IMPLEMENTATION AND PERFORMANCE OF PARALLELIZED elegant [*]

Y. Wang[†] M. Borland, ANL, Argonne, IL 60439, USA

Abstract

The program elegant is widely used for design and modeling of linacs for free-electron lasers and energy recovery linacs, as well as storage rings and other applications. As part of a multi-year effort, we have parallelized many aspects of the code, including single-particle dynamics, wakefields, and coherent synchrotron radiation. We report on the approach used for gradual parallelization, which proved very beneficial in getting parallel features into the hands of users quickly. We also report details of parallelization of collective effects. Finally, we discuss performance of the parallelized code in various applications.

INTRODUCTION

The accelerator simulation code "elegant" [1] has been undergoing parallelization through an element-by-element strategy. As explained in more detail below, we used a gradual parallelization approach [2] that permitted users to take advantage of parallelization as it proceeded. It also allowed continued work on the serial version through frequent merging of the parallelized code into the serial code base. Automated regression testing (testing against prior results) and cross-comparison of the serial and parallel versions were used extensively to avoid introduction of errors.

As reported earlier [2], for simulations dominated by single-particle optics elements, parallelized elegant (Pelegant) achieved near optimal speedup with up to 512 CPUs on the BlueGene/L supercomputer at Argonne National Laboratory (ANL). The first version of Pelegant, released in 2006, had most of the single-particle dynamics elements parallelized. This has proved very beneficial for research programs at the Advanced Photon Source (APS), e.g., modeling of the pulsed crab cavity scheme for short x-ray pulse production [3].

Recent developments focus on parallelizing the elements with collective effects, such as short-range longitudinal wake, coherent synchrotron radiation, and the element with transverse space-charge effects. These have proven essential in development and modeling of concepts for energy recovery linac (ERL) upgrades to the APS [4] and modeling related to the International Linear Collider (ILC) damping ring.

PARALLELIZATION STRATEGY

The parallelization is done through the master/slaves model. The master is responsible for I/O operations, distributing workload to the slaves, and tracking in serial elements. All the elements are divided into four classes according to their properties and the stage of parallelization. The tracking routine in Pelegant keeps track of whether the particles reside on the master or slaves. By looking at the next element classification, it can determine whether to scatter the particles, gather the particles, or retain them where they are.

1. Parallel element: only the slave processors will do the tracking. Each slave is responsible for a portion of the particles. No communication is required among the slaves or between the master and slaves. Elements of this type involve single-particle dynamics only.
2. MP (multiprocessor) algorithm: the master will participate in the tracking, but without doing any intensive computations itself. In most cases, the master collates statistical data from the slaves and then shares it with them. Further computations are then performed by the slaves.
3. Uniprocessor element: must be done by master (for now) and modifies particle coordinates. The first (last) such element in a sequence of uniprocessor elements requires that the master gather (scatter) all the particles from (to) the slaves.
4. Diagnostic: same as the uniprocessor element, but doesn't change particle coordinates. An example would be using the WATCH element to dump raw particle data to a file. In this case, the master must gather but need not scatter.

As nearly 80% of the elements have been parallelized, the chance of frequent gather and scatter operations is small. Good speedup can be expected even with a mixture of parallel and serial elements, provided Pelegant is used properly. For example, particularly in ring simulations, it is often possible to gather certain serial elements together to avoid multiple gather/scatter cycles.

PARALLELIZATION OF COLLECTIVE EFFECTS

Most of the collective effects in elegant make use of histograms. For example, for the short-range longitudinal wake, elegant first determines all particle arrival times, which are stored in a histogram as $I(t)$. Next,

[*] Work supported by the U.S. Department of Energy, Office of Science, Office of Basic Energy Sciences, under Contract No. DE-AC02-06CH11357.

[†] ywang25@aps.anl.gov

it convolves the histogram with the user-supplied wake potential $W(t)$ to get the wakefield of a bunch distribution $V(t)$. For each particle, $V(t)$ is interpolated to get change in longitudinal momentum p_z. The parallel version uses the same basic approach. After computing the time coordinates of their particles, the slaves share their minimum and maximum time values so that the global minimum and maximum may be determined. These values are used by the slaves to produce individual processor histograms $I_i(t)$. After making these histograms, the N slaves use a collective operation to compute and share $I(t) = \Sigma_{i=0}^{N-1} I_i(t)$. Each slave then computes $V(t)$ and applies it to the particles it controls. We used functions from the Message Passing Interface (MPI [5]) to perform operations such as global minimum, global maximum, and global summation of arrays. Elements parallelized in this fashion include short-range longitudinal wakes/impedances (`WAKE` and `ZLONGIT`), short-range transverse wakes/impedances (`TRWAKE` and `ZTRANSVERSE`), longitudinal cavity modes (`RFMODE` and `FRFMODE`), transverse cavity modes (`TRFMODE` and `TFRFMODE`), and coherent synchrotron radiation (`CSRCSBEND` and `CSRDRIFT`).

