High-density Mobile Cloud Gaming on Edge SoC Clusters

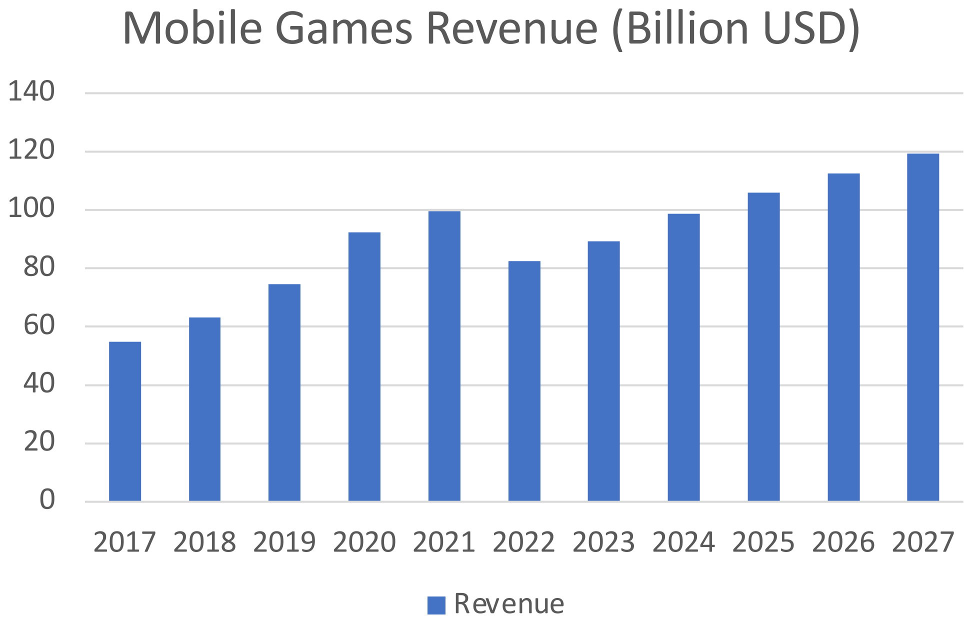
Li Zhang, Shangguang Wang, Mengwei Xu

Beijing University of Posts and Telecommunications (BUPT)

北京邮电大学
Beijing University of Posts and Telecommunications

Mobile Games

- Mobile games: A popular and portable form of entertainment on daily smartphones
- Huge and growing market: An estimate of 100 billion USD revenue globally



Source: https://www.statista.com/outlook/dmo/digital-media/video-games/mobile-games/worldwide

Mobile Games: Huge Resource Requirements

- Better gaming experiences call for huge hardware resources.
- Games are becoming "bigger" and "more complex"; fully load the latest, powerful mobile processors.

These new, resource-consuming mobile games retire old smartphones sooner or later!

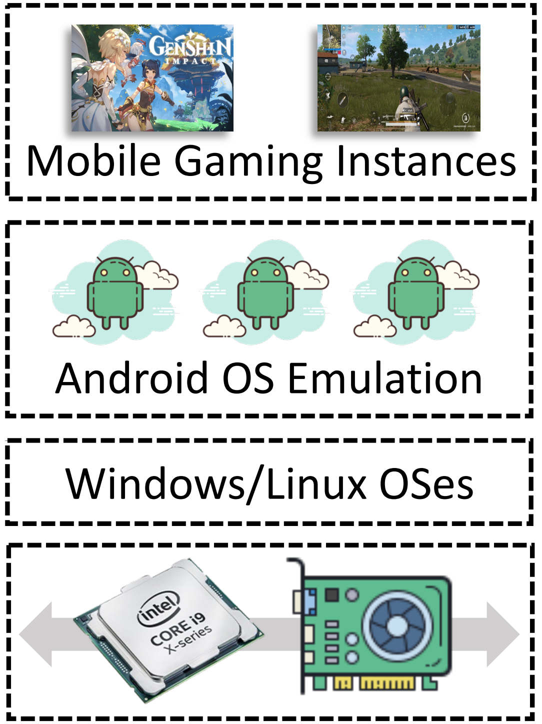
Mobile Cloud Gaming Services

- Better gaming experiences call for huge hardware resources.
- Games are becoming “bigger” and “more complex”; fully load the latest, powerful mobile processors.

- ✓ Instant access
- ✓ High compatibility
- ✓ Immersive game experience
- ✓ Reduced hardware cost

Mobile Cloud Gaming Infrastructure

- **Traditional approach:** Mobile environment emulation on Intel/ARM CPUs with server-level GPUs (e.g., NVIDIA GPUs)



Traditional Approach

- ❑ Pros: Share the same hardware as other general workloads
- ❑ Cons:
  - Performance loss: OS emulation required
  - Low flexibility, huge human efforts: Require game reengineering to solve compatibility and performance issues
  - Limited game availability: Game developers may not provide app packages for other hardware architecture (e.g., x86)

Mobile Cloud Gaming Infrastructure

- **System-on-Chip Clusters:** Group multiple mobile processors inside a server; provide identical mobile environments as on user smartphones.

