Performance Evaluation of WRF Meteorology Model in Parallel Processing Systems

MohammadReza Majma
Islamic Azad University, Pardis Branch
m_majma@pardisiau.ac.ir
Hossein Pedram
Amirkabir University of Technology, Computer Engineering Department
pedram@aut.ac.ir
Bijan Fallahi
National Cloud Seeding Research Center (NACSER)
bijan.fallah@gmail.com
Sanaz Almassi
Iran University of Science and Technology, Computer engineering Department
Sanazalmassi@comp.iust.ac.ir
Paper Reference Number:
Name of the Presenter: Mohammd Reza Majma

Abstract

Parallel processing and distributed systems, have efficient role in different sciences such as meteorology, medicine, nuclear, physics, chemistry, etc. These sciences need parallel processing systems based on cluster to solve and run their algorithms with a higher speed. Some tools and models exist for parallel programming such as shared memory, message passing, oriented model, hybrid model, java, but message passing by MPI library plays an essential role in cluster systems. In this paper, we have presented WRF model with a message passing interface. We have implemented parallel algorithm of this model in a real cluster and evaluated the results in comparison with different forecasting models. Experimental results demonstrated that by increasing the number of processors to 24, run time decreases from 2500 seconds to 600 seconds, however the performance decrease to 17%.

Key words: Cluster, Interface WRF model, Parallel Processing, Message Passing, Meteorology.

1. Introduction

By arising of computers, human beings tried to achieve more processing power and nowadays processing systems have a serious role in different sciences such as: meteorology, medicine, nuclear, physics, and chemistry. Experts use different tools to achieve this goal, such as using VLSI technology, architecture techniques to increase processing speed such as vector processors and super scalar and taking advantage of several processors to execute a task.

Cluster computing[1] which means using two or more computers together to achieve flexibility, scalability and more processing power is not a new field of calculation; however it is mostly being used when users frequently need parallel and distributed calculations[2].

The requisite to improve and upgrade the components of a parallel processing system in one hand and the speed of software components with high performance and availability one the other hand, makes these systems an effective tool which is greatly used in physics and Meteorology; in a manner that many algorithms in these fields are being executed in cluster environments to be solved with a higher speed.

Meteorology models witch are powerful and exact tools to forecast weather and atmosphere conditions have the same characteristics to use cluster computing facilities to speed up. In this paper we have run, examined and evaluated an exact meteorology forecasting model, called WRF on a real distributed computing system. In the second section of this paper, we will investigate parallel and cluster systems and parallel programming model will be discussed. In third section WRF meteorology model will be introduced and in the final section, this model will be evaluated based on achieved advantages on a real cluster.

2. Approach to Parallelism

2.1. Cluster Computing Systems

A cluster is a type of parallel or distributed processing system, which is consist of a collection of interconnected stand-alone computers cooperatively working together as a single, integrated computing resource. Their taxonomy is based on how their processors, memory, and interconnect are laid out. The most common systems are [3]:

- *MPP (Massively Parallel Processing):*

An MPP is usually a large parallel processing system with a shared-nothing architecture. It consists of several hundred processing, interconnected through a high-speed interconnection network/switch. These systems have high cost and low performance/price ratio.

- *SMP (Symmetric Multi-processor):*

SMP systems today have from 2 to 64 processors and can be considered to have shared-everything architecture. In these systems, all processors share all the global resources available (bus, memory, I/O system); a single copy of the operating system runs on these systems. These systems have a low performance/price ratio but suffer from scalability.

- *CC-NUMA (Cache-Coherent Non uniform Memory Access):*

CC-NUMA is a scalable multiprocessor system having cache-coherent non uniform memory access architecture. Like an SMP, every processor in a CC-NUMA system has a global view of all of the memory. This type of system gets its name (NUMA) from the non-uniform times to access the nearest and most remote parts of memory.

- *Distributed systems:*

Distributed systems can be considered conventional networks of independent computers. They have multiple system images, as each node runs its own operating system. The individual machines in a distributed system could be combinations of MPPs, SMPs, clusters, or individual computers. They have low performance in computation and parallel processing.

- *Cluster Systems:*

For parallel computing purposes, a cluster will generally consist of high performance workstations or PCs interconnected by a high-speed network. The main feature of cluster is to gather high performance and High availability computing together.

