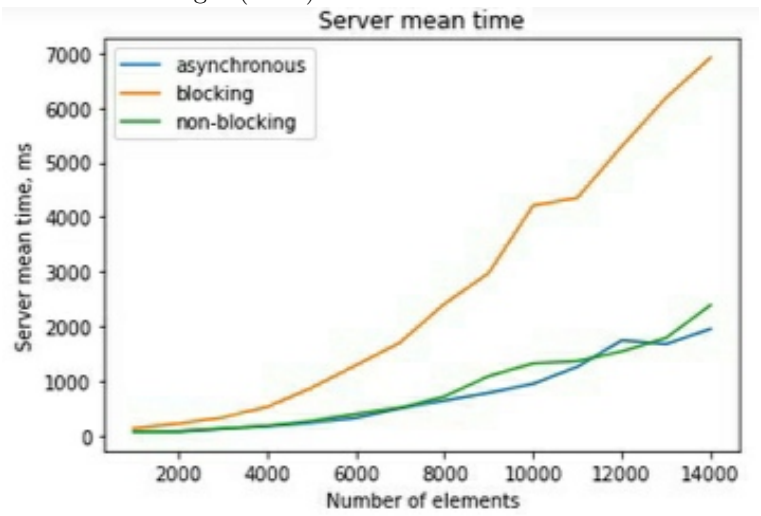
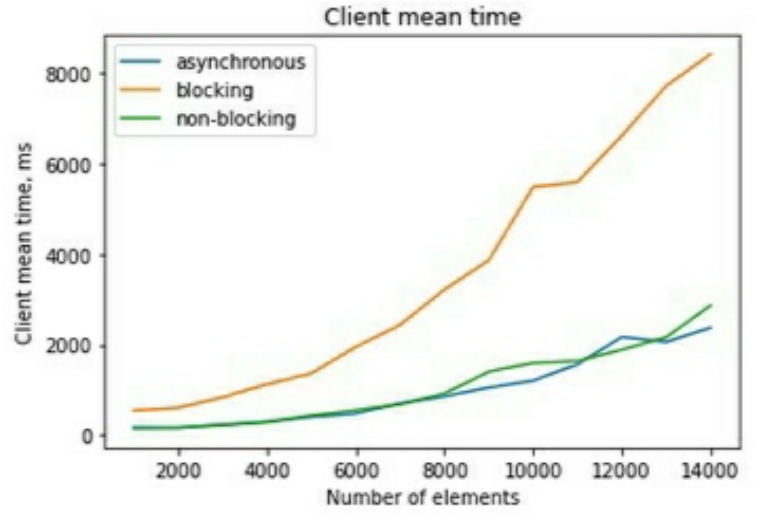
1 Parameter: number of elements(N)

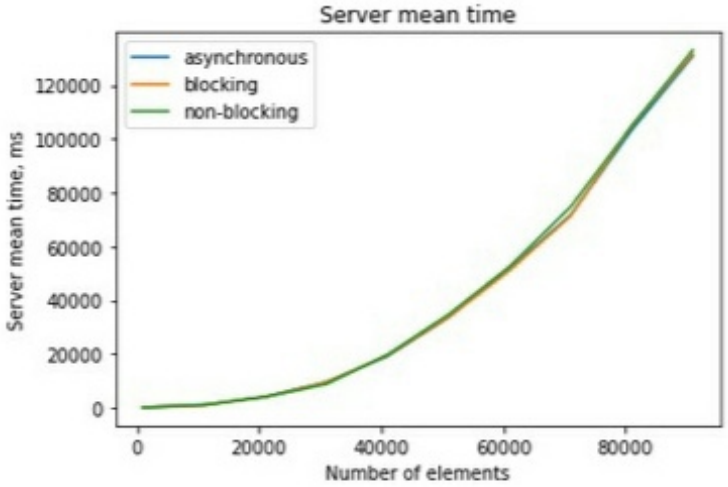
number of messages(X)=1,
number of clients=10,
number of working threads(M)=5,
time between messages (delta)=200ms

