



DEEP
LEARNING
INSTITUTE



A large, semi-transparent graphic of a neural network or complex graph structure is positioned in the background. It consists of numerous small green dots connected by thin lines, forming a dense web of triangles. A vertical column of these dots and lines is centered, while the rest of the structure is more organic and scattered.

Ejercicios del tercer módulo del DLI

Pertenecientes al workshop
“Fundamentals of Accelerated Computing with CUDA C/C++”

Punto de partida del profiler: addVectorsInto

```
cudaMallocManaged(&a, size);
cudaMallocManaged(&b, size);
cudaMallocManaged(&c, size);
```

```
initWith(3, a, N);
initWith(4, b, N);
initWith(0, c, N);
```

```
addVectorsInto<<<numberOfBlocks, threadsPerBlock>>>(c, a, b, N);
```

```
checkElementsAre(7, c, N);
```

```
cudaFree(a);
cudaFree(b);
cudaFree(c);
```

Ejercicio 1:

Comparar con prebúsqueda en el kernel

```
cudaMallocManaged(&a, size);
cudaMallocManaged(&b, size);
cudaMallocManaged(&c, size);
```

```
initWith(3, a, N);
initWith(4, b, N);
initWith(0, c, N);
```

```
cudaMemPrefetchAsync(a, size, deviceId);
cudaMemPrefetchAsync(b, size, deviceId);
cudaMemPrefetchAsync(c, size, deviceId);
addVectorsInto<<<numberOfBlocks, threadsPerBlock>>>(c, a, b, N);
```

```
checkElementsAre(7, c, N);
```

```
cudaFree(a);
cudaFree(b);
cudaFree(c);
```

Ejercicio 2: Comparar con inicializar en la GPU (y adelantar la prebúsqueda)

```
cudaMallocManaged(&a, size);
cudaMallocManaged(&b, size);
cudaMallocManaged(&c, size);
```

```
cudaMemPrefetchAsync(a, size, deviceId);
cudaMemPrefetchAsync(b, size, deviceId);
cudaMemPrefetchAsync(c, size, deviceId);
initWith<<<numberOfBlocks, threadsPerBlock>>>(3, a, N);
initWith<<<numberOfBlocks, threadsPerBlock>>>(4, b, N);
initWith<<<numberOfBlocks, threadsPerBlock>>>(0, c, N);
cudaDeviceSynchronize();
```

```
addVectorsInto<<<numberOfBlocks, threadsPerBlock>>>(c, a, b, N);
```

```
checkElementsAre(7, c, N);
```

```
cudaFree(a);
cudaFree(b);
cudaFree(c);
```

Ejercicio 3: Comparar con inicializar en GPU y prebúsqueda al devolver los datos a CPU

```
cudaMallocManaged(&a, size);
cudaMallocManaged(&b, size);
cudaMallocManaged(&c, size);
```

