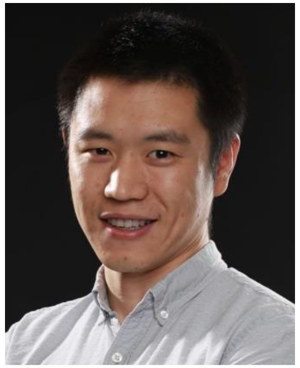
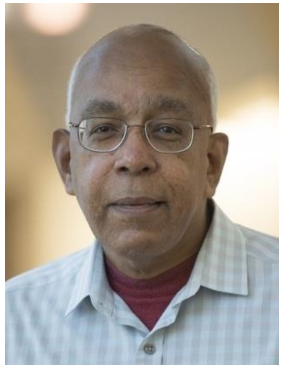
Joint Task Offloading and Routing in Wireless Multi-hop Networks Using Biased Backpressure Algorithm





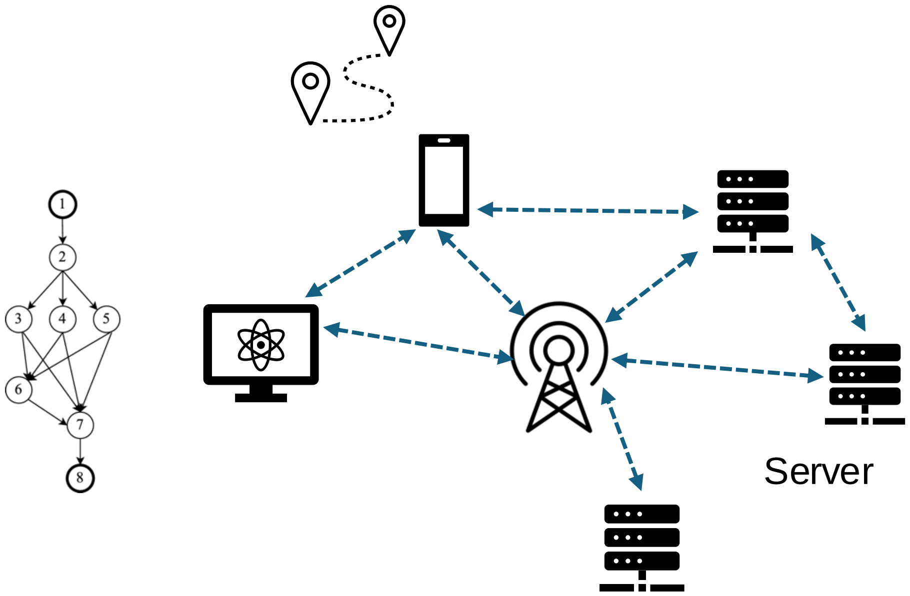



Zhongyuan Zhao*, *Jake Perazzone*†, *Gunjan Verma*†, *Kevin Chan*†, *Ananthram Swami* †, *Santiago Segarra**