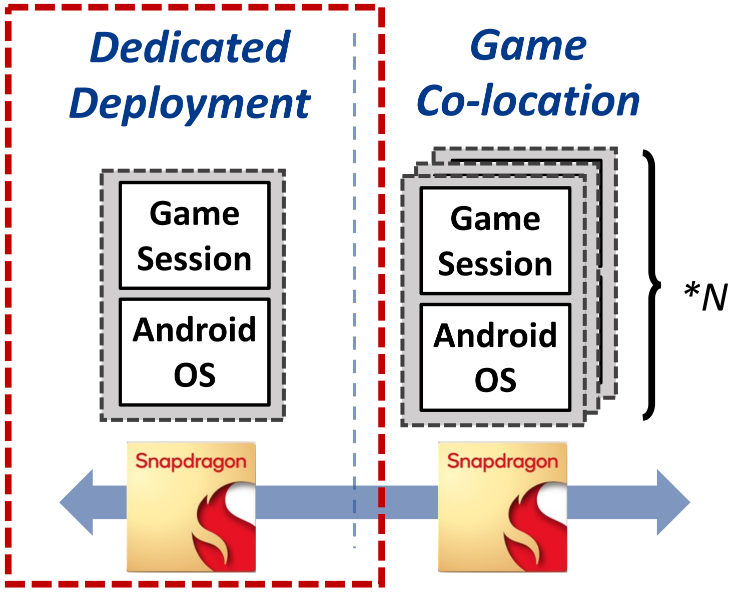
System-on-Chip Clusters

- ❑ The same mobile context: No OS/game modification required
- ❑ Easy of deployment: Games are optimized for a single mobile processor

What are the drawbacks of using SoC Clusters?

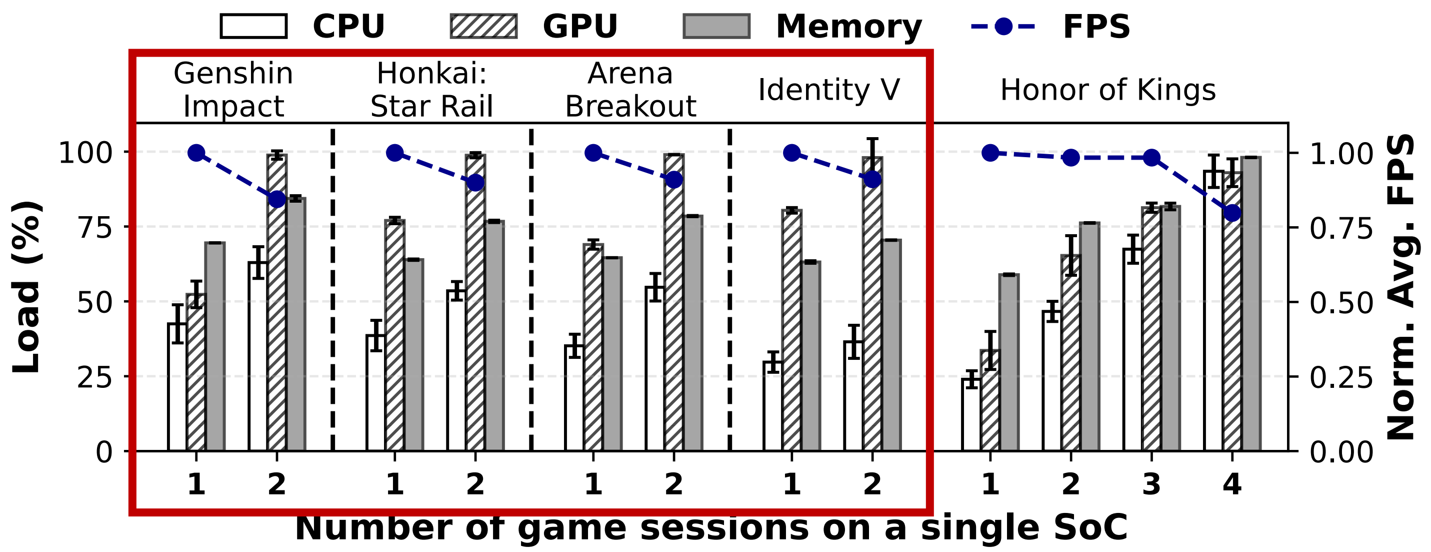
Low Game Deployment Density

- Conservative game deployment methods
  - **Dedicated deployment:** Deploy one game instance per mobile SoC.
  - **Game co-location:** Co-locate multiple game instances on the same mobile SoC through pre-profiling.



Low Game Deployment Density

- Conservative game deployment methods
  - **Dedicated deployment:** Deploy one game instance per mobile SoC.
  - **Game co-location:** Co-locate multiple game instances on the same mobile SoC through pre-profiling.
- Experiment on five commercial mobile games