Collective effects often involve beam moments, computation of which can also be readily parallelized using MPI calls. An example of this is transverse space charge in a storage ring. A transverse space-charge kick element (`SCMULT`) using K.Y. Ng's formula was added to `elegant` recently [6]. For the parallel version of this element, most of the communication overhead happens when calculating the rms sizes, while the remaining part can be calculated for all the particles on each of the processors independently.

PERFORMANCE EXAMPLES

In [2], we showed the near optimal efficiency for the crab cavity simulation on up to 512 CPUs of the BlueGene/L supercomputer at ANL. Good parallel efficiency can be expected for simulations dominated by single-particle dynamics elements, as the particles track through elements independently without any significant communication overhead. In contrast, it is notoriously difficult to achieve good performance for a simulation dominated by collective-effect elements, as intensive communication overheads are involved during tracking. Good speedup can still be achieved through careful implementation/optimization by the developers, combined with thoughtful use of `Pelegant`.

Energy Recovery Linacs

Energy recovery linacs (ERLs) are a concept of great interest for an upgrade of the APS [4]. The simulations involve a number of elements involving collective effects. As explained above, the algorithm for computing the wakefield requires `Pelegant` to sum the individual processor histograms to create a global histogram. These types of calculations are realized with the so-called all-to-all communication pattern in MPI, which can be one of the most time-consuming communication patterns. The good news is the communication overhead is only proportional to the number of bins for the histogram, i.e., the amount of communication will not be increased when the number of particles is increased. As a result, the larger the number of particles, the greater the share of running time that will be occupied by useful computation, and the greater the efficiency. (Here we neglect the communication overhead from particle scattering and gathering, as most of the relevant elements have been parallelized at this point.) Of course, simulating with a large number of particles is exactly what `Pelegant` is designed to do.

Table 1: Speedup for ERL simulation on BlueGene/L

	250k particles		**1 million particles**	
CPUs	**Time**	**Speedup**	**Time**	**Speedup**
1	30:52:14	1.0	123:25:56	1.0
32	01:08:48	26.9	04:05:50	30.1
64	00:38:44	47.8	02:05:38	59.0
128	00:23:11	79.9	01:07:03	110.5
256	00:17:01	108.8	00:38:09	194.2
512	00:13:55	133.1	00:24:19	304.7

We did two experiments to evaluate the performance of `Pelegant` with an ERL simulation on the BlueGene/L supercomputer at ANL. We first ran the ERL simulations with 250k particles, and observed the speedup of 133 on 512 CPUs. The result for the first experiment can be found in the second and third columns of Table 1. With the same lattice, we did another series of simulations with 1 million particles with the result shown in the last two columns of Table 1. (The time spent for the serial version is estimated from the result of 250k particles.) The speedup is very close to the optimal speedup when the simulations are run on 32 or 64 CPUs, which is typical of the resources available on small clusters. The speedup for this simula-

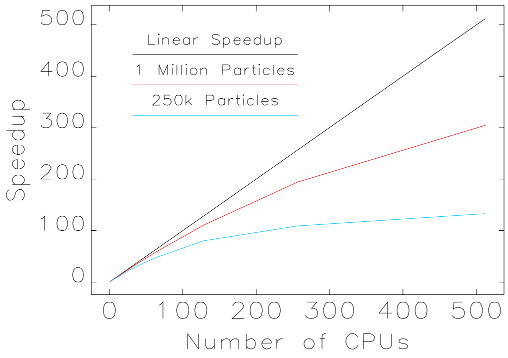

Figure 1: Speedup of the ERL simulation on BlueGene/L supercomputer. The black line shows the optimal speedup; the blue and red lines show the speedup of the simulations with 250k particles and 1 million particles respectively.

1-4244-0917-9/07/$25.00 ©2007 IEEE

tion is 304 on 512 CPUs, which is more than double of the speedup achieved for the simulation of 250k particles.

