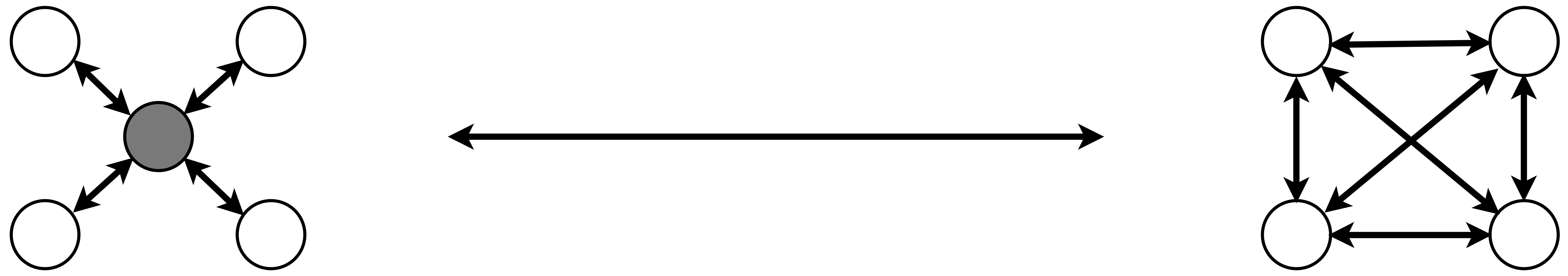
Lecture 8

Hybrid Architectures

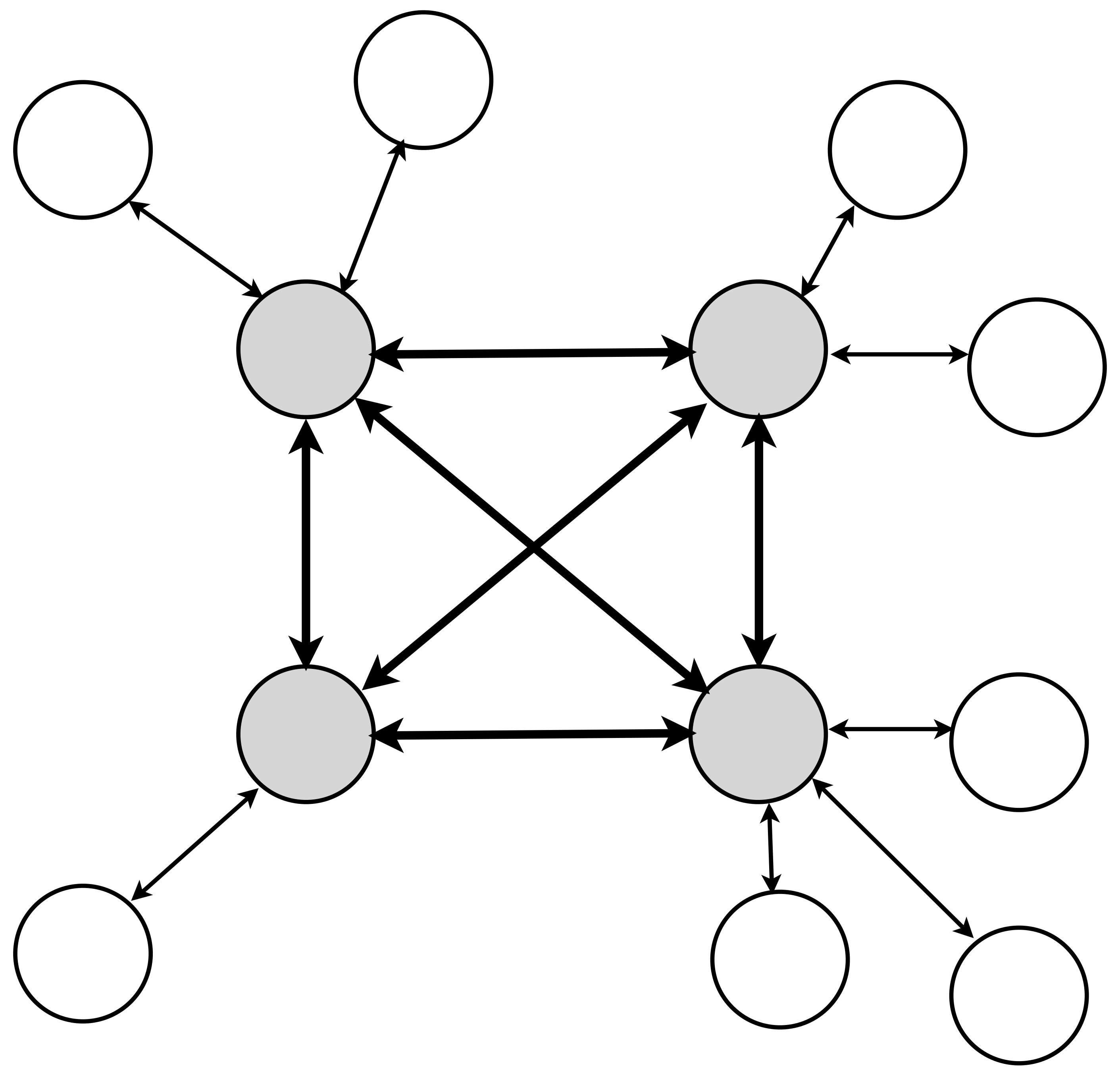


single
centralized
server

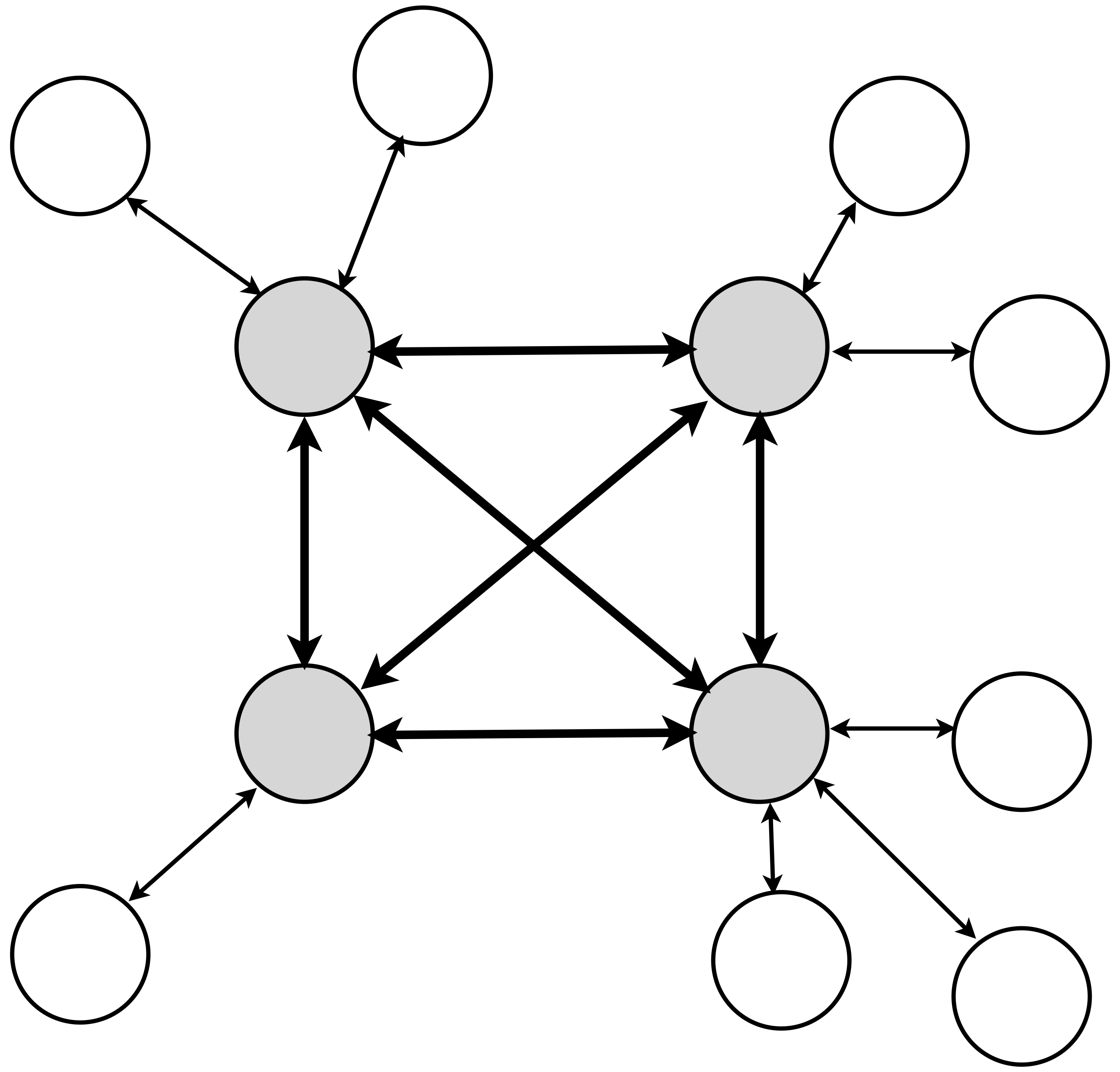
completely
decentralized

Mirrored Servers Architecture

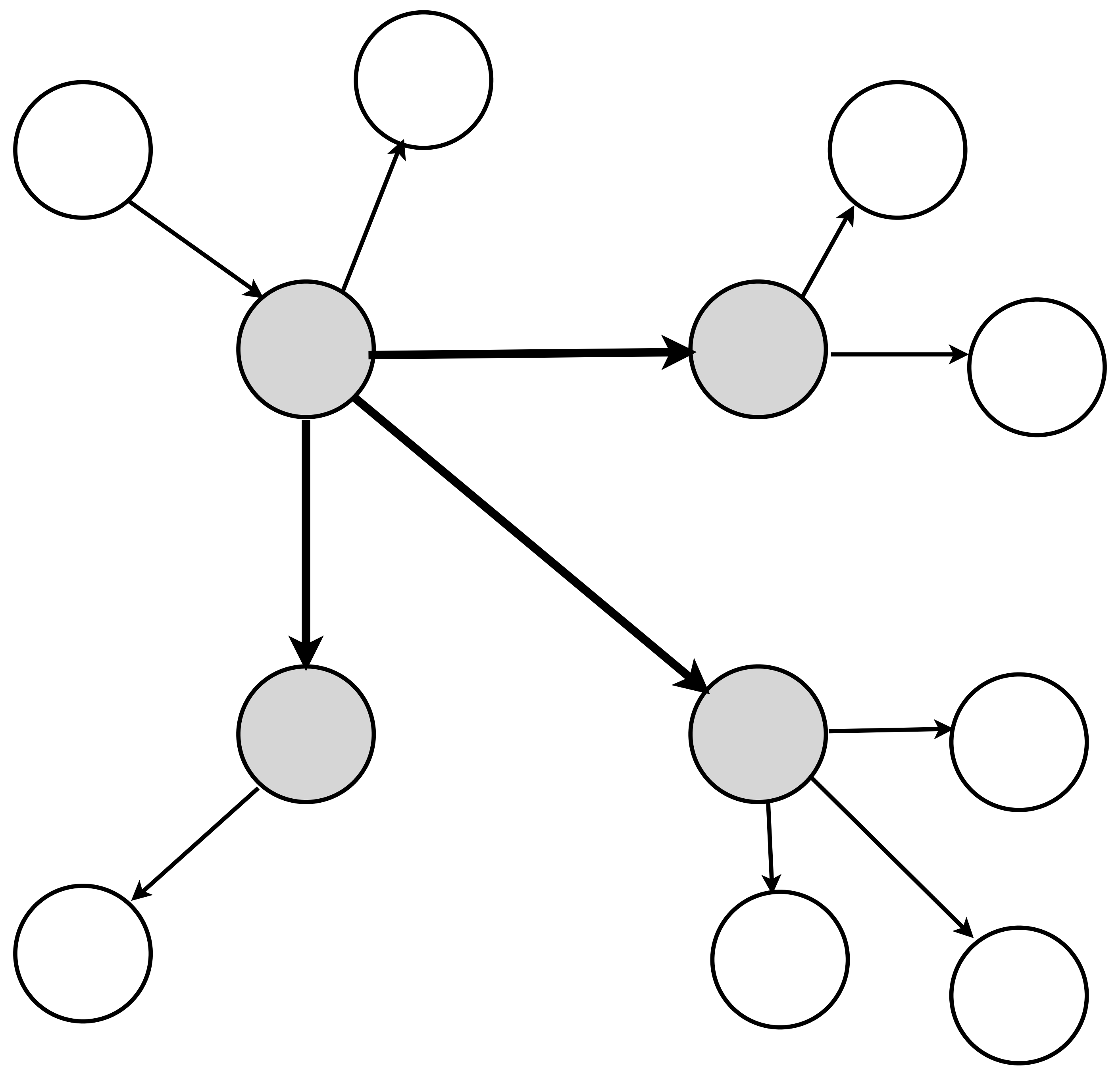