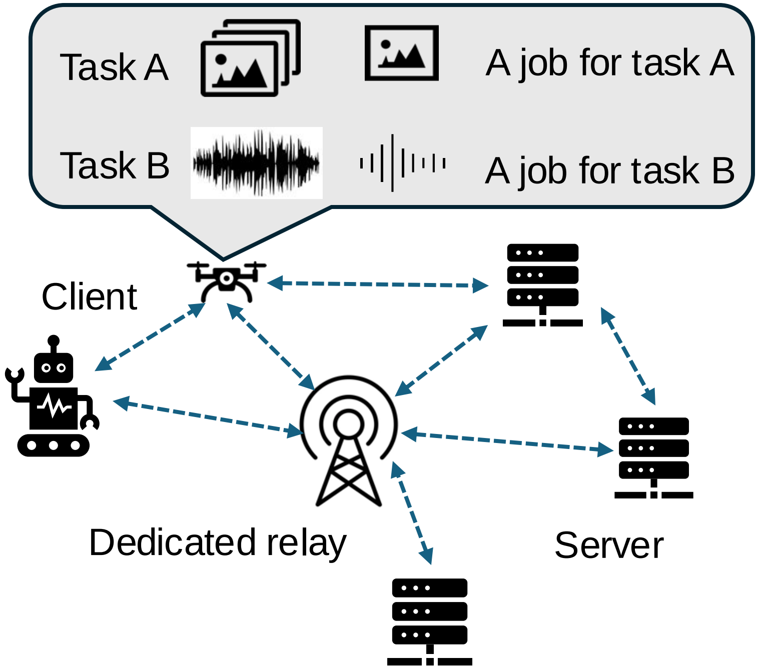
*Rice University, USA

† US Army's DEVCOM Army Research Laboratory, USA

One-time vs recurrent task offloading

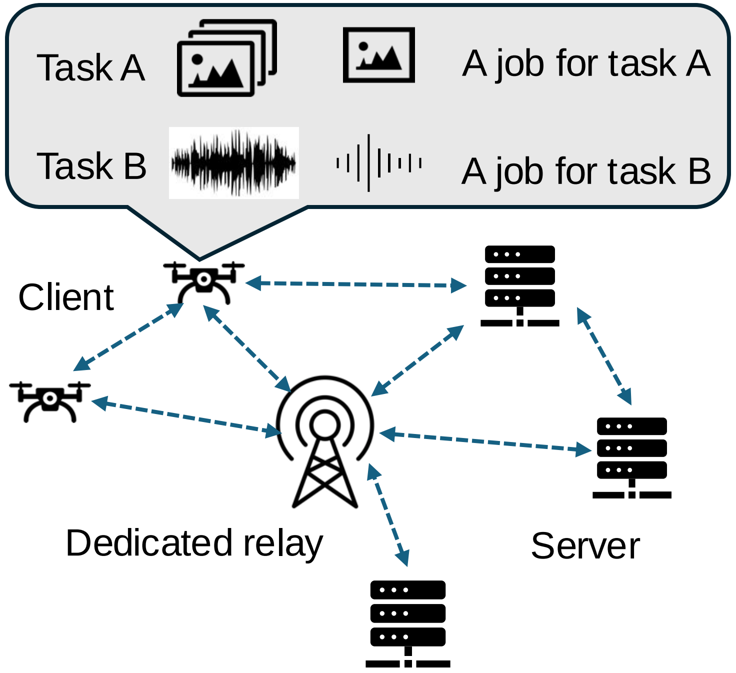


One-time computation offloading
- Path finding
- Scientific computing
- Data analysis



Recurrent computation offloading
- Video surveillance & analytics
- Autonomous robots
- Environmental, health monitoring

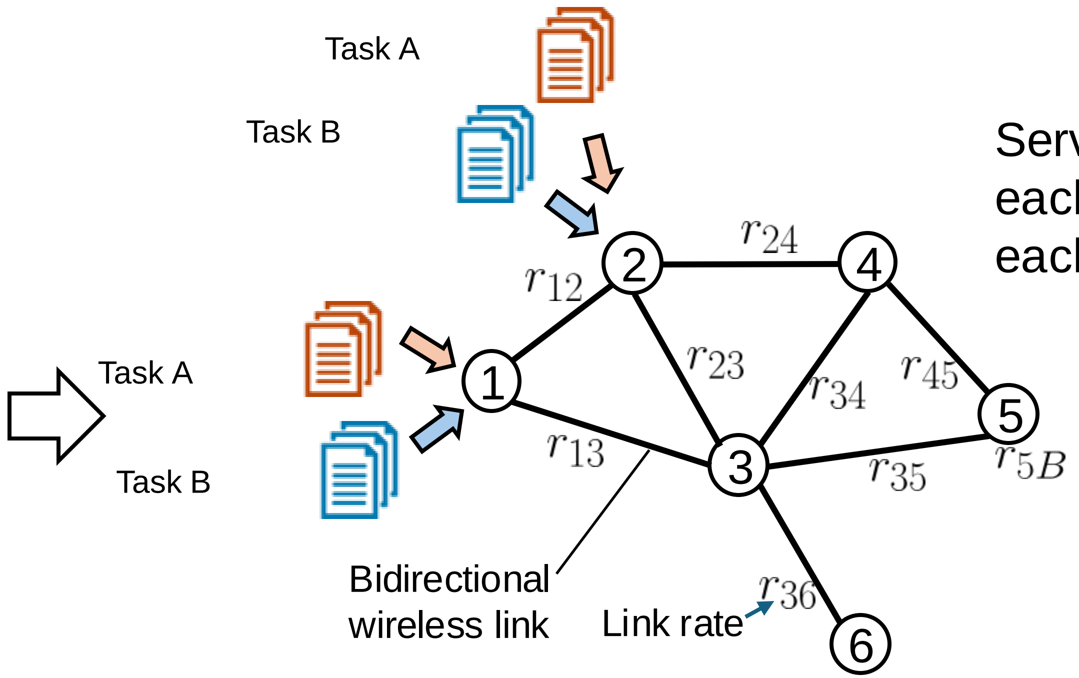
Recurrent task offloading



Task offloading in a wireless multihop network

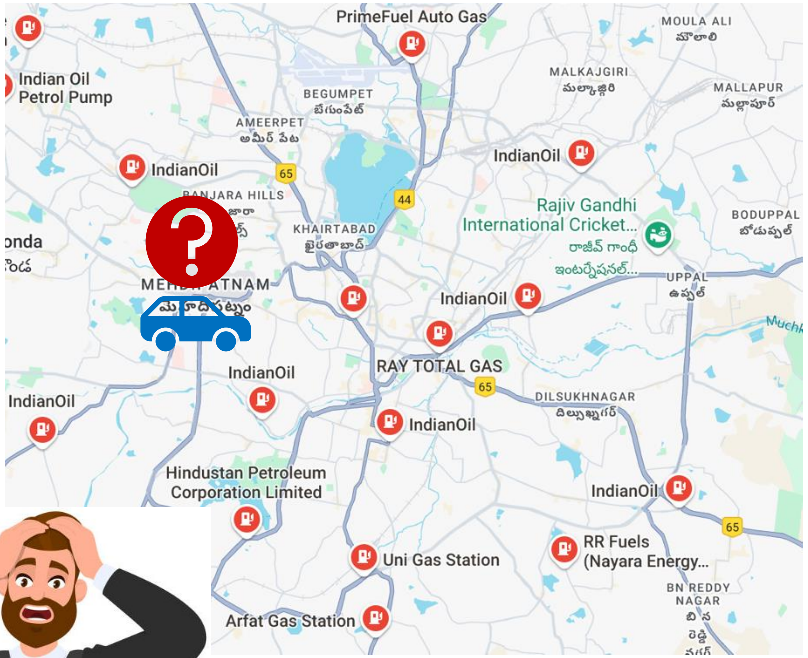
Job arrival rate of each task initialized by each client

