Document :

**Abstract:**

Customised implementation of GUI for SU2 with measurement to optimise the topology of the application.

**Description:**

Interconnected systems with the parallel threading framework running in multiple cores between the shared systems and one core used for GUI performance which displays the performance of the real time application SU2. The GUI also allows the probing for different analysis for SU2 application and displays the efficiency and performance of parallel computation and how the performance changes when we increase no.of nodes.

**Application and Optimisation:**

**SU2 Fluid Dynamics**

**Experiment:**

Two laptops are inter connected via crossover cable using SSH. Laptop contains 4 cores with 2 threads per core, so conceptually 8 CPUs work in both Laptops.one core will be used for running the GUI (made using Python3 the Tkinter Libraraies) which will display various metrics of the application say the parallel threads running,computational time,latency, and various metrics related to the application. Options present in the GUI will help in setting the priorities of the threads to observe the differences as well as see the share of CPU and Memory usage.

Input files format :

**SU2 (Stanford University Unstructured )**

**Configuration Files:**

The configuration file is a text file that contains a user’s options for a particular problem to be solve with the SU2 suite. It is specified as an input upon execution of SU2 components.The SU2 configuration file name typically carries a name of the form filename.cfg. The file extension .cfg is optional (this is our own convention), and the prefix can be any valid string with no spaces;

**Ex:**config.cfg, su2-config.cfg, and flow\_config.cfg are all suitable file names.

The configuration file consists of only three elements:

**Options :** An option in the file has the following syntax: option\_name = value, where option\_name is the name of the option and value is the desired option value. The value element may be a scalar data type, a list of data types, or a more complicated structure. The “=” sign must come immediately after the option\_name element and is not optional. Lists of data types may be formatted for appearance using commas, ()-braces, {}-braces, and []-braces, though this is not required. Semicolons are semantically relevant for several option types and may not be used as convenience delimiters. SU2 will exit with an error if there are options in the config file which do not exist or if there are options with improper formatting.

**Comments :**On a given line in the file, any text appearing after a % is considered a comment and is ignored by SU2. Additional % signs after the first on a given line are not significant.

**White space :**Empty lines are ignored. On text lines that define options, whitespace (tabs,spaces) can be used to format the appearance of the file.

**Mesh Files :**

SU2 mainly uses a native mesh file format as input into the various suite components. Limited support for the CGNS data format has also been included as an input mesh format. CGNS support can be useful when it is necessary to create complex geometries in a third-party mesh generation package that can export CGNS files. A converter from CGNS to the nativeformat is also built into SU2.

**SU2 Native Format (.su2)**

In keeping with the open-source nature of the project, SU2 relies mostly on its own native mesh format. The format is meant to be simple and readable. A description of the mesh and some examples are below.

**Description :**

The SU2 mesh format carries an extension of .su2, and the files are in a readable ASCII format. As an unstructured code, SU2 requires information about both the node locations as well as their connectivity. The connectivity description provides information about the types of elements (triangle rectangle, tetra hedron, hexahedral, etc.) that make up the volumes in the mesh and also which nodes make up each of those elements. Lastly, the boundaries of the mesh, or markers, are given names, or tags, and their connectivity is specified in a similar manner as the interior nodes.)

Output file format :

**Tecplot :** Tecplot supports an ascii input file format.The first and most basic data type is the ASCII DAT data format (\**.dat*). This is a simple human-readable or -writeable format and is perfect for smaller data sets when working between multiple applications. However, when data sets become large, the inefficiency of ASCII data storage quickly becomes apparent and a binary format should be considered.

**Paraview :**Is a graphical user interface based on VTK, hence, data can be processed without writing code. Since Paraview implements a client-server architecture around the VTK modules, not all VTK functions is supported.

\*.vtk file format uses ASCII header lines that separate the data sections. The data itself can be ASCII or binary.

Software used:

1.Two laptops windows or linux

2.GUI python3 and Tkinter

3.OpenMP/MPI

4.Tecplot and Paraview

5.SSH Communication