Multiple servers, each replicates the complete game states and serves clients from different geographical regions.   Each client connects to one server.



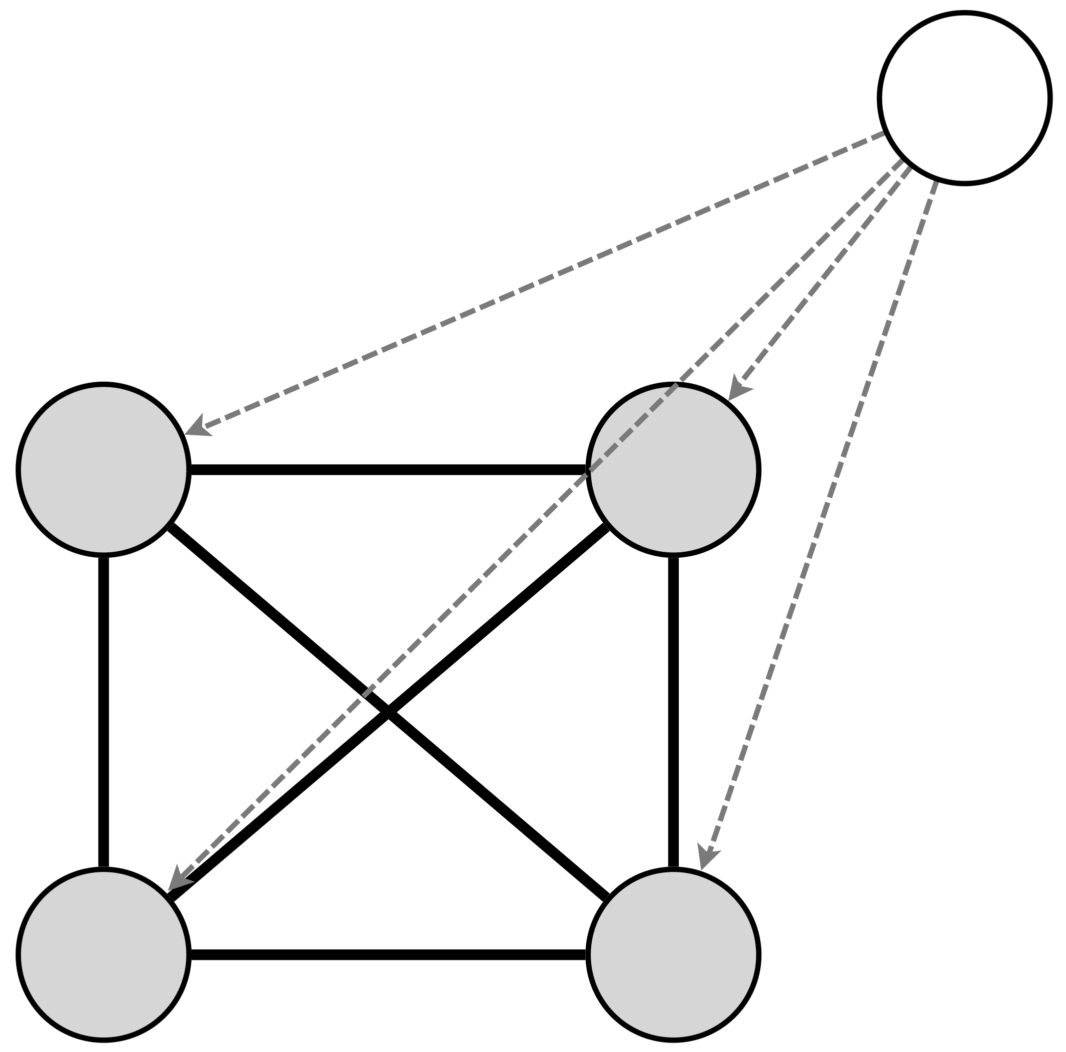
Servers may be connected with high speed, provisioned networks, reducing synchronization latency among the servers.



