UiT
THE ARCTIC
UNIVERSITY
OF NORWAY

Department of Computer Science, Faculty of Science and Technology

Multi-Locality Based Local and Symbiotic Computing for Interactively fast On-Demand Weather Forecasting for Small Regions, Short Durations, and Very High-Resolutions

Bård Fjukstad

A dissertation for the degree of Philosophiae Doctor – May 2014

UNIVERSITY OF TROMSØ

Multi-Locality Based Local and Symbiotic Computing for Interactively fast On-Demand Weather Forecasting for Small Regions, Short Durations, and Very High-Resolutions

Bård Fjukstad

A dissertation for the degree of Philosophiae Doctor

Department of Computer Science
Faculty of Technology and Science
UiT The Arctic University of Norway

Abstract

Modern weather forecasting is based on forecasts from huge numerical models executed on large supercomputer clusters in large computer centers. Production deadlines for forecast and available computational resources are the main factors for having to limit the forecast region size, spatial resolution, forecast duration and detail level of the weather model used. Modern numerical models currently used by meteorological services exhibit characteristics of weak scaling. The time to produce a forecast of fixed size decreases less and less with increasing number of CPU cores used. Disregarding communication costs, will how fast a computer cluster can compute a forecast be limited by how fast a single core on a node can compute one time-step for a single mesh point in the model. Adding more CPU cores allows a forecast for a larger region to be computed within the same time period. Forecasts for a small region can be computed using few CPU cores as fast or faster than a large region using many cores.

The publicly available weather forecast from national weather services as well as from other public services has a set of characteristics. The forecasts cover large regions with a resolution not high enough to include important features and detailed flow patterns in regions with complex terrain. Excepting the National Weather Service in the USA, the services make public only a small subset of available forecast parameters. Forecasts for large areas are computed using batch systems at fixed intervals using whatever observations are available at the fixed starting times. When a user request a forecasts, the latest forecast may typically be up to 6 - 12 hour old. It is not computed on-demand when the user requests the forecast. Visualizations from weather services are often from a static set of types, styles and features and are created for a web browser or apps on commodity devices. Most visualization are not suitable for specialized devices, like a display wall or a smart watch.

In this dissertation, tradeoffs are made in model parameters to identify the highest resolution being meteorologically sound and still computable on commodity hardware. The dissertation identifies also the benefits resulting from combining multiple overlapped very high-resolution forecasts. The architecture, design and implementation for commodity platforms for two systems are described.

The first system is for computing on-demand very high-resolution interactively fast forecasts. Forecasts can be produced with a resolution better than what is publicly available, for small regions and with short duration, on modern personal desktop computers. The forecasts can be produced for any location on the globe, with a user specified resolution.

The second system is for combining local weather forecasts done by multiple users with overlapping regions. The forecasts can be shared between neighbors using a developed collaborative system. Combining shared forecasts enables estimation of forecast uncertainty, thereby increasing the value of the individual forecast. The combined forecasts also allows for computing a measurement of trust between peers.

The produced forecast allows for individual visualization, and may contain all possible parameters and vertical levels from the model. A set of experiments was conducted on the prototype systems to document their characteristics.

By reducing the problem size, a numerical forecast computation can be moved from a supercomputer to a personal computer. This dissertation documents that the forecasts are produced faster, on-demand, at a higher resolution and are meteorologically sound. The forecasts can realistically be produced on-demand for any location on the globe.

This dissertation presents the principle of short traveled data and computations. One realization of this principle is locally produced weather forecasts. The produced forecasts have a local applicability, are computed and stored locally and have only a very limited dependency on third parties and external networks. A set of computational models are described including forecast uncertainty estimation, embarrassingly distributed computations, symbiotic collaborative weather forecasting, forecast visualization using augmented reality and trust establishment using local data.

The dissertation has a number of concussions including; A modern numerical model can be executed with at least 1 km resolution for a limited area providing forecasts with details not available in other forecasts today. Combining multiple overlapping very high-resolution forecasts provides information usable for error or uncertainty estimation. The architecture utilizes the locality principle and the short traveled computations and data principle. Experiments show that the typical execution times range from 2 minutes for a 10 km forecast to 15 minutes for a 1 km forecast computed using a quad-core Intel processor with hyper-threading enabled.

The architecture for the collaborative symbiotic weather forecast systems is functional without requiring prior-knowledge of peers. All peers are found on-demand, scaling well because of the limited number of peers needed.

The simplicity of exchanging forecasts also demonstrates that given these constrains, a fully decentralized exchange of forecast is viable in the sense that it will lead to forecasts that are more useful. While the simple way of finding peers succeeds in localizing all possible peers in seconds in a laboratory setting, no experiments have been conducted outside the laboratory.

Acknowledgments

Thanks to the Director General of the Norwegian Meteorological Institute, Professor Anton Eliassen. A generous proposal made the economics of a Research Fellow position at the University possible.

Thanks to my advisors Professor Otto Anshus and Associate Professor John Markus Bjørndalen who believed in this project the whole time, and continues to impress me with their knowledge and overview. In particular, Otto Anshus showed a way of thinking outside the box and at the same time keep the long lines together.

Thanks to the knowledgeable Professors and Researchers at the Department of Computer Science and in particular Tore Brox-Larsen, Head of the Department, for good coffee and very good and often humorous discussions.

Thanks to the Svein Tore Jensen and Jan Fuglesteg, staff at IFI without whom nothing is done in the proper and correct way. Thanks to the Technical staff at IFI, Maria Wulff Hauglann, Ken-Arne Jensen, Jon Ivar Kristiansen and Kai-Even Nilssen, whose knowledge and good humor can provide relief for any problems.

Thanks to my fellow Ph.D. students at IFI whose discussions inspiring. Also thanks to the two former Research Fellows, Dr. Daniel Stødle and Dr. Tor Magne Stien Hagen who showed the way and were major inspirations for starting this Ph.D.

A huge thanks to my family. My wife, Monica, and children Bjørn and Erik for their support and creative help. Erik made the front cover artwork and Bjørn helped with the typography of this dissertation and notification of new things coming. An understanding wife, who is advising many Ph.D. students was a major benefit for this work and did provide interesting discussion points and a strong incentive to bring this to a successful completion.

Thanks to the rest of my family for support. A special thought to Maria who showed the rest of us the value of small and large things in life.

The illustrations and figures in this dissertation is created by the author if not otherwise noted. Photos from the fire in Flatanger are used with permission from NRK and the Norwegian Police. The front cover artwork is created by Erik Fjukstad.

This work have in parts used equipment and resources supported by the Norwegian Research Council, projects No. 159936/V30, SHARE - A Distributed Shared Virtual Desktop for Simple, Scalable and Robust Resource Sharing across Computers, Storage and Display Devices, and No. 155550/420 - Display Wall with Compute Cluster.

Contents

List of Tables

List of Figures

List of Abbreviations

API	Application Programming Interface.
AR	Augmented Reality.
DAS	Device Attached Storage.
DEM	Digital Elevation Model.
DHT	Distributed Hash Table.
ECMWF	European Centre for Medium-Range Weather Forecasts.
EPS	Ensemble Prediction Systems.
GB	Gigabyte. 10^9 bytes.
GFS	Global Forecast System, US National Weather Service global weather prediction system.
GIS	Geographical Information Systems.
GPS	Global Positioning Systems.
GRID	Grid computing, the collection of computer resources from multiple locations to reach a common goal.
HCI	Human Computer Interface.
HIRLAM	High Resolution Limited Area Model. Also a consortium developing this model..
ISP	Internet Service Provider.
KML	Keyhole Markup Language, XML notation for geographic annotation and visualization for Google maps and Google Earth.
LAM	Limited Area Model. Meteorological models covering a region.
MOU	Memorandum of Understanding.
MPI	Message Passing Library.

MPixel	10^6 pixels.
NAS	Network Attached Storage.
NAT	Network Address Translation.
TB	Terrabyte. 10^{12} bytes.
TCP/IP	Transmission Control Protocol/Internet Protocol.
UAV	Unmanned Aerial Vehicle.
URL	Uniform Resource Locator, sometimes called URI, Uniform Resource Identifier from Tim Berners-Lee's RFC 1630 dated 1994.
UTC	Coordinated Universal Time, former GMT, Greenwich Mean Time.
VM	Virtual Machine.
VNC	Virtual Network Computing.
WMO	World Meteorological organization.
WMS	Open Geospatial Consortium's Web Map Service specification.
WRF	Weather Research and Forecasting Model.

Part I

PhD dissertation

1

Introduction

The main tool of modern weather forecasting is the numerical model. Models for the atmosphere and ocean are huge software projects with several hundred thousand lines of code. The 2.0 version of the Weather Research and Forecast model, WRF, had around 250 000 lines of code in 2004 [74], and are still being developed and extended. The models are executed on large supercomputers in large computer centers. The spatial and vertical resolution of these models is often adjusted to the computational resources available. Most forecasting centers have deadlines for when the forecasts must be made available to the users. Before a model is started, observations of current conditions have to be collected from the region of interest. Some models have global coverage and must therefore wait a long time before observations from around the globe is collected. The waiting time is in meteorological terms referred to as the *cut-off time* for the model. The combination of the cut-off time, deadlines for products and compute resources available, will limit the spatial resolution and duration of the forecasts produced. Managing the compute resources often requires the models to be executed in fixed schedules using batch-type systems.

The stored output from a model is often tightly controlled and a numerical model can produce many different parameters depending on the users need. Examples of trade-offs that must be made are; How often during the forecast period will the model produce output (every 10 min, every hour?), which parameters are included in the output, which vertical levels are included and what aggregated parameters are included. The trade-off may also include factors as available disk space and bandwidth from the supercomputer to the data storage.

Weather forecasts for the whole globe are today freely available on the Internet. What data that is made available is tightly controlled by the weather service providers. One example of an open source of weather data is the NOMADS, NOAA Operational Model Archive and Distribution System [90] which already in 2006 had 16 TB of data available online. The NOMADS system is on of the most open sources of data and still limits the number of available parameters.

This dissertation presents a system where numerical weather forecasts are produced on-demand close to the user, the data is stored on the users own systems and the data is manipulated and visualized on the devices and systems under the users control.

The user chooses a location using either the GPS in mobile devices or manually, and a forecast for a small region around this location, and for a short duration is produced on the users own commodity computer. The user may choose what parameters that are made available for visualization and the visualization can be customized for the users particular needs.

Users may exchange forecasts for either extending the forecast region, or the forecast duration. With forecasts overlapping in space and time, a novel way of estimating forecast uncertainty is possible. The statistical measurements used for uncertainty estimation can also be used for establishing a measure of trust between the users exchanging forecasts. The developed systems utilize open source and freely available software for most of the user interface and visualization. Custom software has been developed for initiating and controlling computations, data storage, data exchange and data visualization.

Numerical Weather forecasting was selected as a domain for study based on the authors background with almost 20 years of experience from operational weather forecasting. Weather forecasting is also important on both local and global scale and impacts the economy and the society in many ways. Public safety is the ultimate goal of most national weather services.

1.1 Geographical Location

UiT The Arctic University of Norway is situated on the island of Tromsø, at 69.68°N and 18.98°E. The area around is dominated by deep fjords, steep mountains and at times very bad weather. Figure 1.1 illustrates the terrain. The city of Tromsø is indicated by the arrow.