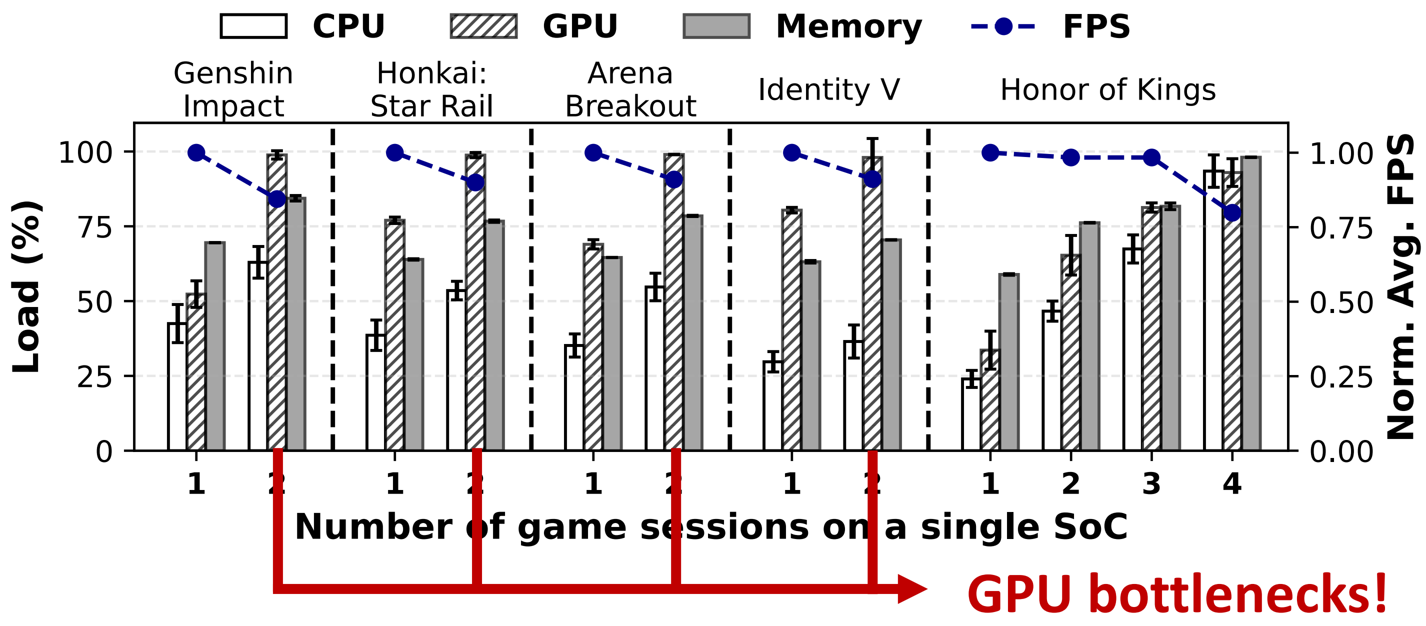
❑ Four out of all five games can only run one game session.

❑ A huge resource waste when only one game session is running.

Wasted resources: > 50% CPU and > 25% GPU

Low Game Deployment Density

- Conservative game deployment methods
  - **Dedicated deployment:** Deploy one game instance per mobile SoC.
  - **Game co-location:** Co-locate multiple game instances on the same mobile SoC through pre-profiling.
- Experiment on five commercial mobile games



- ❑ Four out of all five games can only run one game session.
- ❑ A huge resource waste when only one game session is running.
- ❑ Limited GPU resources per SoC bottleneck game deployment density.

Goal of this Work

- Our goal: Run more mobile game within limited hardware resources of mobile SoCs.
- Similar to the goal of traditional cloud gaming systems!

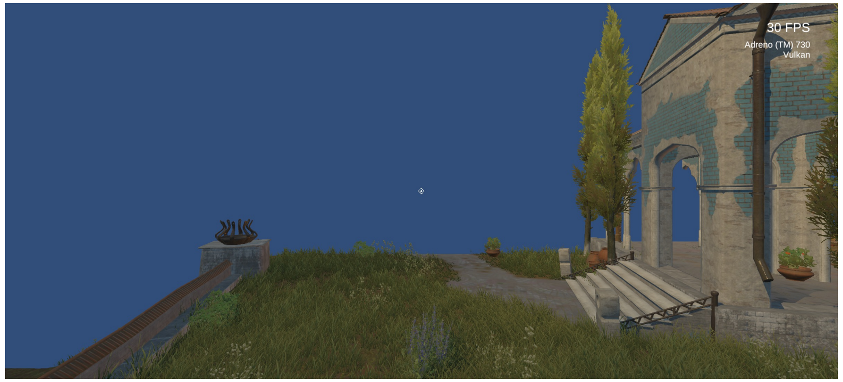
How well do previous cloud gaming systems perform here?

- Their approach: They partition complete game instances, but in the cloud, they all consume a bunch of resources.
  - Run a full game copy.
  - Run a partial game instance, which still consumes a lot of resources.

Revisit Prior Game Partitioning Designs

- *[ASPLOS'20] Coterie: Exploiting Frame Similarity to Enable High-Quality Multiplayer VR on Commodity Mobile Devices*
- Split the whole game world into a near part and a remote part.

