Analysis with Paraver & Dimemas

Methodology

Objective

The objective of this session is to present some examples of analysis methodology supported by the configuration files.

This methodology should in the future be automated by integrating Paraver and Dimemas with profile presentation tools and rule based systems.

Distribution of configuration files

An analysis methodology is essentially a tree of measurements to perform that lead us to a good understanding of the behavior of our application or the identification of bottlenecks. A typical search will iterate through successive sets of hypotheses and validations down the tree till we understand the behavior or have no further information in the available data to validate an hypotheses.

The standard Paraver distribution contains a large set of directories and configuration files. At the outermost levels, directories try to group configurations based on the type of information they look at. Internally we typically separate directories for configuration files that that show a specific timeline view and directories for configuration files that compute a profile or histogram. The major directories are:

- General:
- MPI: targeting the measurement of the MPI behavior.
- OpenMP:
- ...
- Methodology: this directory tries to group the basic cfgs that should be used as as initial steps of an analysis.

Methodology: basic profiles

In this section we will present some recommendations on how to proceed in an analysis.

The first question to address is what is the parallel efficiency and whether it is determined/limited by load balance or communication.

- Load **Iberia-128-CA.chop1.1it.shifted.prv.gz** and then **cfgs/methodology/parallel_efficiency.cfg**.
- The entry "Average" towards the end of the second column tells you how much **parallel efficiency** you are getting. 100% would be ideal. 90% would mean you are loosing 10% of potential performance.
- The entry "Maximum" in this same column measures **communication efficiency**, how much of that performance loss is due due to MPI. A value of 100% means that MPI and communication in general

is NOT limiting the application performance. a value of 90% means that you are loosing 10% because of the communication

- The value at the end of the second column (Avg/Max) actually measures the degree of global **load imbalance** in the application. A value of one is perfect load balance. A value of 0.9 means you are loosing 10 % of performance with respect to an ideally balanced execution. You can see the distribution of the percentage of useful computation by the different threads in the corresponding entries of this column.
- The parallel efficiency is actually the product of these two factors.

If the application does show some load imbalance, you may wonder whether it is computational load imbalance or due to other factors.

- Load **cfgs/methodology/instructions_profile.cfg**. The entry "Avg/Max" at the end of the second column tells you how well is the computational load (number of instructions) balanced across processors. A value of. 1 would represent ideal balance. A value of 0.9 would mean you would be loosing 10% of potential performance if IPC was uniform across all processes due to computational load imbalance.

An important metric to look at reports the performance of the sequential computation phases. If the parallel efficiency is good, this may be the limiting factor. If it is not good, imbalances in IPC may cause the imbalance in execution time. Sometimes, imbalance in IPC may compensate computational load imbalance.

- Load **cfgs/methodology/IPC_profile.cfg:** The second column reports the average IPC achieved by each process in the computation bursts. In powerPC970 case, values above 1 are reasonably good.
- The entry "Avg/Max" measures the imbalance in IPC. A value of 1 is ideal. a value of 0.9 would mean you would loose 10% of potential performance if the application was perfectly balanced fro the computational point of view.

If the IPC is not good or well balanced, you may want to look at cache misses.

If the communication time seems to be a problem it may actually not be due to communication but to local (or microscopic) load imbalances or serialization. In order to identify this effect, a Dimemas simulation is required. You will need to convert the file to .dim, and simulate with an ideal target architecture.

- Load **cfgs/methodology/parallel_efficiency.cfg** on the trace generated with a the **ideal Dimemas simulation**. The entry "Maximum" in the second column specifies the actual value of Microscopic load balance.
- The actual value of communication efficiency can be computed by dividing the elapsed time of the ideal Dimemas simulation to the elapsed time of the original Paraver trace.

Methodology: detailed profiles

The above metrics are computed for the whole program. You may want to have the same metrics for different computational phases. You will need to cluster the paraver trace and **load it**.

- Configuration file **cfgs/methodology/2dp_cid.cfg** will show the same type of information as the parallel-efficiency file, but now each column will represent on of the relevant computation phases of the program (clusters)..

- If you change the statistic to “Sum Bursts” and the Data Window to Instructions and repeat the analysis you will get the computational load balance for each cluster in the “Avg/Max” entry.
- If you change the statistic to “Average value” and the Data Window to instructions per cycle, you will get the IPC imbalance for each cluster in the “Avg/Max” entry.

Methodology: histograms

If parallel efficiency is bad due to load imbalance and you want to know how is that load imbalance distributed and where it shows up.

- Load **cfgs/methodology/computation_duration_histogram.cfg:** Ideally, vertical lines should appear, showing that all processes spend take same time for all computation phases. If you pop up the control window you will see the time distribution. If you are interested on a special range of durations where the histogram shows a wide stripe where the duration of a computation phase is different for different processors you may click on the “Open Control Window Zoom 2D” and select the range of durations and processors you are interested. You will get a timeline for only the range of durations and processors you have selected.

If parallel efficiency is good, you may want to look at how the IPC distributes along the different computation phases

- You may also load **cfgs/methodology/IPC_histogram.cfg** This is a histogram of the IPC of the useful computation phases. The reason is that different phases may have different IPCs and may be interesting to identify poorly performing phases even if the average is good. You can pop up the useful_IPC timeline (control window) to see the distribution along time.