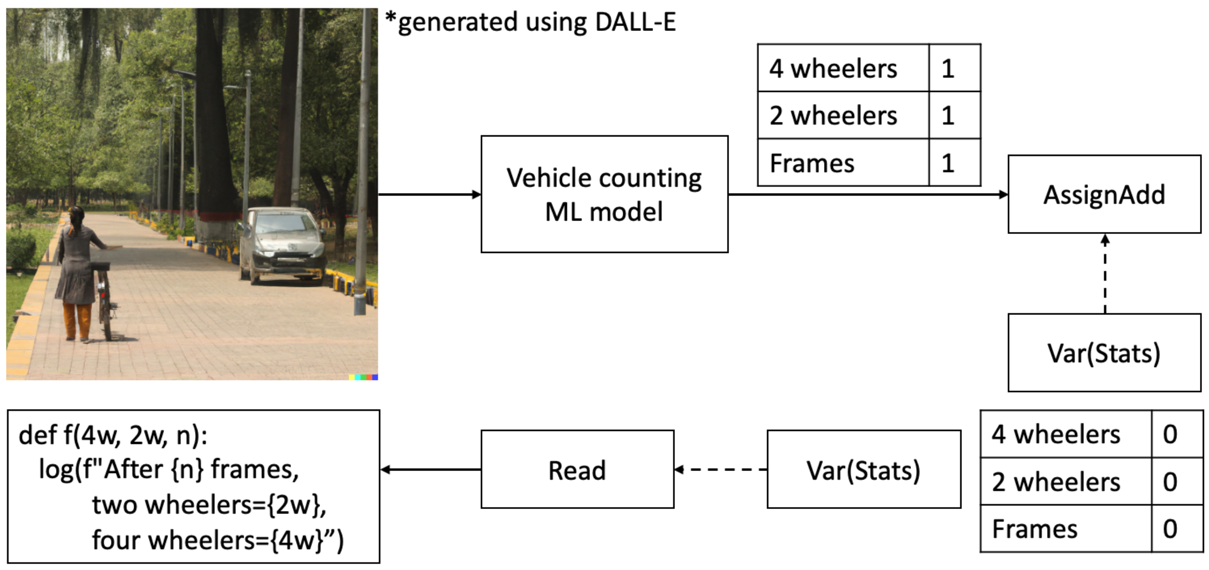
Name: ______________________ Entry number: _______________________

Q1 [15 marks] **TensorFlow**

Let us say that you want to count the number of four wheelers and two wheelers from two cameras in IIT Delhi during Rendezvous 2024. The Rendezvous organizers have heard a lot about TensorFlow and insist that you use it for this project.

Since your team knows about the operational semantics of TensorFlow, you came up with the following program. You plan to directly use the TensorFlow graph API.



4 wheelers	1
2 wheelers	1
Frames	1

4 wheelers	0
2 wheelers	0
Frames	0

```
def f(4w, 2w, n):
  log(f"After {n} frames,
        two wheelers={2w},
        four wheelers={4w}")
```

The graph above feeds the camera frames as tensors to a vehicle counting ML model. When a frame is input, the model generates a 3x1 tensor representing the number of four wheelers, two wheelers, and 1 respectively. For the example image above, the model outputs (1, 1, 1). This tensor is added to the Stats variable which is initialized to (0,0,0).

Stats variable is also read and sent to a function `f`. This function does not produce any output tensors but just logs its input tensor to a fault tolerant storage. For instance, it might print "`After 1 frames, two wheelers=1, four wheelers=1`".

For the questions below, please assume that this graph is connected to two traffic cameras that feed it input frames.

Q1.1 [3 mark] Let us say that this graph is running on a single machine with a CPU and a GPU. How might you place this graph? Use the CPU and the GPU.