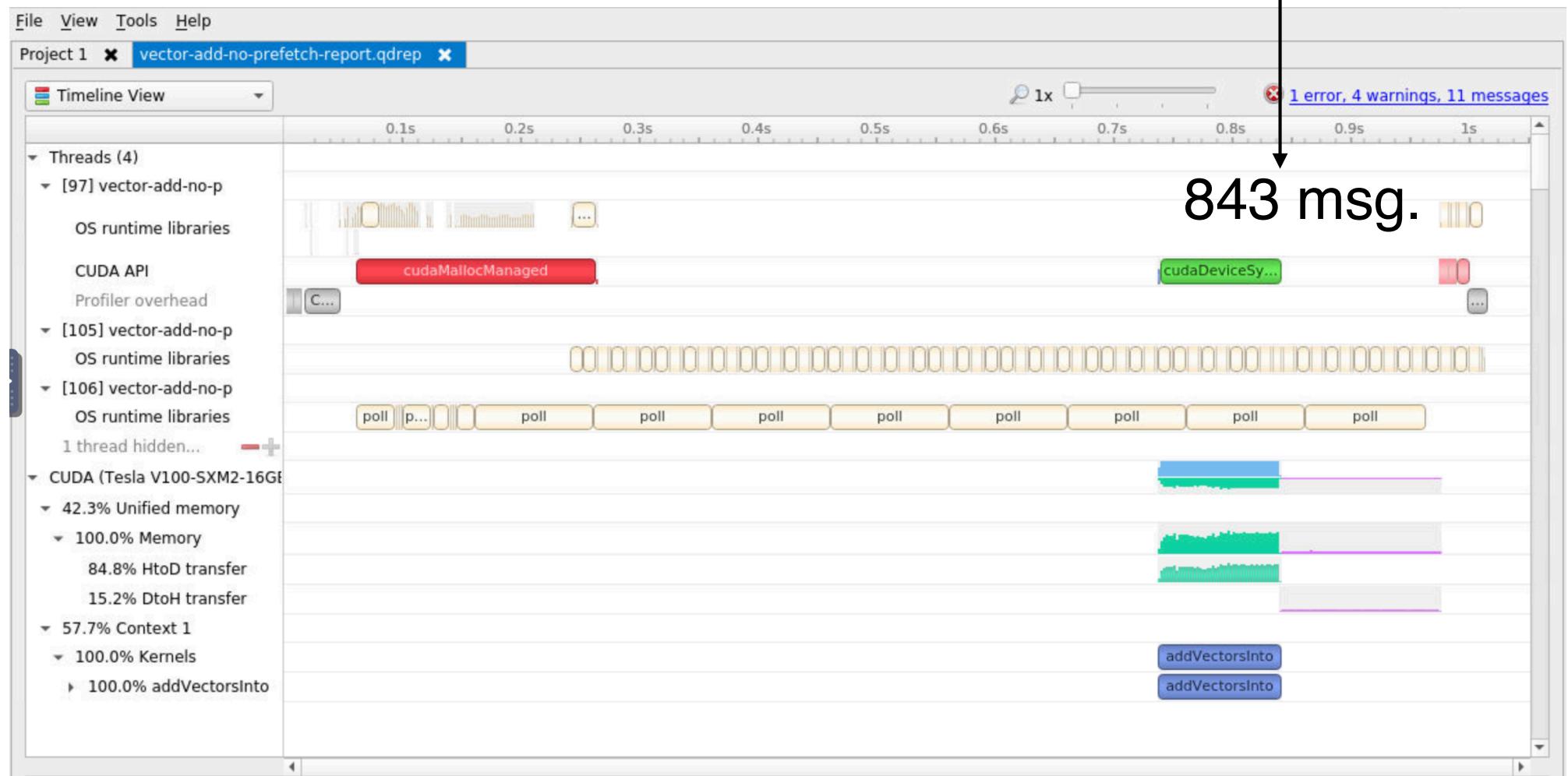
```
cudaMemPrefetchAsync(a, size, deviceId);
cudaMemPrefetchAsync(b, size, deviceId);
cudaMemPrefetchAsync(c, size, deviceId);
initWith<<<numberOfBlocks, threadsPerBlock>>>(3, a, N);
initWith<<<numberOfBlocks, threadsPerBlock>>>(4, b, N);
initWith<<<numberOfBlocks, threadsPerBlock>>>(0, c, N);
cudaDeviceSynchronize();
```

```
addVectorsInto<<<numberOfBlocks, threadsPerBlock>>>(c, a, b, N);
```

