The best cache result of astar.trace is C = 14 B = 6 S = 8 . First of all, since the value of C is 14 and sum of other two value is 14, it shows that fully associative set performs more efficient with maximum cache size. So, it is able to host any memory block, which means that maximum amount of data (B=6) can be stored without any separation. As a result, its AAT is 7.516603ns.

The best cache result of bzip2.trace is C = 14 B = 6 S = 8. This case also shows that fully associative is more efficient for cache. The reason is same as above. As a result, its AAT is 2.155919 ns.

The best cache result of mcf.trace is C = 14 B = 6 S = 8. This case also shows that fully associative is more efficient for cache. The reason is same as above As a result, its AAT is 2.155919 ns.

The best cache result of perlbench.trace is C = 14 B = 6 S = 8. This case also shows that fully associative is more efficient for cache. The reason is same as above . As a result, its AAT is 2.155919 ns.

By the cache results of the traces, in this simulation, fully associative set shows that it performs more efficiently that other sets. However, in real situation, the fully associative set is not always guaranteed to have more efficient performance. The reason is that fully associative set requires searching all single block to find a correct block. Therefore, it can cause time inefficiency in actual situation.

Another thing about the result, each result shows different AAT. This is caused by misses(especially about writeback). Writeback requires updating modified data. Therefore, it needs time to go back to update data. Therefore, according to the result, there are most writeback, occurred in astar.trace test.