A Model For the Time Complexity of the Time Stepper w/ MPI

Max Ruth

```
$Assumptions = {τ ∈ Integers, τ > 0};
(* Assumptions about the variables we are working with. In particular,
we want any Sums to evaluate correctly*)
```

Description of the problem

There are a few parameters that we have to keep track of in this problem. They are:

```
Clear[Nx, Ny, L, M, T]
Nx; Ny; (* Number of processors in the x and y directions *)
L; (* Length of the x or y direction of the domain (it's square!) *)
M; (* Number of grid points the x or y direction *)
T; (* Total time for which the problem is run *)
τ; (* Number of time steps taken per MPI Communication *)
CFL; (* The CFL Condition *)
h; u; v; (*The things we are solving for: height, x velocity, y velocity *)
α; β; (* Coefficients in the α-
 β model of communication in MPI. Assume β has units that work with floats naturally *)
γ; (* The time required for one predict-correct time step at a single point *)
δ; (* The time to calculate the speed at a single point *)
```

Derived from these parameters, we can find a few important things:

```
dx = L/M; (* The distance between grid points *)
dt = CFL/Max[u, v] dx; (* The time step (I think. It's something like this) *)
mx = M/Nx; my = M/Ny;
(* Number of grid points belonging to each cell (ignoring any integer nonsense) *)
mg = 4 τ; (* Number of ghost grid cells on one side in one dimension *)
Nt; (* Number of time steps taken. This isn't know a priori,
but it is independent of the number of processors (assuming dx is fixed). However,
it does depend on the CFL condition and dx, so it is in the "derived" category. *)
```

The algorithm that we are performing goes like this:

for i = 1 : Nt/τ

Communicate ghost cells

Find local time step

Allreduce for global time step

Performe the time step operation 2 τ times

To analyze the complexity of the algorithm, we need to think about the cost of each of these operations individually

Finding Time Step Costs

Here, we calculate the cost of doing a block of time steps **per processor**.

---

1. Ghost Cell Communication Cost

The variable we are interested in is:

```
cg; (* Cost of ghost cell communication *)
```

To perform the ghost cell communication, we have to communicate with all four neighbors. The first two communications are to the left and right, and communicate blocks of size ng mx. The last two communications to the top and bottom have to include the corners as well, so they communicate blocks of size ng (my + ng). Plugging this into the α-β model, we have

In[27]:=
```
cg = 2 (α + β mg mx) + 2 (α + β mg (mg + my)) // Collect[#, {α, β}, FullSimplify] &
```

Out[27]= $4\,\alpha + 8\,\beta\,\tau\left(M\left(\frac{1}{Nx} + \frac{1}{Ny}\right) + 4\,\tau\right)$

---

2. Find Local Time Step

The cost of this is simply

In[43]:=
```
clts = δ mx my;
```

---

3. Allreduce for Global Time Step

This is the term that has the worst scaling with the number of processors. Assuming a reasonable algorithm, this should be done via some sort of divide-and-conquer, giving an approximate cost

In[45]:=
```
car = α Log[Nx Ny];
```

---

4. Perform the time-step operation 2τ times

Using the cost to predict-correct for a single node γ, we can find the cost to calculate a layer to be

In[37]:=
```
clayer = γ (mx + 2 k) (my + 2 k);
```

where k is the number of ghost cells that are being calculated for the next step. That is, the first step after communicating, we have to calculate the value at mg – 2 ghost cells in each direction. After that step, we calculate mg – 4, and so on until the final step where we calculate 0 ghost cells. In total, this becomes the sum

In[38]:=
```
cts = Sum[clayer, {k, 0, mg - 2, 2}] // Collect[#, τ, Simplify] &
```

Out[38]=
$$\frac{2\left(3 M^2+8\,\mathrm{Nx}\,\mathrm{Ny}-6 M(\mathrm{Nx}+\mathrm{Ny})\right)\gamma\,\tau}{3\,\mathrm{Nx}\,\mathrm{Ny}}+8\left(-4+M\left(\frac{1}{\mathrm{Nx}}+\frac{1}{\mathrm{Ny}}\right)\right)\gamma\,\tau^2+\frac{128\,\gamma\,\tau^3}{3}$$

As an example, if we communicate every time step ($\tau = 1$), we have:

In[39]:=
```
cts /. τ → 1 // FullSimplify
```

Out[39]=
$$\frac{2\left(M^2+8\,\mathrm{Nx}\,\mathrm{Ny}+2 M(\mathrm{Nx}+\mathrm{Ny})\right)\gamma}{\mathrm{Nx}\,\mathrm{Ny}}$$

Finding Total Cost and Average Cost per Time Step

The total cost of the algorithm comes to multiplying the cost per time step block by the number of blocks, i.e.