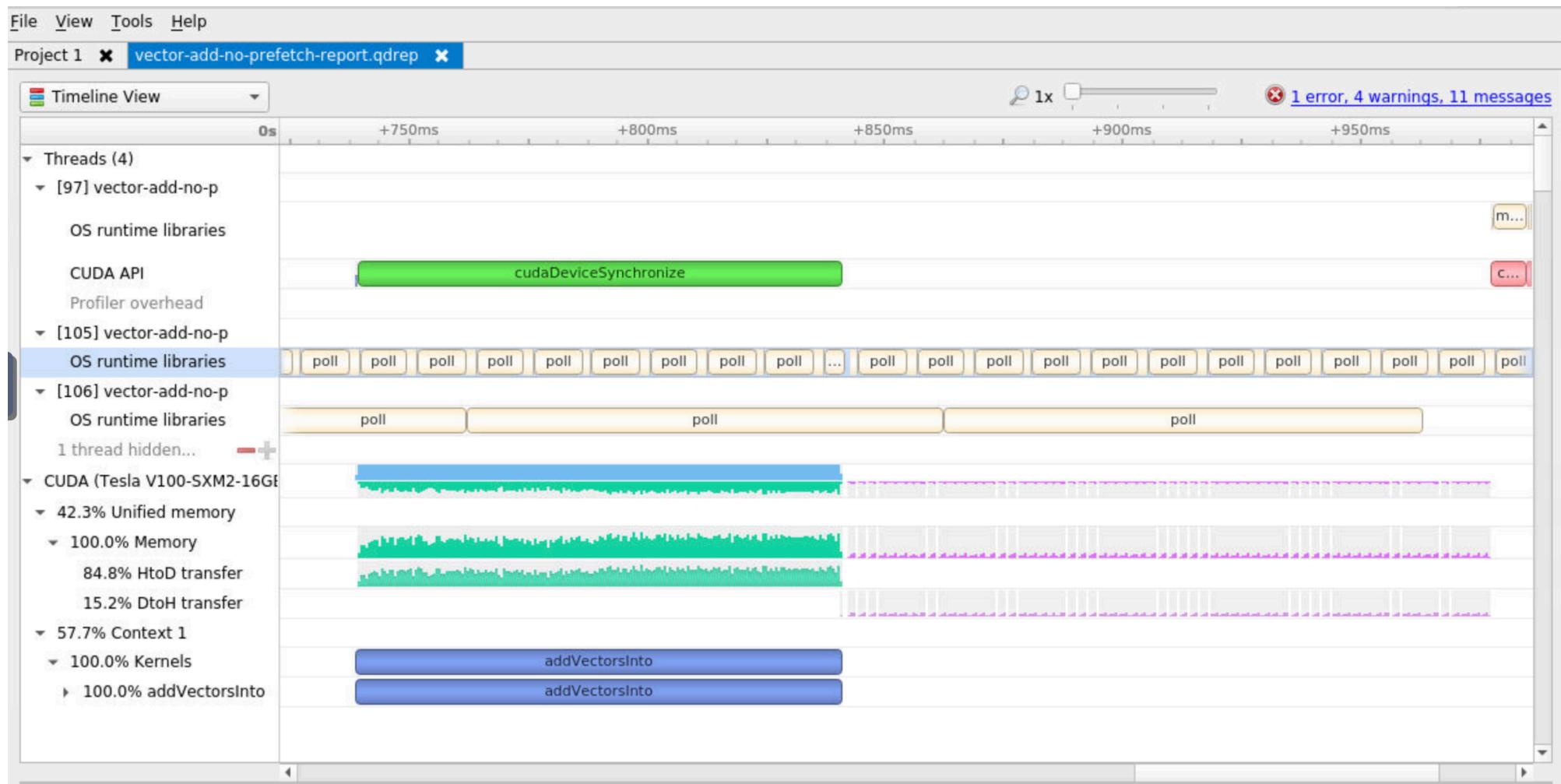
```
cudaMemPrefetchAsync(c, size, cudaCpuDeviceId);
checkElementsAre(7, c, N);
```

```
cudaFree(a);
cudaFree(b);
cudaFree(c);
```