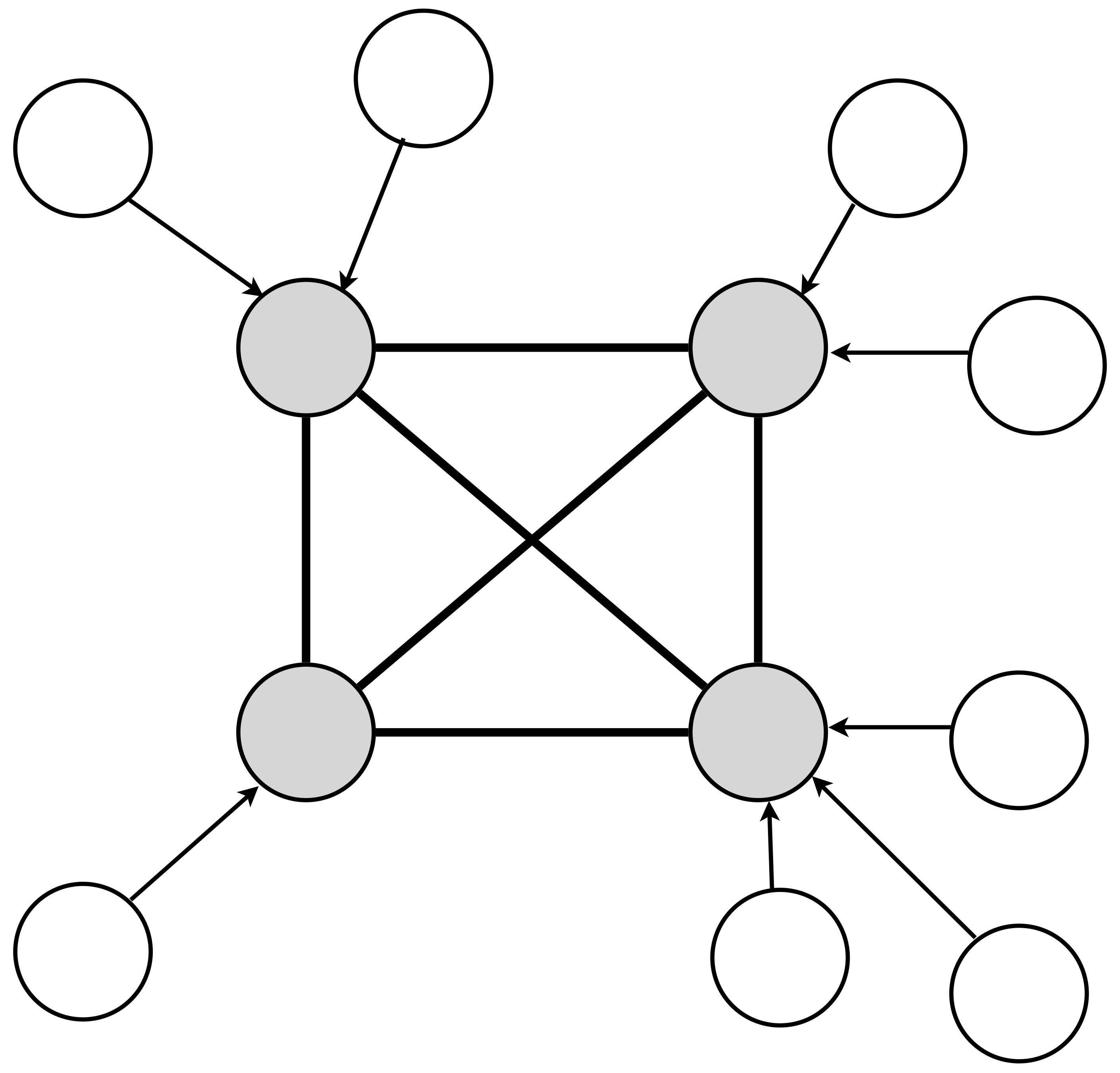
An event from a client is sent to its server. The server then forwards the event to other servers. Each server updates the states of the game, and forward the new states to all relevant clients.



Client usually connects to the server with the shortest RTT (subjected to capacity)



In what order should events be processed?



Advantage: No single point of failure. If a game server is down, client can reconnect to another replicated server.

Advantage: Network capacity is increased. Good if network bandwidth is a bottleneck.

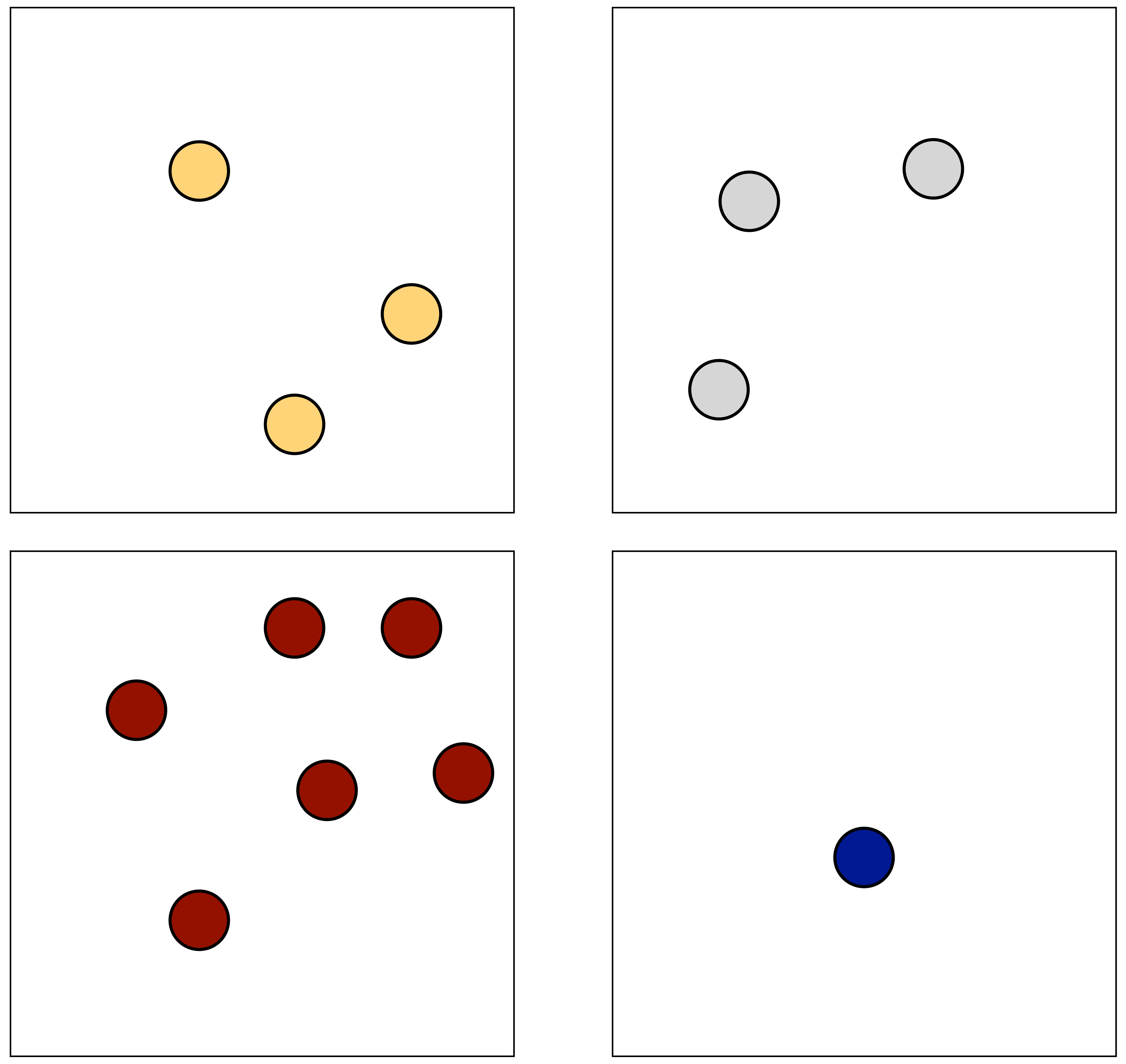
Advantage: Lower latency between client and server.

Drawback: Higher latency between clients

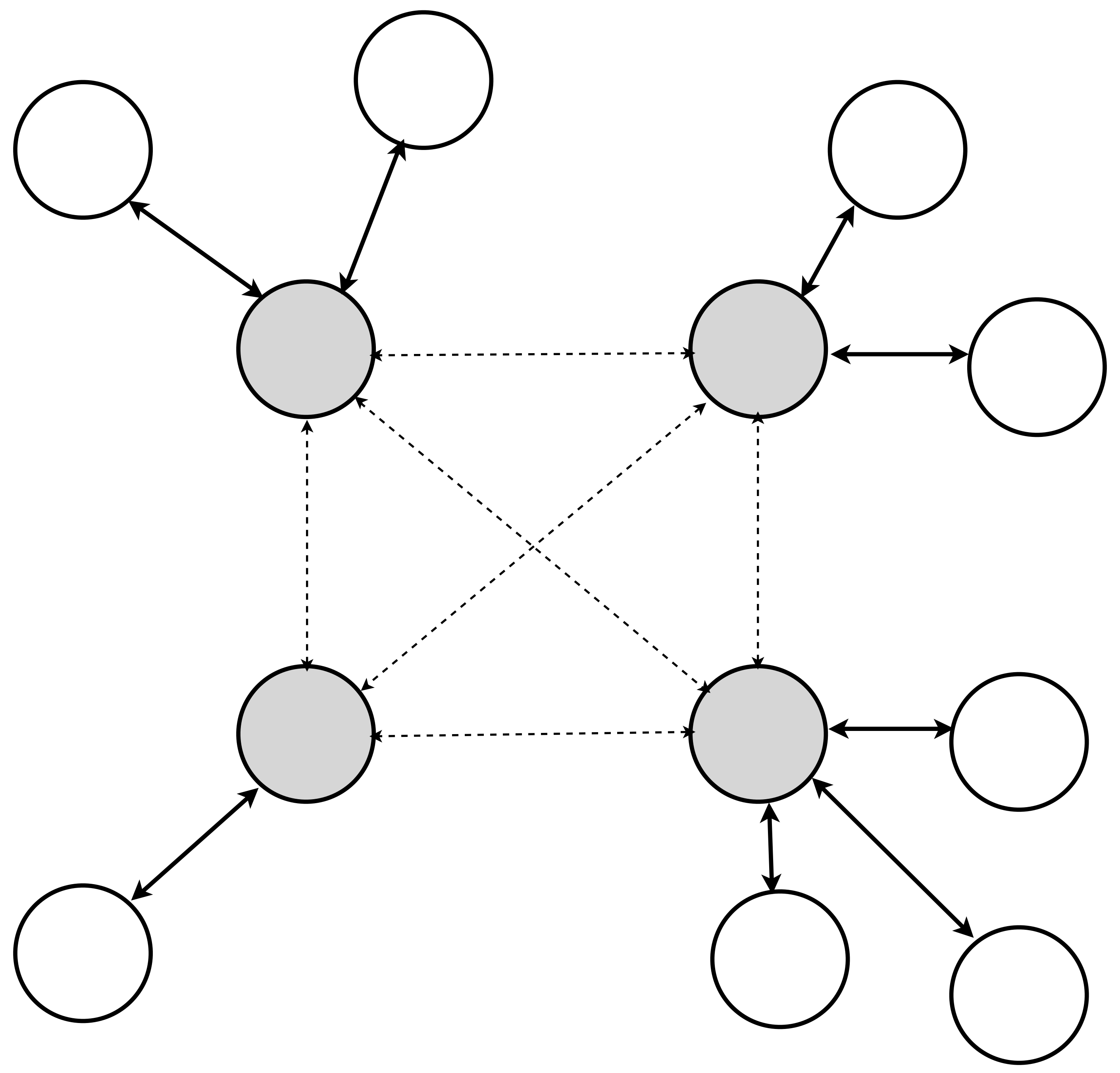
Drawback: Does not scale to large world (if computation is an overhead)