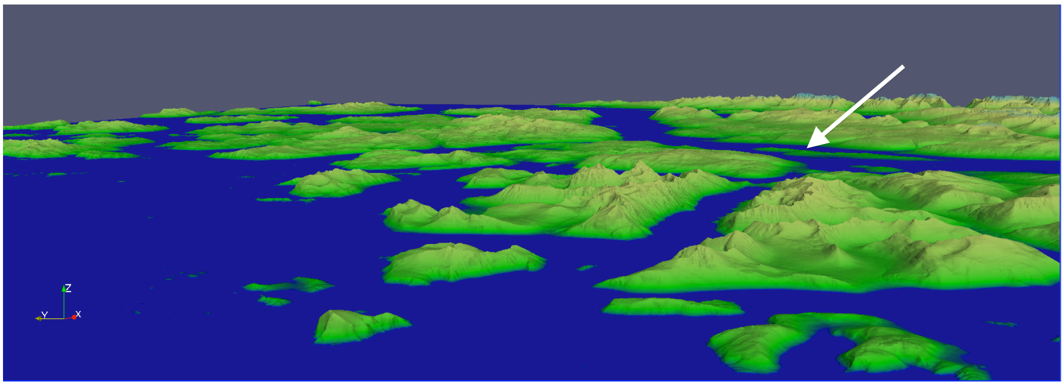


Figure 1.1 *The Topography around the island of Tromsø. Tromsø is indicated by the arrow. The figure is a rendering of 10* m *resolution DEM data from the Norwegian Mapping Authorities.*

The livelihood in this area has traditionally been based on farming and fisheries, with an increasing offshore oil industry activity. Many activities are very weather dependent, both for personal and commercial use. Activities for improving forecasting are often focused in extreme and seldom occurring dangerous weather although some industries may be impacted by often occurring weather. The building industry is one example of an activity where often-occurring weather may be problematic. Large cranes lift heavy equipment and material at building sites. This is only possible when the wind speed is below some equipment specific height. A detailed forecast for the building site may help planning the activity and avoid dangerous situations.

The complex topography in an area is a challenge when forecasting weather and especially wind. Large differences in wind speed and direction over short distances are typical. Differences of 50-90% can often be see over distances less a few kilometers. Figure 1.2 illustrates high local variability using a time series of observations from two measuring stations on the Tromsø Island, only approx. 2 km apart.

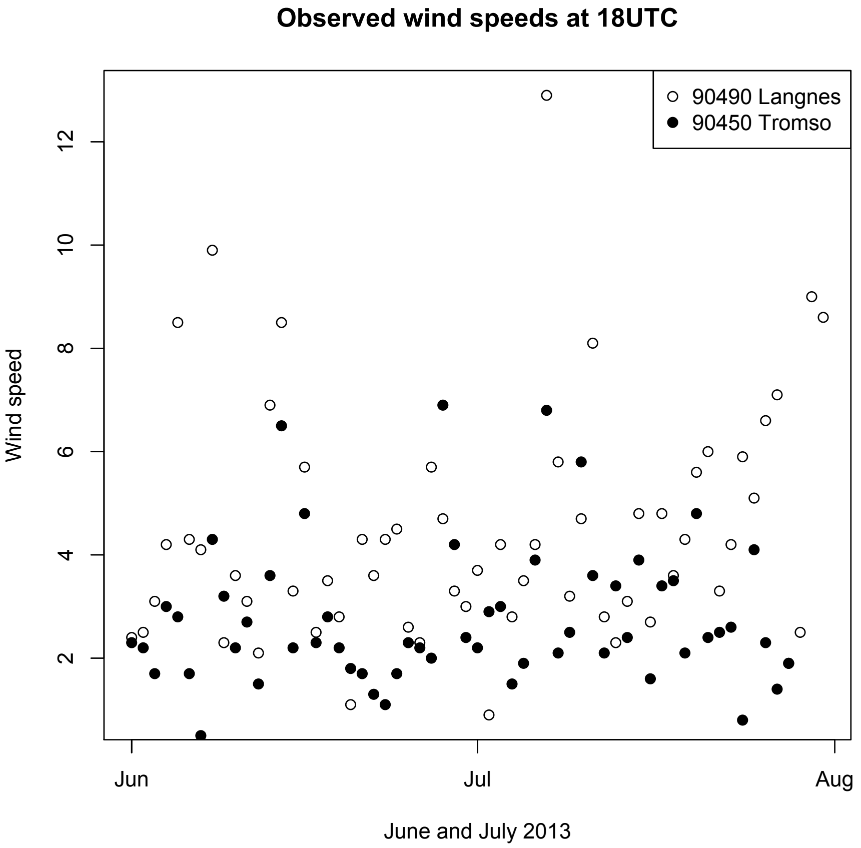


Figure 1.2 *Wind speed observations from two stations on the Tromsø island.*

1.2 Modern Weather Forecasting

The 1904 paper by Vilhelm Bjerknes [13] (1862-1951) is often cited [48] as the first demonstration of using equations based on Newton's second law for calculating the state of the atmosphere some time in the future, given knowledge of the state at the present. Two Norwegian meteorologists and scientists, Fjørtoft and Eliassen [19, 20, 85], were in 1949 part of an experiment to use the newly developed ENIAC computer at Aberdeen Maryland for calculating the future weather. It took 24 hours to do a 24-hour forecast [86], but the experiment represents the beginning of modern weather forecasting using computers. A short introduction to numerical weather forecasting is included in Chapter 3.

The Norwegian Meteorological Institute, MET Norway, currently executes the operational model suite on the Vilje [81] supercomputer in Trondheim, Norway. The current, Mars 2014, highest resolution large-region model, is the 11 km HIRLAM[1] that will soon enter pre-operational testing [59]. This model executes on 12 nodes on the Vilje computer, using 384 logical cores. With a 2-hour cut-off time and an approximate 45 minutes execution time, the forecast will be available around 3 hours after observational time. The cut-off time is the time any meteorological data center has to wait while synoptic, at the same time; observations are collected using the WMO network. For a smaller area centered on Scandinavia a version of the AROME[2] model is executed with 2,5 km resolution using 190 nodes and 6080 logical CPUs on the Vilje computer cluster. This models uses a short cut-off time because of constraints on when products from the model has to be available, and have a execution time of approximate 30 minutes. Both models use a fixed mesh location and have therefor the geophysical background pre computed and available.

These forecasts cover Scandinavia and parts of the surrounding sea areas. The highest resolution AROME forecast does not cover the whole region for which Norway is responsible for producing weather forecasts.

Norway is a member of the ECMWF[3], European Centre for Medium-Range Weather Forecasts, and uses the global models available as initial meteorological conditions for the national limited area models, LAMs. The ECMWF is an independent intergovernmental organization supported by 34 European states. ECMWF is both a research institute and an operational service, producing numerical weather forecasts to its member states. The ECMWFs main global model have a resolution around 16 km and is expected to introduce a 10 km mesh in 2015 [31].

1.2.1 Computers used by MET Norway and the ECMWF

The Vilje [81] supercomputer in Trondheim, Norway, is an SGI Altix ICE X system with dual Intel ES-2670 eight core CPUs on each node, giving 16 physical and 32 logical cores with hyper-threading enabled, a total of 1404 nodes with 22 464 cores. MET Norway have a fixed part of this computer available for its many numerical models.

[1] http://hirlam.org

[2] http://www.cnrm.meteo.fr/arome/

[3] http://ecmwf.int

Table 1.1 *Old and new computer clusters at the ECMWF*

	IBM Power cluster	**Cray XC30 cluster**
Processor	IBM Power 7	Intel Ivy Bridge
Cores per chip	8	12
Clock frequency	3.837 GHz	2.7 GHz
Power	200 W	135 W
IBM phase 1 vs. XC30 electrical group configurations		
Cabinets	24	2
Nodes	272	384
Cores	8 704	9 200
Power req.	1165 kW	176 kW
Completed clusters		
Total sustained perf.	∼70 teraflops	∼210 teraflops
Compute nodes	739	∼3 500
Compute cores	23 648	∼84 000

From: Mike Hawkins, 25th Meeting of the Computing Representatives Meeting, 13-15 November 2013, ECMWF [53].

The ECMWF in Reading, England, has just started operational use of two new Cray XC30 clusters that each allows for a sustained 200 Teraflops when executing the main forecast application. The ECMWF has always had supercomputers very high in the TOP500 list[4]. The previous generation used IBM Power 775 clusters and was ranked at 51 and 52 place. Each of the two older clusters have 24 576 CPU cores. A comparison of the new and old cluster configuration illustrates the direction of current High Performance Computing, HPC. Table 1.1 lists some of the differences. Notable is the increase in number of cores per chip and the reduction in clock frequency. The new system outperforms the old system because of a combination of the increased number of cores per chip and more efficient interconnects. The new processor has a 20% less sustained performance [53] on a per-core basis.

1.3 On-Demand Weather Forecasts

The Norwegian Meteorological Institute and the Norwegian Broadcasting, NRK, launched in September 2007 [104] one of the first sites for open-access meteorological data in Europe. The data is available under the Norwegian license for public data, NLOD, and the Creative Commons attribution 3.0 Norway license. The available forecasts covers the whole globe with varying level of details. This site has now regularly over 3 million unique users each week, with a peek of 6.9 million unique users in week 22 of 2013 [60]. Around 50 % of the users are from Norway, 25 % from Sweden and the rest is from all over the globe.

One of the main features of the yr.no web site is the availability of accessing meteorological data using a HTTP REST API [36] available at `http://api.met.no`. Forecasts

[4] `http://www.top500.org/list/2013/11/`

may be retrieved using named locations or latitude-longitude coordinates. The returned forecast will have different content depending on the location. Locations within Norway will have more parameters and more details than forecast in e.g. South Africa. The forecast will in all cases be the interpolated values from the originating mesh used by the Meteorological Institute, currently a 2,5 km resolution mesh for Scandinavia, and around 16 km elsewhere. The forecast models are executed at fixed times each day and the data available from yr.no is the latest forecasts available at all times. For Scandinavia, the update-frequency is four times per day, for the rest of the world twice per day.

All models used in the described systems above cover fixed areas, have fixed execution schedule and make available a small subset of all data produced by the models used. New observations are not included until the next scheduled execution of the model. The models are all limited by at least two factors. 1) The available computational resources. 2) The time before a forecast has to be available for the users.

The forecasts presented above are computed at regular scheduled times and is presented in a pre-processed manner. The reason for this is efficiency and speed of presenting data. The latest forecast available might have been computed many hours ago. Forecasts for the whole globe is always produced, regardless of actual usage.

This dissertation presents a system for producing forecasts on-demand and only for regions where there is actual interest. The forecasts are available for visualization after a few minutes, not instantaneous. What parameters can be visualized is user selectable, allowing for parameters not available elsewhere.

1.4 Sharing Weather

Many apps and web sites with weather forecasts are adding functionality to share[5] forecast and observations on social networks. In addition, some national weather services and other organizations have launched web services where meteorological data can be reported, in exchange for detailed forecasts. One example is `http://www.shareweather.com/`. There are also a very large number of automatic weather stations available on the Internet, one example is the University of Tromsø's own weather station with web camera at `http://weather.cs.uit.no/`.

The size of observational data produced by individuals are usually small and therefore easily measured and exchanged. Forecasts may involve significantly more larger data size.