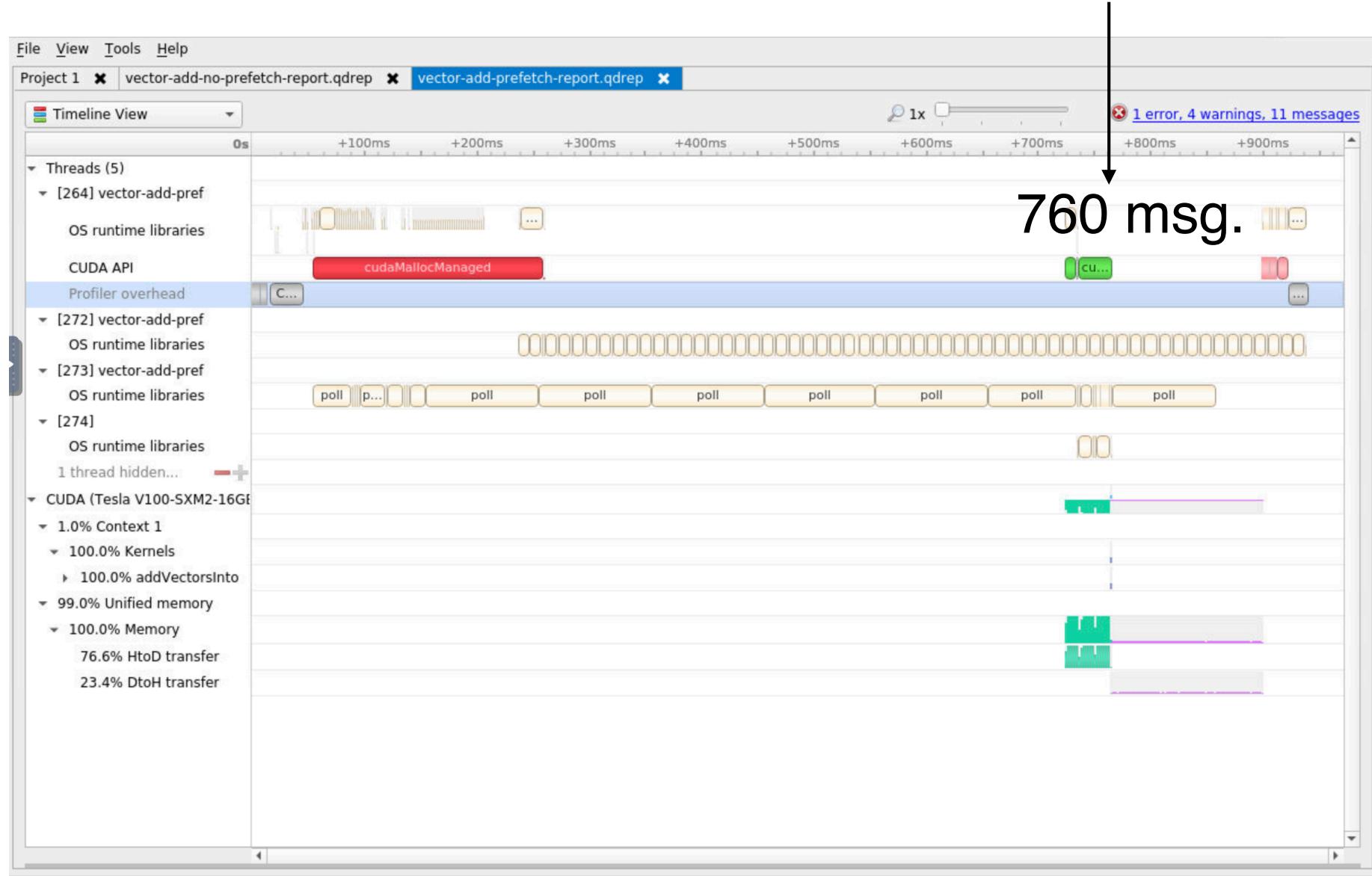
Profiler para el punto de partida