Zoned Servers Architecture

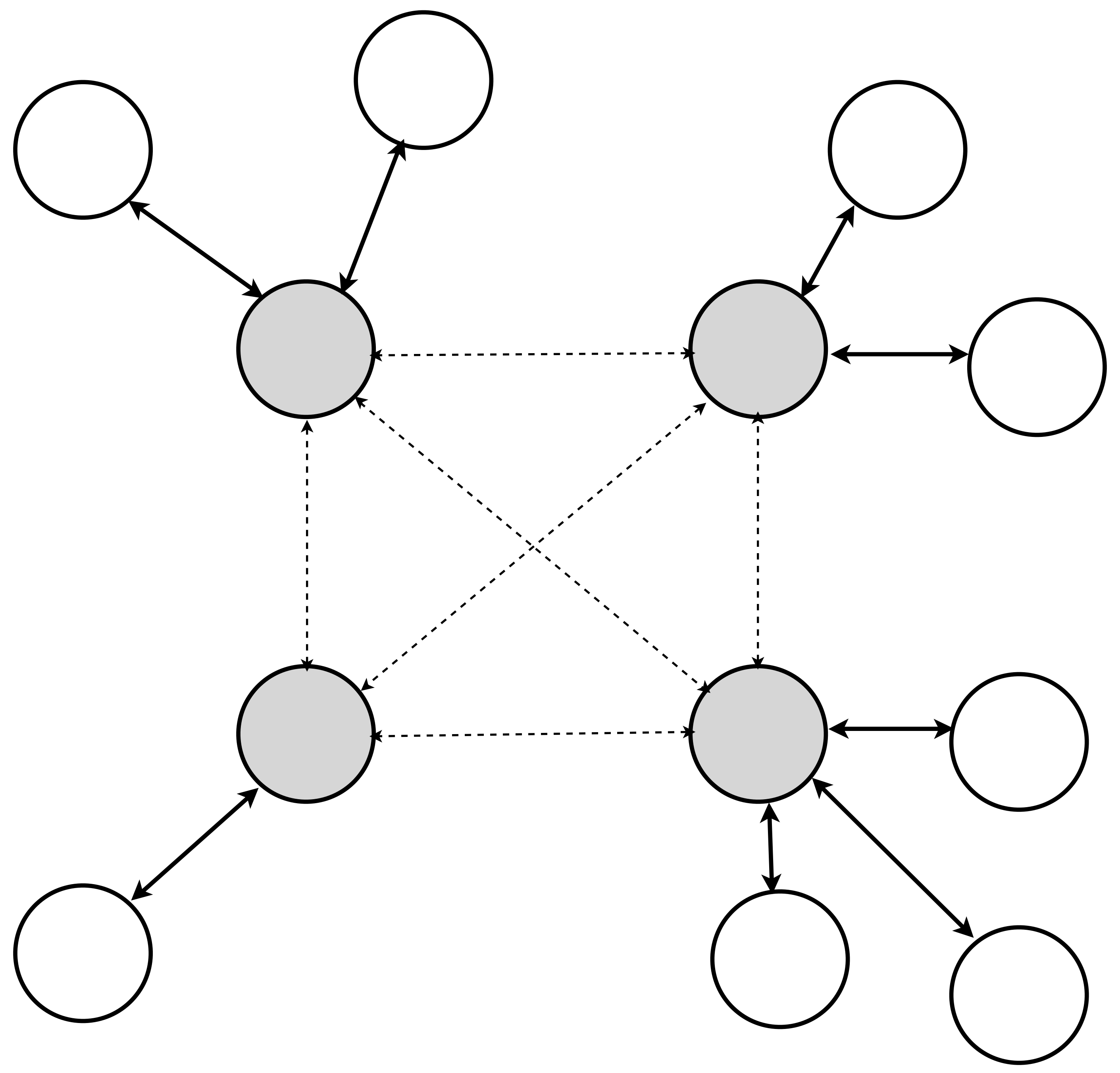
Divide the game world into independent regions.
One server is in charge of one region.



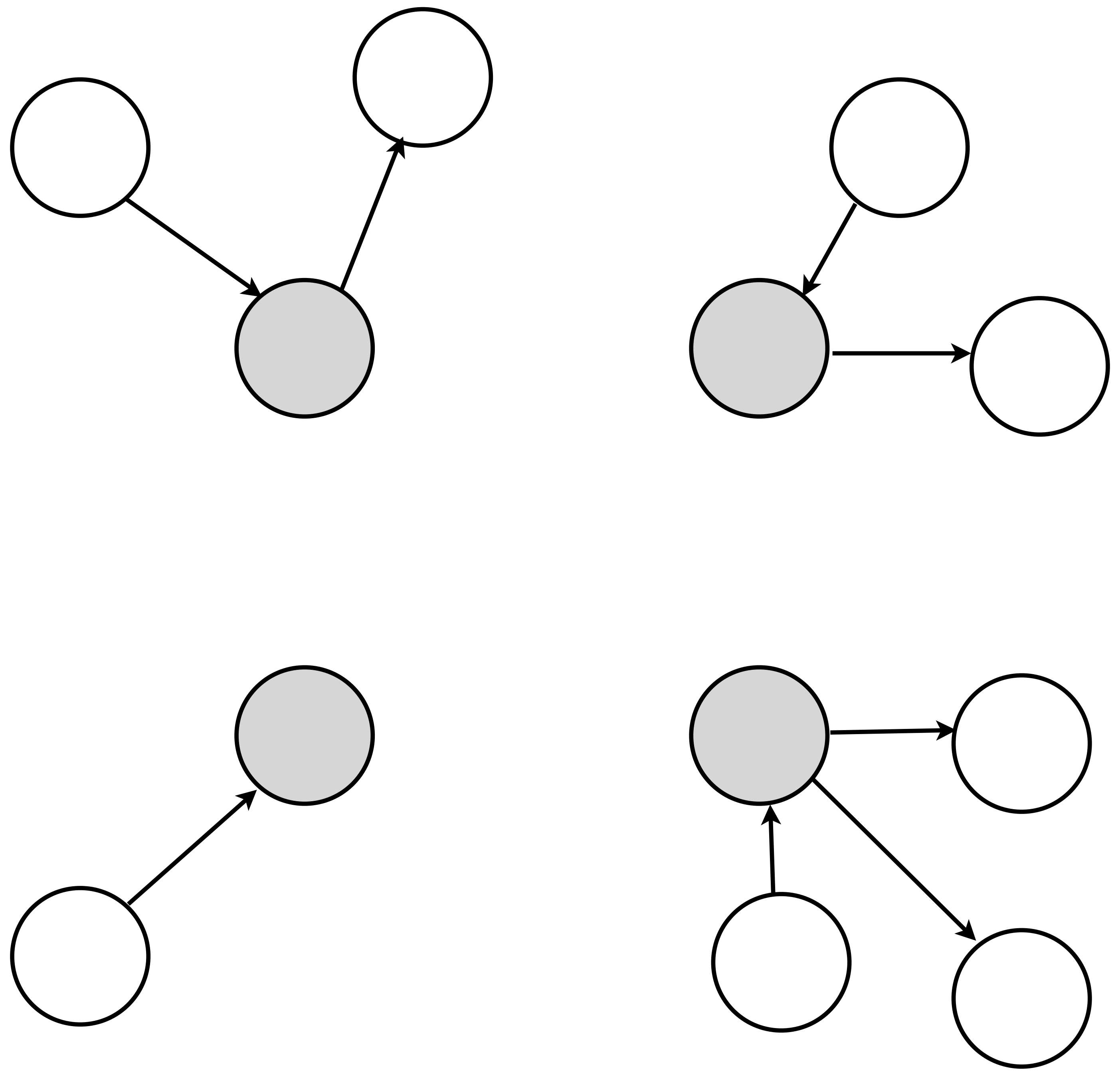
A client connects to the server serving its current region, and is handed off to another server when it moves to another region.