Based on distributed processing power in clusters, high availability and fault tolerance are important parameters to keep performance on a pleasant level. In these systems, failure of a node can lead to other processes and even the whole task’s failure. Here check pointing and process migration are the solutions of this problem. The general application of check pointing and process migration has shown its increasingly important role in fault tolerance. Check pointing approaches save the application state to reliable storages periodically. However Periodic recording is costly in both accessing time and storage space.

Process migration [4] is an optimized form of check pointing with redistribution of processes and immediate restart. Process migration includes providing higher availability, simplifying reconfiguration, and utilizing special machine capabilities.

In some cases such as MPICH-V2 [5] when a fault occurs, process recovers in initial node by Check pointing mechanism. But in real systems, if the node fails, the process cannot recover and fails. So the task stops. To prevent this problem, process migrates to news destination by dynamic method [6] or by intelligent agents [7]. Thus, the process will continue to be available even after machine failure or disconnection.

2.2. *Message Passing Interface*

Programs need resources and due to architecture and physical limitations and the necessity of higher processing speed, parallel processing is massively needed. Some tools and models have been presented for parallel programming such as shared memory, message passing, oriented model, hybrid model, java [8]. In this criterion, message passing is the most efficient, widely used, programming paradigm on distributed memory systems.

MPI is a useful library used in cluster systems; this library helps programmers to change their programs from serial to parallel form with a standard structure. In MPI the whole transmission is done by programmer and the compilers do nothing. Although there is other libraries such as Madeleine III and MPICH-G2 witch are compared in [9].

We should mention that parallelism performance is limited to base algorithm. It means that if the main algorithms do not have the capability to be parallelized, In this case parallelism has no effect on performance and speed up. Fortunately WRF model's algorithm which is mentioned and presented in this paper could be parallelized with MPI. As we mentioned in this section, the main problem in cluster systems is interconnection. So based on different levels of parallelism, programs are mostly parallelized in large grain scale. In [10] we parallelized Monte Carlo Algorithm with MPI-1 in large grain level.

3. *WRF Meteorology Model*

The development of the Weather Research and Forecasting (WRF) modeling system is a multi-agency effort intended to provide a next-generation mesoscale forecast model and data assimilation system that will advance both the understanding and prediction of mesoscale weather and accelerate the transfer of research advances into operations [11]. The model is being developed as a collaborative effort among the following organizations:

- *NCAR* Mesoscale and Micro scale Meteorology (MMM) Division [12]
- *National* Oceanic and Atmospheric Administration's (NOAA) [13]
- *National* Centers for Environmental Prediction (NCEP) [14]

This model is fully compressible, flexible and effective in parallel computing and can be used in scales of hundred meters to thousands of kilometers. It could be used in ideal experimental cases as well. Parameterization of atmospheric events, data assimilation, forecast research, real-time NWP, regional climate research, hurricane research, coupled-model applications and teaching are other aspects of WRF model.

3.1. WRF Model Architecture

WRF model consists of several operational programs such as real.exe and ideal.exe and an integrating program wrf.exe and a nesting program ndown.exe. Geographical and atmospheric data as the first guess will be prepared by WPS. This is a none-hydrostatic model (with hydrostatic option) with horizontal Arakawa-c grid staggering. Second and third order Ronuge-kutta time integration scheme and second to sixth order schemes for convection in vertical and horizontal gird are used. Time-split small step for acoustic and gravity-wave modes are used as well. The main core of WRF model is consist of several primary executing programs such as ideal.exe, real.exe and wrf.exe which is used for numerical integration and ndown.exe for one-way nesting [15] .different components of model is illustrated in (fig.1).

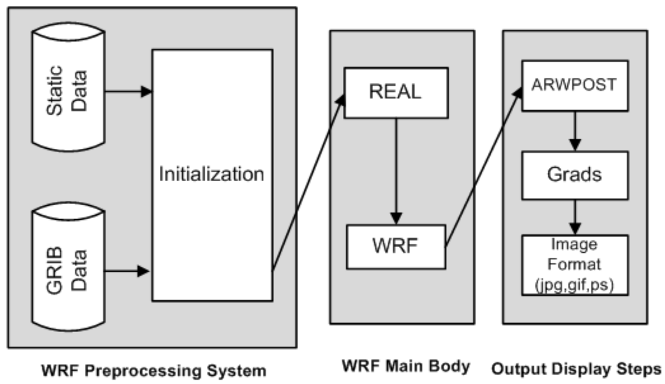

Fig 1: WRF model components

The equations (1-6) are cast in flux form using variables that have conservation properties, following the philosophy of Oyama [16]:

$$F_U = \partial_t U + (\nabla . Vu) - \partial_x(p\phi_\eta) + \partial_\eta(p\phi_x) \quad (1)$$

$$F_V = \partial_t V + (\nabla . Vv) - \partial_y(p\phi_\eta) + \partial_\eta(p\phi_y) \quad (2)$$

$$F_W = \partial_t W + (\nabla . Vw) - g(\partial_\eta p - \mu) \quad (3)$$

$$F_\Theta = \partial_t \Theta + (\nabla . V\Theta) \quad (4)$$

$$0 = \partial_t \mu + (\nabla . V) \quad (5)$$

$$0 = \partial_t \phi + \mu^{-1}[(V . \nabla\phi) - gW] \quad (6)$$

The right-hand-side (RHS) terms FU, FV, FW, and F_θ represent forcing Terms arising from model physics, turbulent mixing, spherical projections, and the earth's rotation. Φ is geo potential;μ is mass per surface and θ is potential temperature.

Defining the prognostic variables in the ARW solver as $\phi=(U,V,W,\Phi,\mu',\phi',Q_m)$and the model equations as $\phi_t = R(\phi)$, the RK3 integration takes the form of:

$$\Phi^* = \Phi^t + \frac{\Delta t}{3} R(\Phi^t) \quad (7)$$

$$\Phi^{**} = \Phi^{t} + \frac{\Delta t}{2} R(\Phi^{*}) \quad (8)$$

$$\Phi^{t+\Delta t} = \Phi^{t} + \Delta t R(\Phi^{**}) \quad (9)$$

Where Δt is the time step for the low-frequency modes (the model time step. more information about algorithms and order schemes used in WRF model are discussed in [14].

In order to use WRF for forecasting, one should train it for the specific region of interest. Besides using the Global Models such as GFS (Global Forecast System) or FNL (Final), station, radar, satellite, upper-level stations and other data must be used in data assimilation to get the best results [17]. The next step would be Ensemble Systems to achieve the best forecasts. All these processes could not be operational without considering the high performance processors.

3.2. WRF implementation in parallel mode

We Build a commodity cluster consist of three SMP nodes with star topology to run WRF Model on it. Each node of this cluster have two 2.2 GHZ Xeon processors with 8 kernels, full cache and three memories of 2 GB capacity. Theses nodes are being connected with a Giga Ethernet network. The switch is used in network's third layer and is manageable. We have used Red hat Linux 5.3 (RHEL 3.5) with core version 2.6. We have also used 2.1.3.2 MPICH version.

To run the model before installation, operating system must be ready. The listed software must be installed on the system: FORTRAN compiler or C compiler, Pgilinux86-711, mpich2-1.0.7, netCDF-3.6.2, HDF5-1.8.1, jasper-1.701, ncarg-4.4.1, zlib-1.2.3, Perl5.04, ARWpost, Grads-2 and RIP4. Then we installed WRF model and all its components, such as post processing system, ARW post and preprocessing system, WPS. To visualize the output, Vis5D, NCL, Grads and other available software have be used either.

4. Results and Analysis

We ran WRF model on different processors and kernels and the results are illustrated in (fig. 2).

As it's distinguished in the figure, by increasing the number of processors, the algorithm executed with a higher speed. This speed up has direct relation with the number of processors, but this relation is not linear. It means that by increasing the number of processors, the speed do not increase in the same manner.

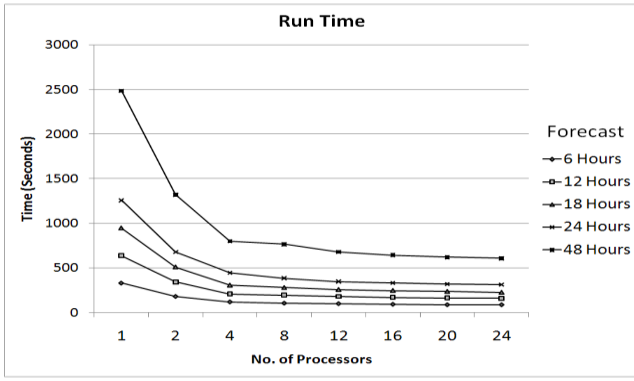

Fig 2: the comparison of run time in different situations

We evaluated speed up as compared with number of processors and the results are being shown in (fig.3). The results indicate that by using two processors instead of one, the speed increases to 1.8; but when the number of processors is more than 16, the speed increases to less than half fold.

