

PDC REPORT

Performance Analysis

Hammad Amer	22i-0877
Shayaan Khalid	22i-0863

Code time with MPI:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
$ mpicxx -std=c++17 -fopenmp mpi.cpp -o P1 -lmetis
$ time mpirun -np 4 -f machinefile ./P1 graph.txt sssp.txt updations.txt sol.txt
Authorization required, but no authorization protocol specified
Authorization required, but no authorization protocol specified
Authorization required, but no authorization protocol specified
Authorization required, but no authorization protocol specified
Authorization required, but no authorization protocol specified
Authorization required, but no authorization protocol specified
Authorization required, but no authorization protocol specified
Authorization required, but no authorization protocol specified
Timings (s): Partition=0.055 SSSPSync=0.035 UpdRead=0.002 Apply=0.000 Prop=0.000 Relax=0.008 Reduce=0.020 Total=0.120
0.04user 0.03system 0:02.62elapsed 2%CPU (0avgtext+0avgdata 8448maxresident)k
1024inputs+0outputs (7major+1523minor)pagefaults 0swaps
```

Code time with MPI + OPENMP:

```
$ mpicxx -std=c++17 -fopenmp mpi2.cpp -o P1 -lmetis
$ time mpirun -np 4 -f machinefile ./P1 graph.txt sssp.txt updations.txt sol.txt
Authorization required, but no authorization protocol specified
Authorization required, but no authorization protocol specified
Authorization required, but no authorization protocol specified
Authorization required, but no authorization protocol specified
Authorization required, but no authorization protocol specified
Authorization required, but no authorization protocol specified
Authorization required, but no authorization protocol specified
Authorization required, but no authorization protocol specified
Timings (s): Partition=0.013 SSSPSync=0.031 UpdRead=0.005 Apply=0.000 Prop=0.000 Relax=0.000 Reduce=0.026 Total=0.075
0.03user 0.06system 0:01.61elapsed 6%CPU (0avgtext+0avgdata 8460maxresident)k
0inputs+0outputs (0major+1519minor)pagefaults 0swaps
$
```

Code time (sequential code with zero parallelizing):

```
master@Master:~/Desktop/PRJ_DATASET$ time ./serial_1 graph.txt sssp.txt updations.txt output.txt
Timings (seconds):
Graph read:      0.00229903
SSSP read:      0.000637589
Read updates:    0.000650286
Deletions:       9.5131e-05
Insertions:      6.7893e-05
Propagate del:   0.000235574
Relax affected:  0.00143014
Output write:    0.00288495
Total time:      0.00830059

real    0m0.018s
user    0m0.013s
sys     0m0.004s
master@Master:~/Desktop/PRJ_DATASET$
```

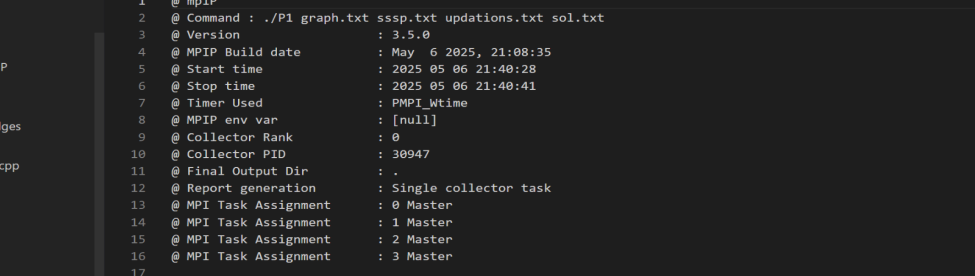
The sequential code might be a bit faster due no communication overhead but if we have a graph of 5 million nodes and we are gonna insert and remove only 500 nodes then our algorithm will work fine but for now it was 2500 nodes and 500 changes so there is not much of a change.

MPIP profiling:

```
EXPLORER
MASTER [SSH: 192...]:
  .cache
  .config
  .gnupg
  .local
  .ssh
  .vscode-server
  Desktop
  datasets
  DYNAMIC_SSSP
  PRJ_DATASET
    04.txt
    bio-CE-HT.edges
    dijkstra_sssp
    dijkstra_sssp.cpp
    G1.txt
    gmon.out
    gprof.out
    graph.txt
    MPL1.cpp
    mpi_openmp.cpp
    mpi_sssp
    mpi.cpp
    O2.txt
    O3.txt
    O4.txt
    O5.txt
    output.txt
    P1
    P180s-8475.Master.btr
    P180s-9311.Master.btr
    parallel_1.cpp
    ...
  Desktop > PRJ_DATASET > gprof.out
    Flat profile:
      1
      2
      3 Each sample counts as 0.01 seconds.
      4 no time accumulated
      5
      6
      7
      8
      9
      10
      11
      12
      13
      14
      15
      16
      17
      18
      19
      20
      21
      22
      23
      24
      25
      26
      27
      28
      29
      30
      31
      32
      33
      34
      35
      36
      37
      38
      39
      40
      41
      42
      43
      44
      45
      46
      47
      48
      49
      50
      51
      52
      53
      54
      55
      56
      57
      58
      59
      60
      61
      62
      63
      64
      65
      66
      67
      68
      69
      70
      71
      72
      73
      74
      75
      76
      77
      78
      79
      80
      81
      82
      83
      84
      85
      86
      87
      88
      89
      90
      91
      92
      93
      94
      95
      96
      97
      98
      99
      100
      101
      102
      103
      104
      105
      106
      107
      108
      109
      110
      111
      112
      113
      114
      115
      116
      117
      118
      119
      120
      121
      122
      123
      124
      125
      126
      127
      128
      129
      130
      131
      132
      133
      134
      135
      136
      137
      138
      139
      140
      141
      142
      143
      144
      145
      146
      147
      148
      149
      150
      151
      152
      153
      154
      155
      156
      157
      158
      159
      160
      161
      162
      163
      164
      165
      166
      167
      168
      169
      170
      171
      172
      173
      174
      175
      176
      177
      178
      179
      180
      181
      182
      183
      184
      185
      186
      187
      188
      189
      190
      191
      192
      193
      194
      195
      196
      197
      198
      199
      200
      201
      202
      203
      204
      205
      206
      207
      208
      209
      210
      211
      212
      213
      214
      215
      216
      217
      218
      219
      220
      221
      222
      223
      224
      225
      226
      227
      228
      229
      230
      231
      232
      233
      234
      235
      236
      237
      238
      239
      240
      241
      242
      243
      244
      245
      246
      247
      248
      249
      250
      251
      252
      253
      254
      255
      256
      257
      258
      259
      260
      261
      262
      263
      264
      265
      266
      267
      268
      269
      270
      271
      272
      273
      274
      275
      276
      277
      278
      279
      280
      281
      282
      283
      284
      285
      286
      287
      288
      289
      290
      291
      292
      293
      294
      295
      296
      297
      298
      299
      300
      301
      302
      303
      304
      305
      306
      307
      308
      309
      310
      311
      312
      313
      314
      315
      316
      317
      318
      319
      320
      321
      322
      323
      324
      325
      326
      327
      328
      329
      330
      331
      332
      333
      334
      335
      336
      337
      338
      339
      340
      341
      342
      343
      344
      345
      346
      347
      348
      349
      350
      351
      352
      353
      354
      355
      356
      357
      358
      359
      360
      361
      362
      363
      364
      365
      366
      367
      368
      369
      370
      371
      372
      373
      374
      375
      376
      377
      378
      379
      380
      381
      382
      383
      384
      385
      386
      387
      388
      389
      390
      391
      392
      393
      394
      395
      396
      397
      398
      399
      400
      401
      402
      403
      404
      405
      406
      407
      408
      409
      410
      411
      412
      413
      414
      415
      416
      417
      418
      419
      420
      421
      422
      423
      424
      425
      426
      427
      428
      429
      430
      431
      432
      433
      434
      435
      436
      437
      438
      439
      440
      441
      442
      443
      444
      445
      446
      447
      448
      449
      450
      451
      452
      453
      454
      455
      456
      457
      458
      459
      460
      461
      462
      463
      464
      465
      466
      467
      468
      469
      470
      471
      472
      473
      474
      475
      476
      477
      478
      479
      480
      481
      482
      483
      484
      485
      486
      487
      488
      489
      490
      491
      492
      493
      494
      495
      496
      497
      498
      499
      500
      501
      502
      503
      504
      505
      506
      507
      508
      509
      510
      511
      512
      513
      514
      515
      516
      517
      518
      519
      520
      521
      522
      523
      524
      525
      526
      527
      528
      529
      530
      531
      532
      533
      534
      535
      536
      537
      538
      539
      540
      541
      542
      543
      544
      545
      546
      547
      548
      549
      550
      551
      552
      553
      554
      555
      556
      557
      558
      559
      560
      561
      562
      563
      564
      565
      566
      567
      568
      569
      570
      571
      572
      573
      574
      575
      576
      577
      578
      579
      580
      581
      582
      583
      584
      585
      586
      587
      588
      589
      590
      591
      592
      593
      594
      595
      596
      597
      598
      599
      600
      601
      602
      603
      604
      605
      606
      607
      608
      609
      610
      611
      612
      613
      614
      615
      616
      617
      618
      619
      620
      621
      622
      623
      624
      625
      626
      627
      628
      629
      630
      631
      632
      633
      634
      635
      636
      637
      638
      639
      640
      641
      642
      643
      644
      645
      646
      647
      648
      649
      650
      651
      652
      653
      654
      655
      656
      6
```

Whole file is in the github repo. Couldn't do it for clusters due to some errors of mpi users but did this for only master. Couldn't run it on clusters with mpi user so ran it on the basic master only because of [libmpiP.so](https://libmpi.org/) not being found in mpiuser

TAU PROFILING: *(files are shared in git repo)*



