Jacket MGL

Back to Jacket Documentation, Jacket HPC

- This Jacket addon is available for purchase.
- The MATLAB® Parallel Computing Toolbox is required for Jacket MGL.

Contents

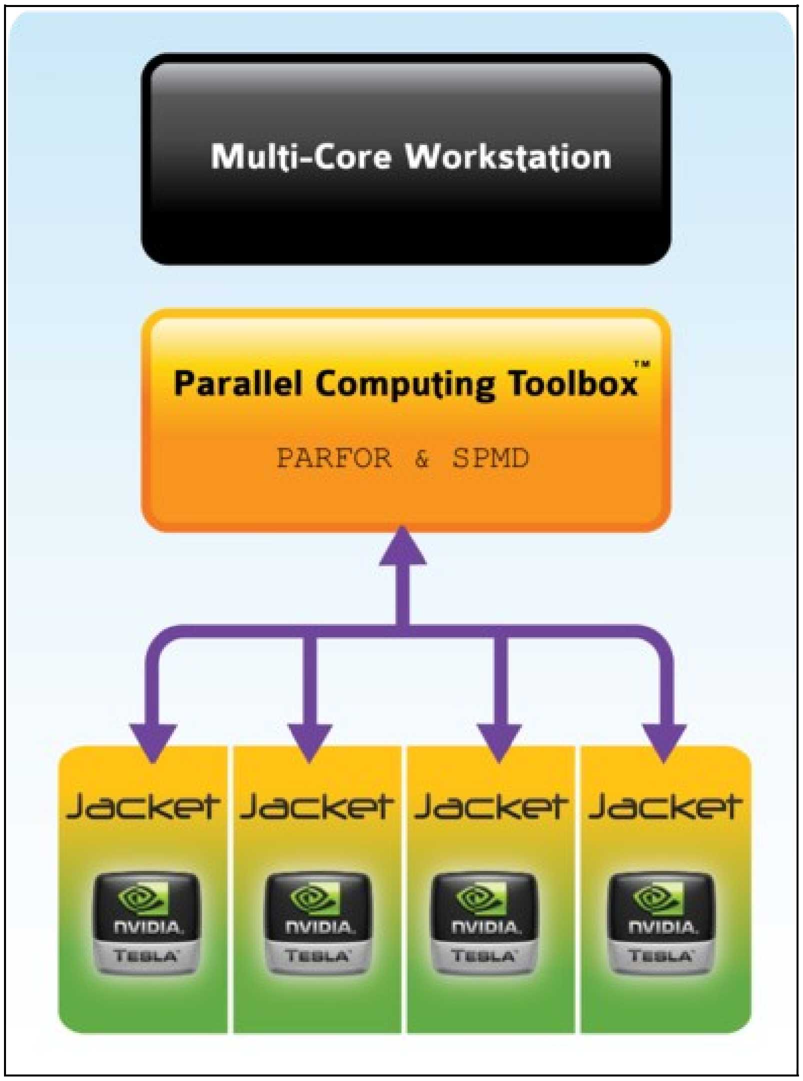
Introduction

Jacket includes functionality to seamlessly utilize multiple GPUs either in the same machine via the MATLAB® Parallel Computing Toolbox. With the simple addition of well known parallel constructs such as PARFOR and SPMD, pre-existing code may be dispatched across all GPUs and CPUs in a single machine. In many cases, little to no code revision is required to take advantage of this new parallel computing capability.

Jacket's ability to span computation across multiple GPUs allows for an unprecedented ability to transparently scale GPU and CPU computing resources. Additional GPUs added to a host may now be instantly utilized without a code modification - simply increment the number of MATLAB workers via the MATLAB command prompt. MATLAB paired with Jacket is the easiest and most scalable solution for GPU computing available.

The Jacket Multi-GPU license **Jacket MGL** enables single node, multi-GPU systems to run Jacket code. It is available with every Jacket trial license, along with an example application available in the installation directory, `<jacket_root>/examples/mgl_example`.

The Jacket High-Performance Computing license Jacket HPC enables multi-node, multi-GPU systems to run Jacket code.