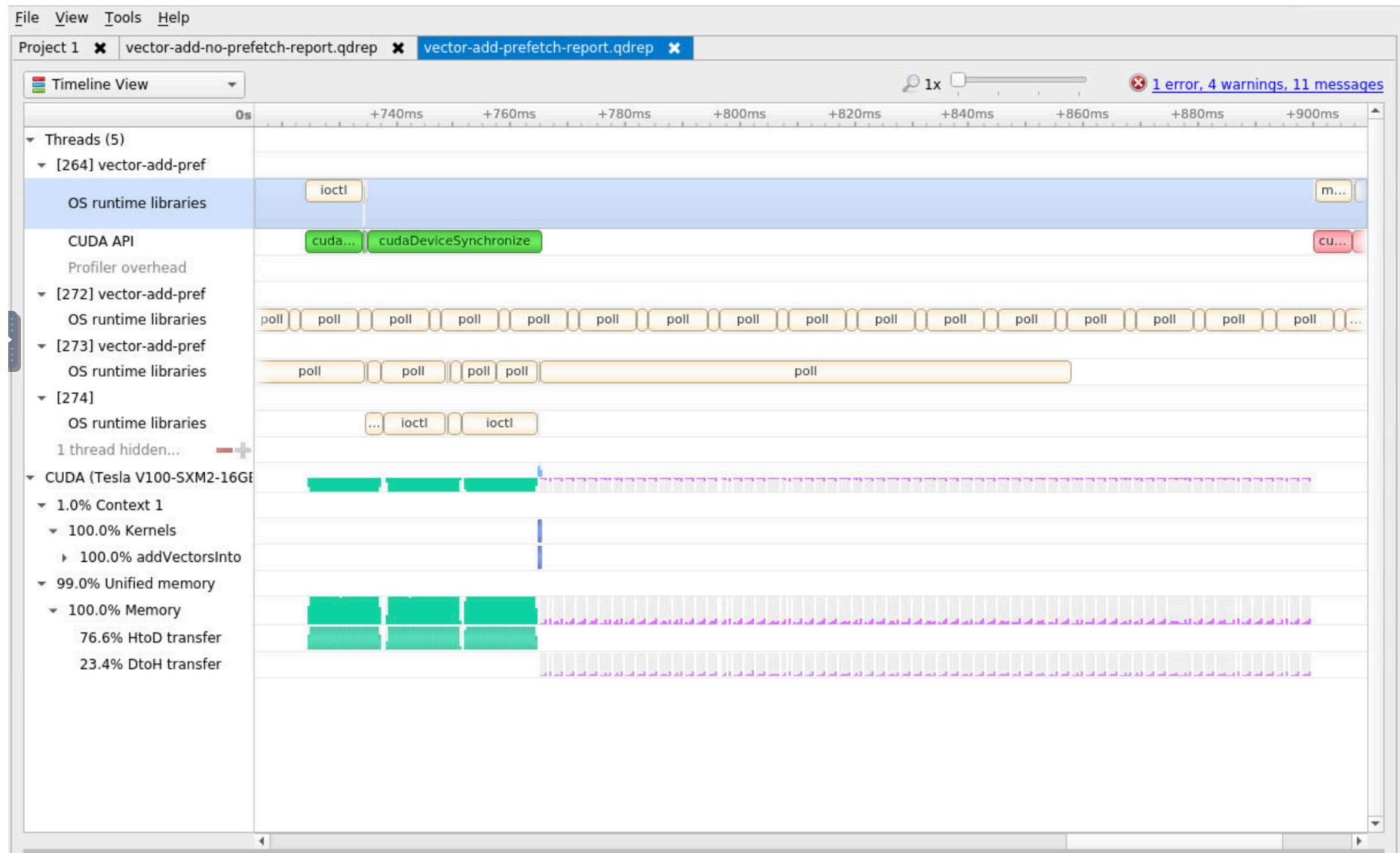
Profiler para el punto de partida (zoom)



