Example:
  - $V_n(t) = \{1,3,6,15\}, V_{n-1}(t) = \{5,6,8,9\}, V_{n+1}(t) = \{1,3,4,5\} \ => \ V_n^s(t) = \{2,7\}$
  - $V_n(t) \cup V_n^s(t) = \{1,2,3,4,5,6,7,8,9,15\}$
  - Find $M_n(1,9)$, which represents the minimum total size of a contiguous set of views covering the range from view 1 to view 9 **(Maybe view 1,4,7,9)**

Competitive Ratio Analysis

- Define $\Delta_n = \max_{(i,j)} \{j - i + 1 \ (mod\,|V|) : M_n(i,j) \leq Z\}$ as the **maximum number of views** (including synthesized views) that can be hit in the cache

- The view popularity (the distribution of the user request) follows Zipf distribution, denoted by $f_n(i, z, |V|) = \frac{1/i^z}{\sum_{n=1}^{|V|} (1/n^z)}$
  - i is the preference rank of a view
  - z is the Zipf factor

Competitive Ratio Analysis (Case 1: $\Delta_n = |V|$)

- Consider the instance with cache state $V_n(0)$ corresponding to Δ_n, which view $[i, j]$ are in $V_n(0)$ or can be synthesized by cached view in $V_{n-1}(0), V_n(0), V_{n+1}(0)$
- *OPT* would be able to synthesizes all views and directly return to the user without cache miss
- *ALG* always find a view i (for $j - D \leq i < j$) fitting the minimum cost by exploring the chance to synthesize the view in the range $[j - D, j - 1]$
- Since $|V| = \Delta_n$, *ALG* generates an optimal cache storing the views to cover the views $\{1,2,\ldots,|V|\}$, such that there is no cache miss at any time

Competitive Ratio Analysis (Case 2: $\Delta_n < |V|$)

- There are at most $\Delta_n - (B-1)$ kinds of request that would not cause any cache miss
- Example: $(B = 5, \Delta_n = 9)$ 5 kinds of requests would not make cache miss

- Since the view popularity follows Zipf distribution, the percentage of views that cache hits occur is at most $\sum_{i=1}^{\Delta_n-(B-1)} f_n(i, z, |V|)$
  - The percentage of views that **cache misses** occur is at least $p_n = 1 - \sum_{i=1}^{\Delta_n-(B-1)} f_n(i, z, |V|)$

Competitive Ratio Analysis (Case 2: $\Delta_n < |V|$)

- We first derive an upper bound of the percentage of views for which any cache miss does not happen, to obtain the lower bound of optimal solution OPT
- The cost with optimal policy is **at least** $\left[(c_m + \underline{c_{n,f}} + \underline{c_{n,s}}) \cdot p + \underline{c_{n,s}} \cdot (1-p)\right] \cdot T$
- OPT also has a fundamental lower bound $\underline{c_{n,s}} \cdot \left(\lceil \frac{B-1}{D} \rceil + 1\right) \cdot T$ (those views are in LEO_n) with no cache miss
- **The lower bound of** OPT is

$$max\left\{ \left(\left[(c_m + \underline{c_{n,f}} + \underline{c_{n,s}}) \cdot p + \underline{c_{n,s}} \cdot (1-p)\right] \cdot T \right), \left(\underline{c_{n,s}} \cdot \left(\lceil \frac{B-1}{D} \rceil + 1 \right) \cdot T \right) \right\}$$

Competitive Ratio Analysis (Case 2: $\Delta_n < |V|$)

- For each request range $[h, l]$ consisting of B views, considering each view would cache miss $(c_m \cdot B)$
- Also the inter-satellite links would not help to cache hit
- The number of fetched views is at most $\lceil \frac{B-1}{D} \rceil + 1$ in each time slot
- The cost of the solution with ALG is at most

$$\underbrace{\left(c_m \cdot B + \overline{c_{n,f}} \cdot (\lceil \frac{B-1}{D} \rceil + 1) + h \cdot \overline{c_{ISL}} \cdot (\lceil \frac{B-1}{D} \rceil + 1) + \overline{c_{n,s}} \cdot (\lceil \frac{B-1}{D} \rceil + 1)\right) \cdot T}_{\text{The upper bound of } ALG}$$

Competitive Ratio Analysis

$$\frac{\text{ALG}}{\text{OPT}} \le \frac{(c_m \cdot B + \overline{c_{n,f}} \cdot (\lceil \frac{B-1}{D} \rceil + 1) + h \cdot \overline{c_{ISL}} \cdot (\lceil \frac{B-1}{D} \rceil + 1) + \overline{c_{n,s}} \cdot (\lceil \frac{B-1}{D} \rceil + 1)) \cdot T}{\max\{\left(\left[(c_m + \underline{c_{n,f}} + \underline{c_{n,s}}) \cdot p + \underline{c_{n,s}} \cdot (1-p)\right] \cdot T\right), \left(\underline{c_{n,s}} \cdot (\lceil \frac{B-1}{D} \rceil + 1) \cdot T\right)\}}$$

$$\le \frac{(c_m \cdot B + c_m \cdot (\lceil \frac{B-1}{D} \rceil + 1) + h \cdot c_m \cdot (\lceil \frac{B-1}{D} \rceil + 1) + c_m \cdot (\lceil \frac{B-1}{D} \rceil + 1)) \cdot T}{\max\{(\underline{c_{n,s}} \cdot p + \underline{c_{n,s}} \cdot (1-p)) \cdot T, (\underline{c_{n,s}} \cdot (\lceil \frac{B-1}{D} \rceil + 1) \cdot T)\}}$$

$$\le \frac{(3+h) \cdot c_m \cdot B \cdot T}{\max\{(\underline{c_{n,s}} \cdot T), (\underline{c_{n,s}} \cdot \frac{B}{D} \cdot T)\}} = \min\left\{\frac{(3+h)c_m B}{\underline{c_{n,s}}}, \frac{(3+h)c_m D}{\underline{c_{n,s}}}\right\}$$

- Second inequality holds because $c_m > \overline{c_{n,f}}$ and $c_m > \overline{c_{n,s}}$
- The last inequality holds because $\frac{B}{D} \le \frac{B-1}{D} + 1 \le \lceil \frac{B-1}{D} \rceil + 1 \le B$

If $D \ge 1$