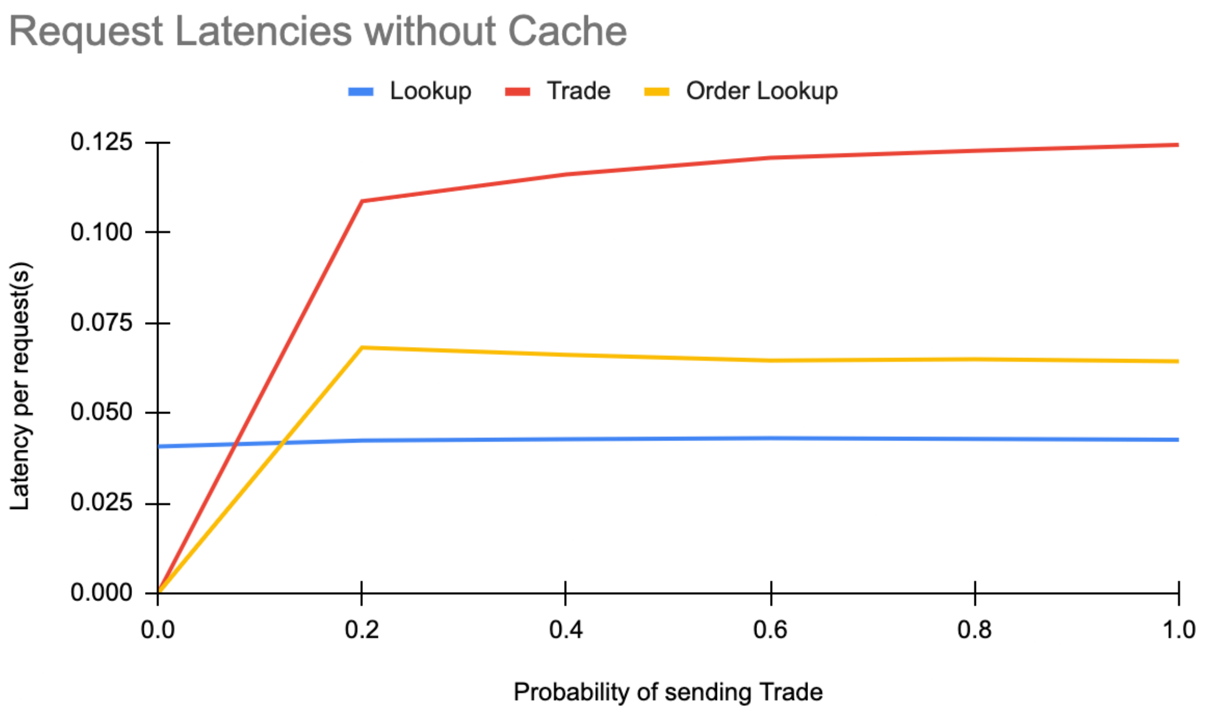
Evaluation document

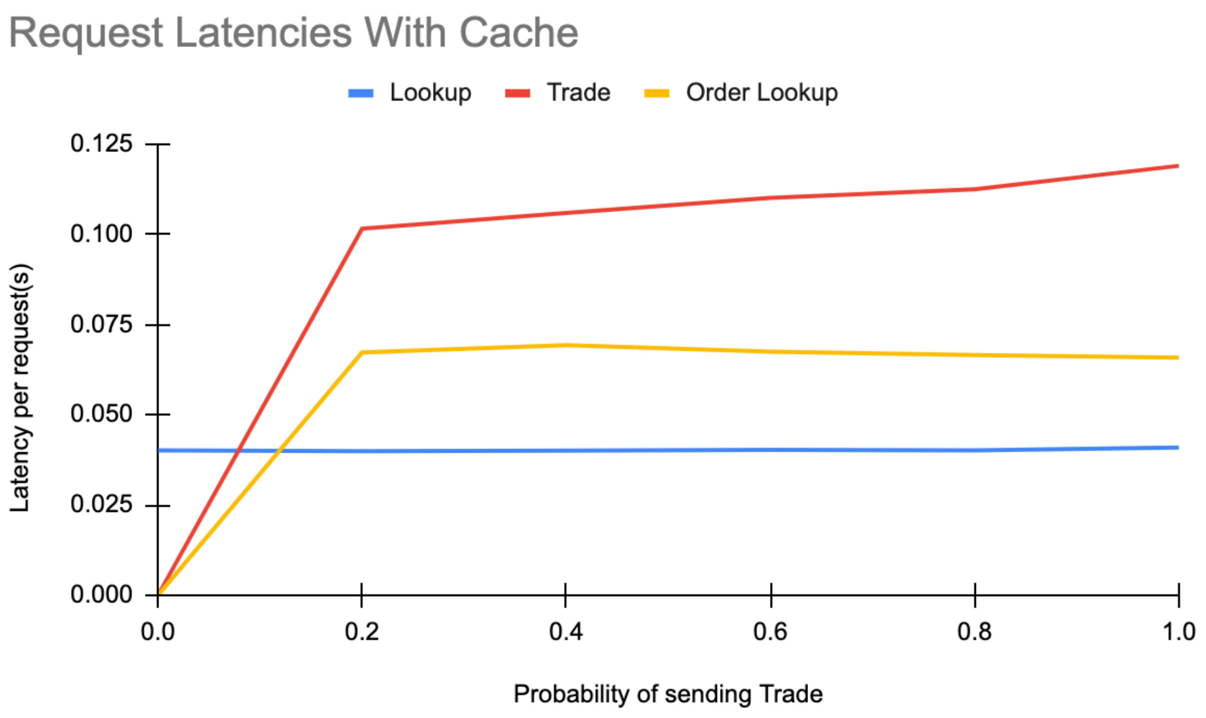
Latencies for different types of requests vs Trade request probability, with and without cache:

<u>On AWS:</u>

	Without Cache		
Probability of sending Trade	**Lookup**	**Trade**	**Order Lookup**
0	0.0407028	0	0
0.2	0.0423588	0.1086648	0.0680956
0.4	0.0427312	0.1161238	0.0661278
0.6	0.042993	0.1207018	0.06451
0.8	0.0427914	0.1226352	0.0648932
1	0.0425602	0.1243004	0.0642896

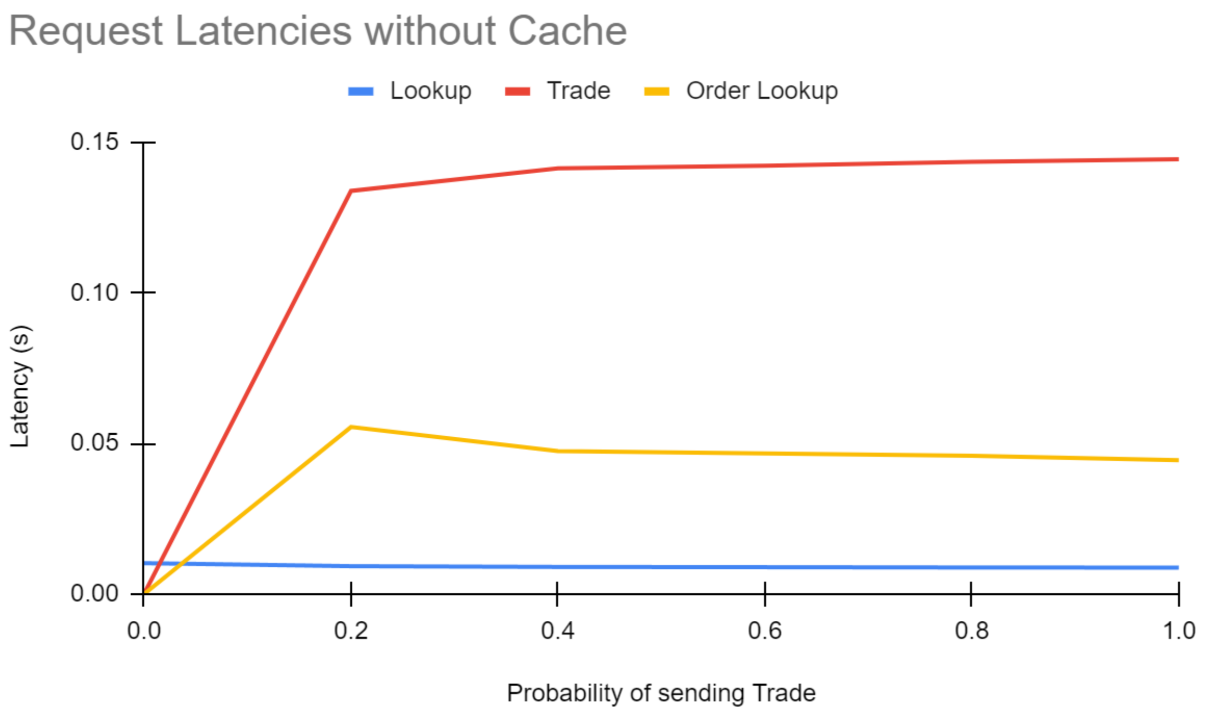


With Cache			
Probability of sending Trade	**Lookup**	**Trade**	**Order Lookup**
0	0.0400958	0	0
0.2	0.0398722	0.1015398	0.0672562
0.4	0.0400326	0.1058938	0.0692586
0.6	0.0402664	0.1100602	0.0674358
0.8	0.040108	0.112457	0.0665594
1	0.0409348	0.1189362	0.0657868