A culture of sharing information within meteorology has been the basis for the success of the WMO, World Meteorological Organization, a UN organization. Weather observations are shared between nations for mutual benefit. The WMO operates the Global Telecommunication System, GTS, for exchanging meteorological information between nations and organizations through the World Weather Watch, WWW[6], program. The purpose of the WWW is to enable efficient meteorological services in all

[5] `http://lifehacker.com/yahoo-weather-adds-ipad-support-new-animations-and-sh-1484182899`

[6] `http://www.wmo.int/pages/prog/www/index_en.html`

countries. The WWW was established in 1963 for making meteorological and related environmental observations and data available to the member countries. The GTS has a global scope and predates the Internet and is dedicated for sharing meteorological data.

Sharing of meteorological data on the Internet can be divided into two classes. Observations, including web cameras, are shared by individuals and organizations. Forecasts are traditionally shared between national meteorological services. Putting up a weather station and camera and connecting to the Internet requires only easily available hardware and does not demand much in terms of network connectivity.

Elevant [32] studied the motivation behind sharing meteorological observations and computational resources using observations from the Citizen Weather Observer Program, CWOPs, and SETI@home. The main motivators were identified as social acceptance and intrinsic motives. The results implied that reciprocity might be of considerable importance.

Forecasts are shared between weather services and from weather services to users. Users may share observations with each other and some of these may be used by the weather services. To our knowledge, no user-produced forecasts are currently shared between users or with weather services.

1.5 Locality

The publicly available data today comes from centralized systems. All originating data are stored remotely on storage controlled and owned by the weather services. In this dissertation, a system for local production and storage is presented.

The locality principle is one of the fundamental principles of computing. The principle is also often refereed to as locality of reference. Peter Denning [27] presents a historical view on the use of the locality principle from the design of CPUs to the late 20th century focus on edge servers like Akamai on the Internet. Newer research by Gupte et. al. [49] have considered the locality of reference in a peer-to-peer setting, where similar data elements maps to nearby identifiers when storing data elements in a Distributed Hash Table, DHT. Locality of reference is also important for systems like distributed databases [83].

The locality principle or locality of reference, own much of its popularity on the speedup and increased efficiency obtained. This is frequently because of good re-use of data or instructions already in caches. This temporal and spatial locality [28] is expected to be valid on a system level, for individually computations.

1.6 Building Prototypes

The prevalent research method used by the HPDS research group[7] at the Institute of Computer Science is to develop the idea of a system followed by developing one or more architectures, one or more designs and one or more implementations. Experiment can

[7] `http:\http://hpds.cs.uit.no/`

be performed using the developed systems for establishing characteristics, performance and problems. By actually building systems, the very complex interaction between the always-developing hardware and software can be investigated and usability of different approaches can be measured and validated.

The results in this dissertation have come from a gradual process involving building systems using the lessons learned from the previous system. Constantly exploring the implications of what is possible and what is not possible, guided by tests of implementations and experiments for establishing performance and utilization.

1.7 Problem Statements

The research problems in this dissertation are:

1. Increase resolution, decrease area, and decrease duration of a numerical atmospherical model using available background data with the purpose of finding the highest resolution being meteorologically sound.

2. Combine multiple local very high-resolution forecasts with the purpose of determining the meteorological benefits.

3. Build a system for computing on-demand very high-resolution interactively fast meteorologically sound forecasts with the purpose of identifying the characteristics and performance of an architecture, design and implementation done for a commodity platform.

4. Build a system for combining local weather forecasts with the purpose of identifying the characteristics and performance of an architecture, design and implementation done for a commodity platform.

1.8 Contributions

A more extensive discussion of the claimed contributions of this dissertation is given in Chapter 12. Here are some of the contributions listed. The contributions are organized into *principles*, *models*, *artifacts*, *facts* and *lessons learned*.

- Principles
  - **A numerical forecast computed on a super computer can be moved to a personal computer by reducing the problem size by O**(C_{SC}/C_{PC}).
    - C_{SC} - Number of cores on a super computer.
    - C_{PC} - Number of cores on a personal computer.
  - **Short traveled data and computations.**
- Models for
  - **Embarrassingly distributed computations**

  - **Symbiotic collaborative weather forecasting**
  - **Scalable Distributed Weather Forecasting**
  - **Uncertainty and error estimations of weather forecasts based on collaborative exchange of forecasts**
  - **Forecast visualization using augmented reality**
  - **Computing Trust in Locality Based Collaborative Weather Forecasting Systems**

- Artifacts
  - Prototypes
    - **A system for on demand production on very high resolution numerical weather forecasts**
    - **A system for collaborative weather forecasting using peer-to-peer exchange of forecasts between unknown neighbors**
    - **A system for visualization of forecasts using a user-location centric view**
    - **Augmented Reality System for Visualization of Forecasts using a User-Location Centric View**
  - Output from the prototypes
    - **Local Forecasts with Three Predetermined Resolutions**
    - **Amalgamated forecasts from collaborative weather forecasts**
    - **Estimation of Uncertainty in Wind Speed Forecasts**
    - **Visualization of Weather Forecasts**
- **Facts**
- **Lessons learned**
  - Suitable Trade-Offs for Sound Forecasts
  - Better weather forecasts
  - Scalable distributed weather forecasting
  - Localized collaborative weather forecasting
  - On-Demand Weather Forecasts for Achieving Safety
  - Visualization of Meteorological Data on a Display Wall
  - Miscellaneous Applications

1.9 Included Papers

The papers in Table 1.2 provide the basis for this dissertation. All papers have been submitted for peer-review and accepted for publication. The roles of author and co-authors have followed the criteria of the Vancouver Convention.

The additional papers represent side issues that were also studied as part of the work, but are not significant parts of the contributions.

Table 1.2 *Papers included in this dissertation*

Paper	Title	Presented at
1	**Interactive Weather Simulation and Visualization on a Display Wall with Many-Core Compute Nodes**. Bård Fjukstad, Tor-Magne Stien Hagen, Daniel Stødle, Hoai Phuong Ha, John Markus Bjørndalen, and Otto Anshus. [41]	PARA 2010, Reykjavik, Iceland
2	**Embarrassingly Distributed Computing for Symbiotic Weather Forecasts**. Bård Fjukstad, John Markus Bjørndalen, and Otto Anshus. [40]	ICCS 2013, Barcelona, Spain
3	**Accurate Weather Forecasting Through Locality Based Collaborative Computing**. Bard Fjukstad, John Markus Bjorndalen, and Otto Anshus. [39]	CollaborateCom 2013, Austin, USA
4	**Uncertainty Estimation and Visualization of Wind in Weather Forecasts**. B Fjukstad, John Markus Bjørndalen, and Otto Anshus. [37]	IVAPP 2014 Lisbon, Portugal
	Additional Papers	
1	**Nine Years of the Tromsø Display Wall**. Otto J Anshus, John Markus Bjørndalen, Daniel Stødle, Lars Ailo Bongo, Tor-Magne Stien Hagen, Yong Liu, Bård Fjukstad, and Lars Tiede. [6]	CHI 2013, Paris, France
2	**SAFE-WEATHER: User Specified, Rapidly Produced, On-Demand, Very High-Resolution Numerical Weather Forecasts**. Bård Fjukstad, John Markus Bjørndalen, and Otto Anshus. [38]	Poster, Arctic Frontiers, 2012, Tromsø, Norway

1.10 Summary of the Papers

1.10.1 Interactive Weather Simulation and Visualization on a Display Wall with Many-Core Compute Nodes

This paper was presented at the Para 2010: State of the Art in Scientific and Parallel Computing in Reykjavík, Iceland, June, 2010.

This paper describes a system for production of on-demand numerical weather forecasts and visualization of these on the 22 MPixel Tromsø Display Wall. The paper describes an interactive system for visualizing the globe, with zoom and many layers of detail. The interactive system selects a center location and starts the execution of the numerical forecast model on any of two compute clusters. The results are transferred to a visualization computer, the wanted parameter are visualized using the current zoom factor and the results are sent to the Display Wall for rendering on top of the current view. This paper describes executing the numerical model on a single 8-core node on the Stallo supercomputer. The bottlenecks in the system are found to be the execution of the numerical model on a compute cluster, and the visualization computer. Using a spatial resolution of 10 km, the total time from selection of center location to the visualized parameter are available on the Display Wall was found to be around 3 min.

Citation: Bård Fjukstad, Tor-Magne Stien Hagen, Daniel Stødle, Hoai Phuong Ha, John Markus Bjørndalen, and Otto Anshus. Interactive Weather Simulation and Visualization on a Display Wall with Many-Core Compute Nodes. In State of the Art in Scientific and Parallel Computing (Para 2010) —Program Schedule and Short Abstracts (Updated version published in LNCS 7133, February 2012), pages 142—151. University of Iceland, 2012.

1.10.2 Embarrassingly Distributed Computing for Symbiotic Weather Forecasts

This paper was presented at the International Conference on Computational Science in Barcelona, Spain, June 2013.

This paper describes the initial system for executing a numerical model on a desktop computer and sharing forecasts with geographical neighbors. This paper introduces the DSWF, Distributed Symbiotic Weather Forecast, system. Results for executing the models at 10, 3 and 1 km resolution are shown. In addition, the first notion of error estimation is introduced.

Citation: Bård Fjukstad, John Markus Bjørndalen, and Otto Anshus. Embarrassingly Distributed Computing for Symbiotic Weather Forecasts. In Proceedings of the International Conference on Computational Science, ICCS, pages 1217—1225, June 2013.

1.10.3 Accurate Weather Forecasting through Locality Based Collaborative Computing

This paper was presented at the 8th IEEE International Workshop on Trusted Collaboration (TrustCol 2013) in conjunction with the 9th International Conference on Collaborative Computing, Networking, Applications and Worksharing in Austin, USA, October 2013.

This paper develops the CSWF, Collaborative Symbiotic Weather Forecast, system. The three—tier approach with the global, the local and the collaborative symbiotic forecast is introduced. This paper focuses on the collaboration aspect of the weather forecasting system.

Citation: Bard Fjukstad, John Markus Bjorndalen, and Otto Anshus. Accurate weather forecasting through locality based collaborative computing. In Collaborative Computing: Networking, Applications and Worksharing (Collaboratecom), 2013 9th International Conference Conference on, pages 571—578, 2013.

1.10.4 Uncertainty Estimation and Visualization of Wind in Weather Forecasts

This paper was presented at the 9th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications in Lisbon, Portugal, January 2014.

This paper focuses on the visualization of the uncertainty estimation generated by the CSWF system. Differences in visualization on smaller displays and large display walls are discussed and two different solutions for visualizing many glyphs on the same plot are illustrated. In addition, visualization of time-series of forecasts is introduced.

Citation: Bård Fjukstad,John Markus Bjørndalen, and Otto Anshus. Uncertainty Estimation and Visualization of Wind in Weather Forecasts. IN PRESS. IVAPP 2014, pages 1—8, 2014.

1.10.5 Additional Papers

Nine Years of the Tromsø Display Wall

This paper was presented at the PowerWall International Workshop on Interactive, Ultra High-Resolution Displays hosted by the ACM CHI, Paris, France, April 2013. Bård Fjukstad is a co-author of this paper and the contribution is according to the Vancouver Convention requirements of co-authorship.

This paper describes the history and status of the Tromsø Display Wall. Visualizing weather forecasts are one of several themes.

Citation: O. J. Anshus, J. M. Bjørndalen, D. Stødle, L. A. Bongo, T.-M. S. Hagen, Y.

Liu, B. Fjukstad, and L. Tiede, Nine Years of the Tromsø Display Wall, presented at the POWERWALL International Workshop on Interactive, Ultra-High-Resolution Displays, part of the SIGCHI Conference on Human Factors in Computing Systems, 2013, pp. 1-6.