(a) Original game view before partition

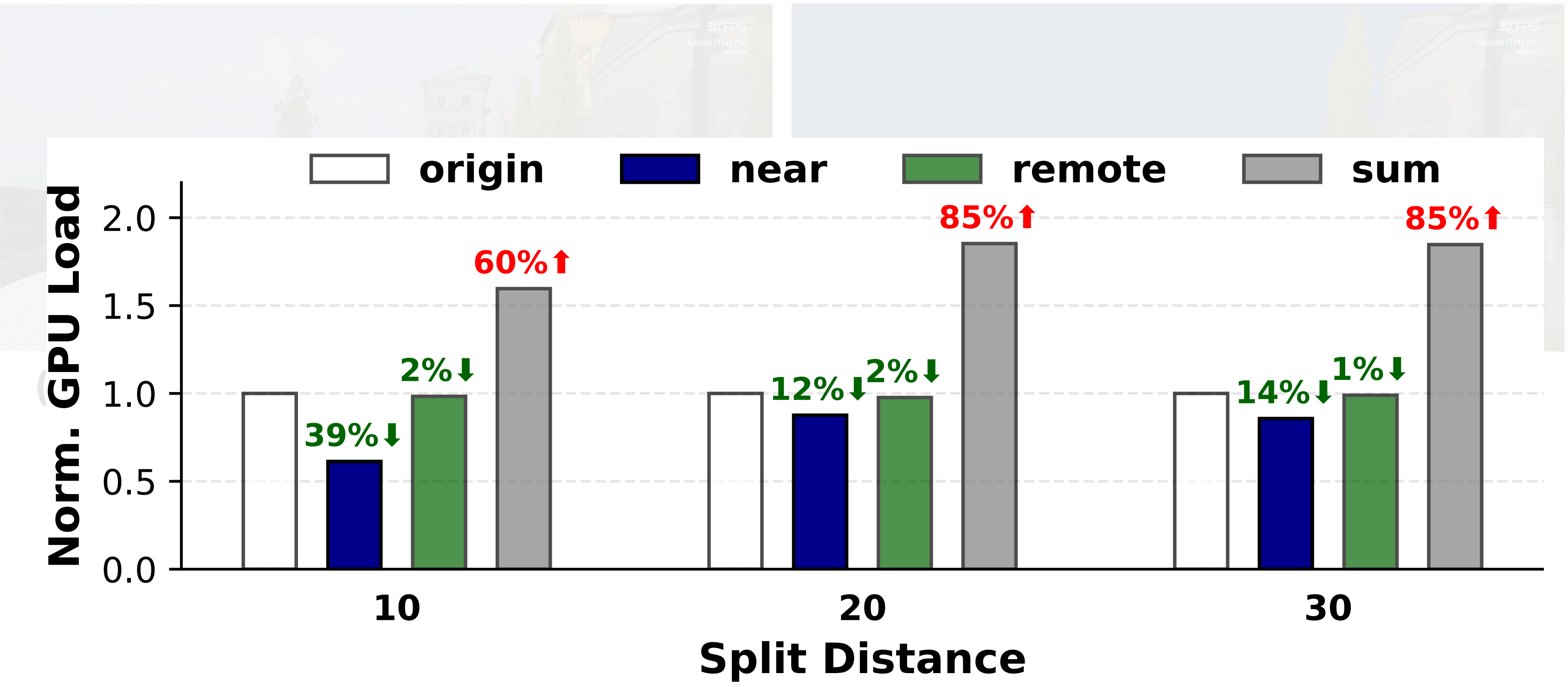


(b) Near-part game view after partition

(c) Remote-part game view after partition

Revisit Prior Game Partitioning Designs

- *[ASPLOS'20] Coterie: Exploiting Frame Similarity to Enable High-Quality Multiplayer VR on Commodity Mobile Devices*
- Split the whole game world into a near part and a remote part.



- ❑ Trivial GPU load reduction at the remote game part.
- ❑ Even higher accumulated GPU load after game partitioning.

Revisit Prior Game Partitioning Designs

- *[ASPLOS'20] Coterie: Exploiting Frame Similarity to Enable High-Quality Multiplayer VR on Commodity Mobile Devices*
- Split the whole game world into a near part and a remote part.

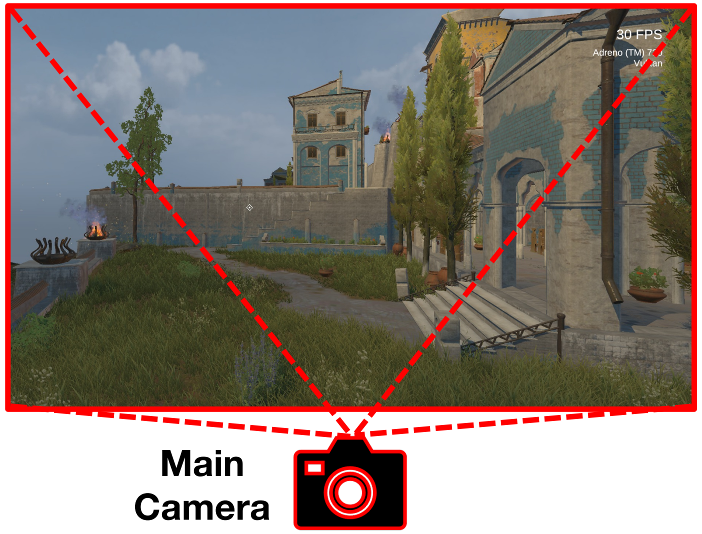
(a) Original game view before partition **(b) Near-part game view after partition** **(c) Remote-part game view after partition**

Preserve the optimizations brought by the default graphics rendering pipeline.

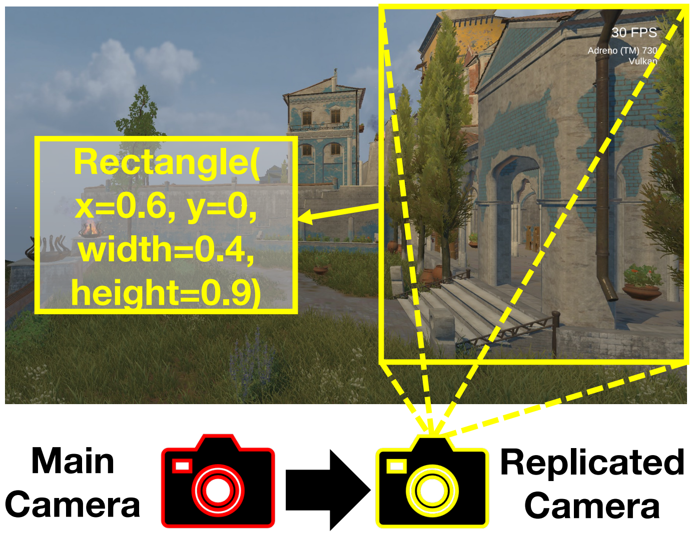
The occluded areas are unnecessarily rendered after game partitioning.