Servers seldom communicate with each other.



Event from a client is sent to its server, which is then forwarded by its server to all relevant clients in the same region.



Drawback: Scale to large world (just add server)

Drawback: Single point of failure

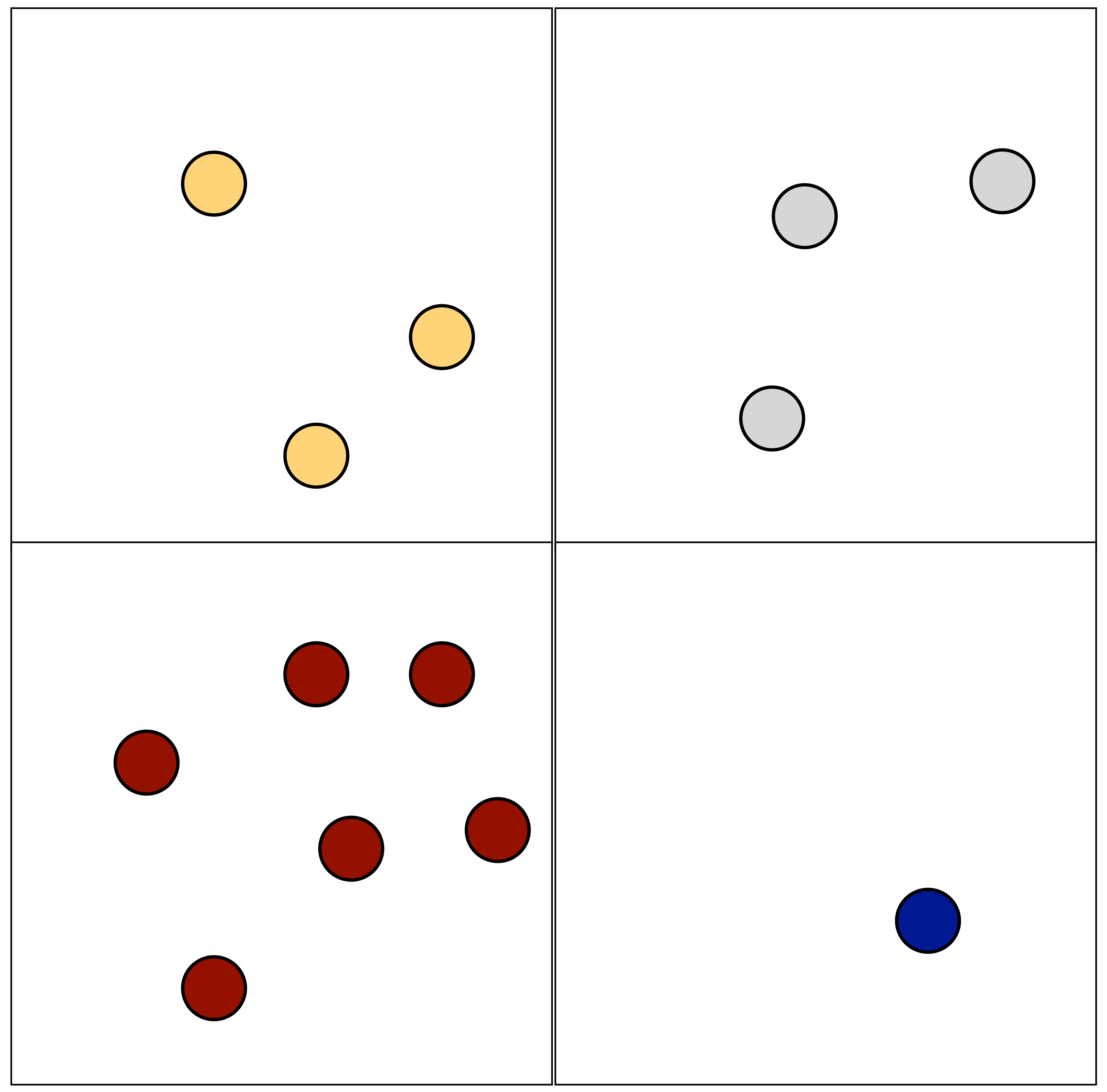
(but can deploy mirrored servers within a zone)

Drawback:

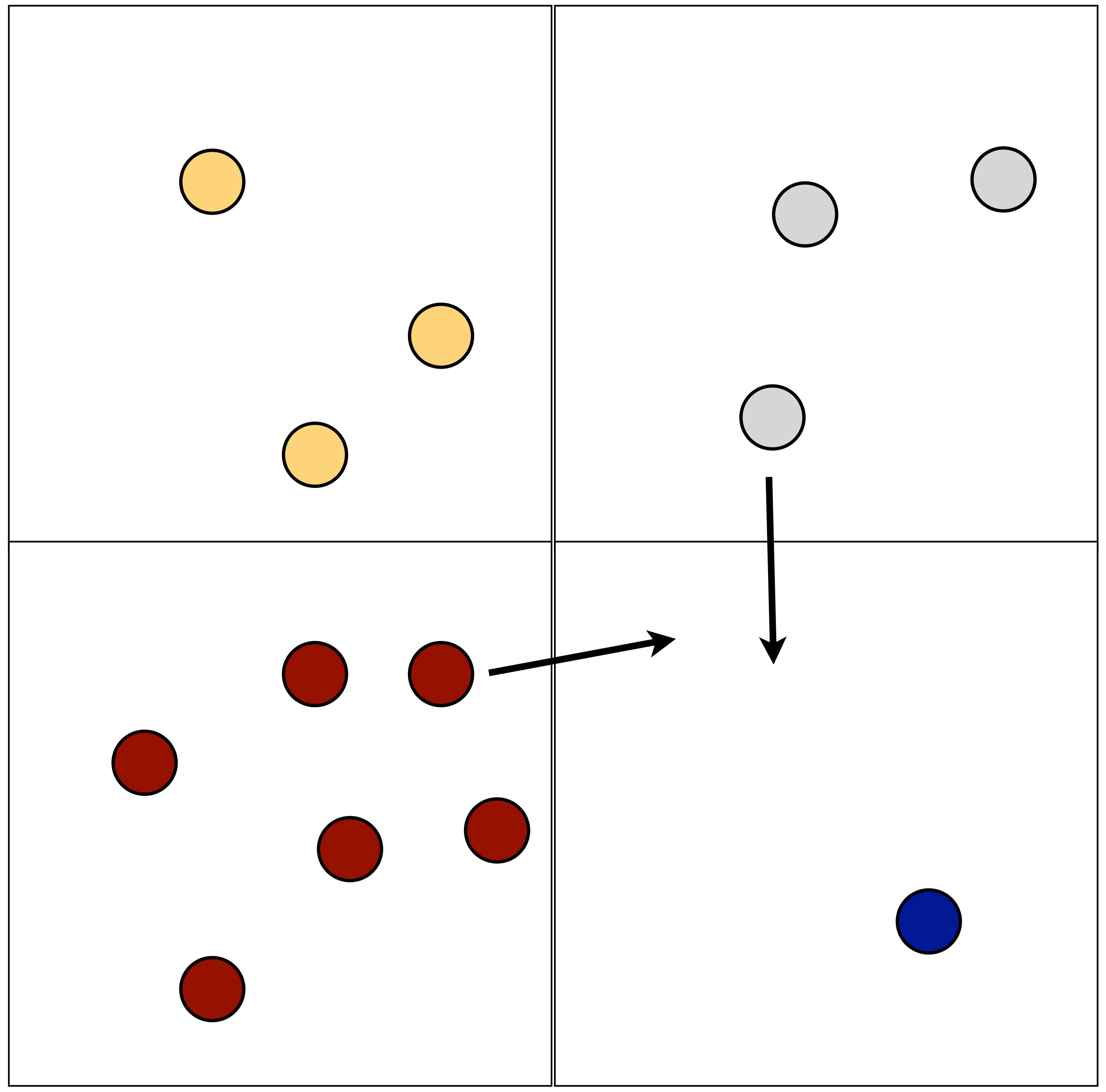
Constrained game design

Supporting Seamless Game Worlds

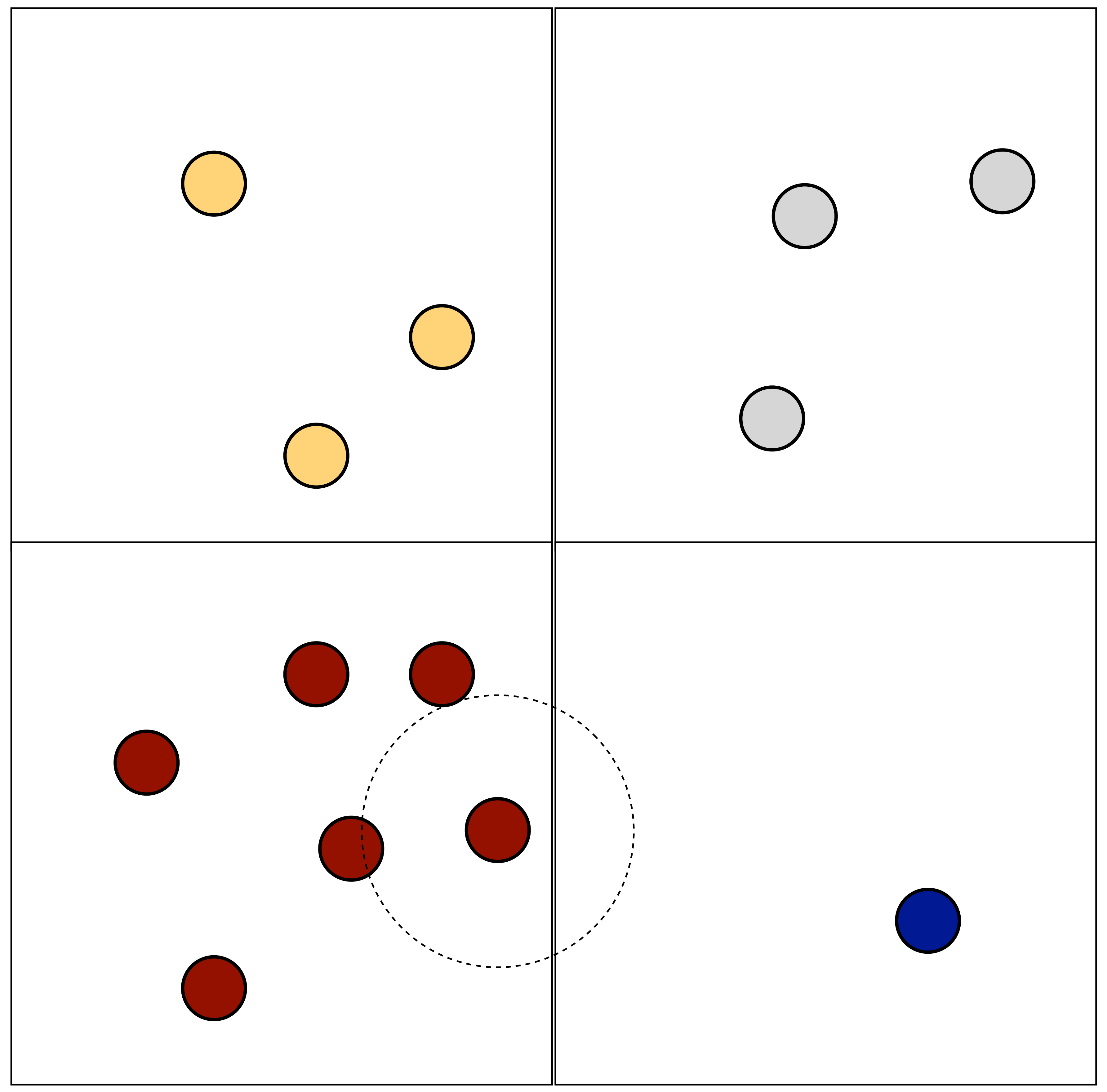
Divide the game world into regions.
One server is in-charged of one region.



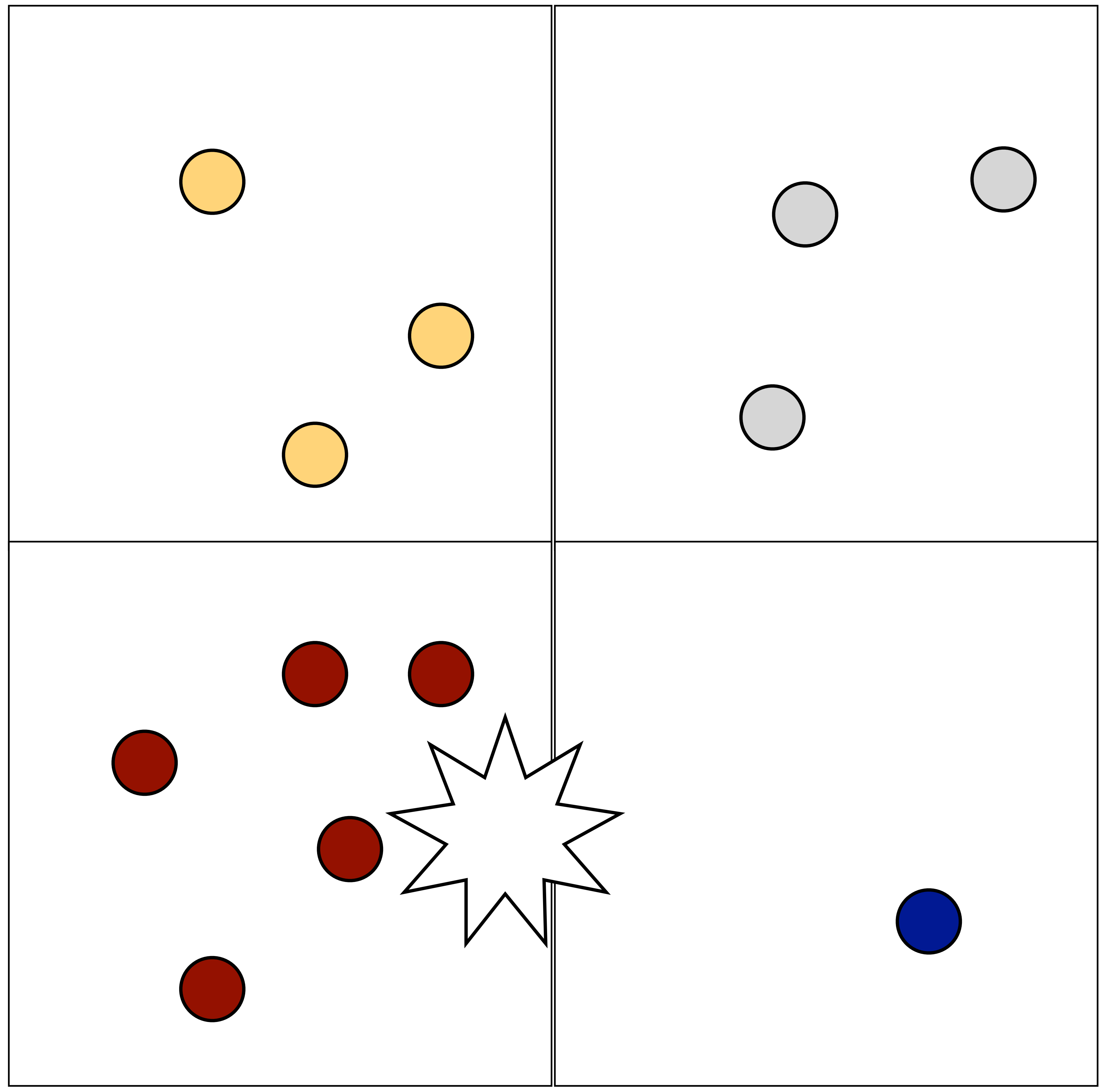
Player may move around the world seamlessly



A client may see events and objects from neighboring regions.