SAFE-WEATHER: User Specified, Rapidly Produced, On-Demand, Very High-Resolution Numerical Weather Forecasts

This paper was presented at the science session of the Arctic Frontiers 2012 conference. Bård Fjukstad is the main author of this paper.

This poster is an illustration of one possible practical use of a system for producing high-resolution on-demand weather forecasts, using a Polar Low as an example use-case. This system would be available for execution on platforms, ships and control-centers, and would produce a much higher resolution forecast in a short period, than what is currently publicly available.

Citation: Bård Fjukstad, John Markus Bjørndalen, and Otto Anshus. SAFE-WEATHER: User Specified, Rapidly Produced, On—Demand, Very High—Resolution Numerical Weather Forecasts. In Arctic Frontiers, Tromsø, Norway, January 2012.

1.11 Assumptions and Limitations

The following limitations and assumptions have been made during the work with this dissertation. Some of the assumptions will be clear after reading the details in the coming chapters.

- A single numerical atmospheric model have been used in this work. It is assumed that the results would be very similar if using other local area models that can be executed using a comparable computer system and with a similar organization of model instance execution.

- Most numerical atmospheric models utilize the classical interpretation of the locality principle seeking optimal access patterns in matrixes and other techniques. Using a large number of nodes on a super computer may introduce problems related to the reliability of nodes and operational stability and service levels. These themes will not be investigated in this dissertation but it should be noted that a small and localized computation can easily be restarted.

- The default numerical atmospheric model setup generated by the WRF Portal[8] are assumed a reasonable good starting point for executing the model and the model setup generated was used throughout this work. No research into finding the optimal setup for the available hardware is done in this dissertation.s

[8] http://esrl.noaa.gov/gsd/wrfportal/WRFPortal.html

- The model was successfully built on a single Linux system at the Department of Computer Science and the executable were used on all Linux computers within the department.
  - The model was built with support for distributed memory (dm) using MPI and without support for shared memory (sh).
  - The model was executed without using high speed interconnects for the main part of the work presented in this dissertation. Exceptions are noted in the descriptions of the implementation.
  - It follows from the previous items that issues connected to optimizing the model was not discussed in this dissertation.
  - The pre-processing was executed sequentially on one host using one process. The pre-processing was executed every time the model was executed.
  - No local measurements were used, i.e a local analysis is not used in the model. A daily set of background meteorological data was used for generating a forecast valid for the same day.
- Security has a major role in distributed systems and collaboration systems. Most of this aspect is not investigated in this dissertation. The domain, localized weather forecasting, allows for a novel way of verifying trust. This aspect is expanded upon.
- A visualization application for iPad or iPhones is developed using an Augmented reality approach. The suitability of the HCI elements is not further investigated in this dissertation.
- The necessary background meteorological data used in the developed prototypes is assumed available for download prior to the request for a forecast. A user has to trust the source of this data, usually public services with high visibility, many users and a high volume of traffic. This traffic may not be randomly distributed over a period, as each participating user can decide when to do this. The service delivering these background data is assumed to have sufficient capability for handling this traffic.

1.12 Organization of the Dissertation

The rest of this dissertation is organized as follows. Chapter 2 presents various computational approaches for personalized and localized computations. Chapter 3 is a brief introduction to numerical weather forecasting. Chapter 4 presents a way of doing numerical weather forecasting on commodity hardware. Chapter 5 presents a way of adding value to forecasts using multiple overlapping local forecasts. Chapters 6 to 10 develop systems for on-demand interactive very high-resolution weather forecasting, and visualization of forecast and collaborative weather forecasting. Chapter 11 contains two case studies. Chapter 12 details the claimed contributions of this dissertation. Chapter 13 contains some related works. The final Chapters contain conclusions and future work.

2

Personal Computations

New approaches for executing large computational tasks and storing data for individuals are emerging as technologies are developed. In this Chapter, computation of a numerical atmospheric model for a small region and short duration, and video editing are used as examples of computational demanding tasks for individuals. These tasks have slightly different hardware and organizational requirements. Data may be produced, consumed and shared from different locations and through different systems.

Restricting the discussion in this Chapter to what approaches would normally be available for individuals excludes for example the use of compute clusters. The discussion is also focused on user-initiated production of content and the subsequent use. The different approaches will be discussed in terms of architectural characteristica, capabilities, division of concerns and responsibilities.

2.1 Own Device/Home Computer

The computational power available on personal devices and computers have increased rapidly following the popular interpretation of Moore's law [77], the processing performance is doubling every 18 months. Today, early 2014, mobile phones like the iPhone 5S have the equivalent processing power of an Intel Core 2 Duo, ca. 2008, at 2530 MHz (2 cores)[1]. Table 2.1 contains details using the Geekbench 3[2] scores for comparing different generation devices and capabilities. First two entries are comparing the processor in the mobile device with past desktop devices; last two are comparing the hardware used in this work, to the latest Apple Mac Pro. The Geekbench 3 score is one of several similar ways of measuring computational capabilities; other indexes would give similar but not identical results. The current capabilities of computational power in a mobile device imply that this class of devices would be capable of executing a full numerical atmospheric model, if all supporting libraries were available. The

[1] From http://browser.primatelabs.com/processor-benchmarks Score 2457 vs. 2530

[2] http://www.primatelabs.com/geekbench/

Table 2.1 *Geekbench 3 scores*

Device	Year	Scores	Cores
iPhone 5S	2013	2457	2
Intel Core 2 Duo T9400	2008	2685	2
Intel Xeon W3520	2010	7935	4
Intel Xeon E5-1650 vs	2013	18400	6

Geekbench 3 64 bits, multi-cores scores. Calibrated against a baseline score of 2500 (which is the score of an Intel Core i5-2520M @ 2.50 GHz). Higher scores are better, with double the score indicating double the performance.

computational power is still notably less than available on a current desktop computer represented by the last two entries in Table 2.1. Model details will be discussed in Chapter 6, and the hardware used in this work in Section 4.4.1.

The best video editing hardware is currently the Apple Mac Pro[3]. Most media reports focuses on personalized computation and in particular video editing and other creative work, as natural on this platform.

Data storage is also increasing in capacity in an exponential way. Hard disks with storage capacity of several TB are now a commodity product. This makes relatively large storage available for homes and small businesses. Examples of products available in this category include NAS and DAS devices. Storage on mobile devices are still in the lower hundreds of GB range, for example 64 GB of storage on an iPhone. This limits what can be stored on these devices.

A suggested hardware setup for a system for executing the tasks like a numerical model or video editing would include some modern multi-core CPU, 4 to 12 cores, with as much RAM as possible, 8 to 32 Gb. The system would contain a modern GPU card and be connected to the Internet with bandwidth in the 5 to 100s MB/seconds range. The system would have a few TBs disk space available. This level of system is currently available from any larger home electronic outlet. Later in this dissertation, similar hardware will be documented sufficient for executing a numerical model. The system is assumed sufficient for home video editing. Connecting users' mobile devices to a home computer for sharing storage and computational resources requires dedicated software today. Few systems are commercial or freely available, most related to multimedia consumption on multiple devices.

There is a trend of adding networked devices to homes, for example, Phillips Hue[4] light controllers, Smart TVs[5] and smarter Set-top boxes like Apple-TV, Chromecast and Media servers. These devices are connected to the local network in the residence, usually with Internet connection and provide a limited set of services. Often these devices are controllable from apps on mobile devices, smart phones etc. Several areas

[3] `http://store.apple.com/us/buy-mac/mac-pro`
[4] `http://www.meethue.com/`
[5] `http://www.samsung.com/no/consumer/tv-home-theatre/tv/smart-tv`

are seeing much research, like personal health care and the Internet-of-things. These are outside the scope of this dissertation and will not be investigated further, but will of course have implications on future architectures.

The architecture for a home system for computations could be represented with a single box labeled *computer*. A design would add the network and Internet connectivity and the implementation could be solved using a home computer from major electronic outlets.

2.2 GRID Computing

GRID computing [43] is used as a label on a *Super Virtual Computer* where geographically distributed resources are made available using standardized middleware. Two examples are the Globus Toolkit [44] and the Minimum intrusion Grid [11][6]. A GRID is often intended for non-interactive and a high-performance orientation use, with batch-oriented systems for submitting workloads. GRID computing offers a distributed solution for executing a numerical atmospherical model. Using GRID computing would split the concerns of controlling and implementation between the groups responsible for maintaining the GRID resources and the user. The GRID systems assume that the user handles most issues with setting up the computation, background data and data retrieval after the computations. The GRID may provide additional support for data storage and visualization.

The division of concerns using GRID computing would be fairly close to the situation when using home computers. Most issues related to starting up, controlling the execution and post-processing and storage would be the users own responsibility. The GRID provides some of the computational infrastructure. Movement of data and software into the GRID and afterwards back to the users may require significant network capacity. A GRID would be acceptable for executing a numerical model, but would not be suitable for interactive video editing.

The architecture of a system using GRID computing would involve several elements. First the home computer where the application and data are initially stored, the the connection to a GRID computer front-end and finally the GRID cluster. All need to be connected and management software executed everywhere. This makes GRID computing a system for more *professional* use, in some sense.

2.3 Peer To Peer

Androutsellis-Theotokis and Spinnellis [5] propose a definition of Peer-to-peer.

> *Peer-to-peer systems are distributed systems consisting of interconnected nodes able to self-organize into network topologies with the purpose of sharing resources such as content, CPU cycles, storage and bandwidth, capable of adapt-*

[6] `http://www.migrid.org`

> *ing to failures and accommodating transient populations of nodes while maintaining acceptable connectivity and performance, without requiring the intermediation or support of a global centralized server or authority.*
> (Androutsellis-Theotokis and Spinnellis)

Peer-to-peer systems have peers that are connected for varying periods. The system needs to be able to handle new and leaving peers, usually implying that routing information has to be maintained by sending information between peers with high enough frequency.

Peer-to-peer systems have mainly being used for content sharing or distribution. One example of using peer-to-peer systems for (legal) content distribution is the Norwegian Broadcasting, NRK, which distributes some of the programs using Bit Torrent[7]. The complete *Hurtigruten minutt for minutt* (The Coastal Express, minute by minute, a 134 hour continuous broadcast) download uses 600Gb of disk space. NRK uses peer-to-peer distribution on this material as a way of reducing the bandwidth demands on their system. The peer-to-peer system distributes the network load over all participating nodes. Peer-to-peer systems may also share computational resources between peers.

Peer-to-peer systems are suitable for one of the two tasks discussed in this Chapter. Computation of the numerical model is possible to divide into suitable large tasks and distributed for execution. This requires the application, the implementation for the numerical model, to be installed on all peers. Video editing is a task suitable for peer-to-peer only in special cases. A crowd-sourced movie is an example of a task that would fit the peer-to-peer model.

The architecture related to computations in a peer-to-peer system would specify a single computer with one or more possible connections to a varying set of peers. The design would require functionality for disconnection and reconnection of peers, and a system for distribute a computational task in parts, ensuring that all parts are executed and results collected. The implementation could specify one of several network layouts used, ie. DHTs or trees.

2.4 Cloud Services

The definition of "Cloud Computing" is slightly dependent on the reporter. In this dissertation, the NIST-definition is used.

> *Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources .... that can be rapidly provisioned and released with minimal management effort or service provider interaction.* [24] (NIST)

One main difference between GRID and Cloud services is that the former is a result of collaboration within or between groups, and the later is either commercial services

[7] `http://nrkbeta.no/bittorrent/`

from single vendors or private cloud-based services within businesses. Some federated cloud solutions are also available.

