KapitanPOP Manual

Oleksandr Rudyy

June 24, 2020

1 Metrics Computation

This section accurately describes how KapitanPOP computes its metrics from the data contained in the HDF5 trace file. First, it starts with a depiction of how raw metrics are extracted, i.e. *Useful Computation Time*, evolving gradually to the higher level metrics, like *Parallel Efficiency*. Before continuing, though, it is convenient to give a solid definition to some very used terms in order to avoid confusions.

- **Useful**: this adjective refers to any value belonging to the *Running* state defined by Extrae. In other words, *useful* is anything that belongs to user code, or in other words, that is not part of system's -the kernel- or runtime's -OpenMP, pthreads, MPI- code. Input/Output or I/O can also be considered as *useful*, this will depend of how Extrae was configured when tracing.

- **Maximum**: refers to the maximum existing value when grouping the data by application, process and thread. For example, *Maximum Time in MPI* is the maximum time one process -MPI rank- has spent inside MPI compared to the rest of processes.

- **Runtime**: is the elapsed time since the start of an application to its end as recorded by Extrae.

- **Ideal**: this adjective refers to any value that has been computed from a Dimemas trace with ideal network -zero latencies, infinite bandwidth.

- **Reference**: is used for scalability purposes and refers to the metric coming from the run with less number of cores.

Raw Metrics

- Runtime
- Useful Compute Time
  - Total Useful Compute Time
  - Maximim Useful Compute Time
  - Average Useful Compute Time
- Total Useful Instructions
- Total Useful Cycles
- Ideal Times
  - Ideal Runtime
  - Maximum Ideal Useful Compute Time

Runtime

The *runtime* is extracted from Paraver's header, thus it is already provided and not computed.

Useful Compute Time

The Useful Compute Time is the time the application spends running user code. It is computed from *States* records where state is equal to *Running*.

$$\begin{gathered}\text{Total Useful Compute Time} = \\ \sum StateRecord_{time\_end} - \sum StateRecord_{time\_start} \\ \text{where } StateRecord_{state} = \text{Running}\end{gathered} \tag{1}$$

$$\begin{gathered}\text{Max. Useful Compute Time} = \\ \max(\sum StateRecord_{time\_end} - \sum StateRecord_{time\_start}), \\ \text{where } StateRecord_{state} = \text{Running, groupby(appID, taskID, theadID)}\end{gathered} \tag{2}$$

$$\begin{gathered}\text{Average Useful Compute Time} = \\ \text{Total Useful Compute Time} \; / \; \text{number of threads}\end{gathered} \tag{3}$$

Total Useful Instructions

$$\begin{gathered}\text{Total Useful Instructions} = \\ \sum EventRecord_{EvValue}, \\ \big(\text{where } EventRecord_{EvType} = PAPI\_TOT\_INS \wedge \\ EventRecord_{time} = StateRecord_{time\_end}, \text{where } StateRecord_{state} = \text{Running}\big) \\ \text{groupby(appID, taskID, theadID)}\end{gathered} \tag{4}$$

Total Useful Cycles

$$\begin{gathered}\text{Total Useful Cycles} = \\ \sum EventRecord_{EvValue}, \\ \big(\text{where } EventRecord_{EvType} = PAPI\_TOT\_CYC \wedge \\ EventRecord_{time} = StateRecord_{time\_end}, \text{where } StateRecord_{state} = \text{Running}\big) \\ \text{groupby(appID, taskID, theadID)}\end{gathered} \tag{5}$$

Ideal Times

Ideal Runtime and *Maximum Ideal Useful Compute Time* are computed the same way as *Runtime* and *Maximum Useful Compute Time* with the difference that the data used comes from a Dimemas trace simulating an ideal network.

1.1 Multiplicative Performance Metrics

- Parallel Performance Metrics
  - Load Balance
  - Communication Efficiency
  - Transfer Efficiency
  - Serializtion Efficiency

  - Parallel Efficiency

- Global Efficiency

- Scalabilities

  - IPC Scalability
  - Frequency Scalability (GHz)
  - Strong
    * Speedup
    * Computation Scalability
    * Instruction Scalability
  - Weak
    * Load Increase Factor (LIF)
    * Speedup
    * Computation Scalability
    * Instruction Scalability

- Serial Performance Metrics

  - Average IPC
  - Average Frequency

Parallel Performance Metrics

$$\begin{gathered}\text{Load Balance} = \\ \text{Average Useful Compute Time} \;/\; \text{Max. Useful Compute Time} * 100\end{gathered} \tag{6}$$

$$\begin{gathered}\text{Communication Efficiency} = \\ \text{Max. Useful Compute Time} \;/\; \text{Runtime} * 100\end{gathered} \tag{7}$$

$$\begin{gathered}\text{Transfer Efficiency} = \\ \text{Ideal Runtime} \;/\; \text{Runtime} * 100\end{gathered} \tag{8}$$

$$\begin{gathered}\text{Serialization Efficiency} = \\ \text{Communication Efficiency} \;/\; \text{Transfer Efficiency} * 100\end{gathered} \tag{9}$$

$$\begin{gathered}\text{Parallel Efficiency} = \\ \text{Load Balance} * \text{Communication Efficiency}/100\end{gathered} \tag{10}$$

Global Efficiency

$$\begin{gathered}\text{Global Efficiency} = \\ \text{Parallel Efficiency} * \text{Computation Scalability}/100\end{gathered} \tag{11}$$

Scalabilities

$$\begin{gathered}\text{IPC Scalability} = \\ \text{Average IPC / Reference Average IPC} * 100\end{gathered} \tag{12}$$

$$\begin{gathered}\text{Frequency Scalability (GHz)} = \\ \text{Average Frequency / Reference Average Frequency} * 100\end{gathered} \tag{13}$$

Strong

$$\text{Speedup} = \text{Reference Runtime / Runtime} \tag{14}$$

$$\begin{gathered}\text{Strong Computation Scalability} = \\ \text{Reference Total Useful Compute Time/Total Useful Compute Time} * 100\end{gathered} \tag{15}$$

$$\begin{gathered}\text{Strong Instruction Scalability} = \\ \text{Reference Total Useful Instructions/ Total Useful Instructions} * 100\end{gathered} \tag{16}$$

Weak

LIF (Load Increase Factor) is the factor by which the load (amount of work) increases. Ideally, it should be equal to the increase in number of processes/threads.

$$\begin{gathered}\text{LIF} = \\ \text{Number of threads / Reference Number of Threads}\end{gathered} \tag{17}$$

$$\text{Speedup} = \text{Reference Runtime / Runtime} * \mathit{LIF} \tag{18}$$

$$\begin{gathered}\text{Weak Computation Scalability} = \\ \text{Reference Total Useful Compute Time / Total Useful Compute Time} * \mathit{LIF} * 100\end{gathered} \tag{19}$$

$$\text{Weak Instruction Scalability} = \\ \text{Reference Total Useful Instructions} \;/\; \text{Total Useful Instructions} * LIF * 100 \tag{20}$$

Serial Performance Metrics

$$\text{Average IPC} = \\ \text{Total Useful Instructions} \;/\; \text{Total Useful Cycles} \tag{21}$$

$$\text{Average Frequency (GHz)} = \\ \text{Total Useful Cycles} \;/\; \text{Total Useful Time (ms)}/1000 \tag{22}$$