Service rate of each server for each type of task



Connectivity Graph

Offloading and routing decisions



- **Where**: pick a destination,
  - server, gas/charging station
- **How**: find a route to destination
- Easy?
  - ★☆☆☆☆ [Greedy] let's go to the **nearest** one, on the **shortest** route
  - ★★☆☆☆ [Strategy 1] Check status of road and destinations, then decide
  - ★★★☆☆ [Strategy 2] Guess the future network status, then decide
  - ★★★☆☆ [Strategy 3] Centralized scheduler

Goal: reduce travel time, avoid congestion (on the road or at the destination)

Strategy 1: centralized joint offloading & routing

- Mixed-integer programming (MIP)
  - Collect full network state
  - Offline, batch operations
  - Joint decisions for all tasks
  - Ignore queueing and interference
  - Integer rate assignment → NP-hard
  - Poor scalability (comm. & compute)

  [Kiamari, 2022], [Li, 2022], [Dai, 2022]

- Linear relaxation for recurrent tasks
  - Model task as divisible liquid flow
  - Fractional probabilistic rate assignment
  - Can be solved quickly

  [Feng, 2021], [Funai, 2019], [Liu, 2020]

flow rate assignment for task type c of client m, on link e

Minimize total costs (latency)

$$\{f_m^c(e)\}^* = \underset{\{f_m^c(e)\}\in\{\mathbb{R}^+\}}{\operatorname{argmin}} \sum_{m\in\mathcal{M}}\sum_{c\in\mathcal{C}} \delta_m^c \quad \text{(1a)}$$

Linear cost (latency)

$$\text{s.t. } \delta_m^c = \sum_{e\in\mathcal{E}} f_m^c(e)u(e) + \sum_{v\in\mathcal{V}} g_m^c(v)u^c(v)\ , \quad \text{(1b)}$$

Flow conservation

$$\lambda_m^c \mathbb{1}(m{=}v) + \sum_{i\in\mathcal{N}(v)} f_m^c((i,v)) = g_m^c(v) + \sum_{i\in\mathcal{N}(v)} f_m^c((v,i)), \quad \text{(1c)}$$

In-flow = out-flow

$$\lambda_m^c = \sum_{v\in\mathcal{V}} g_m^c(v)\ ,\ \forall\ m\in\mathcal{M}, c\in\mathcal{C}\ , \quad \text{(1d)}$$

Server capacity

$$\psi(v) \geq \sum_{c\in\mathcal{C},m\in\mathcal{M}} g_m^c(v)h^c(v),\ \forall\ v\in\mathcal{V}\ , \quad \text{(1e)}$$

Link capacity

$$\psi(e) \geq \sum_{c\in\mathcal{C},m\in\mathcal{M}} f_m^c(e),\ \forall\ e\in\mathcal{E}\ , \quad \text{(1f)}$$

$$h^c(v), u(e), g_m^c(v), u^c(v) \in \mathbb{R}^+, \forall\ e\in\mathcal{E}, m, v\in\mathcal{V}, c\in\mathcal{C}, \quad \text{(1g)}$$

Non-negative costs

Ignore interference

RICE UNIVERSITY

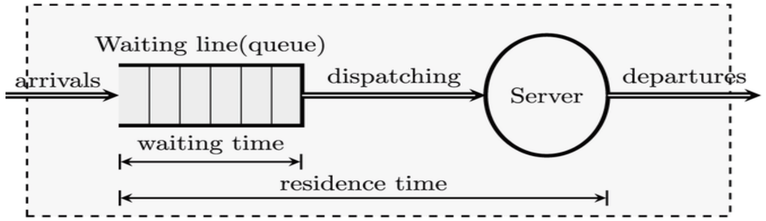
DEVCOM ARMY RESEARCH LABORATORY

Strategy 2: separated offloading and routing

- Distributed solutions
  - Clients collect real-time state of servers
  - Online, asynchronized decisions
  - First decide destination, then find path
  - Consider queueing and interference
  - Various decision granularity
  - Limited scalability (comm. overhead)

[Kamran, 2022], [Lin, 2020], [Jiang, 2023], [Liu 2019], [Bi 2021], [Zhao 2024]

- An ideal benchmark
  - Clients know real-time servers' state at no cost
  - SOTA routing in wireless multihop networks
    - Backpressure offloading
    - SP-BP routing
  - Maximum queue stability