Figure 1 plots the data from Table 1 for comparison. We can find that the parallel efficiency is significantly improved for the simulations of 1 million particles. These experimental results validate our hypothesis, given at the beginning of this section, that better efficiency can be achieved with a larger number of particles. Thus, it is recommended to allocate sufficient computational workload (particles) to use `Pelegant` on a supercomputer efficiently. Put another way, the number of processors has to be chosen to reflect the workload.

Simulation with Transverse Space-Charge Effects

We also did a performance evaluation for the simulation with space-charge effects, which is an important research subject for the ILC damping ring [6]. For these simulations, most of the calculations can be done independently on each of the processors except for the rms size calculations. The communication overhead for these rms size calculations is relatively trivial, as only local moments of the coordinates, instead of all the coordinates of particles, need to be passed between processors. As a result, a good speedup can be expected.

Table 2: Results for damping ring simulations with space-charge effects and 1 million particles on the Apex cluster

CPUs	Time (hours)	Speedup
1	38.12	1.0
10	3.77	10.1
20	1.33	28.8
30	0.9	42.3
40	0.68	56.0
50	0.56	68.4
60	0.46	83.5

Table 2 gives the simulation result with 1 million particles on the 64-core Apex cluster at APS. The simulation time is reduced from 38 hours to less than half an hour. We observed super-linear speedup due to the good fit in the cache (1 MB per core) and enough computational workload assigned for each of the processors. In another series of simulations with 100k particles, the speedup rolls off after 50 processors. The number of CPUs for the maximal speedup reached is decreased when the number of particles is reduced, as the communication overheads become dominant when a relatively small computational workload is assigned to each processor. This shows again that an appropriate computational load is necessary to achieve good parallel efficiency with `Pelegant`.

CONCLUSION

The gradual strategy we used to parallelize `elegant` has been found to be both feasible and practical. Accelerator physicists are able to take advantage of the state-of-the-art computing facility with `Pelegant` on several on-going research and operations-related projects. It not only saves a lot of valuable time for the researcher, but also provides the possibility of running large simulations, which are not practical with serial `elegant`.

The performance can strongly depend on the ratio of computation load to inter-processor communication for a particular problem, and the amount of output operations (which have not been parallelized at this time). The user can expect better efficiency with larger computational workload and minimal output. Also, the scalability depends on the hardware of the cluster, which includes the cache size of each CPU, the speed of the network, the time required for synchronization, and so on. We found the synchronization time for an all-to-all communication on BlueGene/L is much shorter than the other clusters for the same number of CPUs. This is crucial to achieve good performance, especially when there is a relatively small computational workload on each of the processors.

For an ERL, the simulation time for 1 million particles was reduced from 123 hours to 24 minutes on 512 CPUs of the BlueGene/L supercomputer at ANL. The parallel efficiency for this type of simulations varies from 60% (for 512 CPUs) to 94% (for 32 CPUs). In a simulation including the transverse space-charge effects with 1 million particles, `Pelegant` reduced the time from 38 hours to less than half an hour on 60 CPUs of the Apex cluster at APS. Although `Pelegant` shows good efficiency for simulations with both single-particle dynamics and collective-effect simulations, users should be advised that good scalability can be expected only if there is sufficient workload on each of the CPUs, i.e., the number of particles should be large. Otherwise, the communication overhead between processors will negate the speedup gained from using multiple processors.

REFERENCES

[1] M. Borland, “elegant: A Flexible SDDS-Compliant Code for Accelerator Simulation,” Advanced Photon Source LS-287, September 2000.

[2] Y. Wang and M. Borland, “Pelegant: A Parallel Accelerator Simulation Code for Electron Generation and Tracking,” Proceedings of 12th Advanced Accelerator Concepts Workshop, AIP Conf. Proc. 877, 241-247 (2006).

[3] M. Borland *et al.*, “Planned Use of Pulsed Crab Cavities for Short X-ray Pulsed Generation at the Advanced Photon Source,” these proceedings.

[4] M. Borland *et al.*, “Configuration, Optics, and Performance of a 7-GeV Energy Recovery Linac Upgrade for the Advanced Photon Source,” these proceedings.

[5] W. D. Gropp, E. Lust, and R. Thakur, *Using MPI-2: Advanced Features of the Message-Passing Interface*, MIT Process, Cambridge, MA, 1999.

[6] A. Xiao *et al.*, “Direct Space Charge Calculation in elegant and its Application to the ILC Damping Ring,” these proceedings.