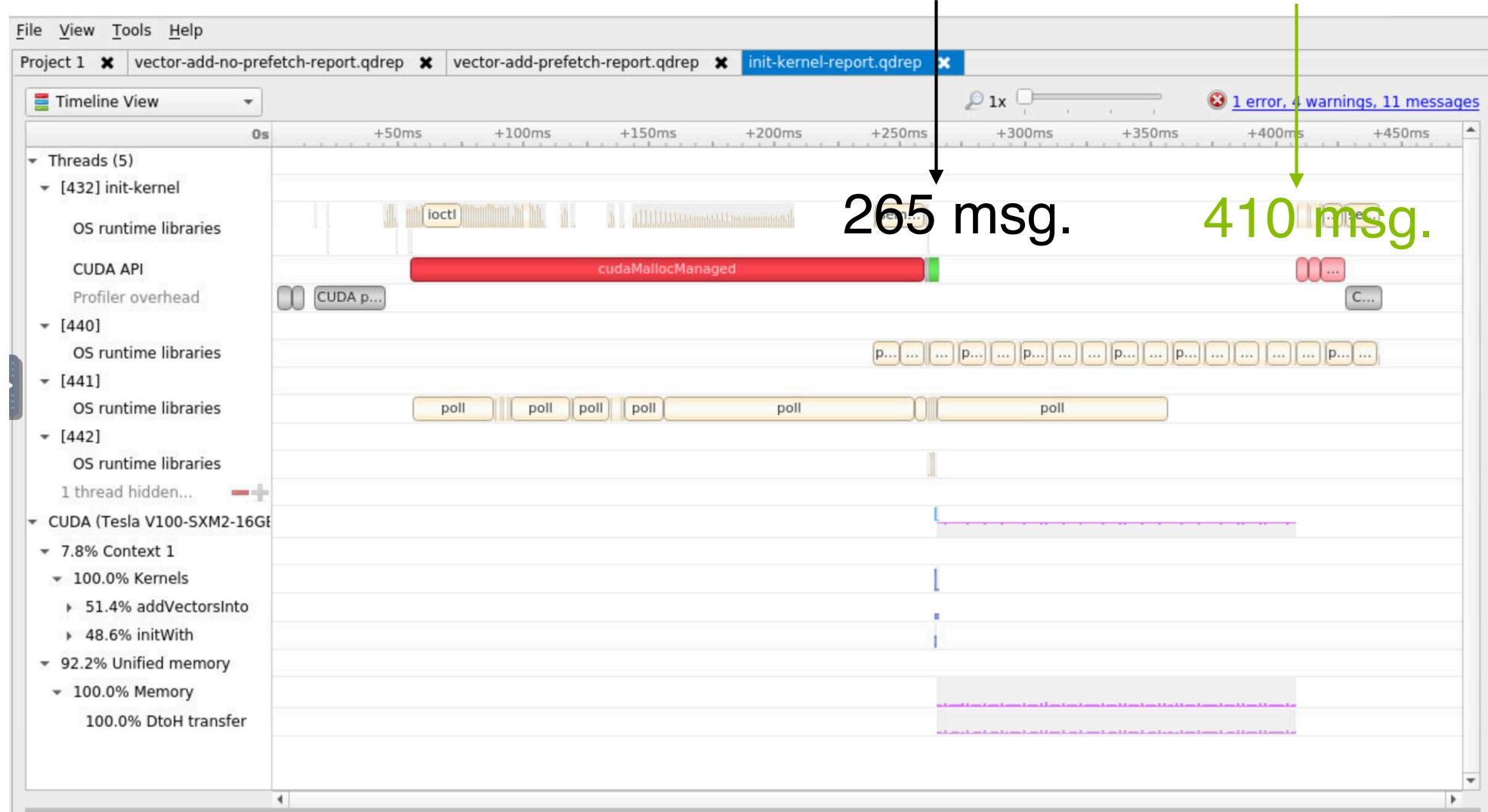
Ejercicio 1: Profiler para la prebúsqueda en el kernel