Our System: *SFG*

- A simple yet efficient partitioning method: Partition graphics rendering workloads before rendering (like the sort-first rendering[1])
- More flexibility: Use an abstracted rectangle to represent the target area for rendering; Runtime adjustment



(a) Full game view rendered by the original main camera



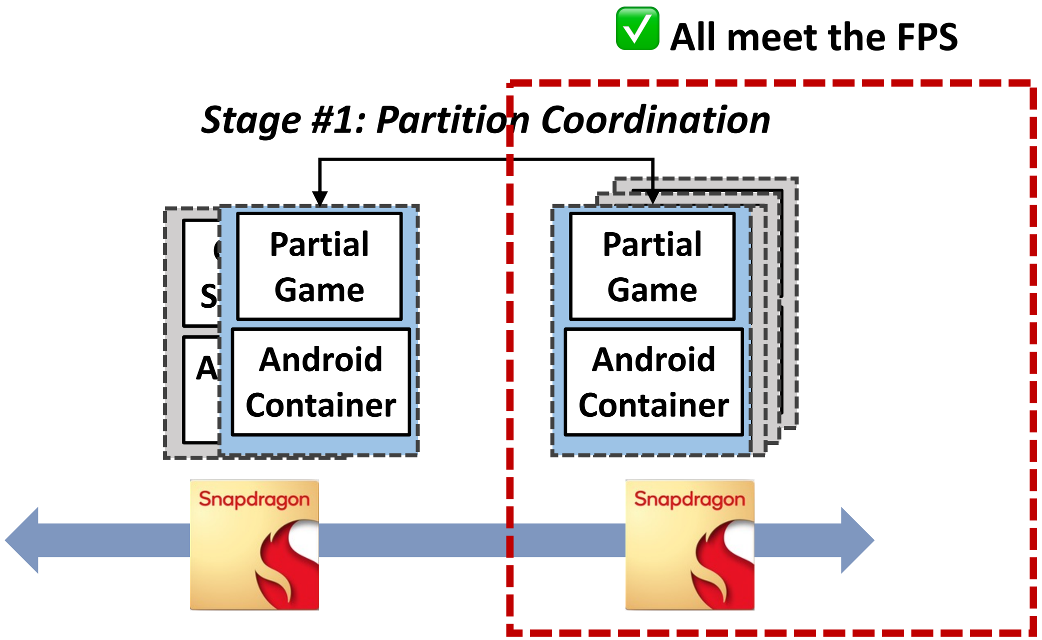
(b) Partial game view rendered by the replicated camera

[1] Steven Molna et al. A sorting classification of parallel rendering. IEEE computer graphics and applications,1994.

Our System: *SFG*

- NPU-enhanced game partition coordination to handle game usage dynamics
- Assumption: Render native frames first; then use frame super-resolution on SoC NPUs if there is no GPU resource left
- Approach: A two-stage coordination

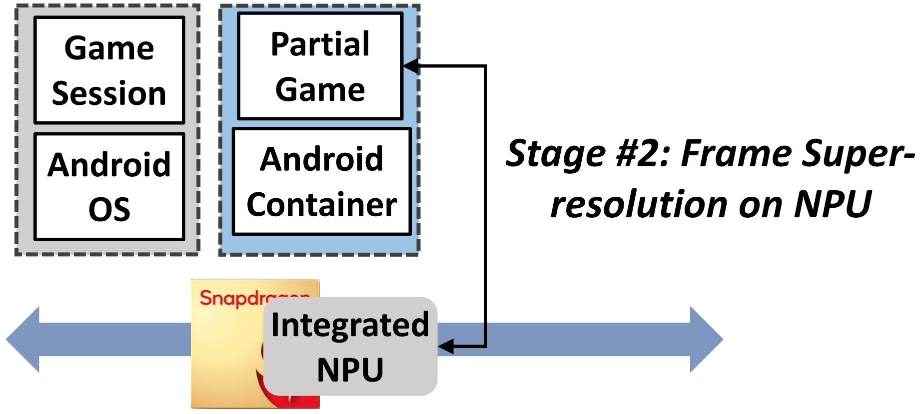
  - ❑ Stage #1: Shifting GPU rendering workloads to make all game sessions on one of the SoCs meet the target (every 500 ms).



Our System: *SFG*

- NPU-enhanced game partition coordination to handle game usage dynamics
- Assumption: Render native frames first; then use frame super-resolution on SoC NPUs if there is no GPU resource left
- Approach: A two-stage coordination

  - ❑ Stage #1: Shifting GPU rendering workloads to make all game sessions on one of the SoCs meet the target (every 500 ms).
  - ❑ Stage #2 (optional): Apply frame super-resolution on a partial game session if game sessions on one of the SoC do not meet the FPS.