SP-BP: shortest-path biased Backpressure routing and scheduling*

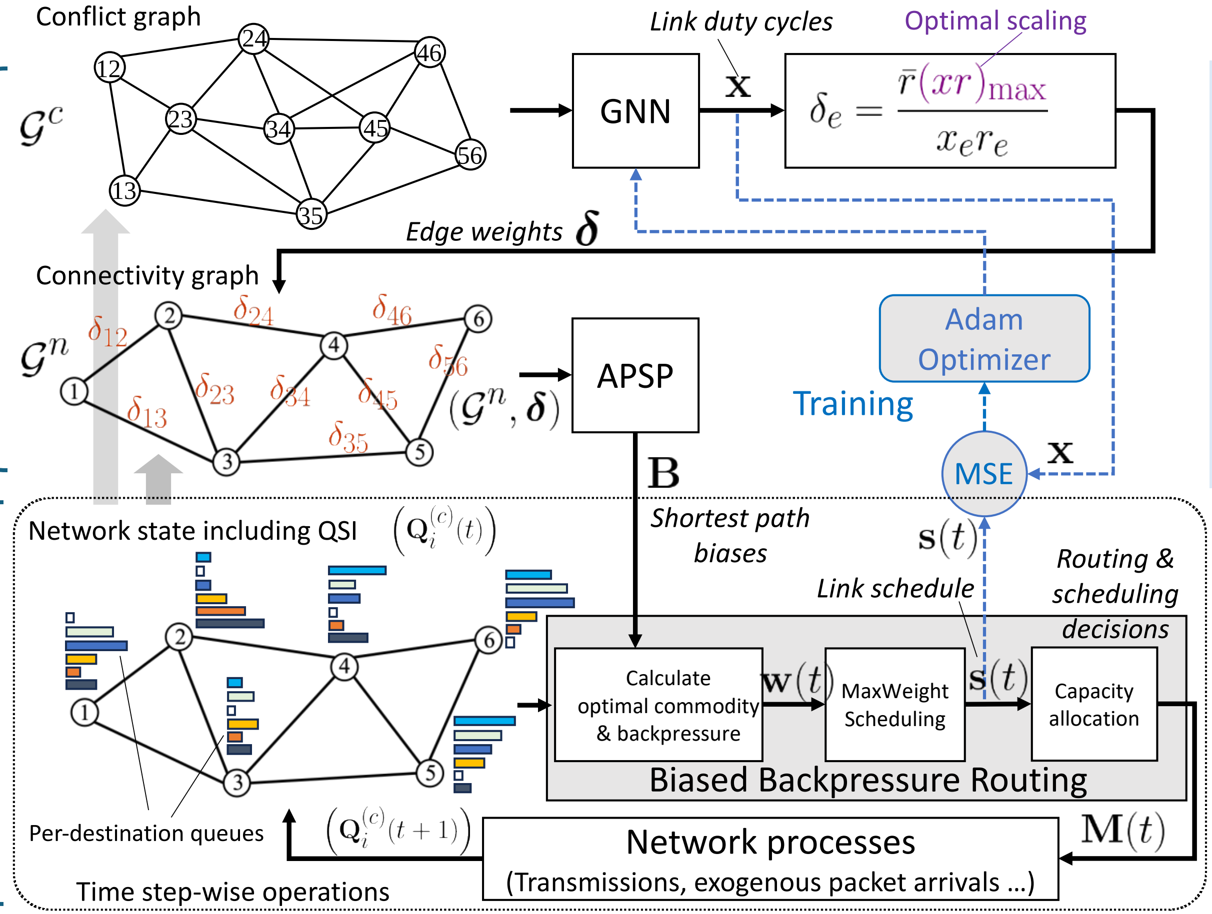
SP_SP-BP:
offload a job to the server with the shortest queue, then SP-BP routing takes the job to the server

SPBP_SP-BP:
offload a job to the server with the shortest queue + processing bandwidth,
then SP-BP routing takes the job to the server

SP-BP: shortest-path biased backpressure

One-time shortest path bias computing

Fully distributed routing and scheduling



Per-destination queueing system

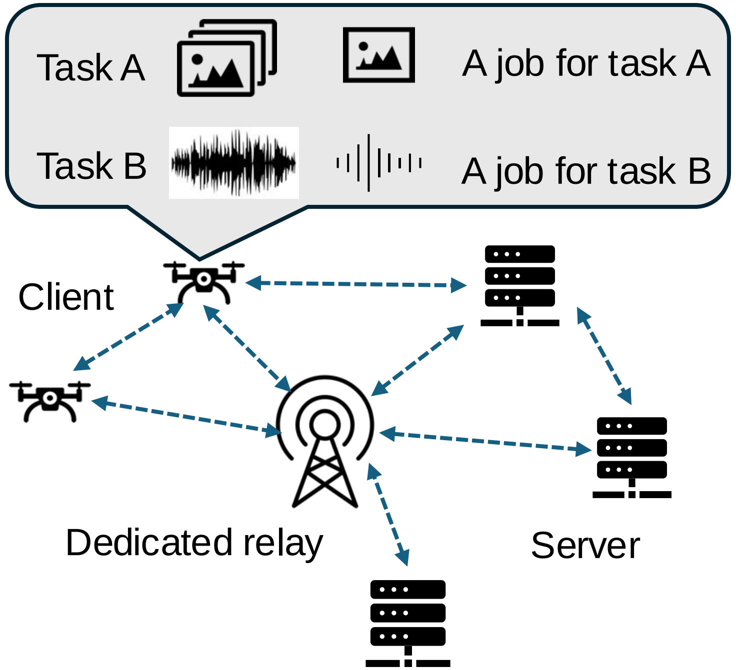
SP-BP routing and scheduling*, considers both the shortest path and congestion state,

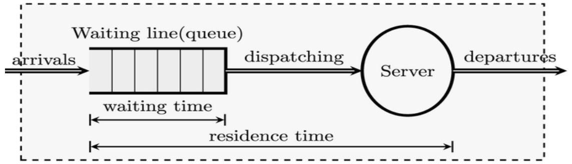
- minimize latency
- maximum queue stability (throughput optimality)
- Mitigate last packet problem
- Minimize startup time, random walking