Q1.2 [3 marks] (True/False) Can the graph print the following? **Assume that the machine never crashes for Q1.2.**

(a) *[1 marks]*

```
After 1 frames, two wheelers=1, four wheelers=1
After 1 frames, two wheelers=1, four wheelers=1
After 2 frames, two wheelers=2, four wheelers=3
```

____

(b) *[1 marks]*

```
After 3 frames, two wheelers=3, four wheelers=3
After 1 frames, two wheelers=1, four wheelers=1
After 2 frames, two wheelers=2, four wheelers=3
```

____

(c) *[1 marks]*

```
After 1 frames, two wheelers=1, four wheelers=1
After 3 frames, two wheelers=3, four wheelers=3
```

____

Q1.3 [3 marks] Let us say now we want to get separate stats from the two traffic cameras. For this, let us say that the vehicle counting ML model produces a 4x1 tensor for each input image, i.e, it also produces a camera ID 1 or 2 to identify between the two cameras. Please modify the graph to calculate and print `Stats1` for the traffic camera with ID 1, `Stats2` for the traffic camera with ID 2, and `Stats` for the total of both the traffic cameras.

Name: ______________________ Entry number: _______________________

Q1.4 [3 marks] Let us say that this graph is running on two machines each with a CPU and a GPU. How might you place this graph? Use both the CPUs and GPUs.
Hint: Remember that parts of the graph can be duplicated to do “concurrent execution”.

Q1.5 [3 marks] Upon running your graph above, we observe the following log statements for the log file of traffic camera 1 stats:

```
...
After 193 frames, two wheelers=311, four wheelers=772
After 194 frames, two wheelers=312, four wheelers=775
After 156 frames, two wheelers=276, four wheelers=657
After 157 frames, two wheelers=279, four wheelers=659
...
```

Explain how one might see this log file.

Name: ________________________   Entry number: _________________________

Q2. [20 marks] CIEL and scalability
We looked at the following program to compute the longest common subsequence.

```
import ray
ray.init()

@ray.remote
def lcs(X, Y, bleft, bup, bsize, Lleft, Lup, Ldiag):
  # Declaring the array for storing the dp values
  L = [[None] * bsize for i in range(bsize)]

  def l(i, j):
    if i >= 0 and j >= 0:
      return L[i][j]
    if i < 0 and j < 0:
      return Ldiag[bsize+i][bsize+j] if Ldiag is not None else 0
    if i < 0:
      return Lleft[bsize+i][j] if Lleft is not None else 0
    return Lup[i][bsize+j] if Lup is not None else 0

  # Note: L[i][j] contains length of LCS of X[0..i-1] and Y[0..j-1]
  for i in range(bsize):
    for j in range(bsize):
      left = bleft*bsize + i
      up = bup*bsize + j
      if left == 0 or up == 0:
        L[i][j] = 0
      if X[left-1] == Y[up-1]:
        L[i][j] = l(i-1, j-1) + 1
      else:
        L[i][j] = max(l(i-1, j), l(i, j-1))
  return L

# Driver code
if __name__ == '__main__':
  S1 = # input string 1 with length=30,000
  S2 = # input string 2 with length=30,000

  m, n, bsize = 30000, 30000, 10000
  X, Y = ray.put(S1), ray.put(S2)
  f = [[None] * (n//bsize + 1) for i in range(m//bsize + 1)]

  for bleft in range(0, m//bsize):
    for bup in range(0, n//bsize):
      fleft = f[bleft-1][bup] if bleft > 0 else None
      fup = f[bleft][bup-1] if bup > 0 else None
      fdiag = f[bleft-1][bup-1] if bleft > 0 and bup > 0 else None
      f[bleft][bup] = lcs.remote(X, Y, bleft, bup, bsize,
                                 fleft, fup, fdiag)

  L = ray.get(f[m//bsize-1][n//bsize-1])
  print(f"Length of LCS is {L[bsize-1][bsize-1]}")
```

Name: ______________________ Entry number: _______________________

Q2.1 [2 marks] Draw the task DAG for the above program.
Hint: Notice that the program computes 9 2D array objects each of size 10,000 x 10,000. You can refer to these 9 array objects as $L_{0,0}$, $L_{0,1}$, … $L_{2,2}$ respectively.