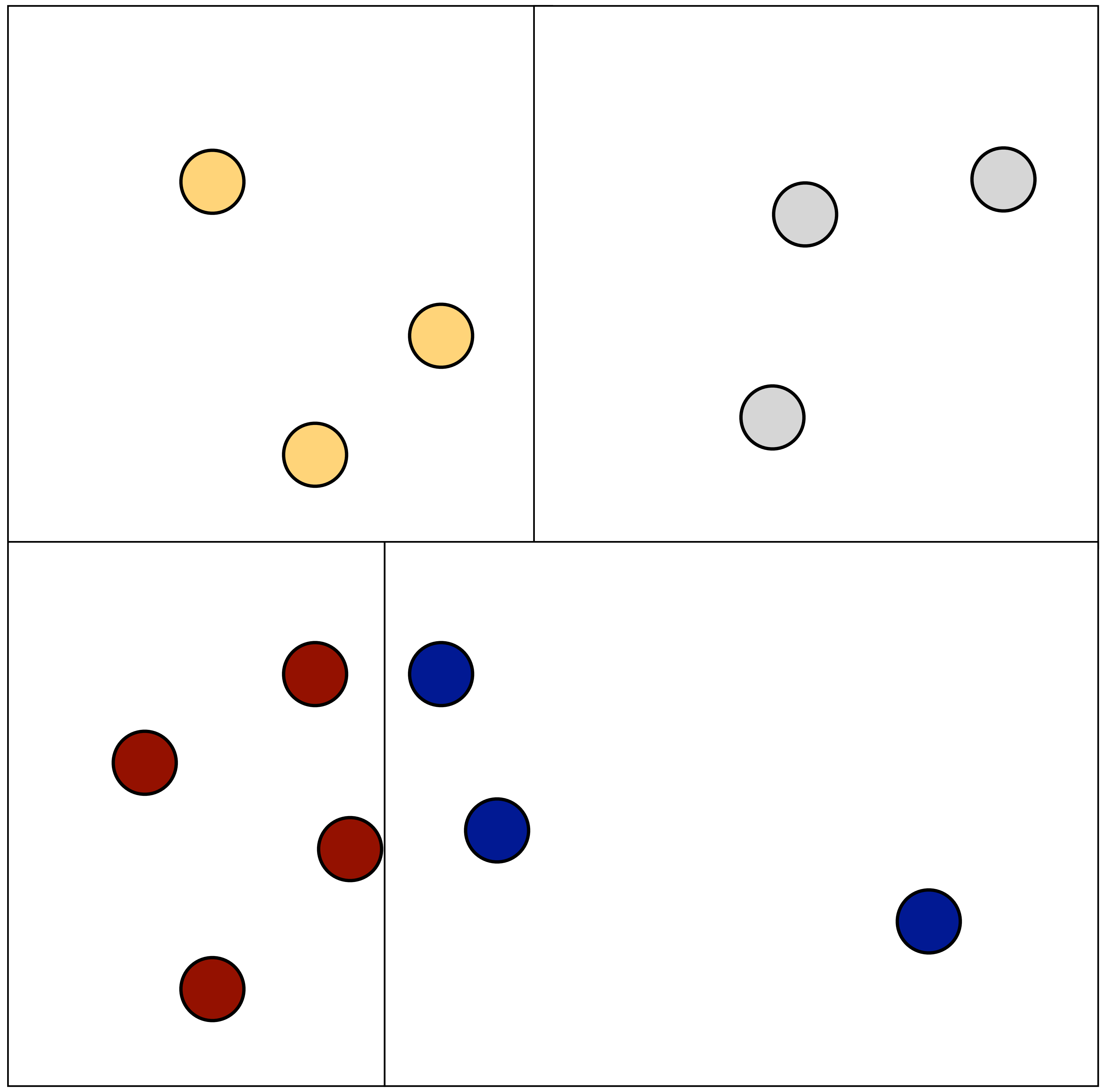


Events occurring in one region may affect objects in another region.

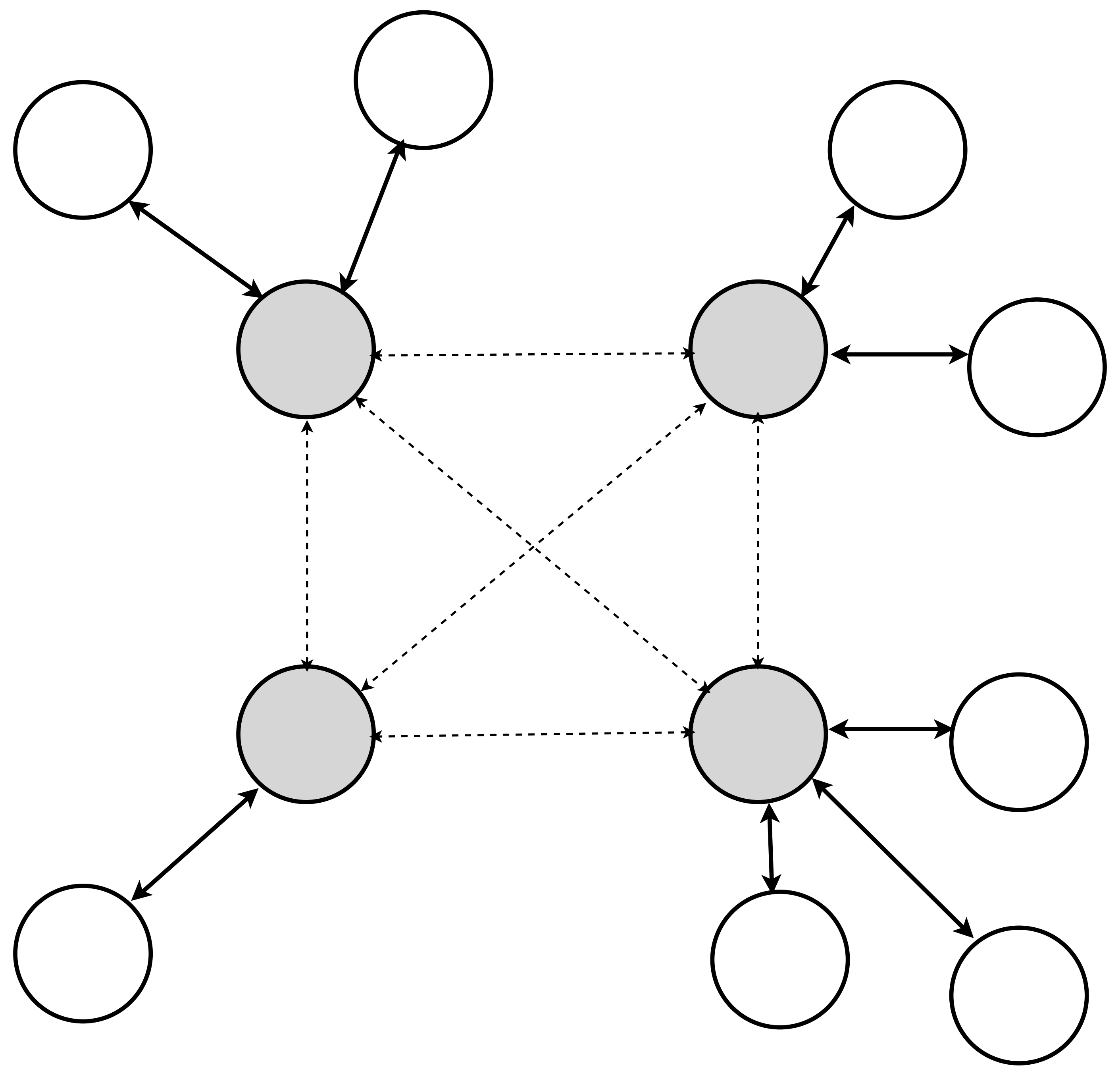


Advantage: Partitioning is transparent to the players. No artificial constraints in game design.

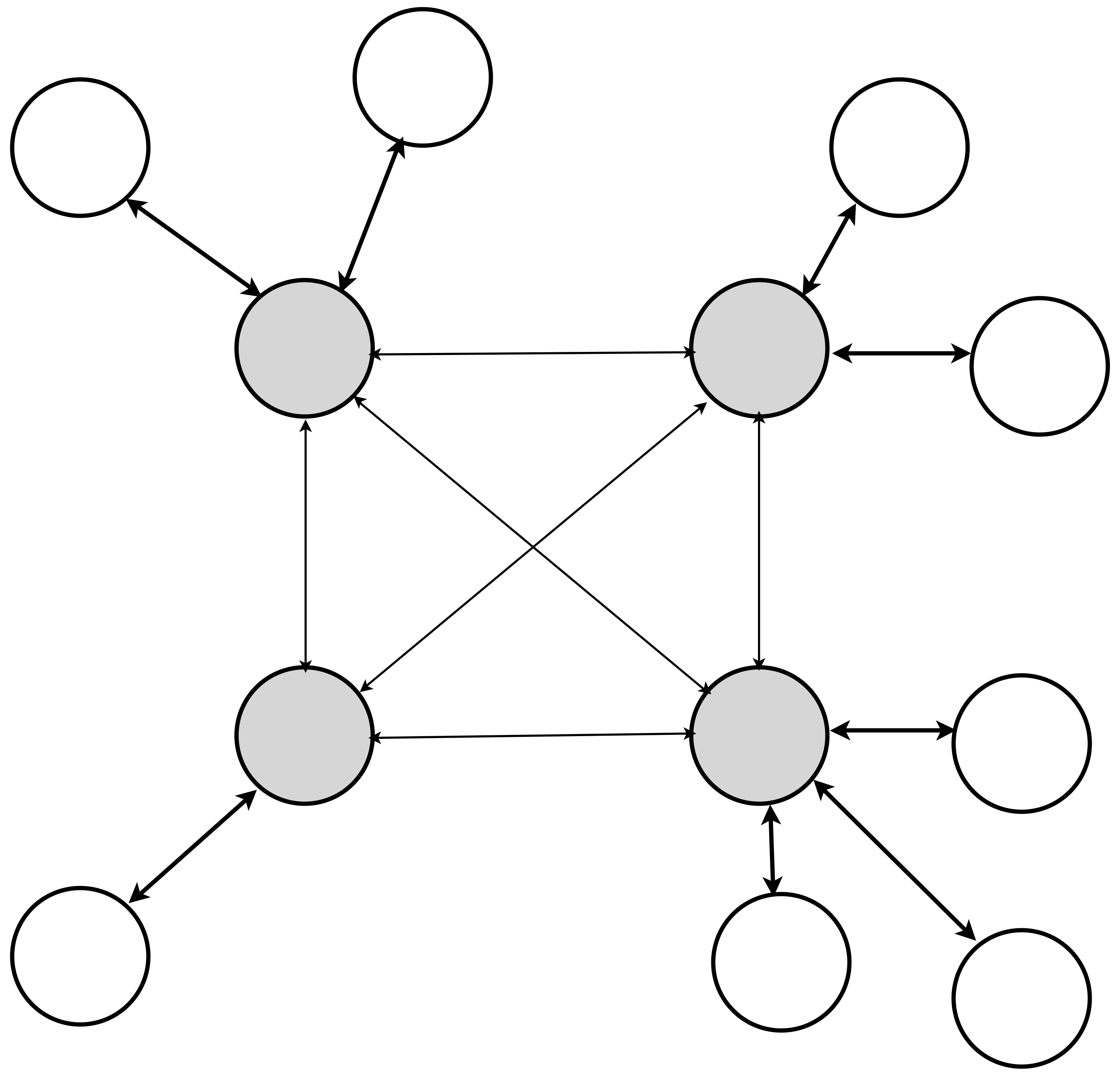
Advantage: We can now resize the regions to load-balance among the servers, improving server utilization.