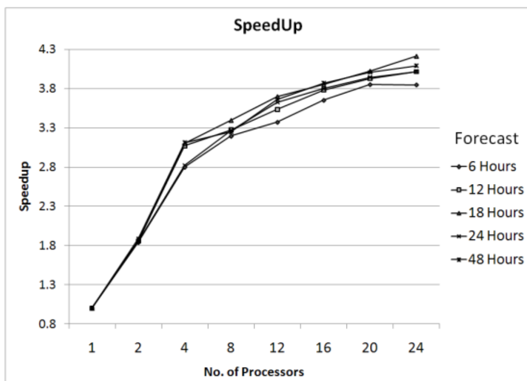

Fig 3: Speed up diagram

Using Amdahl rule in equation (10), we calculated optimized time during increasing the number of processors.

$$S(n) = \frac{S(1)}{1 - f + \frac{f}{n}} \tag{10}$$

In this equation S(1) is the program running speed with one processor and S(n) is the same program running speed, while running simultaneously on n parallel processors. F parameter is the percentage of program code witch could be run in a parallel form. In WRF model, the whole code have the capability to be paralleled; witch means that f=100%. Afterwards we calculated standard derivation for ideal and normal real time with equation (11) and the results are shown in figure 4.

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2} \tag{11}$$

In this equation is the standard derivation and is the data average which is estimated from equation 12.

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i = \frac{x_1 + x_2 + \cdots + x_N}{N} \qquad (12)$$

As it is observed in (fig. 4), the standard derivation in ideal and real time in 6 hours of forecast is approximately the same. The run time and node communication in this forecast is low, hence the network overhead decreases and that's why the results are almost the same.
As the forecasting time increases, standard derivation and run time averagely increases and the node communication and network overhead gets higher. This fact leads to variance between ideal and real time, so that in 48 hours of forecast, the standard derivation is about 800 seconds, where the ideal time is 600 seconds. This variance greatly appears when the model runs on 16 or more processors. We can decrease this variance by increasing band width and using LAN Teaming. Although the variance rate don't turn into zero but approximately decreases.

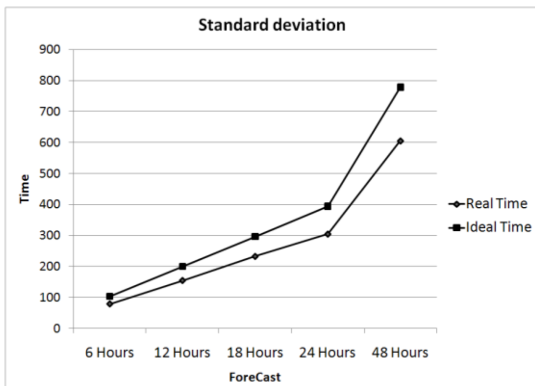

Figure 4: the standard derivation comparison between real and ideal time

We calculated performance by dividing speed up to number of processors. As it's obvious in table.1 by increasing the number of processors in 48 hour of forecast, performance decreases up to 17%. The performance deduction is because of intercommunication of nodes and processes and using most system's resources. To achieve more performance almost 3.4 of system resources should be used, although this strategy leads to more run time and cost.

No. of process	Forecast				
	6 Hours	12 Hours	18 Hours	24 Hours	48 Hours
1	100%	100%	100%	100%	100%
2	91.96%	92.84%	93.01%	92.58%	94.15%
4	69.97%	76.73%	77.58%	70.57%	77.77%
8	39.93%	40.91%	42.46%	40.84%	40.58%
12	28.10%	29.44%	30.84%	30.22%	30.55%
16	22.82%	23.63%	24.06%	23.75%	24.19%
20	19.26%	19.64%	20.13%	19.73%	20.03%
24	16.04%	16.73%	17.56%	16.74%	17.05%

Table 1. Performance comparison in different forecasts

5. Conclusions

In this paper, we ran WRF witch is a new model in meteorology, using message passing interface in parallel processing system based on cluster. We designed and implemented a commodity cluster to achieve maximum performance and implemented different middle wares on it. We designed parallelism in task level with large grain scale. Then by reducing node communications, we used the most of processing ability of processors to run the model. Running this algorithm is based on parallel programming model and MPICH2 library.
The main goal to run this meteorology model on designed commodity cluster is to optimize run time and achieve maximum speed up. So we distributed task overhead on different processors and kernels. By increasing the number of processors in 48 hours of forecast, the algorithm run time decrease from 2485 seconds to 607 seconds. Although the system's performance decrease to 17.05%.