Jacket MGL: Getting Started

Parallel computing in MATLAB is built around the concept of workers. The number of workers is declared by using the command MATLABPOOL. Although a user can create any number of workers, the optimal performance is only achieved when there is one CPU core dedicated for each worker. For example, on a quad-core machine, it is recommended to create four workers only. The similar philosophy applies to workers assignment to GPUs. The total number of workers in the compute pool should be equal to the number of GPUs present in the system.

Requirements

The following requirement exists for Jacket MGL:

- **MATLAB with Parallel Computing Toolbox (PCT)**. MATLAB and PCT provide users the language and tools for programming parallel MATLAB applications. Interested users should make sure they are familiar with the basic use cases supported by the toolbox. MATLAB documentation is the best source of information for PCT documentation.

- **Jacket MGL License**. The Jacket MGL license add-on, with the appropriate number of GPUs.

Testing the setup

In a PCT environment, you can test your configuration with the following example where the host computer is setup as a single node, 2-worker cluster:

```
>> spmd; ginfo; end

Lab 1:
GPU0(enabled) GeForce GTX295, 1212 MHz, 895 MB VRAM, Compute1.3(single)
GPU1(enabled) GeForce GTX295, 1212 MHz, 895 MB VRAM, Compute1.3(single)(in use)

Lab 2:
GPU0(enabled) GeForce GTX295, 1212 MHz, 895 MB VRAM, Compute1.3(single)(in use)
GPU1(enabled) GeForce GTX295, 1212 MHz, 895 MB VRAM, Compute1.3(single)
```

Programming Notes

PARFOR and GFOR are both designed to run many iterations of a FOR loop simultaneously, but they accomplish this in different ways, namely:

- PARFOR: This is a construct that comes with MATLAB's Parallel Computing Toolbox (PCT) and enables splitting computations into different CPU threads (which they call "workers"). This can be accomplished on either a single multicore CPU or via a cluster of connected CPUs. When you think about PARFOR, you should think about CPU-based parallelism. Documentation is here.

- GFOR: This is a construct that comes with Jacket and enable splitting computations amongst the GPU cores. In this process, Jacket automatically tiles out the computations in GPU memory and runs all the iterations in one kernels pass. When you think about GFOR, you should think about GPU-based parallelism. Documentation is here.

Note, that a GFOR loop may be used within a PARFOR loop, but a PARFOR loop may not be used within a GFOR loop.

In order to utilize multiple GPUs, you must have multiple CPU-threads in action, because any given CPU thread can only communicate with a single GPU at a time. Hence, **Jacket MGL** and Jacket HPC require PARFOR to create the multiple CPU-threads. Then each of those CPU threads can use GFOR to exploit the parallelism of each GPU.

For example, `gfor i=1:10000 ...gend` will offload to a single GPU and `parfor i=1:10000 ...end` will offload to multiple CPU cores but will not use any GPUs (unless there are Jacket-specific commands inside the PARFOR loop). Therefore, the PARFOR loop in a 3-GPU machine will not use any of those GPUs and will probably not run faster than GFOR. However, the fastest execution would be achieved with: `parfor i=1:10000 ... gfor i=1:10000 ...gend ...end`.

Programming Limitations

- **Parallel Computing Toolbox (PCT).** There are several programming constructs provided by PCT. Jacket fully supports SPMD and PMODE parallel programming constructs. The PARFOR construct is also supported but with some minor limitations.

- The loop-iterator variable, loop-start and loop-end variables must be MATLAB basic data types and not GPU data types such as GSINGLE or GDOUBLE.
- Any variable created inside a PARFOR loop will be created as a MATLAB type by default, and therefore cannot have a GPU value assigned to it. To solve this problem, it must be pre-allocated prior to the loop. In some cases, it must also be referenced as a value in the loop before the assignment. This is also the standard practice in MATLAB. For example:

```
parfor i=1:10
  n(i) = gsingle(i);
end
```

This will not work because n will be created as a CPU data type. Instead, the following must be done:

```
n = gones(1,10);
parfor i=1:10
  n(i);
  n(i) = gsingle(i);
end
```

The `n = gones(1,10)` pre-allocates the output variable. The `n(i);` causes the PARFOR loop to understand that it is a value from outside the loop and must be brought into the loop (since it's being used), otherwise it tries to assume it is only a temporary value.