Using a cloud service for numerical models is possible and there have been several research papers published using the WRF model. See Chapter 6 for more details on this model. Most have focused on the issue of model performance in a virtualized environment [92, 29]. Using lightweight virtualization is reported to cost 5-10% in performance. Agustsson et. al. in [1] use the WRF model and either an in-house cluster or the Amazon Elastic Compute Cloud (Amazon EC2) for computations. Using resources that can be instantiated and released as demand is fluctuating would have provided a good fit with a varying number of users accessing the system. This would require a VM with a pre-loaded setup of computations. If varying background data are needed, as it often is the case with numerical models, these had to be either pre-loaded in the storage part of the cloud, or downloaded on-demand. Downloading background data would also add to the total run-time for the computational task.

One major issue with cloud-based services is control over the user-generated content. Most services have a service description that states how and when user content are accessed and processed. Most cloud based services gives only limited guarantees on the security and long-term storage of user generated content. Most services have user-level agreements that give the services access to the content, in most cases for the services to be able to provide functionality to the users. These user-level agreements may be changed by the services unilaterally. The recent exposures by Edward Snowdon[8] highlight the issues of who have access to data stored by social networks.

Several Internet services providing results from numerical models are available online. This is the standard product from all weather services. Executing a numerical model remotely is not publicly available as far as extensive Internet searches can establish. Augustsson [1] describes a related service that may use cloud computing. Cloud based online video editing is available from operators like WeVideo[9] and YouTube[10].

YouTube and Facebook are popular social networks where content are produced by the users, but distributed and shared using centralized applications and web-services. Using these cloud based services places little demand on the users infrastructure. Content may be produced on mobile devices and are easily uploaded to the services. For example will most photo and video applications on smart phones have a "share" button for many popular social networks, like YouTube and Facebook. Content are also easily consumed on any devices from the same social networks and are easily shared between registered users. No long-term storage on the user side is required for either service; this is all done "in the cloud". Most services are accessible on many devices, making small demands on computational power or capabilities. Uploading video content may require some network capabilities, and the use of the available content is dependent on the network. Most social networks have applications and web pages that dynamically change the way content are loaded and presented depending on the platform user for consumption.

[8] http://www.youtube.com/watch?v=XHbK96WOwQI

[9] http://www.wevideo.com/

[10] http://www.youtube.com/editor

Table 2.2 *Architectures and characteristics*

System	Architecture	Characteristics	User requirements
Home computers	Single computer	Everything locally	User must organize software, hardware and backup
GRID computing	Remote cluster with front-end	Local and remote execution and local storage	User must provide application and data and move these to and from. Does not support interactivity
Peer-to-peer	Varying number of peers	Local execution, data shared	User must organize own software, hardware and backup
Cloud services	Remote virtual computer	Provider handles everything	No user requirements. Access to content is controlled by service provider. Agreements may change

Architecture of different systems with some of the characteristicas.

The architecture of a cloud based system may use a web browser as an application for accessing everything on a remote computer, or a local application utilizes computational resources on a remote computer. All content and work are stored and computed on the remote system. YouTube and Facebook use both models for accessing content, ie. Apps on mobile devices and browser access on desktop systems. Google Drive and Dropbox are examples of systems where local storage is replicated in a cloud service and files may be edited either using the cloud service or using a local editing tool on the local file. The user license for such services may give the service access to perpetual access and use the stored content also after the user have deleted the content.

2.5 Summary

Table 2.2 presents a brief summary of the architecture and characteristics of the different systems considered in a computational setting.

3

Numerical Weather Forecasting

This chapter contains a brief overview of some of the techniques and issues related to numerical weather forecasting. The chapter is not meant as an extensive overview, but more as a background for some of the issues discussed later in this dissertation. A good book for further reference on this theme is R. Pielke Sr. *Mesoscale Meteorological Modeling* [87]. Wikipedia also have an article on this topic[1].

The basis for numerical weather forecasting is the use of mathematical models for prediction of the future state of the atmosphere. First, an initial state of the system must initialized. The atmosphere is then treated as a fluid, and the basic laws of motion, thermodynamics and preservation of mass can be used for computing the state after a short time period.

The basic laws used, are often referenced to as the *primitive equations*, a set of nonlinear differential equations that must be solved numerically. A set of *prognostic variables* describes the state of the atmosphere. Traditionally this is done by dividing the area of interest by a mesh, in numerical modeling often called a *grid*. In this dissertation *grid* is a computer science related theme and the term *mesh* is therefore used. This is illustrated in the left hand side of Figure 3.1. The mesh is three-dimensional usually with a terrain following vertical levels as illustrated on the right hand side of Figure 3.1. A model may also have some levels that are below ground, to correctly describe the flow of water and heat. The horizontal spacing between each mesh point is the resolution of the mesh. This is often in the range from 1 to 50 km.

The set of equation is initialized with a starting condition, and a set of rates of change for each prognostic variable is calculated. Using these rates of changes, a new value of each prognostic variable is calculated for a short time into the future. This short time period is known as the *time-step*. The length of this time-step is chosen as a function of the mesh resolution to preserve numerical stability. Since the primitive equations also contain such things as acoustic waves, these are often handled separately

[1] http://en.wikipedia.org/wiki/Numerical_weather_prediction

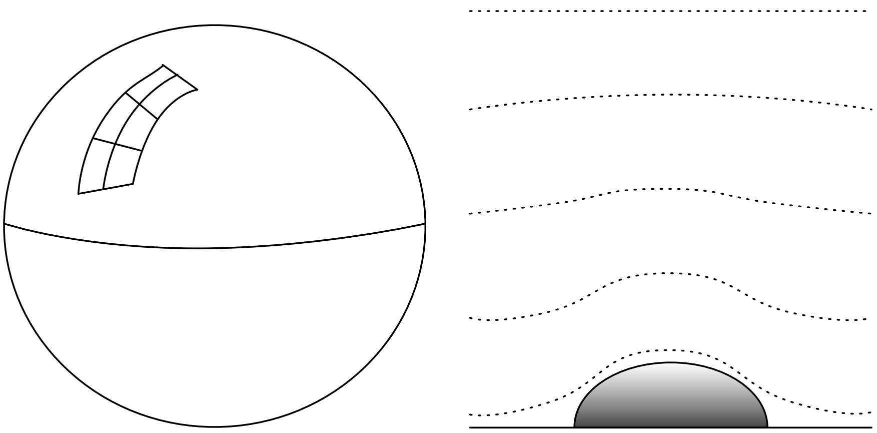


Figure 3.1 *Illustration of numerical forecasting. On the left side, a mesh is placed on the earth, on the right an illustration of a terrain following coordinate system.*

to avoid having to use very short time-steps. Many models also have a choice of which advection scheme to use, that also influences the length of the time-steps. The following equation is usable in 3D models using a third-order Runge-Kutta (RK3) time integration scheme (from [96, p 25]).

$$\Delta t < \frac{C_{r_{theory}}}{\sqrt{3}} \cdot \frac{\Delta x}{u_{max}}$$

Δt is the time-step

$C_{r_{theory}}$ is the Courant number representing how long something will advect in one time step

Δx is the mesh spacing

u_{max} is the maximum velocity expected in the simulations. The maximum velocity expected in jet streams may be up to 100 ms^{-1}.

One rule of thumb from the WRF-ARW model used later in this dissertation, is that the time step (in seconds) should be approximately 6 times the mesh distance (in kilometers). Most models will keep track of some parameter indicating numerical stability, and may stop if certain thresholds are exceeded.

Before starting a model, a set of background data including topography and land use has to be initiated. This initiation uses high-resolution data for constructing a set of data suitable to the actual resolution of the model. It is often necessary to smooth the data to avoid numerical instability. The terrain is often replaced with an envelope version that better represents the effect of the actual terrain on the atmospheric flow, and is smoothed to mitigate numerical instabilities and allow longer time-steps.

The general flow in the atmosphere are described by the equations used in the model, and are integrated numerically. Other physical processes are often parameterized, that is, described using iterative or other techniques for each mesh point in the model. This is often the case for clouds, radiation and precipitation.

A model for a limited area needs lateral boundary conditions. These are usually taken from some lower resolution global or large-scale model. The outer rows and columns are specified at an interpolated boundary value. The next few rows and columns inside the boundary are used for a relaxation zone where the model values are nudged or relaxed towards the large-scale boundary values. The width of this relaxation zone is often a run-time option. Forecast values within this zone should therefore be used with caution. An example from an NCAR technical note [96, p 53] is illustrated in Figure 3.2. The outer mesh points, labeled "1", is kept constant equal to the surrounding background data, and the inner mesh points, labeled "2" through "5", is gradually less and less nudged towards the outer values. The size of the relaxation zone has implications for small area forecasting. The size of the mesh used must ensure that a sufficient large area is inside the relaxation zones to be able to utilize the increased resolution.

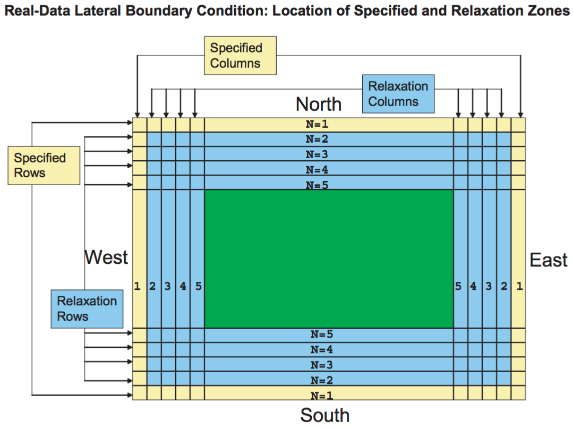


Figure 3.2 *An example of relaxation zones using five rows/columns. From NCAR technical note on the WRF model version 3 [96, p. 53].*

Since most numerical models are intended for execution on large compute clusters, geographical domains are decomposed into smaller areas that fit within single nodes, for optimal usage of shared memory or other parallel programming tools. One illustration of how a model domain may be decomposed is given in Figure 3.3. The domain is first divided into sub-regions for each individual process, and the into sub-sub-regions for each thread in each process.

A model would typically want to execute all shared memory threads on one node and divide the processes up between the available compute nodes in the compute cluster. A model do not have to use shared memory, but will always use some form of message passing for communication and coordination between nodes. Finding the optimal decomposition for each model on each compute clusters requires experimentation and measurements. One example of such experimentation and optimizing is found in Johnson et. al. [62].

A numerical atmospherical model has limitations. A short list of some of these follows:

- The horizontal resolution has a strong influence on what high gradient features like topography, that are sufficiently included in the model.
- The vertical resolution has a strong influence of what flow types can be described by the model.
- Parametrization is a method for representing sub-mesh scale phenomenas in the model. The parameterization used may have known or limiting capabilities depending on the resolution used. One example is using the parameterization of convective clouds versus explicitly resolving the clouds in the model. This is highly dependent on the resolution used, and will vary depending on the latitude.

Most models will also show systematic errors resulting from the lack of resolution for resolving very fine scale flow.