Evaluation

- Implementation
  - A Unity Plugin that can be easily integrated into any Unity-based games
  - Game partitioning: Unity's Camera API
  - Game states and rendering results synchronization: WebRTC
  - Frame super-resolution: The quantized ETDS[1] model; TFLite on SoC NPUs

[1] Chao et al., Equivalent Transformation and Dual Stream Network Construction for Mobile Image Super-Resolution. CVPR 2023

Evaluation

- Implementation
  - A Unity Plugin that can be easily integrated into any Unity-based games
  - Game partitioning: Unity's Camera API
  - Game states and rendering results synchronization: WebRTC
  - Frame super-resolution: The quantized ETDS[1] model; TFLite on SoC NPUs
- Software
  - Games: Five open-source Unity games with varied graphics settings

Game	Resolution	FPS	Feature
Sun Temple	1920 * 1080	30	Infrequent scene switch
Corridor	1280 * 720	30	Fast scene switch
Sewer Mid	1920 * 1080	60	Medium scene switch
Sewer High	2560 * 1440	60	Medium scene switch
Viking Village	1920 * 1080	30	High dynamics, fast scene switch

Evaluation

- Implementation
  - A Unity Plugin that can be easily integrated into any Unity-based games
  - Game partitioning: Unity's Camera API
  - Game states and rendering results synchronization: WebRTC
  - Frame super-resolution: The quantized ETDS[1] model; TFLite on SoC NPUs
- Software
  - Games: Five open-source Unity games with varied graphics settings
  - Game play simulation: Manually recorded interactive scripts powered by Unity's animation system; replayed at game runtime for deterministic interaction.
- Hardware
  - An SoC Cluster consisting of 60 Qualcomm Snapdragon 865 SoCs; Android 10 OS
  - 1 Gbps network bandwidth between individual SoCs

Effectiveness of Game Partitioning Design

- Baseline: Distance-based game partitioning proposed in Coterie
- Our partition design
  - Reduces the GPU load by an average of 15%.
  - Enables running games on two SoCs that cannot be supported on individual ones.

Game	GPU Load: Origin	Partition Method	GPU Load: Partition			
			P1	P2	P1+P2	Co(P1+P2)
Sun Temple	76.1	Distance	57.0	75.4	132.4	92.4 (21.4%↑)
		Ours	55.3	73.8	129.1	74.2 (2.50%↓)
Corridor	48.5	Distance	30.0	41.1	71.1	60.0 (23.7%↑)
		Ours	29.5	36.1	65.6	56.1 (15.7%↑)
Sewer-Mid	72.4	Distance	59.8	72.7	132.5	85.7 (18.4%↑)
		Ours	58.5	56.0	114.5	75.9 (4.83%↑)
Sewer-High	✗	Distance	73.8	✗	✗	✗
		Ours	71.3	70.2	141.5	✗
Viking Village	✗	Distance	80.8	✗	✗	✗
		Ours	82.5	79.1	161.6	✗

Effectiveness of Game Partitioning Design

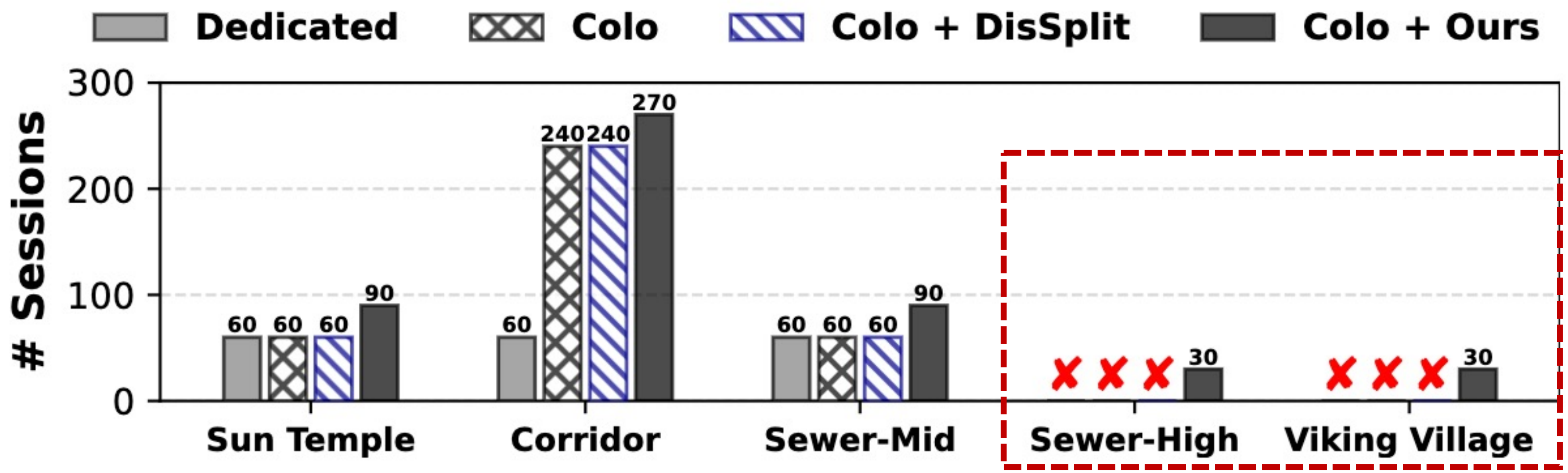
- Baseline: Distance-based game partitioning proposed in Coterie
- Our partition design
  - Reduces the GPU load by an average of 15%.
  - Enables running games on two SoCs that cannot be supported on individual ones.

