CS336 : Parallel & Distributed processing

Project 6 Report

Summary of tasks

For this project the main we visit the computer science classic problem of a bounded buffer in a producer-consumer production line. Our simulation modelled a production line that creates frozen fish dinners. They are passed from worker to worker in a production line. Each worker does something to the fish so that it can be packaged into a somewhat palatable frozen entree. The workers are lined up sequentially and a fish first goes to worker 1 who then passes it onto the next worker . Between each pair of workers is a holding area for the fish. Actual fish-prep work is simulated by calling sleep for a random number of seconds in a range.

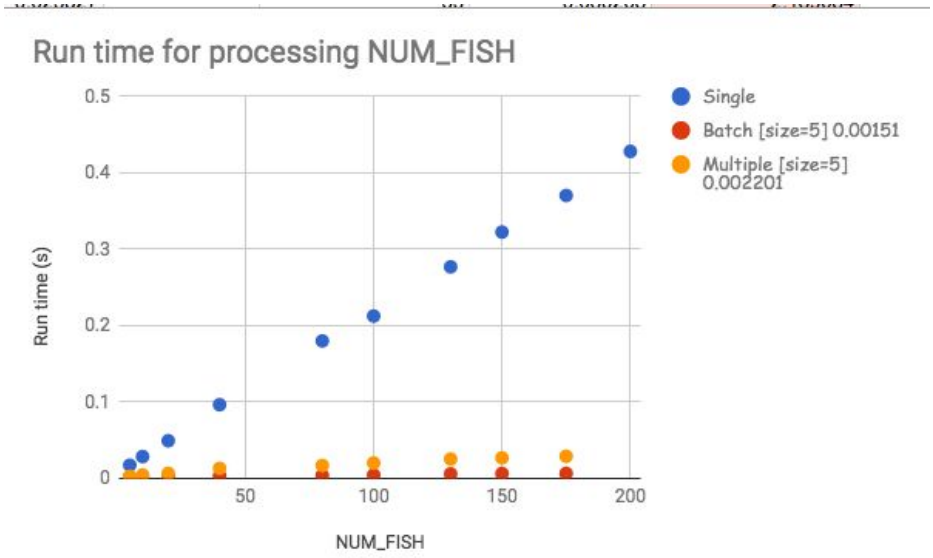
The goal of this project is to write multiple versions of the program and compare the speed of execution, and then interpret the results.

Tasks

1. I created a linear production line whereby each consumer waits for a fish to be on the buffer to consumer, and the producer waits for the holding area to be non-empty to add fish.
2. Second version of the program whereby production is in batches, passing multiple fish at a time onto the buffer/holding area. The buffers between the workers can hold one to BATCH_SIZE fish.
3. Third version of the program, in which the buffers can hold multiple fish, but the workers put in one fish and remove one fish at a time.
4. Analysing the performance of my multiple programs, I firstly compared how well they handle volume for 5 workers in the factory. I add color scaling onto my tables as well to highlight the magnitude perspectively. The results are shown below:

		[NUM_WORKERS=5]	
NUM_FISH	Single	Batch [size=5]	Multiple [size=5]
5	0.017332	0.00151	0.002201
10	0.028592	0.00181	0.002669
20	0.049297	0.002023	0.004482
40	0.096433	0.002552	0.006599
80	0.179913	0.003505	0.013175
100	0.212527	0.003854	0.016928
130	0.276916	0.004556	0.02016
150	0.322452	0.005882	0.025327
175	0.370352	0.006369	0.026792
200	0.428078	0.006481	0.028929

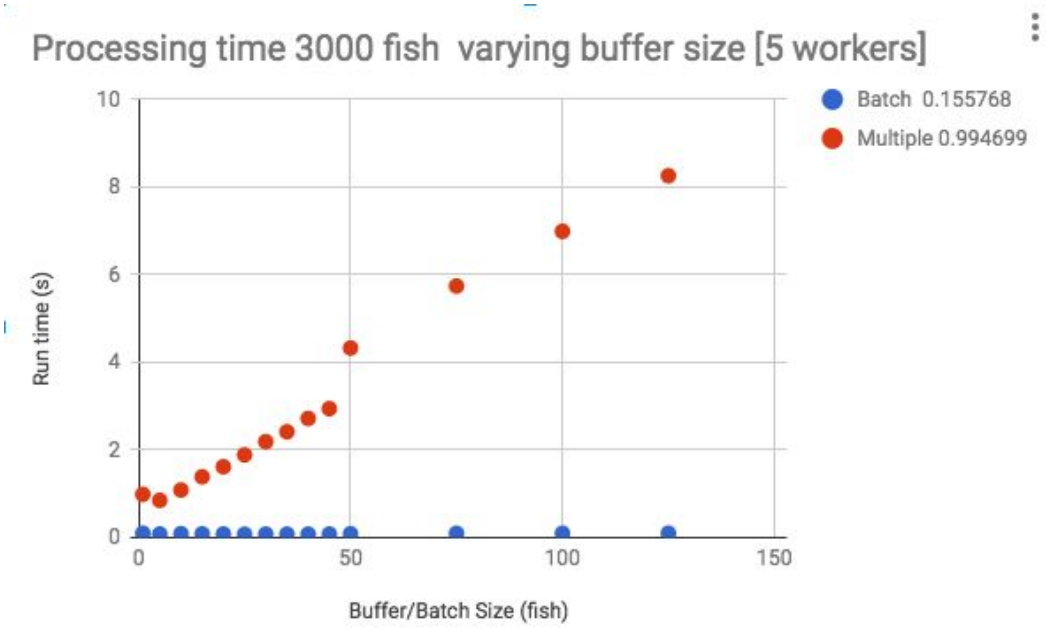
The graph below is the result of this processing:



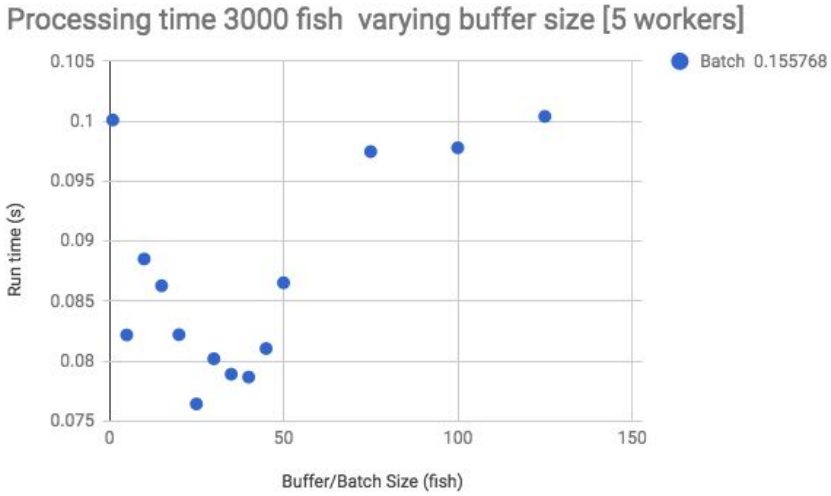
For the first option, clearly that option is an expensive one, as for a larger volumes it's time cost of operation increases linearly, whilst for the latter options the performance ranges on the same order of magnitude.
Secondly, when producing a large volume of 3000 fish, I then compare how the performance varies with the buffer size. This process rules out process 1 as we cannot vary the buffer size for it. The results are below:

Processing 3000 Fish	Total run time (s)	[NUM_WORKERS=5]
Buffer & Batch Size	Batch	Multiple
1	0.155768	0.994699
5	0.100109	0.98622
10	0.08219	0.849092
15	0.088523	1.085181
20	0.086295	1.387746
25	0.082215	1.615934
30	0.076439	1.888289
35	0.080208	2.189084
40	0.07892	2.414385
45	0.078679	2.719239
50	0.081059	2.94289
75	0.086539	4.325907
100	0.097473	5.741024
125	0.097791	6.988759
150	0.100407	8.257468

The graph:

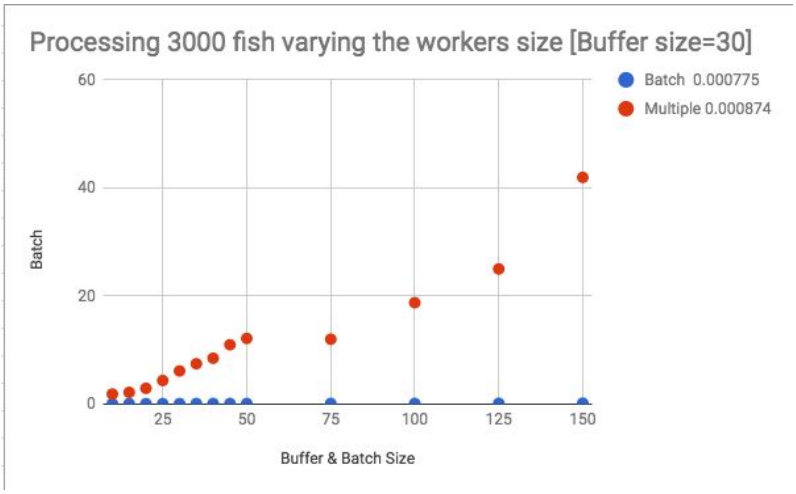


I was expecting that for a buffer holding multiple fish with workers adding as soon as they finish processing would be faster, however it seems updating the buffer/holding is an expensive time step, as increasing the buffer size causes the program to be more time costly. The batch processing is not constant however, as varying the buffer size does not affect the order of magnitude of the values significantly compared to the multiple. An analysis of that range is attached below:



Lastly varying the number of workers in the factory for an optimal buffer size of 30 from the above graph, the results for that production are shown below:

Processing 3000 Fish	Total run time (s)	[BUFFER_SIZE=30]
Buffer & Batch Size	Batch	Multiple
1	0.000775	0.000874
5	0.089804	1.870271
10	0.08495	2.171204
15	0.089616	2.923964
20	0.084799	4.363035
25	0.089688	6.139397
30	0.10466	7.46615
35	0.114276	8.493844
40	0.111819	10.980963
45	0.113116	12.154408
50	0.129889	12.008495
75	0.123454	18.767952
100	0.151629	25.007524
125	0.179864	41.957654
150	0.181846	51.24201
175	0.207801	59.138388
200	0.215128	57.894973



Again here, for the batch production, since updating the buffer is not as frequent as in the multiple workers, here as well that step costs the production time significantly.

Conclusions

From the data above, updating the buffer proved to be expensive and that led to significant production efficiency differences. Batch production by each worker in the factory proved to be the most effective amongst the examined methods.

Extensions

None, However, I implemented a usage parameter which when uncommented tells you the usage for high volume productions. I did not analyse this, but it's pretty cool, please appreciate the effort there.

Collaborators

I worked alone on this project. Thanks to Zhuofan for helping me debug.