GPU Hackathon Attendee Guide

A GPU hackathon is a 5-day coding event in which your team of developers will prepare your application(s) to run on GPUs, or focus on optimizing your application(s) that currently run on GPUs. Your team should consist of three or more developers who are intimately familiar with (some part of) your application, and you will work alongside two mentors with GPU programming expertise. The mentors come from universities, national laboratories, supercomputing centers, government institutions, and vendors.

Note, this document assumes a 5-day event held Monday - Friday from 8 AM - 5 PM each day except Friday, where the event officially ends sometime between 1-3 PM.

Preparing for a Hackathon

This section outlines recommendations gathered from participants and mentors over the years on how to prepare for a hackathon. While each team will proceed differently in practice, we find that teams who follow these suggestions typically have the greatest success.

Team Introductions

In the weeks leading up to the event, the local hackathon organizer will send an email introducing your team to the mentors you will be working with. Your team should then schedule a web meeting to address the following topics:

- Team/Mentor introductions
- Discuss code(s) your team will be working on (give code access where appropriate)
- Determine which compute system(s) your team will use during the event
- Assign any action items based on discussions

This initial meeting will help get your team organized and ensure that everyone is on same page for follow-up discussions.

Obtain Access to the Hackathon Compute System

It is beneficial for all of your team members to obtain access to the compute system(s) they intend to use at the hackathon *before the start of the event*. This gives you the opportunity to familiarize yourself with aspects of the system that might be new to you (e.g. different batch scheduler, job launcher, etc.) and also to get your application compiled and running on the system. This helps to ensure your team is ready to start programming GPUs on the system when you arrive for the event, instead of spending time learning how to use the system.

NOTE: It might not always be possible to obtain access to the compute system before the event due to restrictions by some institutions.

Preparing Your Code to Run on the Hackathon Compute System

Although you might be using your application for production work already, that does not ensure it is ready for the work you will be performing at the hackathon. For example, if your code takes four hours to produce results, it will be difficult to test the many incremental changes you will likely be making to your code during the event. To address this point, this section outlines recommendations for preparing your application for the event.

Ideally, your application should be limited to a few thousand lines of code. If you are working with a much larger application, it is preferable to extract specific kernels or a “mini-app” that contains only the relevant parts of the full application whenever possible. This makes the code more manageable for all your team members (including the mentors - who are likely not familiar with the code) and helps to eliminate potential problems with other parts of the code that are unrelated to the work actually being performed. Once you have reduced application running on the GPUs, you can add the changes you made into the full application to understand how it speeds up (or slows down) the application as a whole.

Your code should also be self contained whenever possible. By eliminating external dependencies (e.g. netCDF), you will not need to rely on specific packages (or specific versions of packages) being available on the hackathon system. To do so, you can include any code needed for external dependencies within (or along side) your application. If this is not possible/practical, you should make the dependency known to the local hackathon organizer ahead of the event so it can be installed. In addition, your build system should be free of any specific system dependencies (e.g. Cray computing environment). Removing these external and system-specific dependencies will make it easier to get your code running on the hackathon system.

It is also important to understand any dependencies that might arise due to your choice of programming model. For example, if you plan to use OpenACC to target GPUs, you will likely want to use the PGI compilers (or now possibly later versions of GCC), so you need to make sure your code compiles with PGI. Doing so before the event helps to ensure your time at the hackathon is actually spent on GPU work (not getting your code compiled). Your mentors should be able to help you identify these types of dependencies.

After making any such changes to your code (e.g., using different compiler, extracting kernels, building your own libraries), you should always confirm that your application still compiles, runs, and (of course) gives the correct results. Ideally, this should be done on the system(s) you will be using during the event. This workflow (make changes - compile - run - check results) will be used frequently during the week of the event, so it is important that the process is efficient. In general, you should configure your application to run in ~30s, and only run it on a single process (if possible).

Understanding Your Code and Simplifying Your Workflow

As mentioned above, it is likely that the mentors working with your team are not familiar with your application. So, it is helpful to have a way of describing your application's program flow to them (e.g. call tree, flow chart, etc.). Helping your mentors (and all team members) understand your code structure before the event can make them more efficient and save you a lot of time during the event.

Having a profile of your application, which shows details about how much time is being spent in different regions of the code, allows you to identify the most beneficial regions to accelerate. For example, you want to ensure you are spending your time accelerating parts of your code that account for a sizeable percentage of the total runtime; optimizing regions that only account for 2% of the total runtime will not gain you much. If you need help, your mentors can likely point you in the right direction on generating a profile. Having this available before the event is helpful in making a plan for the week.

Another important aspect of preparation that is often overlooked is having a way to verify correctness of your results. This is an important part of the hackathon workflow, so arriving with an automatic way of doing so (e.g. a correct results file which can be compared against new results with `diff`) can save more time for development. It is not uncommon for a team to get their code optimized and running blazing fast on the GPUs only to find that it is not giving the correct results!

Attending a Hackathon

During the hackathon, your team will work alongside your mentors on the goals your have (hopefully) set during the pre-hackathon meetings. In addition, there are presentations and morning updates that your team will participate in.

Introduction Presentations

On Monday morning, the first order of business will be for one member of each team to give a short (5-7 minutes / 5-7 slides) presentation introducing their team. Ideally, this should include an introduction of your team members, a brief description of your application, a profile of your code (showing compute-intensive regions), and your goals for the week (parts of your code you will be targeting, anticipated speedup, intended programming model - e.g. OpenACC, CUDA, etc.).

Also, remember that most participants are probably not familiar with your specific domain, so please keep the jargon to a minimum.

Morning Updates (Scrum Sessions)

On Tuesday through Thursday mornings, each team will give a short (3-4 minute / 2-3 slides) update, including

- Progress made since last update
- Goals for the day
- Problems you are currently facing
- Problems you have resolved (that other teams might find useful)

In addition to sharing your progress, these updates are good opportunities to get feedback from other teams and mentors. There is a chance that a problem you are facing has already been encountered and resolved by someone else. Or you might have found and resolved a problem (or reported a bug) that other teams might currently be facing. These hackathons are meant to be cooperative events among all participants, and these update sessions are central to that theme.

Final Presentations

On Friday morning, the teams will finish up their development work for the week and give a final presentation (5-7 minutes / 5-7 slides) detailing their accomplishments; issues they ran into, how they resolved them, speedups they obtained, as well as their closing thoughts on the event and what they learned. The final day typically ends around 1-3 PM to allow participants to arrange travel home.

Feedback and Suggestions

We are always looking for ways to improve these events. If you have feedback you believe would help improve this document or the hackathons themselves, please email Tom Papatheodore (papatheodore@ornl.gov) with your suggestions.