Game	GPU Load: Origin	Partition Method	GPU Load: Partition			
			P1	P2	P1+P2	Co(P1+P2)
Sun Temple	76.1	Distance	57.0	75.4	132.4	92.4 (21.4%↑)
		Ours	55.3	73.8	129.1	74.2 (2.50%↓)
Corridor	48.5	Distance	30.0	41.1	71.1	60.0 (23.7%↑)
		Ours	29.5	36.1	65.6	56.1 (15.7%↑)
Sewer-Mid	72.4	Distance	59.8	72.7	132.5	85.7 (18.4%↑)
		Ours	58.5	56.0	114.5	75.9 (4.83%↑)
Sewer-High	✗	Distance	73.8	✗	✗	✗
		Ours	71.3	70.2	141.5	✗
Viking Village	✗	Distance	80.8	✗	✗	✗
		Ours	82.5	79.1	161.6	✗

✅ ✅ **Deployable on two SoCs!**

End-to-end Game Deployment

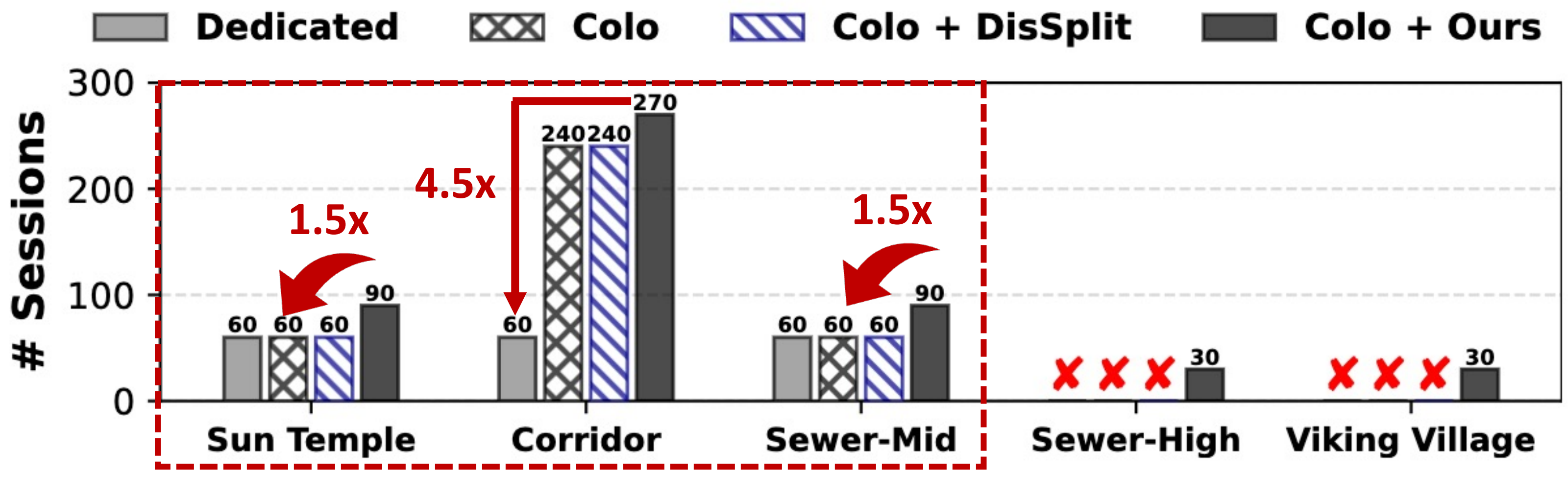
- Baselines
  - Dedicated deployment: One game instance per SoC.
  - Game co-location: One or more game instance per SoCs.
  - Game co-location with distance-based game partitioning
  - Game co-location with our system
- Game deployment density on a whole SoC Cluster (60 SoCs)



❑ Support games exceeding the capacity of one SoC.

End-to-end Game Deployment

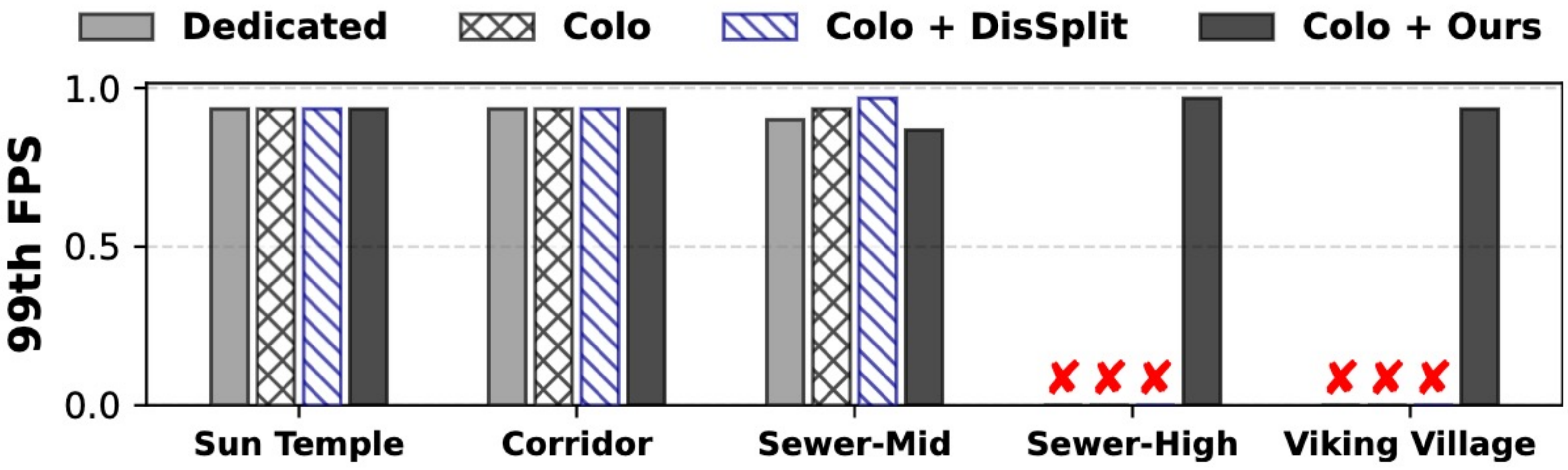
- Baselines
  - Dedicated deployment: One game instance per SoC.
  - Game co-location: One or more game instance per SoCs.
  - Game co-location with distance-based game partitioning
  - Game co-location with our system
- Game deployment density on a whole SoC Cluster (60 SoCs)



- ❑ Support games exceeding the capacity of one SoC.
- ❑ Up to **4.5x** improvement over dedicated deployment.
- ❑ Up to **1.5x** improvement over previous co-location methods.