We will make simplifying assumptions and do "back of the envelope calculations" in the following problems. Let us say that our setup contains 2 workers and a master. Workers are connected with one another over a 2GBps link. It takes 1ns to compute one entry in the 2D array object. We assume that the workers have RDMA-enabled NICs where RDMA stands for "remote direct memory access", i.e, the workers' network interface cards (NIC) can directly download objects from another worker's memory without interrupting the CPU of either worker. We ignore the effects of memory hierarchy, locality, and caching. We ignore the round trip latency to/from master.

Therefore, computing a 2D array object of size 10,000 x 10,000 takes 10,000 * 10,000 * 1ns = 100ms. We assume that each entry in the array is an integer of 4 bytes. So downloading one 2D array object from one worker to another on the 2GBps link takes 10,000*10,000*4/(2*1000,000,000) s = 200ms.

The following shows an execution timeline. Blocks on the CPU timeline show the time taken to calculate a 2D array. Blocks on the NIC timeline show the time to download the object from the other worker.

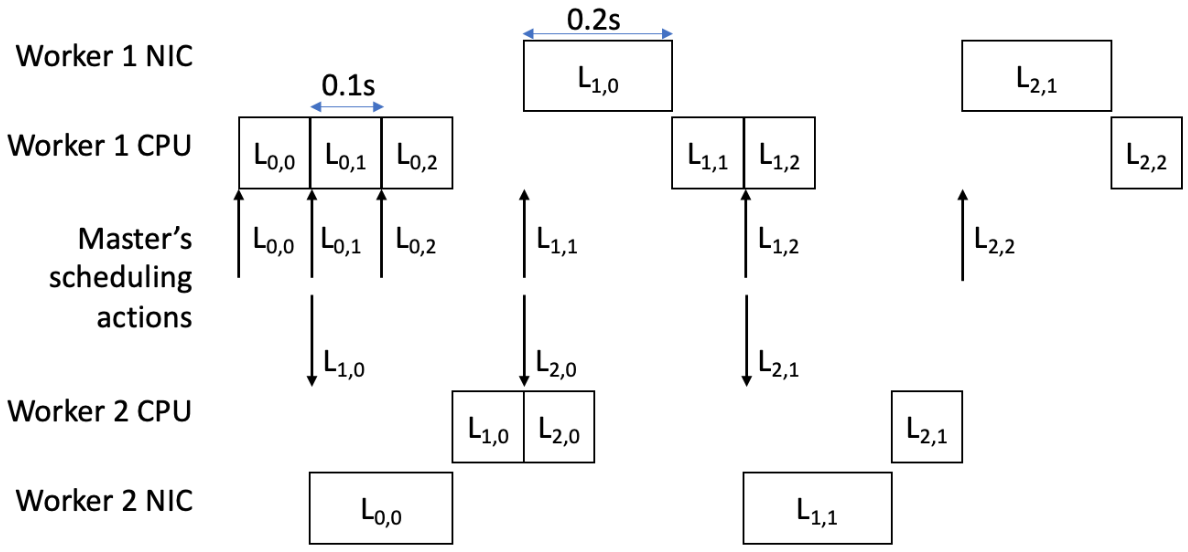


Figure 2A: Execution timeline with a 1ns compute time and a 2GBps link.

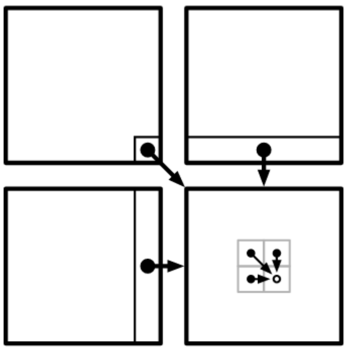
Name: ______________________ Entry number: _______________________

Q2.2 [3 marks] For the execution timeline in Figure 2A, what is the speedup and efficiency? Assume p=2 for two workers, i.e, do not count master in your calculation.