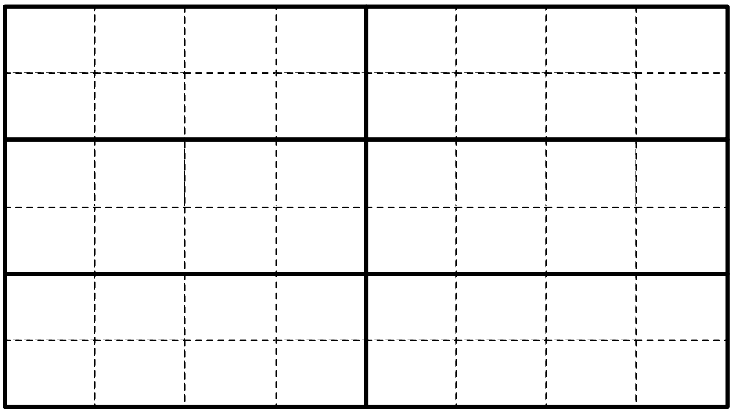


Figure 3.3 *Example of domain decompositioning for a mixed-mode job with six MPI processes indicated by solid lines, and eight OpenMP threads within each process indicated by dashed lines.*

4

Local Area Weather Forecasting

This chapter explores the research problem statement: *Increase resolution, decrease area, decrease duration of a numerical atmospherical model using available background data with the purpose of finding the highest resolution being meteorologically sound.*.

4.1 Idea

The models user in modern weather forecasting may either have global coverage like the models used at ECMWF and NOAA NWS, or regional coverage with higher spatial resolution. Most weather services use several models with different resolutions and coverage.

A modern desktop computer has similar computational power compared to one node on a supercomputer. To be able to execute a numerical atmospheric model on a desktop computer, the model implementation may have to limit the size of the area covered and duration of the forecast.

4.2 Available Models

Several models were considered for use in this dissertation. Table 4.1 describes the status of various models and their availability for use in this work. Table 4.2 contains links to the governing entities for each model. The focus was initially on models created in Europe but was broadened to include models used in Europe because of issues with availability of model and background data.

A major consideration for choosing which numerical model to use is the availability of background geophysical and meteorological data. Most numerical atmospheric models in use in Europe are controlled by consortiums and research groups, and are not freely available. The HIRLAM and AROME models are not publicly available, although some of them could be made available in an academic setting with certain restrictions.

Table 4.1 *Numerical atmospheric models and availability*

Model	Controlled by	License	Availability
HIRLAM	Consortium of national weather services	Conditions set in Appendix 2 of cooperation agreement	Very restricted outside partners
ALADIN	Consortium with national weather services	4th Aladin MOU	Restricted outside partners
AROME	Cooperation between national weather services	Not known	Possible free access for research purposes
WRF	NCAR/UCAR developed	In the public domain	Free

The HIRLAM and ALADIN consortiums are currently cooperating in a joint developmental effort.

Table 4.2 *Numerical atmospheric models references*

Model	URL
HIRLAM	`http://hirlam.org/`
ALADIN	`http://www.cnrm.meteo.fr/aladin/?lang=en`
AROME	`http://www.cnrm.meteo.fr/arome/`
WRF	`http://www.wrf-model.org/index.php`

It was not deemed practical to use the HIRLAM, ALADIN or AROME models based on availability of the model and necessary background data.

In the early trials of the WRF model both data from the US National Weather Service and from the European Centre for Medium range Weather Forecasting, ECMWF, was used. The data from the ECMWF was the reanalysis ERA-40 data, and not real-time current data. From the US daily updated data were available and actual usable forecasts could be made.

All mentioned models would require a specific set of compilers and libraries for compilation. The WRF model can use several different compilers, including the standard compilers on a Linux platform.

The WRF model and background geophysical data is freely available, and the model is flexible in what data to use as meteorological background data. On this basis the Weather Research and Forecast model, WRF [75], was chosen for this work.

4.3 The WRF Model

The version of the WRF model used in this dissertation is WRF ARW-core version 3.1.1. The WRF model is in the public domain and can be used by anyone without any fee or charge[1]. WRF® is a registered trademark of the University Corporation for Atmospheric Research, UCAR.

Building the WRF system requires many additional software libraries, and is not a trivial task. The version used here is explicitly NOT optimized for the current hardware or the meteorological conditions and choices available. The used version is only built with support for message passing using MPI[2], not with support for shared memory.

A single version of the model was built on the front-end for both the local clusters at the Department of Computer Science at the University of Tromsø. This version was used on all computers and clusters at the department, since the requirements in terms of libraries could easily be met on all the local Linux based systems, including in virtual machines on other operating systems. Another version was built on the front-end for the Stallo[3] supercomputer. This version was exclusively used for experiments and testing on the Stallo cluster. Both versions were built for executing a model using daily updated background meteorological data, the so-called *real* mode.

A build of the model produces set of files of executable files, *ndown.exe*, *nup.exe*, *real.exe*, *tc.exe* and *wrf.exe*. Both *real.exe* and *wrf.exe* are executed using *mpirun*, the others as single computer applications. Figure 4.1 illustrates the WRF pre-processing system and the order and input of the various applications that are required. The model proper, *wrf.exe*, is executed after the flow in Figure 4.1.

Listing 4.1 shows a script for executing the model with daily updated background data.

Listing 4.1 Script for executing the WRF pre-processing and model

```
#!/bin/bash
```

[1] http://www.mmm.ucar.edu/wrf/users/public.html

[2] http://www.open-mpi.org

[3] https://www.notur.no/hardware/stallo

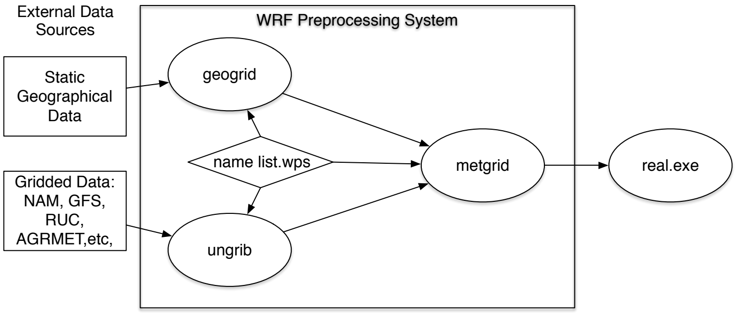


Figure 4.1 *The WRF Pre-processing System. From http://www.mmm.ucar.edu/wrf/users/docs/user_guide_V3/users_guide_chap3.htm.*

```
set -x

rm hosts.list

for i in {1..8}
do
echo "tile-0-3" >> hosts.list
done

./geogrid.exe
./ungrib.exe
./metgrid.exe

mpirun -np 1 -mca btl tcp,self --hostfile hosts.list ./real.exe

time mpirun -np 8 -mca btl tcp,self --hostfile hosts.list ./wrf.exe
```

In this script, first a list of host used for execution is established. In this example, the host named *tile-0-3* is repeated eight times. The *real.exe* is executed using *mpirun* for execution on a single process, i.e. on first host in the file *hosts.list*. The *wrf.exe* is executed using 8 processes on a list of hosts taken from the *host.list* file, in this case, using 8 processes on the same host. Some MPI optimization regarding communication between processes are given on the *mpirun* command line. The execution time for the *wrf.exe* applications is also measured and reported after each run using the *time* command.

The WRF model is built and executed with the limitations listed in Section 1.11 and stated earlier in this chapter. The WRF model divides the workload in a geographical way (in addition to other techniques), trying to keep most computations for a geographical area in a single process, for better use of caches and memory access.

The model implements a check for numerical instability and the used time-steps were after some experiments adjusted to keep the model stable. The version and parameters used were not optimized for the area. Much research within the meteorological community points to problems and solutions. See for example [64, 100, 101, 46, 98] for some cases where the model setup must be adjusted for specific use. High-resolution mesoscale models with mesh sizes of 1000 - 500 m are sensitive to the parameterization scheme chosen. The resolution used were therefore restricted to a lower limit of a mesh using 1000 x 1000 m cell sizes. Using much higher resolutions requires very high-resolution background geophysical data.

4.4 Experiments

A sample WRF session was generated using the WRF Portal[4] software by generating a starting set of controlling files for model execution. This tool generates the *namelist.input* and *namelist.wps* files that control the execution of the pre-processing system and the major parts of the model. The default generated setup of methods, algorithms and schemas used in the WRF model was not changed later in this work. To make informed and meaningful changes the setup would require detailed knowledge of the model and the available schemas and methods.

Several experiments were conducted, verifying that the system successfully could execute the models on all available clusters, one node only, and to ensure that the numerical stability of the model.

To establish the optimal number of local processes used by the model on each platform and for comparing with execution on several nodes, experiments were conducted on each available cluster. The model was executed several times using from 1 to 18 processes. The experiments using more than twice the available node CPU cores were using multiple nodes. An example of a result from such an experiment is found in Figure 4.2. The example is from the Display Wall cluster using the hardware and software listed in Section 4.4.1. The times are measured using the *time* command, and are the total time for a script that executes all pre-processing and model execution. Several measurements for each case were made to avoid effects of varying load on the cluster. In Figure 4.2 five measurements were made for each case. Each measurement is plotted. In most cases, the variation is so small that the individual measurements cannot be distinguished. For the purpose of this experiment, this method is sufficient to identify the best number of processes to use.

From this experiment, exemplified by Figure 4.2, it was decided to use 8 processes for executing the model. This is an expected result when using nodes with Intel quad-core processors with hyper-threading enabled. It should be stressed that this is the optimal for this specific build of the model, with the specific set of options used. The result does not necessarily imply the most optimal use with other builds and configurations.

One experiment was conducted to find a suitable large mesh size for the forecast. As shown in Chapter 3 the relaxation area will reduce the effective mesh area of a forecast. Using a larger mesh size will increase the computational time. Increasing the resolution with the same mesh size will also increase the number of time-steps needed and thereby increase the computational time. Execution time including both pre-processing and model is illustrated in Figure 4.3. The execution time increases linearly with increasing mesh size. In Figure 4.3 a 10 km resolution 12-hour forecast

[4] `http://esrl.noaa.gov/gsd/wrfportal/WRFPortal.html`

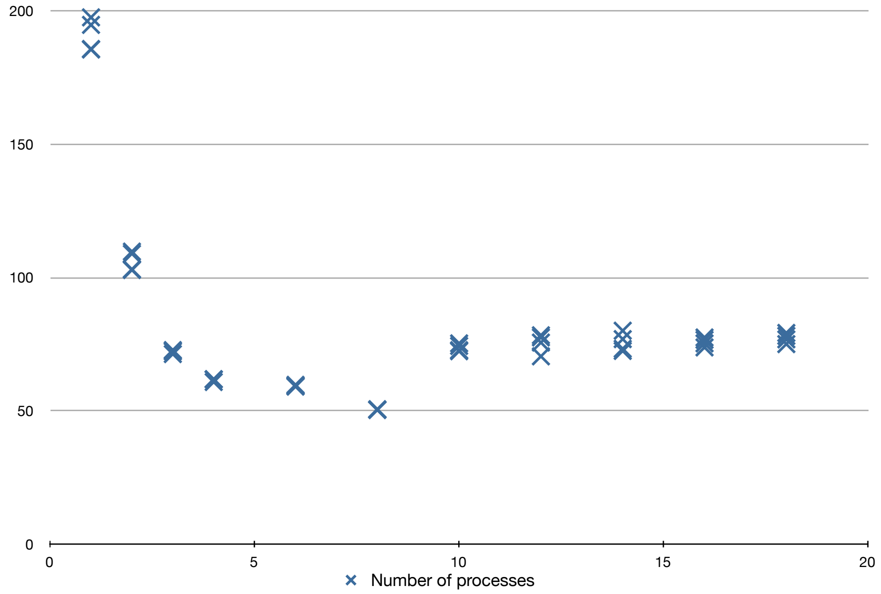


Figure 4.2 *Runtimes in the Display Wall cluster using different numbers of MPI processes on one or more nodes. Up to eight processes are on one node, more processes uses more nodes, up to 8 on each. Times are for executing a 3* km *resolution model for a 6 hour forecast.*

with increasing mesh sizes was computed. The execution time was measured using the *time* command. The measurement includes all processing for executing a model. The results show expected linear relation between the number of mesh points and the time used for computing the model. A mesh size of around 40 x 40 points was found to have an acceptable computational time on the hardware used for the purpose of the work presented in this dissertation.