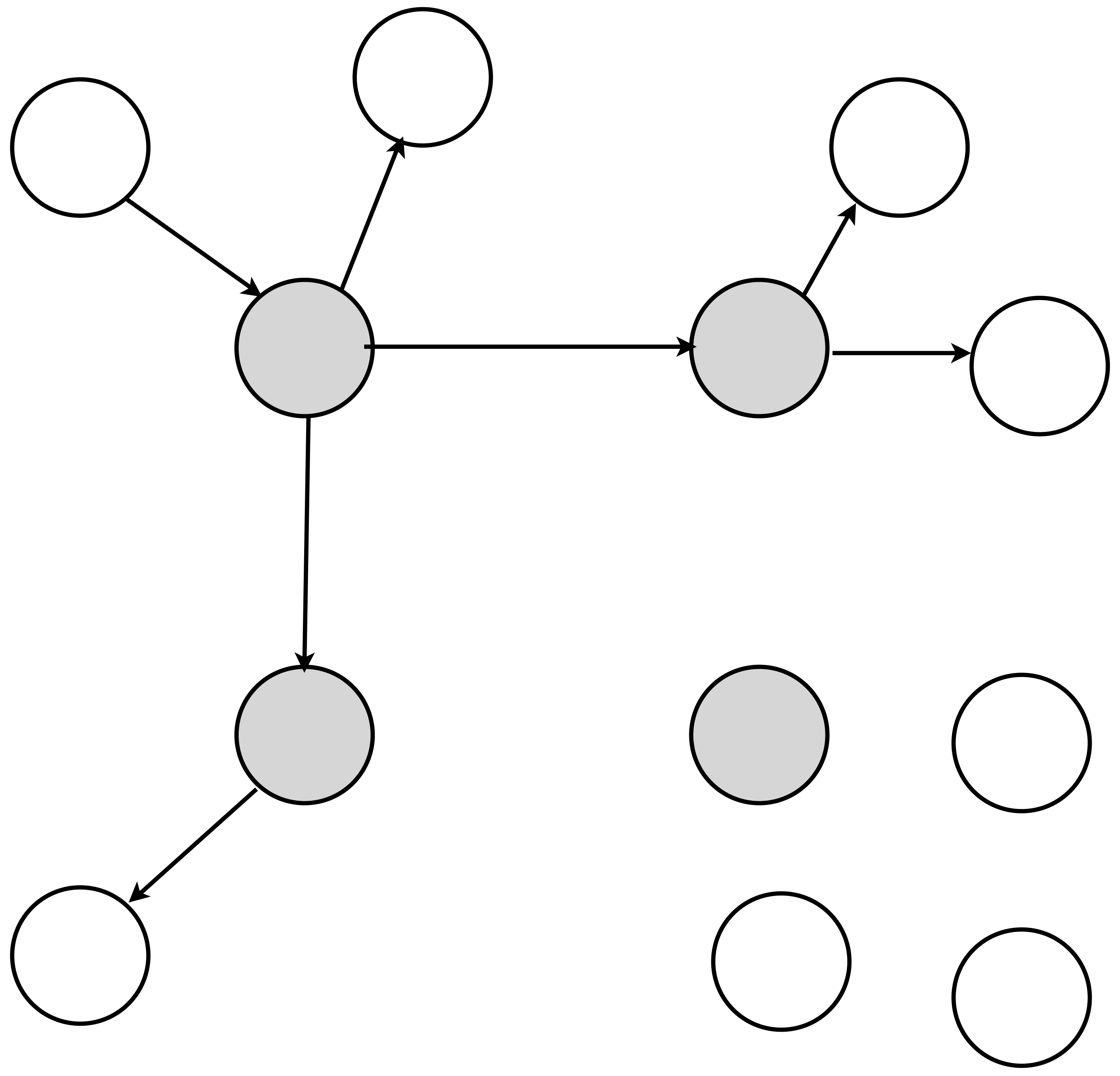
A client connects to the server serving its current region, and is handed-off to another server when it moves to another region.



Server communicates with each other to transfer states, update states etc.

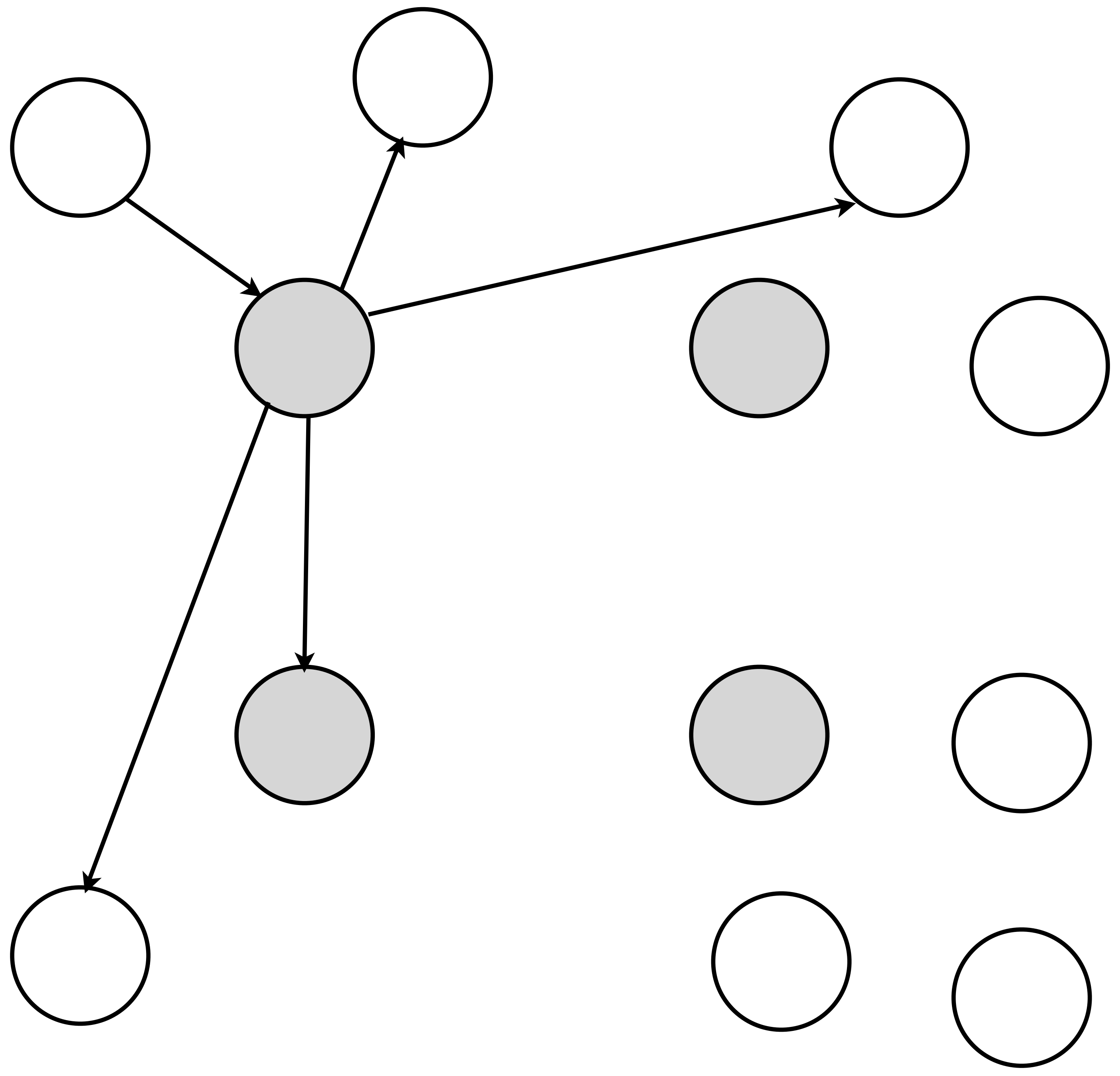


Updates from a client are sent to its server, which then forwards it to all relevant clients in the same region or neighboring regions.



or

A client connects to the servers serving its current region and regions it subscribes to. Updates from neighboring server are sent directly.



Q: How to partition the game world?

Non-trivial: factors include
num of players
communication cost
number of hand offs