End-to-end Game Deployment

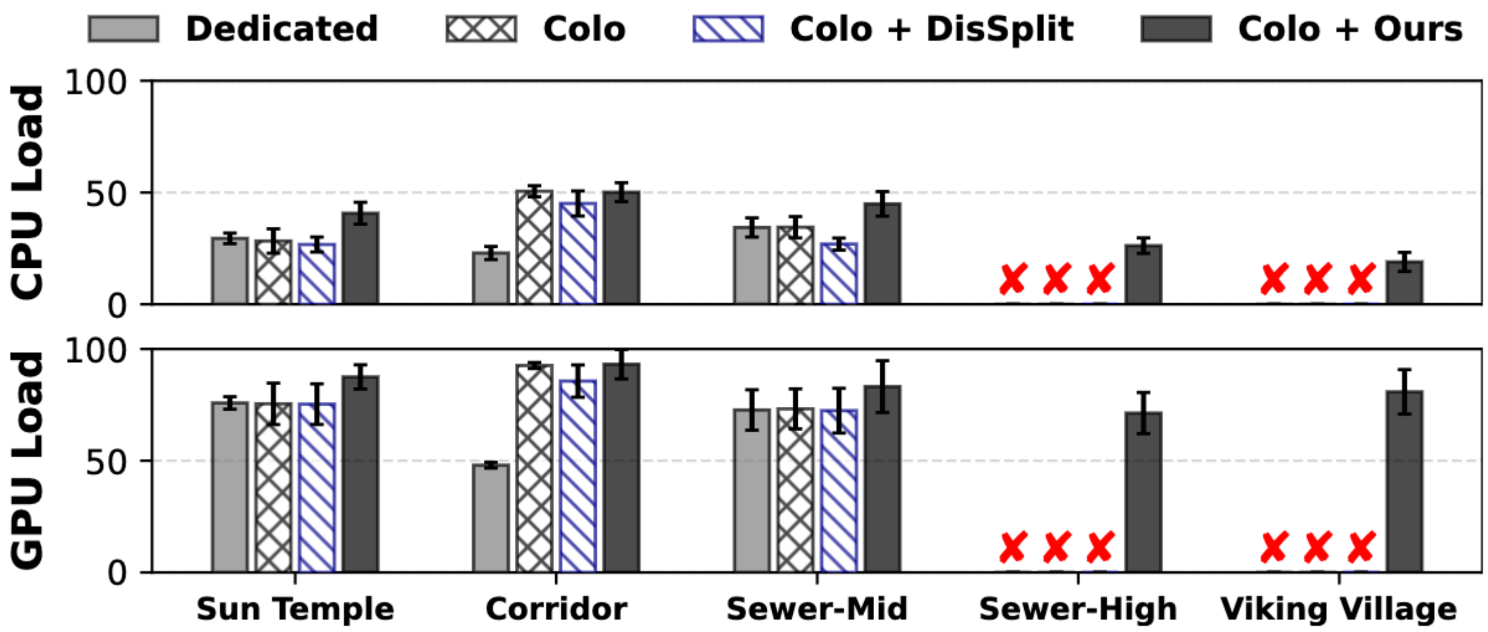
- Game performance (FPS)



Trivial game performance reduction on Sewer-Mid:
Average FPS drops from 54 to 52. (Target FPS: 55)

End-to-end Game Deployment

- Hardware load



- ❑ GPU load: 22% increase compared to dedicated deployment; 7.5% increase compared to game co-location.
- ❑ The average GPU load reaches 97%.
- ❑ The additional CPU costs incurred by duplicate game logic is manageable by a single SoC.

End-to-end Game Deployment

- Frame super-resolution
  - Frame super-resolution is a complementary solution for GPU shortage.
  - 2 out of all 5 games, 16% of all game sessions involve frame super-resolution.

Game	SR Conf	Time Budget	SR Time	Frame Time	Total Time	Frame Quality
Corridor	640x360 x2	33.3 (30 FPS)	16.0	8.9	24.9	33.4
Sun Temple	640x360 x3	33.3 (30 FPS)	18.9	4.76	23.7	29.8

- ❑ The frame super-resolution process can be injected into the frame rendering process (the overall latency is less than the time budget for rendering a frame).
- ❑ Satisfactory frame quality (a PSNR value larger than 30).
- ❑ Mobile NPUs are still fast growing! (15 TOPS on Snapdragon 865 SoC vs. the latest Snapdragon 8 Gen 3)

Conclusion

- Reveal the status quo of mobile cloud gaming on SoC Clusters.
- The first system for high-density mobile cloud gaming on SoC Clusters.
- Two simple yet efficient techniques
  - Pre-rendering game partitioning
  - NPU-enhanced game partitioning coordination mechanism
- Improvement in game deployment density and the ability to support games that cannot be supported by an individual SoC.
- *SFG* Code: https://github.com/lizhang20/SFG

High-density Mobile Cloud Gaming on Edge SoC Clusters

Li Zhang, Shangguang Wang, Mengwei Xu
Beijing University of Posts and Telecommunications (BUPT)