In[46]:=
```
Ctotal = Nt/τ (cg + clts + car + cts) // Collect[#, τ, FullSimplify] &
```

Out[46]=
$$\frac{2\,\mathrm{Nt}\left(6 M(\mathrm{Nx}+\mathrm{Ny})(2\beta-\gamma)+3 M^2\gamma+8\,\mathrm{Nx}\,\mathrm{Ny}\,\gamma\right)}{3\,\mathrm{Nx}\,\mathrm{Ny}}+\mathrm{Nt}\left(32\beta+8\left(-4+M\left(\frac{1}{\mathrm{Nx}}+\frac{1}{\mathrm{Ny}}\right)\right)\gamma\right)\tau+\frac{128}{3}\mathrm{Nt}\,\gamma\,\tau^2+\frac{\mathrm{Nt}\left(4\alpha+\frac{M^2\delta}{\mathrm{Nx}\,\mathrm{Ny}}+\alpha\,\mathrm{Log}[\mathrm{Nx}\,\mathrm{Ny}]\right)}{\tau}$$

The average cost per time step would then simply be

In[47]:=
```
cavg = Ctotal/Nt // Collect[#, τ, FullSimplify] &
```

Out[47]=
$$\frac{2\left(6 M(\mathrm{Nx}+\mathrm{Ny})(2\beta-\gamma)+3 M^2\gamma+8\,\mathrm{Nx}\,\mathrm{Ny}\,\gamma\right)}{3\,\mathrm{Nx}\,\mathrm{Ny}}+\left(32\beta+8\left(-4+M\left(\frac{1}{\mathrm{Nx}}+\frac{1}{\mathrm{Ny}}\right)\right)\gamma\right)\tau+\frac{128\,\gamma\,\tau^2}{3}+\frac{4\alpha+\frac{M^2\delta}{\mathrm{Nx}\,\mathrm{Ny}}+\alpha\,\mathrm{Log}[\mathrm{Nx}\,\mathrm{Ny}]}{\tau}$$

Analysis

What is the best τ?

It seems that we should probably ignore terms that are constant in τ, as these cannot be tuned away via τ. So,

In[48]:=
```
cavganalysis =
 cavg - ((∂τ (τ cavg // FullSimplify)) /. τ → 0) // Collect[#, τ, FullSimplify] &
```

Out[48]= $$\left(32\,\beta + 8\left(-4 + M\left(\frac{1}{Nx} + \frac{1}{Ny}\right)\right)\gamma\right)\tau + \frac{128\,\gamma\,\tau^2}{3} + \frac{4\,\alpha + \frac{M^2\,\delta}{Nx\,Ny} + \alpha\,\mathrm{Log}[Nx\,Ny]}{\tau}$$

Translating this big-O notation, we have

In[50]:= $\texttt{cavg}\tau\texttt{BigO} = O[\tau^2] + O\left[\frac{1}{\tau}\right];$

Because the coefficients of both of these terms are positive, there must be some optimal τ that minimizes cavganalysis. The balance is essentially against latency type costs (i.e. terms with α in them and the time to calculate the time step everywhere in the block) and the extra work required to calculate in the ghost cells (scaling for big τ in the τ^2 term). The actual minimum will depend on all of these parameters however, so it is not so pretty to calculate.

Weak Scaling

For the scaling problems, let's assume that Nx = Ny, τ is fixed, and that we only care about the cost of a time step (and not of the total cost). That is,

In[52]:=
```
cscaling = cavg /. Ny → Nx
```

Out[52]= $$\frac{2\left(12\,M\,Nx\,(2\,\beta - \gamma) + 3\,M^2\,\gamma + 8\,Nx^2\,\gamma\right)}{3\,Nx^2} + \left(32\,\beta + 8\left(-4 + \frac{2\,M}{Nx}\right)\gamma\right)\tau + \frac{128\,\gamma\,\tau^2}{3} + \frac{4\,\alpha + \frac{M^2\,\delta}{Nx^2} + \alpha\,\mathrm{Log}[Nx^2]}{\tau}$$

For weak scaling, we go one step farther, and assume that $M = \mathrm{M0}\,\mathrm{Nx}$:

In[54]:=
```
cweak = cscaling /. M → M0 Nx // FullSimplify
```

Out[54]= $$\frac{4\,\alpha}{\tau} + 8\,\mathrm{M0}\,(2\,\beta - \gamma + 2\,\gamma\,\tau) + \frac{\mathrm{M0}^2\,(\delta + 2\,\gamma\,\tau)}{\tau} + \frac{16}{3}\left(\gamma + 6\,\beta\,\tau - 6\,\gamma\,\tau + 8\,\gamma\,\tau^2\right) + \frac{\alpha\,\mathrm{Log}[Nx^2]}{\tau}$$

We see that the cost per processor actually scales with the number of processors, but only logarithmically due to the Allreduce. We do not expect to see this