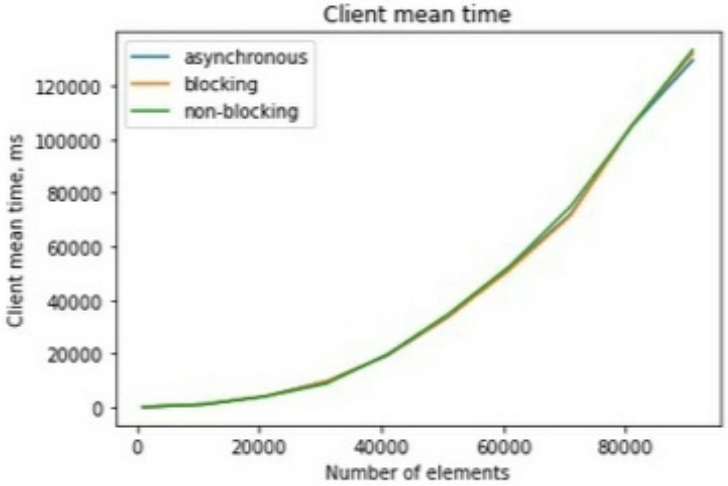




Как правило, если запросы от клиента большие, блокирующая архитектура будет показывать хороший результат (так как считает все сразу и отправит на обработку). Тем не менее, она уступила асинхронной и неблокирующий - частое переключение контекста все-таки играет большую роль (в случае блокирующей на клиентов выделялось 10 потоков, что вместе с чис-

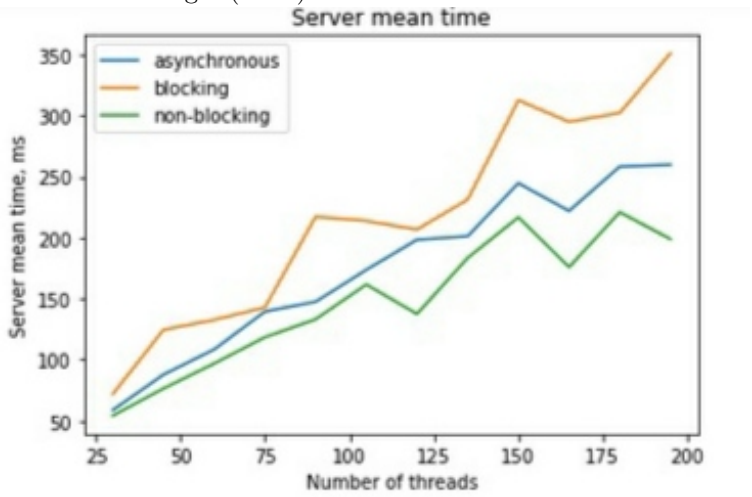
лом потоков в тредпуле сильно превосходит число ядер). А в асинхронной и блокирующей все гораздо лучше с суммарным числом потоков (тоже больше, чем ядер, но не такая большая разница, как в случае с блокирующей), и отработали примерно одинаково. В принципе любой архитектуре хуже, когда много клиентов, чем когда их мало. Но при таких значениях получилось, что жирный минус блокирующей (частое переключение контекста) перевесил минус асинхронной (большое число call-back-ов) и неблокирующей (чтение/запись кусками). При этом если поменять параметры так, чтобы снизить суммарное число потоков (число клиентов теперь совпадает с числом потоков и равно 5), то все архитектуры работают приблизительно одинаково.

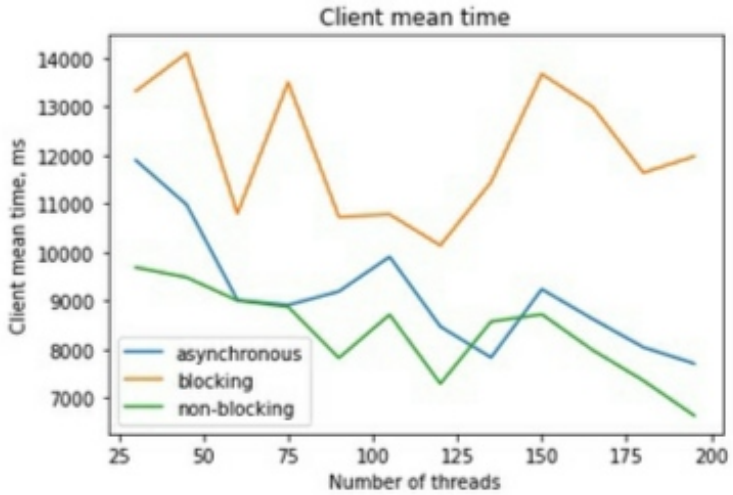




2 Parameter: number of threads(M)

number of messages(X)=50,
number of clients=200,
number of elements (N)=1000,
time between messages (delta)=100ms