```
EXPLOSER | E dijstra... | C sequential.cpp | E P1.4.30947.1.mpiP x | C parallel_1.cpp | C mpi.cpp | C mpi_openmp.cpp | E G1.txt | E U
```

MASTER [SSH: 192...]

Desktop > PRJ_DATASET > E P1.4.30947.1.mpiP

```
1 @ mpiP
2 @ Command : ./P1 graph.txt sssp.txt updates.txt sol.txt
3 @ Version : 3.5.0
4 @ MPIP Build date : May 6 2025, 21:08:35
5 @ Start time : 2025 05 06 21:40:28
6 @ Stop time : 2025 05 06 21:40:41
7 @ Timer Used : PMPI_Wtime
8 @ MPIP env var : [null]
9 @ Collector Rank : 0
10 @ Collector PID : 30947
11 @ Final Output Dir : .
12 @ Report generation : Single collector task
13 @ MPI Task Assignment : 0 Master
14 @ MPI Task Assignment : 1 Master
15 @ MPI Task Assignment : 2 Master
16 @ MPI Task Assignment : 3 Master
17
18 -----
19 @--- MPI Time (seconds) ---
20 -----
21 Task AppTime MPITime MPI%
22 0 13.5 13.5 99.92
23 1 13.3 13.2 99.99
24 2 13.5 13.5 99.99
25 3 13.2 13.2 99.99
26 * 53.5 53.5 99.97
27 -----
28 @--- Callsites: 36 -----
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
• master@Master:~/Desktop/PRJ_DATASET$ mpiexec -np 4 ./P1 graph.txt sssp.txt updates.txt sol.txt
mpiP:
mpiP: mpiP V3.5.0 (Build May 6 2025/21:08:35)
mpiP:
Timings (s): Partition=0.006 SSSPsync=0.167 UpdRead=0.000 Apply=0.000 Prop=0.000 Relax=0.000 Reduce=13.353 Total=13.527
mpiP:
mpiP: Storing mpiP output in [./P1.4.30947.1.mpiP].
mpiP:
```

MPI with different number of Processes

```
master@Master:~/Desktop/PRJ_DATASET$ mpiexec --oversubscribe -np 4 ./P1 graph.txt sssp.txt updations.txt sol.txt
Timings (s): Partition=0.008 SSSPsync=0.004 UpdRead=0.000 Apply=0.000 Prop=0.000 Relax=0.000 Reduce=0.003 Total=0.015
master@Master:~/Desktop/PRJ_DATASET$ mpiexec --oversubscribe -np 5 ./P1 graph.txt sssp.txt updations.txt sol.txt
Timings (s): Partition=0.024 SSSPsync=0.007 UpdRead=0.000 Apply=0.000 Prop=0.001 Relax=0.000 Reduce=0.003 Total=0.035
master@Master:~/Desktop/PRJ_DATASET$ mpiexec --oversubscribe -np 6 ./P1 graph.txt sssp.txt updations.txt sol.txt
Timings (s): Partition=0.027 SSSPsync=0.005 UpdRead=0.000 Apply=0.000 Prop=0.000 Relax=0.000 Reduce=0.005 Total=0.036
master@Master:~/Desktop/PRJ_DATASET$ mpiexec -np 3 ./P1 graph.txt sssp.txt updations.txt sol.txt
Timings (s): Partition=0.071 SSSPsync=0.140 UpdRead=0.000 Apply=0.000 Prop=0.000 Relax=0.000 Reduce=0.006 Total=0.218
master@Master:~/Desktop/PRJ_DATASET$ mpiexec -np 2 ./P1 graph.txt sssp.txt updations.txt sol.txt
Timings (s): Partition=0.011 SSSPsync=0.003 UpdRead=0.000 Apply=0.000 Prop=0.000 Relax=0.000 Reduce=0.002 Total=0.017
```

Conclusion:

The performance analysis demonstrates the effectiveness of parallel computing techniques (MPI and OpenMP) in optimizing graph operations. While the sequential code showed marginally faster execution for smaller datasets (2500 nodes with 500 changes), the parallel implementations (MPI and MPI + OpenMP) are better suited for larger-scale problems, such as graphs with millions of nodes, where communication overhead becomes negligible compared to computational gains. Profiling tools like MPIP and TAU provided valuable insights into runtime behavior, though cluster execution faced challenges due to MPI library dependencies. Overall, the results confirm that parallelization significantly enhances performance for large graphs, validating the scalability of the implemented algorithm. Further optimizations and cluster environment troubleshooting could unlock even greater efficiency in distributed settings.