4.4.1 Experimental Platform

The main cluster used for most experiments in this dissertation was the cluster in the Tromsø Display Wall [6](Additional paper 1). This cluster had the following hardware and software on each node.

- Intel Xeon processor W3550, 3.06 GHz, 8 MB cache, 1066 MHz memory, 4.8GT/s QPI, Quad-Core, HT, Turbo
- 12GBRAM(4x3)DDR31333MHz
- 1TB SATA Hitachi HDS72101
- 600 W power
- Rocks Linux distribution,version 5.4
- gcc version 4.1.2 compiler suite

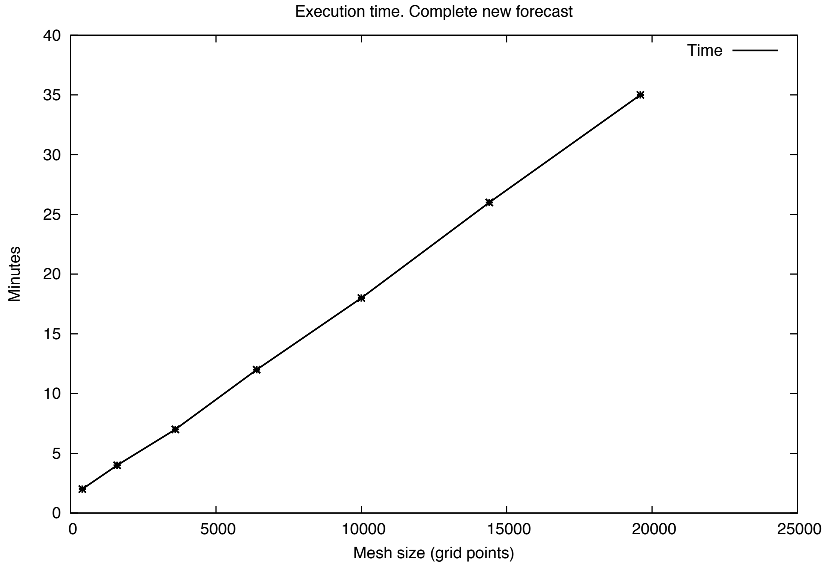


Figure 4.3 *Execution times vs. mesh size. Both pre-processing and model time is included. Mesh sizes from 20x20 to 140x140 points. 12 hour duration forecasts. Computed using one quad core node with hyper-threading enabled and 8 MPI processes.*

- Open MPI version 1.4.3
- Python version 2.7.2.

The hardware is representative of a 2010-2012 era commodity personal computer.

The research presented in Paper 1 [41] used an older compute cluster together with the Stallo supercomputer and an older version of the hardware behind the Tromsø Display Wall. The research using this setup is described in Chapter 7, with details of the hardware and software configuration.

4.5 Results

Experiments were conducted for establishing the highest resolution possible when executing the model on commodity hardware and at the same time being meteorological sound and providing forecast details not available today.

In Section 4.3, the maximum resolution used in this work was limited to 1000 m. The resolution used is limited by the model configuration available without expert guidance, and the available background geophysical data. Higher resolutions are possible but have not been investigated in this dissertation. 1000 m resolution forecasts are for some areas, an order of magnitude better than what is available today.

Table 4.3 *Windspeed forecasts statistics*

Location	Mean Error, ms^{-1}	Correlation	N
90450 TROMSO	-1.50	0.45**	58
90490 TROMSO LANGNES	1.15	0.65**	61

Mean error is the mean of the difference between the observed value and forecasted value. One forecast per day of June and July 2013. ** $p < .01$.

An example of a forecast is illustrated in Figure 4.4. Validating these forecast using instrumental measurements is very difficult as there is only two measurement points within the area, one point at the airport and one point on the top of the Tromsø Island. The locations are indicated by crosses on Figure 4.4. Both places are well away from topographical elements that would have a strong influence in a very high-resolution forecast. An example of two time series of forecasts for the two locations with the corresponding observations is given in Figure 4.5. The time series shows good correlation between the forecasts and the observations. The statistics for these series are given in Table 4.3.

In the illustrated situation, there is an area of calm winds in the fjord to the east of the University. Such features are often observed and Figure 4.4 illustrates that they are reproducible using a model with high enough spatial resolution. Experiments using 10, 3 and 1 km was executed for comparing different spatial resolutions.

The winds around the Tromsø Island are very much influenced by the local topography. Figure 4.4 is a good example of a forecast showing meteorologically sound features that are not publicly available, and thereby giving additional value for the user of the forecast. Many of the interesting features come from the increase in resolution and the increase in the models capability of resolving flow patterns at higher resolution.

The WRF model is used extensively for research and operational forecasting at various resolutions. A large number of studies have validated the soundness of the WRF model [75, 96, 62, 17, 54, and references within]. The work presented here has some similarities to the work of downscaling climate models to higher resolutions. Lo et. al. [70] presented work showing that dynamic downscaling using the WRF model outperformed bi-linear interpolation, particularly for near-surface phenomena in mountainous regions. Executing the WRF model for small areas using low- or medium-resolution meteorological backgrounds is conceptually similar to downscaling climate models.

4.6 Conclusion

The experiments presented in this Chapter and in Paper 1 and 2 documents that a modern numerical atmospherical model can be executed on single commodity computer. The execution times can be controlled by varying the size of the mesh used, the spatial resolution of the model and the length of the forecast. The highest resolution used in the experiments was not the highest technically possible, but does provide meteorologically sound forecasts even when executed for a small area, with details not publicly available at this time.

The way the WRF model is executed using two main configuration files that is shared

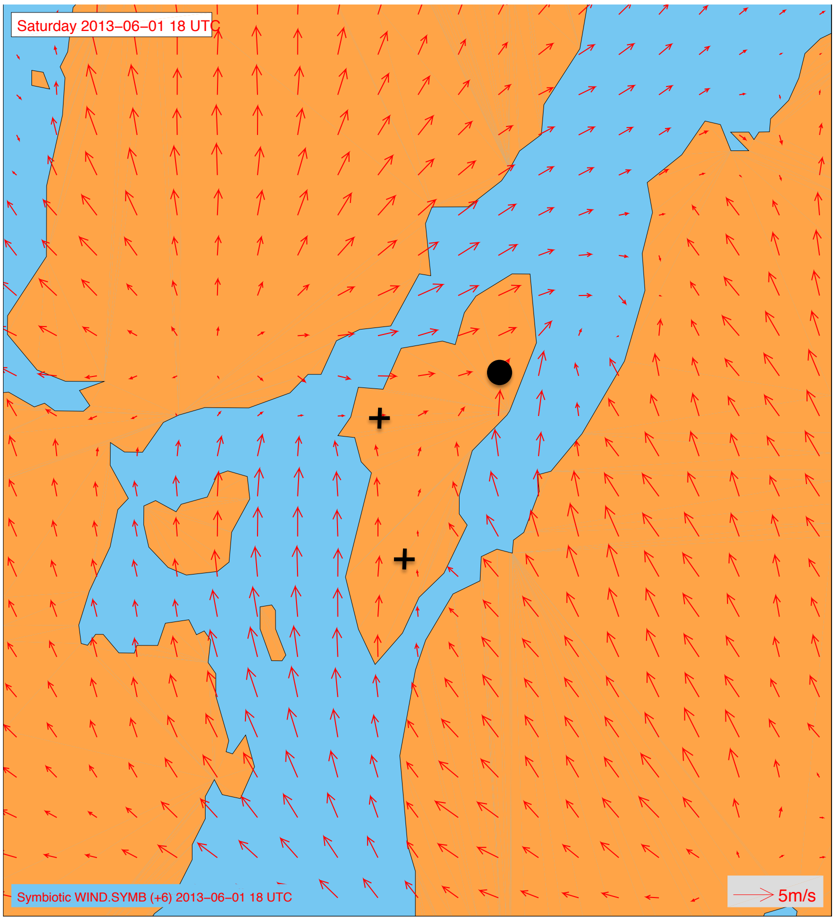

Saturday 2013–06–01 18 UTC
Symbiotic WIND.SYMB (+6) 2013–06–01 18 UTC
5m/s

Figure 4.4 *Example of 1* km *WRF forecast. a 6-hour forecast. Wind arrows in red with legend in the lower right side. North is up in the figure. Crosses indicate locations with observations. Circle indicates location of the University of Tromsø.*

between the pre-processing and main model execution, allows for an architecture of a system utilizing very few user inputs and still positioning the model on any location on the globe.

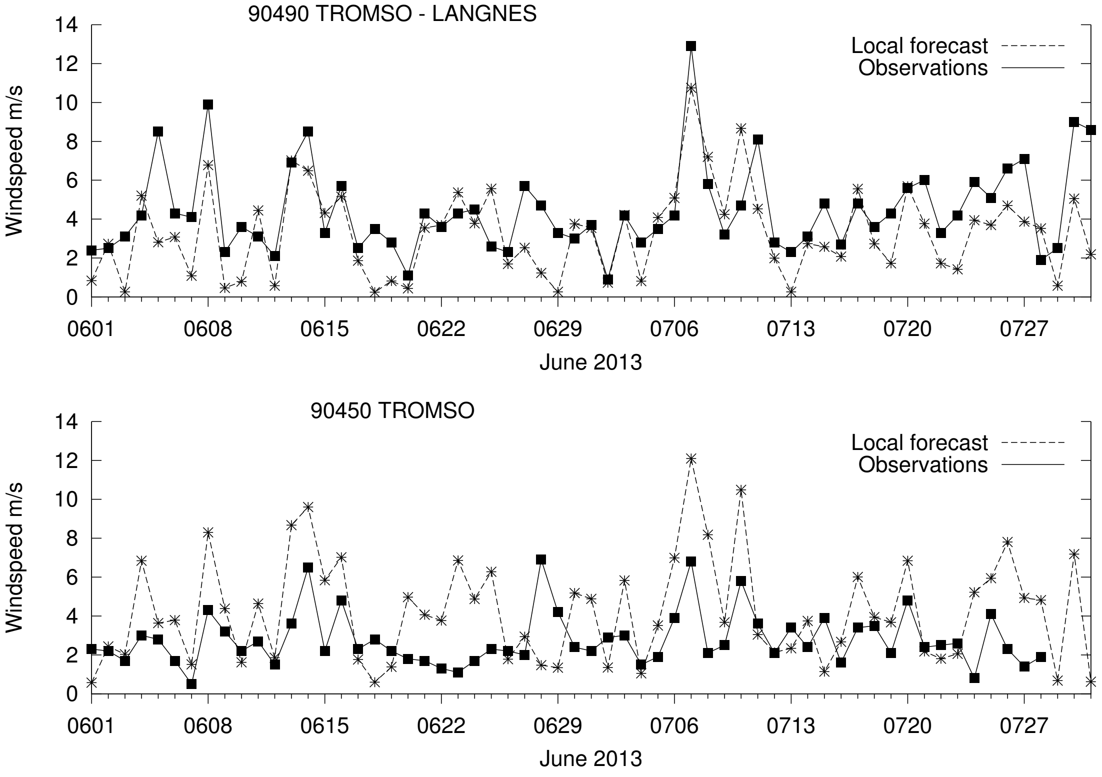


Figure 4.5 *An example of two time-series of forecasts and observations. A 6-hour forecast against the corresponding single wind speed observation. Solid lines and squares are observations. Dashed lines and stars are forecasts.*

5

Symbiotic Forecasts

This chapter investigates issues related to the research statement *Build a system for combining local weather forecasts with the purpose of identifying the characteristics and performance of an architecture, design and implementation done for a commodity platform*.