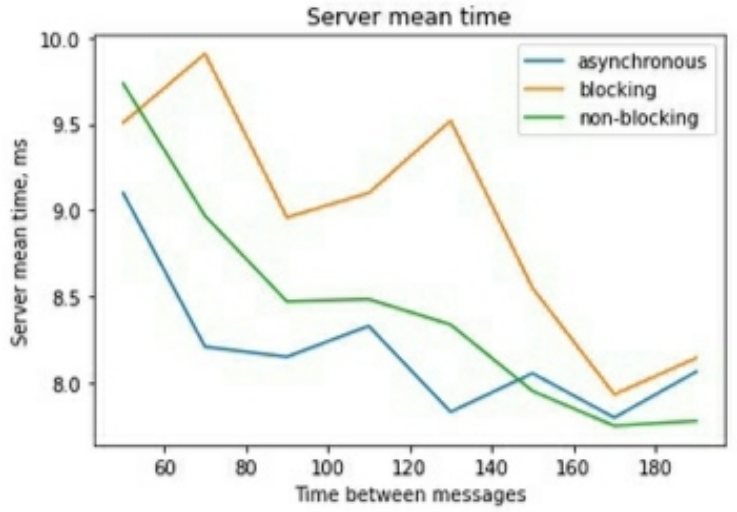


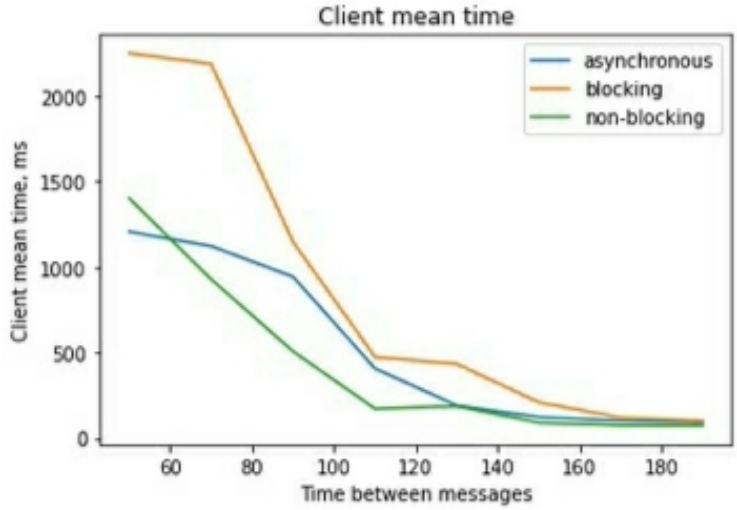


Блокирующей хуже всего, так как клиентов и сообщений много, частое переключение контекста. Асинхронная отработала немного хуже блокирующей (здесь, по сути, время должно было получиться +- одинаковое, так как число потоков одно и то же. В случае клиента, возможно, такое распределение объясняется тем, что call-back-ов много, все в одном потоке, селектор справляется лучше).

3 Parameter: time between messages(delta)

number of messages(X)=30,
number of clients=60,
number of working threads (M)=5,
number of elements (N)=800





Для всех архитектур хорошо, что клиенты редко шлют запросы, разница во времени работы при больших временных промежутках уже не столь значительна. Блокирующая все еще уступает, вероятно, это связано с тем, что размер массива небольшой и чтение в случае неблокирующей и асинхронной происходит быстро (за один раз, а тогда выигрыша у блокирующей нет). Но при этом сообщений много, чтение постоянное, разница лишь в

числе работающих потоков, а тут блокирующая уже проигрывает.