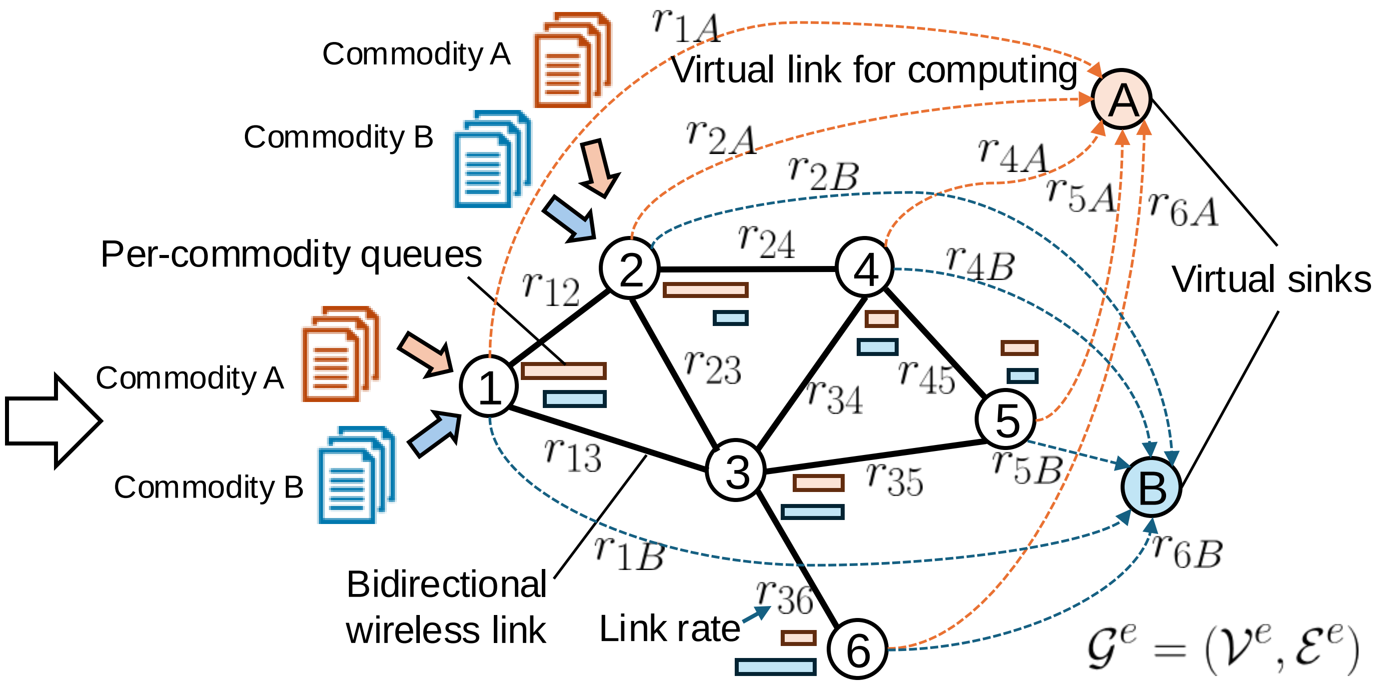
- * Z. Zhao, B. Radojičić, G. Verma, A. Swami and S. Segarra, "Biased Backpressure Routing Using Link Features and Graph Neural Networks," in IEEE TMLCN, vol. 2, pp. 1424-1439, 2024
- M. Neely, E. Modiano, and C. Rohrs, "Dynamic power allocation and routing for time-varying wireless networks," IEEE J. Sel. Areas Commun., vol. 23, no. 1, pp. 89–103, 2005

Our approach

1. Model computing as sending jobs to virtual sinks over virtual links
2. Joint offloading and routing → SP-BP routing



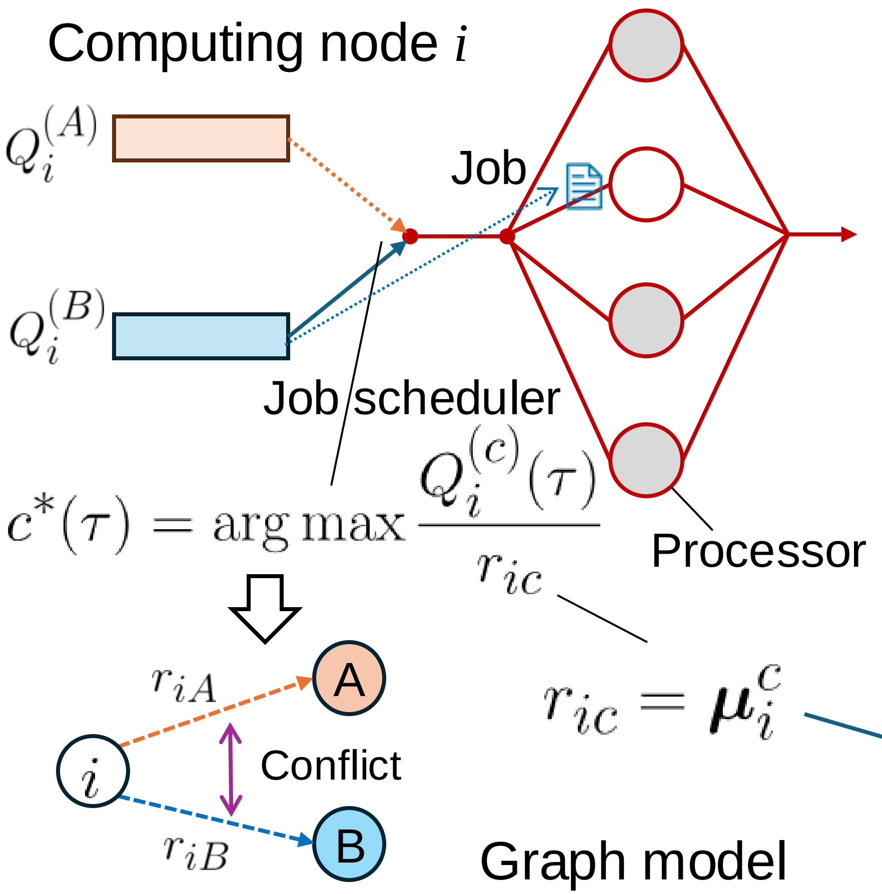




A job can take flexible route to virtual sink (its server is not pre-determined)