Q2.3 [3 marks] Let us say that we upgrade the CPU of workers such that they can now support vectorized instructions. Effectively, each array entry can now be computed 4x faster: in 0.025ns. We further make the network link between workers 10x faster: i.e, 20GBps instead of 2GBps. Redraw Figure 2A for this setup.

Q2.4 [2 marks] For the execution timeline that you drew in Q2.3, what is the speedup and efficiency? Again assume p=2.

We realize that downloading entire 2D arrays from one worker to another is actually quite wasteful. As shown in the right side figure, to compute one 2D array object, the algorithm only requires the bottom-most row from the 2D array above, the right-most row from the 2D array to the left, and just one value from the diagonal 2D array.



So, let us say that the programmer downloads only the required portion of the 2D array objects.

Name: ______________________ Entry number: _______________________

Q2.5 [1 marks] How long does it take to download a row from the 2D array object of 10,000 x 10,000 integer entries on a 20GBps link?
Q2.6 [3 marks] Notice that the download time in Q2.5 is much smaller than the computation time. We can now assume download times to be zero in our calculations. Redraw the execution timeline from Q2.4. Create optimal placement of tasks, i.e, put tasks on workers such that we have minimum total execution time.
Q2.7 [2 marks] What is the speedup and efficiency for the timeline in Q2.6.
Q2.8 [2 marks] Redraw the timeline for Q2.6 assuming that we create 12 2D array objects each of size 7500 * 10000.
Q2.9 [2 marks] What is the speedup and efficiency for the timeline in Q2.8?

Name: ______________________ Entry number: _______________________

***Q3 [9 marks]* True/False questions. If you are unsure about your answer, you may also provide a justification.**

Q3.1 [1 mark] Spark's resilient distributed datasets are immutable.

____

Q3.2 [1 mark] Spark guarantees correctness under faults even when the tasks are non-deterministic.

____

Q3.3 [1 mark] Checkpoints produced by Flink contain the "state" of all the nodes. It is guaranteed that all the nodes were simultaneously in the checkpointed state during the execution.

____

Q3.4 [1 mark] Session windows are all of the same length.

____

Q3.5 [1 mark] Sliding windows do not overlap with one another.

____

Q3.6 [1 mark] False sharing in page-based DSM implementation improves system performance.

____

Q3.7 [1 mark] Reducing page size reduces false sharing in page-based DSM.

____

Q3.8 [1 mark] Assuming no failures and no stragglers, records would typically see lower latency in Flink than in Spark's micro-batch based streaming.

____

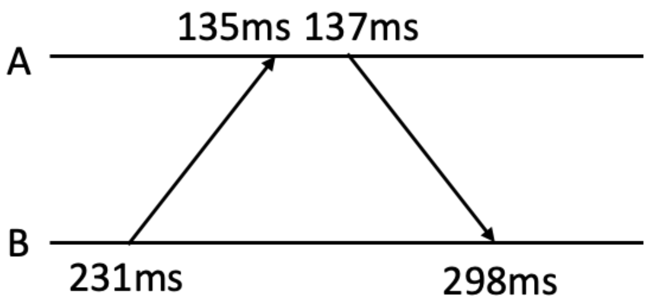
Q3.9 [1 mark] While recovering from a worker failure, records would typically see lower latency in Flink than in Spark's micro-batch based streaming.

____

Name: ______________________ Entry number: _______________________

***Q4 [3 marks]* Network time protocol**

Nodes A and B do the exchange as shown. The figure also shows the local clock time of A and B at send and receive events. Assume that the network time taken from A to B is the same as in B to A.



Q4.1 [1 mark] What is the round trip time between A and B? _________________
Q4.2 [2 marks] By how many milliseconds is A's clock behind B's clock? _________________

Q5 [3 marks] MapReduce

In MapReduce, map tasks write to local disk and reduce tasks write to cluster file system. Let's say we modify this behavior so that the map tasks also write to the cluster file system. What will be the impact of this decision on performance and fault tolerance? Justify your answer.