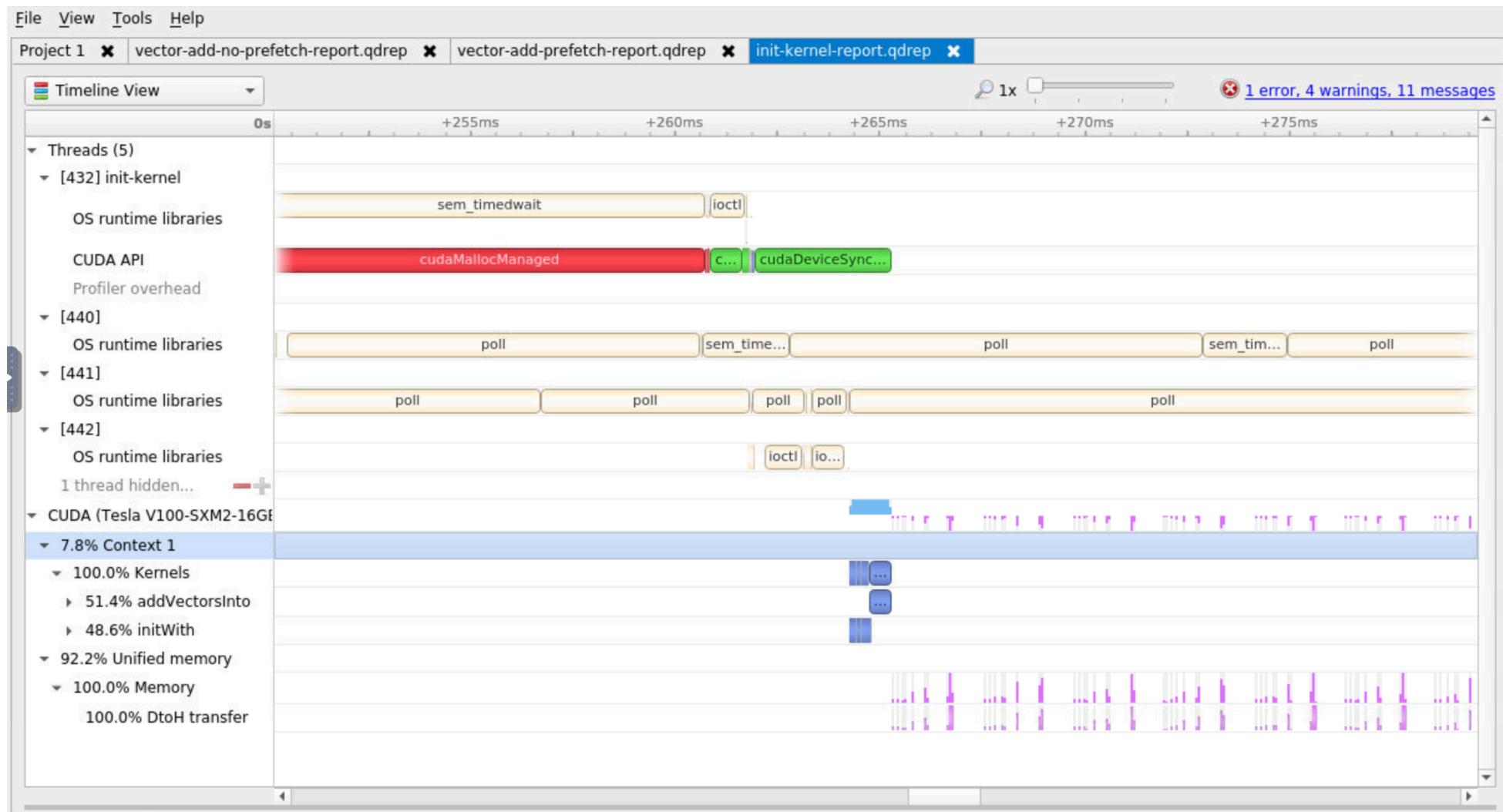
Ejercicio 1: Profiler para la prebúsqueda en el kernel (zoom)