Extended connectivity graph

Job Scheduling for computing nodes

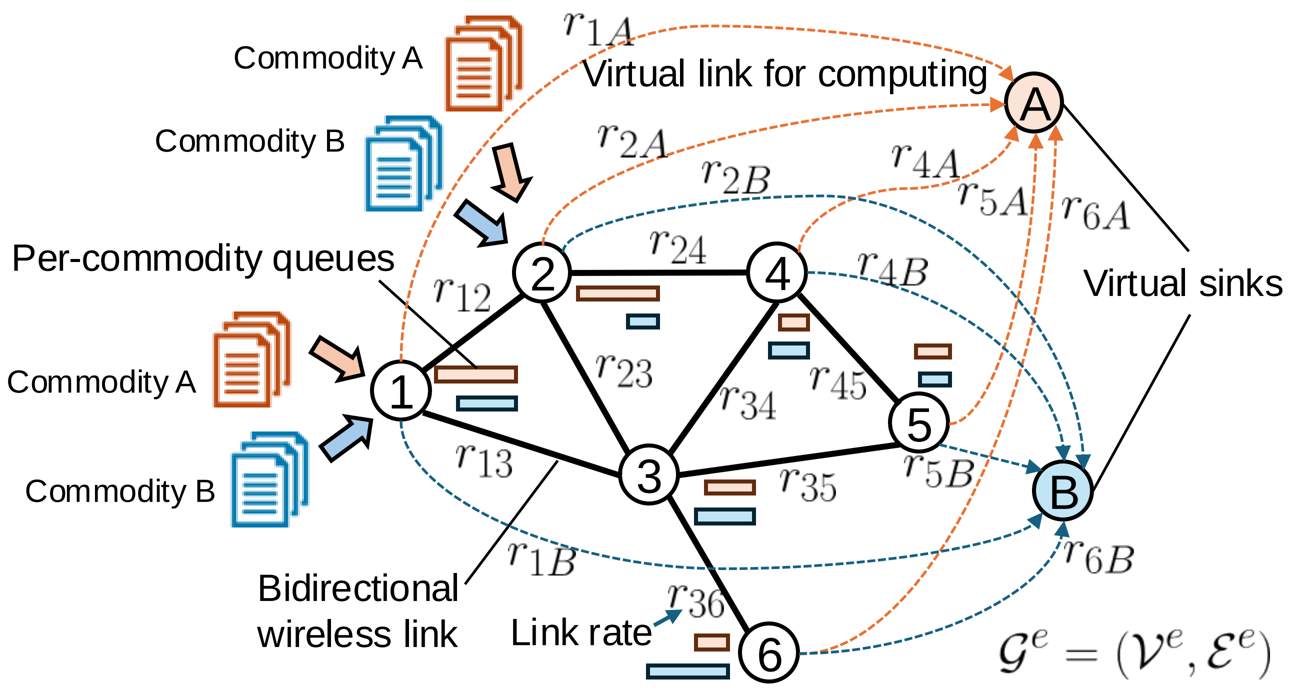


Job scheduling operation
Whenever there is free processor on a server, send it a job from the queue requires the longest time to clear

Virtual link model
Conflict: virtual links on the same server conflicting with each other

Virtual link rate: how many jobs can be processed by the server in a time slot, assuming no other types of jobs