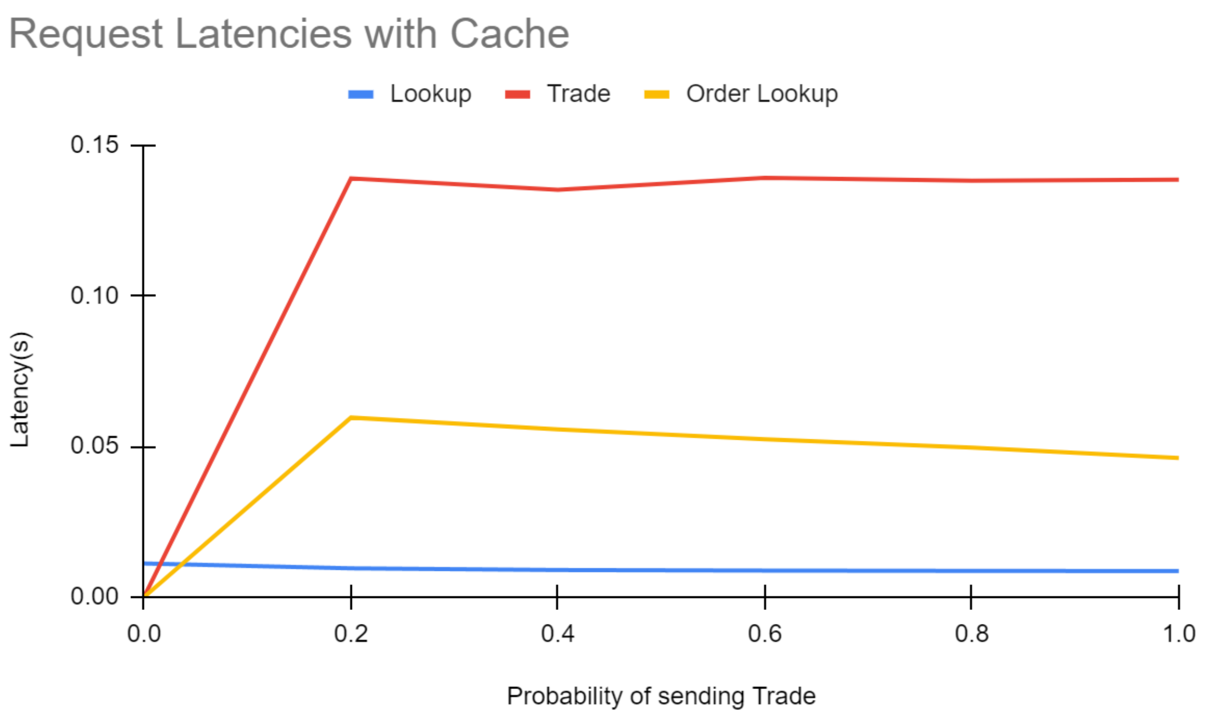


On Local Machine:

Without Cache			
Probability of sending Trade	**Lookup**	**Trade**	**Order Lookup**
0	0.0102562	0	0
0.2	0.0092598	0.13380086	0.05544714
0.4	0.009002	0.14130488	0.04749318
0.6	0.0089242	0.14215442	0.04664216
0.8	0.0088192	0.14342126	0.04591876
1	0.0087714	0.14428708	0.04439406



With Cache			
Probability of sending Trade	**Lookup**	**Trade**	**Order Lookup**
0	0.0111446	0	0
0.2	0.0095366	0.13897366	0.05957788
0.4	0.0089752	0.13518406	0.0556586
0.6	0.0087908	0.13915182	0.05236572
0.8	0.0086904	0.13823212	0.04962942
1	0.0086336	0.13850084	0.04613366



We observe that the lookup latencies are very small compared to trade and order lookup. This is the reason for the graph showing the lookup line almost close to 0. This is expected because in lookup we are calling catalog service directly but in trade requests we need to call order service which inturn calls catalog service, because of all these intermediate calls, trade latency is higher than lookup.

On AWS, we have observed a slight improvement in the performance of lookup requests when the cache is turned on. The maximum improvement was seen for probability 0.2, which was around 6%. This improvement can be mainly attributed to a reduction in the number of invalidate requests, which results in items being preserved in the cache for a longer period of time. We observed a similar trend on local machine

Can the clients notice the failures? (either during order requests or the final order checking phase) or are they transparent to the clients?
The client did not notice any failures as long as at least one order service is running. All the failures are not transparent to the client. We observed cases where a normal replica service is killed - since the leader is still running, the requests are handled fine. And when a leader service is killed, we observe that a new leader is elected and the requests start going to that service, serving client requests without failure.

Do all the order service replicas end up with the same database file?
Each replica has its own database file and we observe that at each point in time, all the databases are in sync (dbs of alive services). Even at the end, we observed that the databases of all replicas are the same.