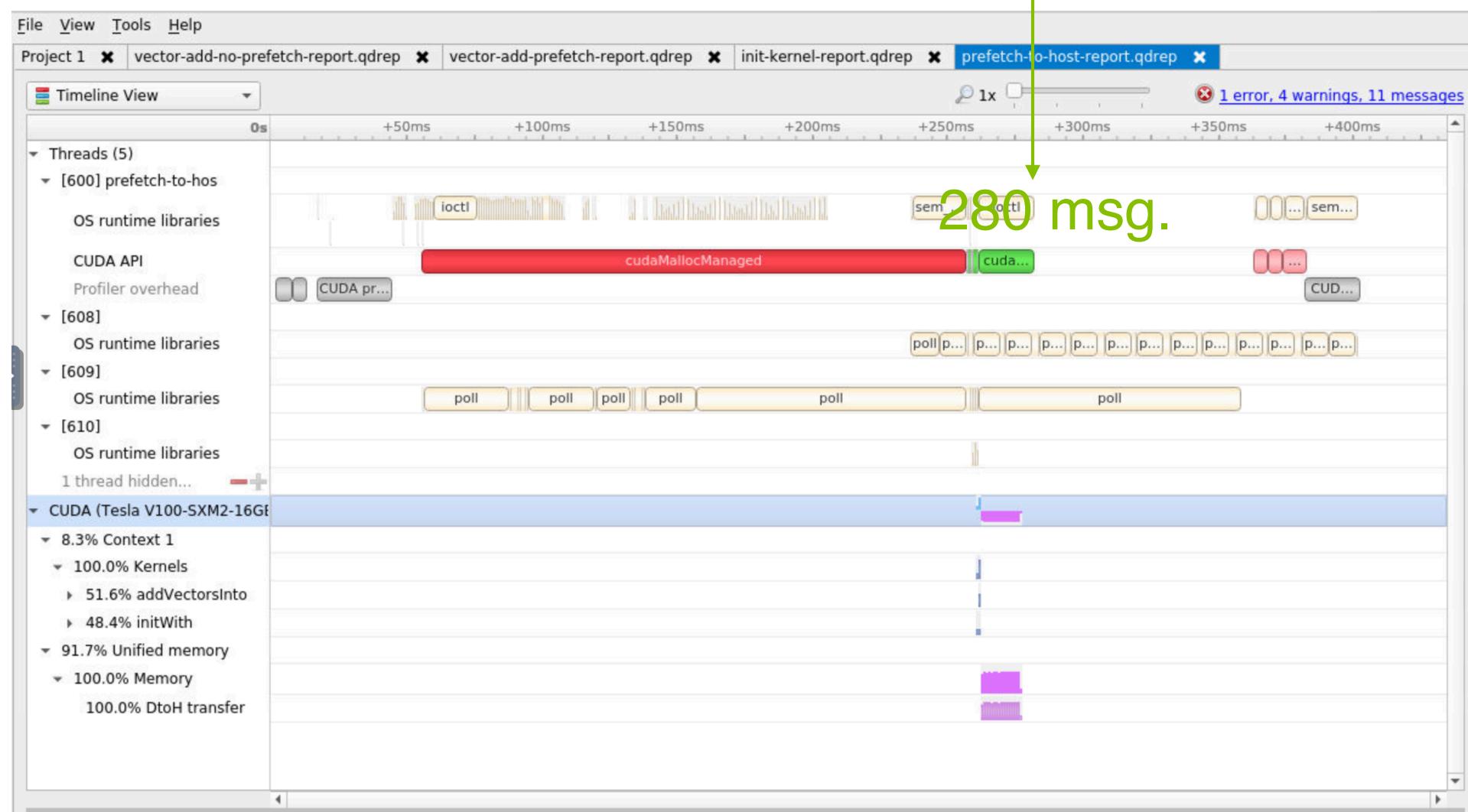
Ejercicio 2: Profiler para inicialización (y prebúsqueda) en la GPU



Ejercicio 2: Profiler para inicialización (y prebúsqueda) en la GPU (zoom)



Ejercicio 3: Profiler para inicialización en GPU y prebúsqueda final en CPU



Ejercicio 3: Profiler para inicialización en GPU y prebúsqueda final en CPU (zoom)