References

[1] Amy Aporia, Rajkumar Buyya, Hai Jin, and Jens Mache, "Cluster Computing in the Classroom: Topics, Guidelines, and Experiences," IEEE, 2001.
[2] Katsuki Fujisawa, Masakazu Kojima, Akiko Takeda, Makoto Yamashita, "High Performance Grid and Cluster Computing for Some Optimization Problems," *Proceedings International Symposium on Applications and the Internet Workshops*, 2004.
[3] R. Buyya, *High Performance Cluster Computing: Systems and Architectures*, Prentice Hall, 1999.
[4] M.R.Majma, S.Almassi, "The Investigation and Analysis of Process Migration," ceic08.
[5] A. Bouteiller, F. Cappello, et al, "MPICH-V2: a fault tolerant MPI for volatile nodes based on pessimistic sender based message logging," *Proceedings of High Performance Networking and Computing (SC2003)*, PhoenixUSA, IEEE/ACM, November 2003.
[6] M.R.Majma, H.Pedram, M.Aminian, "Dynamic Process Migration in the Fault Tolerance Cluster Computing Systems," *16th Iranian Conference on Electrical Engineering (ICEE2008)*, May 2008.
[7] M.R.Majma, H.Pedram, S.Almassi, A.Broumandnia, "Dynamic Process Migration using Intelligent Agents in Fault Tolerant Cluster Computing Systems," *18th Iranian Conference on Electrical Engineering (ICEE2010)*, May 2010.
[8] L. Silva and R. Buyya, *Parallel Programming Models and Paradigms*, High Performance Cluster Computing: Programming and Applications, RajkumarBuyya (editor), Prentice Hall PTR, NJ, USA, 1999.
[9] Daniel Balkanski, Chemnitz Mario Trams, Chemnitz Wolfgang Rehm, "Communication Middleware Systems for Heterogenous Clusters: A Comparative Study," *IEEE International Conference on Cluster Computing (CLUSTER'03)* p. 504.
[10] M.R.Majma, H.Pedram, M.ShahHosseini, A.Hanifi, "The Parallelism of Quantum Monte Carlo Algorithm for Distributed Memory Architecture," *16th Iranian Conference on Electrical Engineering (ICEE2008)*, May 2008.
[11] Skamarock, W. C.: Evaluating Mesoscale NWP Models Using Kinetic Energy Spectra. Mon. Wea., Rev., 132, 3019-3032, 2004.
[12] Wang, W., D. Barker, C. Bruy`ere, J. Dudhia, D. Gill, and J. Michalakes,WRF Version 2 modeling system user's guide, 2004.

[13] NOAAhompage: http://www.ncep.noaa.gov.

[14] Liu, Y., T.T. Warner, J. F. Bowers, L. P. Carson, F. Chen, C. A. Clough, C. A. Davis, C. H.Egeland, S. Halvorson, T.W. Huck Jr., L. Lachapelle, R.E. Malone, D. L. Rife, R.-S. Sheu,S. P. Swerdlin, and D.S. Weingarten: The operational mesogamma-scale analysis and forecast system of the U.S. Army Test and Evaluation Command. Part 1: Overview of the modeling system, the forecast products. J. Appl. Meteor. Clim., 47, 1077–1092,2008.

[15] William C. Skamarock, Joseph B. Klemp, JimyDudhia, and others,NCAR Technical Note, June 2008.

[16] Ooyama K. V., 1990: A thermodynamic foundation for modeling the moist atmosphere,J. Atmos. Sci. 47, 2580–2593.

[17] Bidokhti, M. Farahani, E. Pasandideh and B.Fallah, "A Numerical Model of urban boundary layer for prediction of Urban Air Pollution with complex terrain (Area Tehran)," *12th Iranian Geophysical conference*, February 2006.