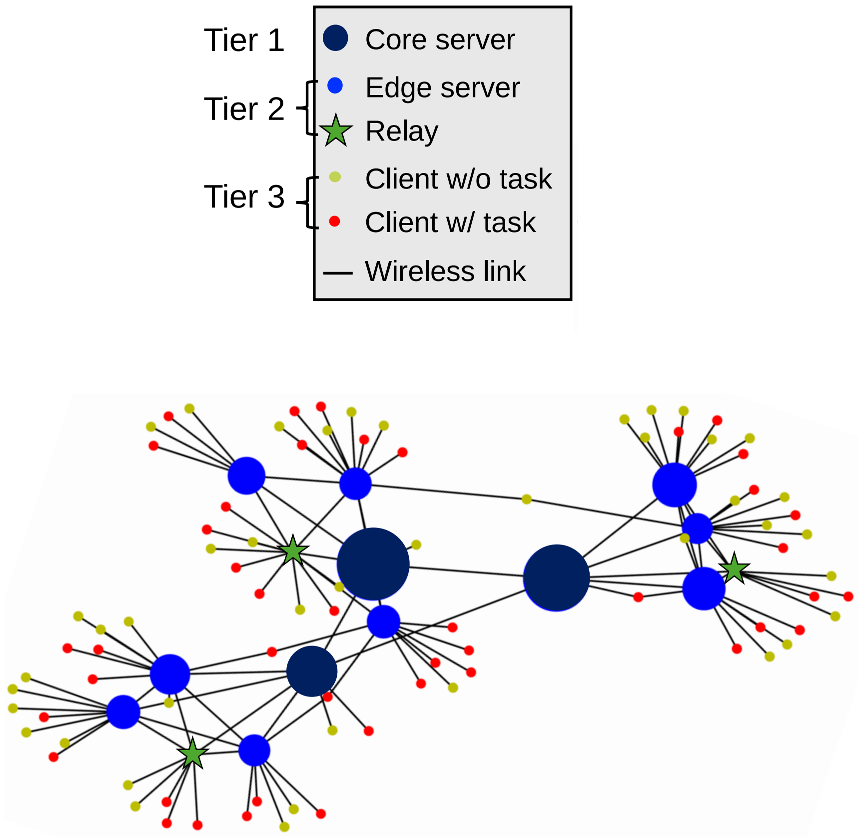
SP-BP on extended connectivity graph



- Computing (virtual links) operate in continuous time
- Physical link scheduling not impacted

Numerical results

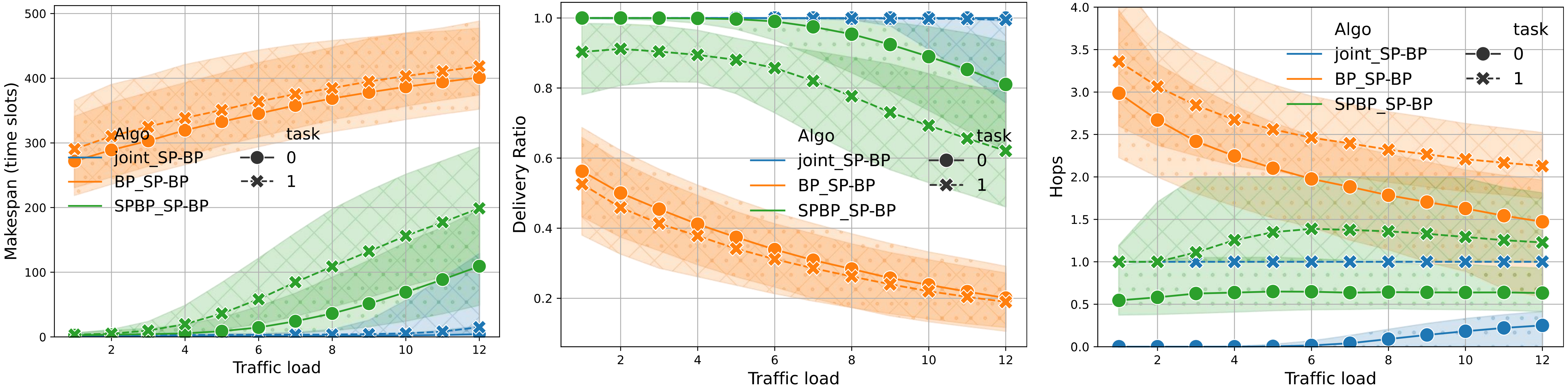
- Random hierarchical wireless multihop networks
  - Network size: 100 nodes,
  - 100 instances = 10 networks X 10 test cases
- Job arrivals: 50% streaming and 50% bursty
  - Arrival time of bursty clients are random
- Time horizon T=1000
- Computing capacity
  - Servers: Pareto distribution (shape=2, scale=8)
  - Clients: Uniform (8, 12)
- Long-term link rates: uniform
- Real-time link rates: truncated Gaussian



Node size indicates computing power

Two types of tasks

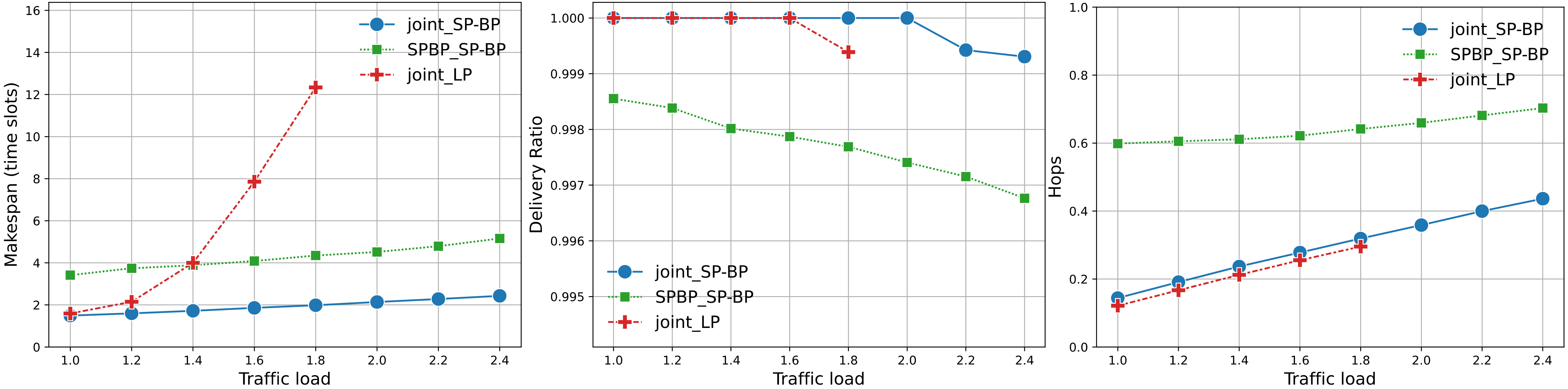
Task 0: can be computed locally and remotely
Task 1: can not be computed by clients



