WHAT IS CLAIMED IS:

1. A computer-implemented method, comprising:

providing, by a computing device, a plurality of dedicated allocations of a cache, a dedicated allocation of the plurality of dedicated allocations being associated with an entity of a plurality of entities, the cache being provided utilizing random access memory;

providing, by the computing device, a shared allocation of the cache, the shared allocation being available for storage to each of the plurality of entities;

receiving, by the computing device from another computing device associated with the entity, a caching request comprising data to be cached; and

based on receipt of the caching request, when the shared allocation of the cache is full:

evicting, by the computing device, an amount of data from the shared allocation of the cache based on identifying the amount of data as being oldest in the shared allocation of the cache;

in response to evicting the amount of data from the shared allocation, migrating, by the computing device, an oldest instance of data from the dedicated allocation associated with the entity to the shared allocation of the cache; and

in response to migrating the oldest instance of data from the dedicated cache, storing, by the computing device, the data received in the caching request in the dedicated allocation associated with the entity without first storing the data in the shared allocation of the cache.

2. The computer-implemented method of claim 1, further comprising:

determining whether a dedicated allocation of the cache is full, the dedicated allocation being associated with the entity; and

when the dedicated allocation of the cache is not full, storing the data received in the caching request in the dedicated allocation associated with the entity.

3. The computer-implemented method of claim 2, further comprising:

when the dedicated allocation of the cache is full:

determining whether the shared allocation of the cache is full;

when the shared allocation of the cache is not full, migrating an oldest instance of data from the dedicated allocation associated with the entity to the shared allocation of the cache; and

storing the data received in the caching request in the dedicated allocation associated with the entity.

4. The computer-implemented method of claim 1, further comprising:

determining a plurality of hit ratios corresponding to each entity of the pluralities of entities, each hit ratio identifying a percentage of caching requests that result in requested data being found in the cache, wherein evicting the amount of data from the shared allocation of the cache is further based at least in part on the plurality of hit ratios.

5. The computer-implemented method of claim 1, further comprising:

receiving a data request for cached data associated with an entity;

determining whether the cached data is stored in the dedicated allocation associated with the entity; and

when the cached data is stored in the dedicated allocation associated with the entity, returning the cached data in response to the data request.

6. The computer-implemented method of claim 5, further comprising:

when the cached data is not stored in the dedicated allocation associated with the entity:

determining whether the cached data is stored in the shared allocation of the cache;

when the cached data is stored in the shared allocation, returning the cached data;

when the cached data is not stored in the shared allocation, returning an indication that the cached data was not found.

7. The computer-implemented method of claim 6, further comprising:

when the cached data is stored in the shared allocation:

determining whether the dedicated allocation associated with the entity is full; and

migrating the cached data to the dedicated allocation associated with the entity when the dedicated allocation associated with the entity is not full.

8. The computer-implemented method of claim 7, further comprising:

when the dedicated allocation associated with the entity is full:

determining whether the shared allocation of the cache is full;

migrating an oldest instance of data from the dedicated allocation associated with the entity to the shared allocation of the cache when the shared allocation of the cache is not full; and

storing the cached data in the dedicated allocation associated with the entity.

9. The computer-implemented method of claim 8, further comprising:

when the shared allocation of the cache is full:

evicting an amount of data from the shared allocation based on identifying the amount of data as being oldest in the shared allocation of the cache;

migrating an oldest instance of data from the dedicated allocation associated with the entity to the shared allocation of the cache; and

storing the cached data in the dedicated allocation associated with the entity.

10. A computing device, comprising:

a processor;

a cache; and

one or more memories storing executable instructions that, when executed by the processor, cause the computing device to:

provide a plurality of dedicated allocations of the cache, a dedicated allocation of the of the plurality of dedicated allocations corresponding to an entity of a plurality of entities, the cache being provided utilizing random access memory of the computing device;

provide a shared allocation of the cache, the shared allocation being available for storage to each of the plurality of entities;

receive, from another computing device associated with the entity, a caching request comprising data to be cached; and

based on receipt of the caching request, when the shared allocation of the cache is full:

evict an amount of data from the shared allocation of the cache based on identifying the amount of data as being oldest in the shared allocation of the cache;

in response to evicting the amount of data from the shared allocation of the cache, migrate an oldest instance of data from the dedicated allocation corresponding to the entity to the shared allocation of the cache; and

in response to migrating the oldest instance of data from the dedicated cache, store the data received in the caching request in the dedicated allocation corresponding to the entity without first storing the data in the shared allocation of the cache.

11. The computing device of claim 10, wherein executing the instructions further cause the computing device to:

identify a hit ratio threshold specifying an amount of data requests that are to result in data being found in the cache;

monitor corresponding hit ratios of data requests submitted by the plurality of entities; and

provide a notification to an entity when a hit ratio associated with the entity falls below the hit ratio threshold.

12. The computing device of claim 10, wherein the amount of data of the shared allocation of the cache is evicted further based at least in part on identifying first data corresponding to an entity with a highest hit ratio of a plurality of hit ratios corresponding to the plurality of entities.

13. The computing device of claim 10, wherein executing the instructions further cause the computing device to:

store the data in a dedicated allocation associated with the entity when the dedicated allocation corresponding to the entity has available space; and

store the data in a shared allocation of the cache when the dedicated allocation corresponding to the entity is full.

14. The computing device of claim 10, wherein executing the instructions further cause the computing device to:

receive, from another computing device associated with an entity of the plurality of entities, a data request identifying data to be retrieved;

search for the data in the dedicated allocation corresponding to the entity; and

provide the data in a data response when the data is found in the dedicated allocation corresponding to the entity.

15. The computing device of claim 14, wherein executing the instructions further cause the computing device to:

search for the data in the shared allocation when the data is not found in the dedicated allocation corresponding to the entity;

provide the data in a data response when the data is found in the shared allocation; and

provide a notification that the data is not found when the data is not found in the shared allocation.

16. The computing device of claim 10, wherein utilizing the plurality of dedicated allocations and the shared allocation decreases an overall latency associated with of data requests submitted by the plurality of entities.

17. The computing device of claim 10, wherein the plurality of dedicated allocations are equal in size.

18. The computing device of claim 10, wherein executing the instructions further cause the computing device to maintain a mapping identifying each of the plurality of entities to a particular corresponding dedicated allocation of the cache.