Gaussian GPU Notes

Using GPUs

When reading these notes, keep in mind that the GPU nodes on Ocelote have one P100 GPU, 28 cores and 256GB RAM.

1. Gaussian 16 can use the NVIDIA P100 GPUs installed on Ocelote. Earlier GPUs do not have the computational capabilities or memory size to run the algorithms in G16. Allowing larger amounts of memory is even more important when using GPUs than for CPUs, since larger batches of work must be done at the same time in order to use the GPUs efficiently (see below).

2. When using GPUs it is essential to have the GPU controlled by a specific CPU and much preferable if the CPU is physically close to the GPU it is controlling. The hardware arrangement can be checked using the nvidia-smi utility. For example, this output is for a machine with 2 16-core Haswell CPU chips and 4 K80 boards, each of which has two GPUs:

```
      GPU0 GPU1 GPU2 GPU3 GPU4 GPU5 GPU6 GPU7 CPU Affinity
GPU0  X    PIX  SOC  SOC  SOC  SOC  SOC  SOC   0-15
GPU1  PIX  X    SOC  SOC  SOC  SOC  SOC  SOC   0-15
GPU2  SOC  SOC  X    PIX  PHB  PHB  PHB  PHB   16-31
GPU3  SOC  SOC  PIX  X    PHB  PHB  PHB  PHB   16-31
GPU4  SOC  SOC  PHB  PHB  X    PIX  PXB  PXB   16-31
GPU5  SOC  SOC  PHB  PHB  PIX  X    PXB  PXB   16-31
GPU6  SOC  SOC  PHB  PHB  PXB  PXB  X    PIX   16-31
GPU7  SOC  SOC  PHB  PHB  PXB  PXB  PIX  X     16-31
```

The important part is the CPU affinity. This shows that GPUs 0 and 1 (on the first K80 card) are connected to the CPUs on chip 0 while GPUs 2-7 (on the other three K80 cards) are connected to the CPUs on chip 1. So a job which uses all the CPUs (24 CPUs doing parts of the computation and 8 controlling GPUs) would use input

```
%cpu=0-31
%gpucpu=0-7=0-1,16-21
```

or equivalently but more verbosely

```
%cpu=0-31
%gpucpu=0,1,2,3,4,5,6,7=0,1,16,17,18,19,20,21
```

This pins threads 0-31 to CPUs 0-31 and then uses GPU0 controlled by CPU 0, GPU1 controlled by CPU 1, GPU2 controlled by CPU 16, etc.

Normally one uses consecutive numbering in the obvious way, but things can be associated differently in special cases. For example, suppose on the other machine one already had one job using 6 CPUs running with %cpu=16-21. Then if one wanted to use the other 26 CPUs with 8 controlling GPUs one would specify:

```
%cpu=0-15,22-31
%gpucpu=0-7=0-1,22-27
```

This would create 26 threads with GPUs controlled by the threads on

CPUs 0,1,22,23,24,25,26, and 27.

3. GPUs are not helpful for small jobs but are effective for larger molecules when doing DFT energies, gradients and frequencies (for both ground and excited states). They are not used effectively by post-SCF calculations such as MP2 or CCSD.

Each GPU is several times faster than a CPU but since on modern machines there are typically many more CPUs than GPUs, it is important to use all the CPUs as well as the GPUs and the speedup from GPUs is reduced because many CPUs are also used effectively (i.e., in a job which uses all the CPUs and all the GPUs). For example, if the GPU is 5x faster than a CPU, then the speedup from going to 1 CPU to 1 CPU + 1 GPU would be 5x, but the speedup going from 32 CPUs to 32 CPUs + 8 GPUs would be 32 CPUs -> 24 CPUs + 8 GPUs, which would be equivalent to 24 + 5x8 = 64 CPUs, for a speedup of 64/32 or 2x.

4. The GPUs can have up to 16 GBytes of memory and one typically tries to have most of this available for Gaussian, which requires at least an equal amount of memory for the CPU thread which is running each GPU. 8 or 9 GBytes works well if there is 12 GByte total on each GPU, or 11-12 GBytes for a 16GByte GPU. Since the program gives equal shares of memory to each thread, this means that the total memory allowed should be the number of threads times the memory required to use a GPU efficiently. For example, when using 4 CPUs and two GPUs each of which has 12GBytes of memory, one should use 4 x 12 GB of total memory, i.e.

%mem=48gb
%cpu=0-3
%gpucpu=0-1=0,2

(or whatever specific CPU and GPU numbers are appropriate to the machine).

5. GPUs on nodes in a cluster can be used. Since the %cpu and %gpucpu specifications are applied to each node in the cluster, the nodes must have identical configurations (number of GPUs and their affinity to CPUs); since most clusters are collections of identical nodes, this is not usually a problem.