SJSU CmpE 275, Fall 2015 Project 1 – Parallel execution for data mining

Project: Climatological Wind Rose generation for Data Mining

The use of historical meteorological data in forecast and research have access to large accumulated data sets. Working with or processing these large data sets is compute intensive. One tool (plot) that is useful to understanding wind speed and direction at a location is the Wind Rose. Calculation of a Rose uses 24 hours of observation data at a location to create a polar plot of the wind direction, magnitude, and frequency. This can be a highly serial task.

Your challenge is to analyze and propose a solution to parallelize data extraction for generating individual Wind Rose plots as well as creating climatological analysis using Wind Rose plots that describe mean conditions for a given day or search data for 'wind persistence' between to arbitrary points. E.g., given point A (lat,lon) find days that the wind would allow a particle to reach point B (lat,lon).

Technical Objectives

The objectives of this project is to look at the effects of serial and parallel (intra-process) performance. This can be extended to inter-node, message passing concepts through domain decomposition (OpenMPI or similar standards/libraries) and/or on-node shared memory. Secondly, your team will need to provide a binding toExpansion of your team's project into deeper explorations and implementations is always supported and encouraged. In determining your team's goals be sure to include requirements described in this write up or discussed in lecture.

Be creative, and take the time to understand what is occurring both technically and conceptually.

Languages, Packages, and Tools

There are a wide range of technologies and tools at your disposal, you should research and determine which tools and libraries are correct for your team's goals. Since the goal of the project to to focus on loop-level parallelization with possibilities of task-based and distributed messaging in a C++ language you should minimally include OpenMP v3 or v4. OpenMP is built into most modern C++ compilers (Clang, gnu, intel, MS).

Computationally, compiled code under most conditions offer significant performance advantages over interpretive and machine portable code (e.g., byte-code, VMs). These portable languages are popular and have a rich base of tools, libraries, and community following. In order for your project to integrate well into this community, you will want to provide a language binding to call your compiled library directly from one of these more-modern languages (e.g., Python, Java, Lua).

References

SJSU CmpE 275, Fall 2015 Project 1 – Parallel execution for data mining

http://sourceforge.net/projects/windrose/
Mesowest data - http://mesowest.utah.edu
MADIS Mesonet archive - https://madis-data.noaa.gov/madisPublic1/data/archive