- BP_SP-BP: worst makespan, delivery ratio, hop-distance
  - Offloading only consider QSI, offload to remote servers
- SPBP_SP-BP:
  - Offloading considers QSI and server capacity, significant improvement
- Joint_SP-BP: far better than separated offloading and routing

Single task type

- Task 0: can be computed locally and remotely
- Scaled down job arrival rates and computing bandwidth by a factor of 8.0
- Communication bandwidth is relatively scaled up



- Easier setting to test joint_LP (linear programming relaxation)
- Joint_LP (unable to find solution for load > 1.8)
  - Matches joint_SP-BP in lowest traffic load
  - Quickly degrade with traffic intensity, due to inter-flow interference
- joint_SP-BP: best, scales well by traffic intensity

Conclusion + future direction

- Joint offloading, routing, scheduling
  - A fully distributed, asynchronized solution
    - Need no real-time server-side QSI
    - Tasks can be initialized at any time
  - Offloading without (prescribed) destinations, routing without route
    - The next step of a job is decided by its current hosting node
    - Long-term global potential + updated QSI in local neighborhood
  - High performance, good scalability, maximum queue stability
- Future directions
  - Context switching cost in job scheduling
  - Multi-stage task (forward/return trip), complex workflows
  - Task/traffic priorities
  - Compatibility with regular data traffics

References

1. S. Müller, H. Al-Shatri, M. Wichtlhuber, D. Hausheer, and A. Klein, "Computation offloading in wireless multi-hop networks: Energy minimization via multi-dimensional knapsack problem," in IEEE Annual Intl. Symp. on Personal, Indoor, and Mobile Radio Communications (PIMRC), pp. 1717–1722, 2015.
2. C. Funai, C. Tapparello, and W. Heinzelman, "Computational offloading for energy constrained devices in multi-hop cooperative networks," IEEE Trans. Mobile Computing., vol. 19, no. 1, pp. 60–73, 2019.
3. B. Liu, Y. Cao, Y. Zhang, and T. Jiang, "A distributed framework for task offloading in edge computing networks of arbitrary topology," IEEE Trans. Wireless Commun., vol. 19, no. 4, pp. 2855–2867, 2020
4. G. Feng, X. Li, Z. Gao, C. Wang, H. Lv, and Q. Zhao, "Multi-path and multi-hop task offloading in mobile ad hoc networks," IEEE Trans. Vehicular Tech., vol. 70, no. 6, pp. 5347–5361, 2021.
5. M. Kiamari and B. Krishnamachari, "GCNscheduler: Scheduling distributed computing applications using graph convolutional networks," in Proc. 1st Intl. Workshop on Graph Neural Netw., pp. 13–17, 2022.
6. X. Li, T. Chen, D. Yuan, J. Xu, and X. Liu, "A novel graph-based computation offloading strategy for workflow applications in mobile edge computing," IEEE Trans. Services Computing, vol. 16, no. 2, pp. 845–857, 2022.
7. X. Dai, Z. Xiao, H. Jiang, H. Chen, G. Min, S. Dustdar, and J. Cao, "A learning-based approach for vehicle-to-vehicle computation offloading," IEEE Internet of Things Journal, vol. 10, no. 8, pp. 7244–7258, 2022.
8. Z. Zhao, J. Perazzone, G. Verma, and S. Segarra, "Congestion-aware distributed task offloading in wireless multi-hop networks using graph neural networks," in IEEE Int. Conf. on Acoustics, Speech and Signal Process. (ICASSP), pp. 8951–8955, 2024.
9. A. Destounis, G. S. Paschos, and I. Koutsopoulos, "Streaming big data meets backpressure in distributed network computation," in IEEE Intl. Conf. on Computer Comms. (INFOCOM), pp. 1–9, 2016.
10. Y. Wang, L. Wang, R. Zheng, X. Zhao, and M. Liu, "Latency-optimal computational offloading strategy for sensitive tasks in smart homes," Sensors, vol. 21, no. 7, 2021.
11. K. Kamran, E. Yeh, and Q. Ma, "DECO: Joint computation scheduling,caching, and communication in data-intensive computing networks," IEEE/ACM Trans. Netw., vol. 30, no. 3, pp. 1058–1072, 2022.
12. R. Lin, Z. Zhou, S. Luo, Y. Xiao, X. Wang, S. Wang, and M. Zukerman, "Distributed optimization for computation offloading in edge computing," IEEE Trans. Wireless Commun., vol. 19, no. 12, pp. 8179–8194, 2020.
13. H. Jiang, X. Dai, Z. Xiao, and A. Iyengar, "Joint task offloading and resource allocation for energy-constrained mobile edge computing," IEEE Trans. Mobile Computing., vol. 22, no. 7, pp. 4000–4015, 2023.
14. C.-F. Liu, M. Bennis, M. Debbah, and H. V. Poor, "Dynamic task offloading and resource allocation for ultra-reliable low-latency edge computing," IEEE Trans. Commun., vol. 67, no. 6, pp. 4132–4150, 2019.
15. S. Bi, L. Huang, H. Wang, and Y.-J. A. Zhang, "Lyapunov-guided deep reinforcement learning for stable online computation offloading in mobile-edge computing networks," IEEE Trans. Wireless Commun., vol. 20, no. 11, pp. 7519–7537, 2021.