The basis for research statement is the findings from the previous chapter that forecasts can be computed on commodity hardware. Given many such forecast all computed for their own location and/or time period, the forecasts could be combined in several ways. Three of these ways are discussed here with the focus on the non-trivial combination. A system for mutually enhancing personalized weather forecast by exchanging and statistically processing individual forecasts is presented.

The work in this chapter is for some parts, inspired by the guidance given in Jakeman et. al. [61] for development and evaluation of environmental models. The guidelines are made for working with new or improved models. In this dissertation, output from an established model is used in a new way for investigating and estimating one aspect of the forecasts produced, and therefore the work is restricted to the relevant parts of the guidelines, [61, section 3.9]. The discussions in the special issue of Environmental Modeling & Software [63] on uncertainty treatment is also relevant, but does not explicitly discuss uncertainty originating from minor differences in how a model represents high gradient features, like topography, between forecasts computed with slightly different center locations. A major difference in the method presented here, is that it does not focus on investigating uncertainty resulting from one or a few parameters, but the combined effect of randomized variation in potentially all surface parameters.

5.1 Idea

Individual computed local numerical forecast have inherent uncertainties originating from several sources. Some are listed below with a short description. The list is not meant to be exhaustive.

> **Model description.** Modern numerical atmospheric models have a very large selection of options on how to represent a feature. Depending on the representation or model description of the feature, slight errors will be present in the results. Early model developers like Charney and Eliassen [19] using very simplistic representation of many features and thought this was the most prominent source of model errors and uncertainties.
>
> **Model resolution.** The atmosphere has movements and phenomena at all scales, from planetary waves with wavelengths in orders of hundreds of km to micro turbulence on mm scale. The spatial resolution in the numerical model determines what phenomena are solvable and may be represented in the model. Usually this is taken to be phenomena's with a scale of 4-6 times the mesh size used in the model. Phenomenas not explicitly resolved in the model may be parametrized.
>
> **Background geophysical data.** The available geographical background data from the WRF model sites have a spatial resolution of $30''$, ≈ 920 m. This is the best global dataset available. Using the scale considerations above, this implies that there are features with size less than around 4 km, is not described with good accuracy. Features smaller than 1 km is not described at all, and both scales will affect the actual weather in an area.
>
> **Background meteorological data.** The background data used for driving the WRF model either have uncertainties from the initial analysis of the current state of the atmosphere or if a short forecast is used, from the model used to produce the background meteorological data.

EPS systems are based on the idea that small differences in the initial meteorological conditions will produce variations in the forecasts produced. The ECMWF [30, 31] currently produce a main global forecast with approximate 16 x 16 km resolution. In addition, 51 global forecasts with a resolution of around 30 x 30 km are produced where each forecast uses a slightly different meteorological initial condition. The difference between the initial conditions used in the forecasts is small and is within the uncertainty of the initial analysis used for the main forecast. All the 51 forecasts in the EPS use the same model mesh and setup. These EPS forecasts are used for statistical-type forecasts, for example the risk of some parameter exceeding certain thresholds. EPS forecasts are used for estimating the uncertainty in the forecasts.

To be able to produce some kind of error or uncertainty estimation for a local personalized forecast, the idea in this Chapter is to combine several forecasts created with slightly different center locations.

5.1.1 Trivial Combination of Local Forecasts

The trivial combination of local forecast can follow two dimensions, time and space. Assuming that several local forecasts each covers a given time period and the same region, the combined forecasts will cover a longer period with or without overlap or gaps. In this way, interesting time periods can be covered, without one user having to generate a forecast for the longer period.

Assuming several local forecast each covers the same period but different regions, the combined forecasts will cover a larger region with or without overlap for this period in time. In this way large regions can be covered with a forecast without one user having to generate a forecast for a potential larger region.

These two methods of combining forecast do not contribute to the estimation of uncertainty in the forecasts. These combinations will extend the forecasts in space and time and may produce a single very high-resolution forecast not elsewhere publicly available.

5.1.2 Combining Overlapping Forecasts

If the forecasts are produced with slightly different mesh locations, the pre-processing will generate slightly different geophysical backgrounds. The variation will be largest for features with high gradients, like topography. The variation in the background data will introduce differences in the forecasts produced. For small regions with strong gradients in topography, this will influence many aspects of the forecasts.

By amalgamating many forecasts with overlapping regions, several statistical properties can be estimated. The variation of a forecasted value around each mesh point represents on of the possible uncertainties in the forecast.

This approach is similar but not identical to the work done by Berner et. al. [10] where the uncertainties in three surface parameters albedo, soil moisture availability and roughness length are randomly perturbed one by one. They also use perturbed initial meteorological conditions; different physics suites ie. ways of representing physical processes in the model, and a scheme for representing the interactions in the model with the unresolved spatial scales. Berner et. al. also use the WRF ARW 3.1.1 model as a basis in their work. Berner et. al. conclude that including some representation of model errors leads to ensemble systems that produces significantly better probabilistic forecasts.

5.2 Architecture and Design

The architecture and design of a system for producing uncertainty estimations from local forecasts is illustrated in Figure 5.1. The architecture describes a system that collects many individual forecasts into one *symbiotic forecast*. The individual forecast are all produced with slightly different center locations and have some overlapping regions. The design allows the symbiotic forecast to be generated utilizing any of the local individual forecasts as a basis. The term *symbiotic forecast* originates from the mutual benefits obtained by exchanging local forecasts and combining theses for creating uncertainty forecasts. This will be further elaborated in Chapter 9.

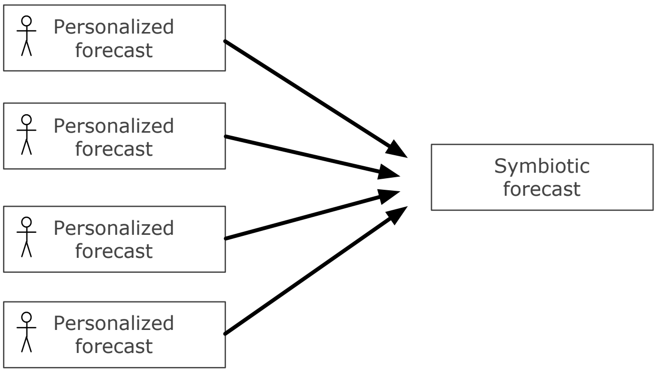


Figure 5.1 *An illustration of the architecture and design for creating symbiotic forecasts.*

5.3 Implementation

At least two approaches can be used for implementing and visualizing symbiotic forecasts. One is to collect all forecast and present all to the user, using each forecast original mesh and location. The second approach is to use a users own personalized forecast as a basis and incorporate the collected forecast into the forecast using the local forecast's mesh.

The first approach can easily be done using the commonly available visualization tools described in Section 8, provided the tools have support for combining data from many files in the same visualization. The second approach provides statistics like count, maximum, minimum, mean and standard deviation for each mesh point and each parameter in the data files. The second approach was implemented as a prototype that creates a new NetCDF file with the amalgamated data for a selected set of parameters.

5.4 Experiments and Results

Two experiments were conducted with the purpose of investigate the characteristics of the proposed method of executing multiple forecasts using the same model but with slightly different center locations.

The first experiment was intended to establish the level of uncertainty of wind speed produced by this method for a given situation. The uncertainty was measured using the standard deviation of the wind speed from all forecasts.

The second experiment executed the same forecasts once every day for a time of two months and retrieved a 6 hour forecasted wind speed for two locations. This experiment was intended to show the daily variation in the uncertainty estimation and allowed for comparison with two measurements of wind speed within the forecast region.

The experiments were conducted using the 28 compute nodes of the Tromsø Display Wall. See 4.4.1 for details on hardware and software setup. The WRF model was executed using a base location and each node in the compute cluster, except one, would add a random displacement within a 10 km radius, giving 1 original forecast and 27 displaced. The original forecast is thereafter used as a basis for the following figures.

Figure 5.2 show how many additional forecasts are available for any point in the original forecast. The maximum number is 27 and is achieved in the center part of the region. Figure 5.3 illustrates the variation the pre-processing generates represented by the standard deviation of the terrain height in the model within a region. The largest variation is found in regions with steep gradients.

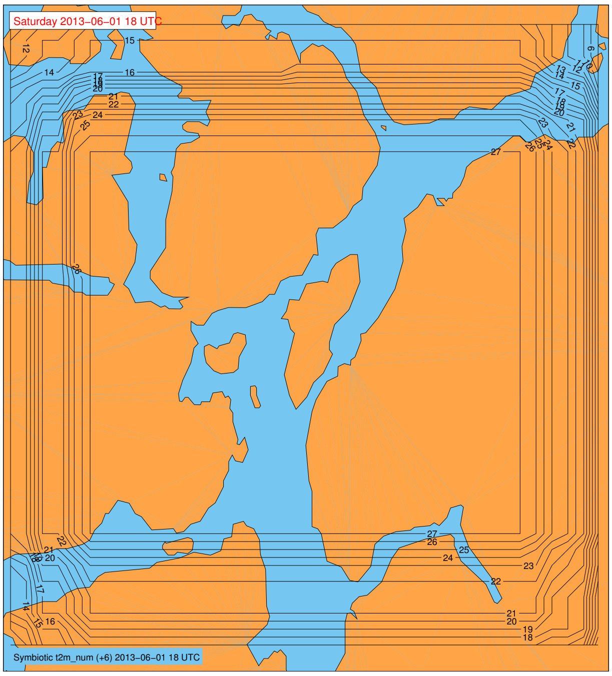

Saturday 2013-06-01 18 UTC
12
14
15
16
21
22
23
24
25
26
27
6
15
17
20
23
24
21
22
27
26
25
24
23
22
21
20
19
18
17
16
14
15
Symbiotic t2m_num (+6) 2013-06-01 18 UTC

Figure 5.2 *Varying number of models used in an amalgamated forecasts within a region. Values range from 6 to 27.*

The effect of variation in terrain in the forecasts is illustrated with Figure 5.4 showing the standard deviation of wind speed in a 6-hour forecast. The mean wind speeds for the same forecast are illustrated in Figure 5.5. Variations in wind speed are in the order of 0.2 to 1.0 ms^{-1} and the mean wind speed of all forecasts is in the rage from 0.5 to 3.0 ms^{-1}. The strongest variation in wind speed is often, but not always found near regions of strong gradients in the wind speed. The largest values if variation in wind speed does not coincide with the largest variations in the terrain representation. This can be seen by comparing Figure 5.3 and Figure 5.4.

Figure 5.6 illustrates combining the local wind forecasts as wind arrows and uncertainty as a contoured colored field. Presenting uncertainty together with the forecasted winds can be helpful for users evaluating both the forecasted wind speed and uncer-

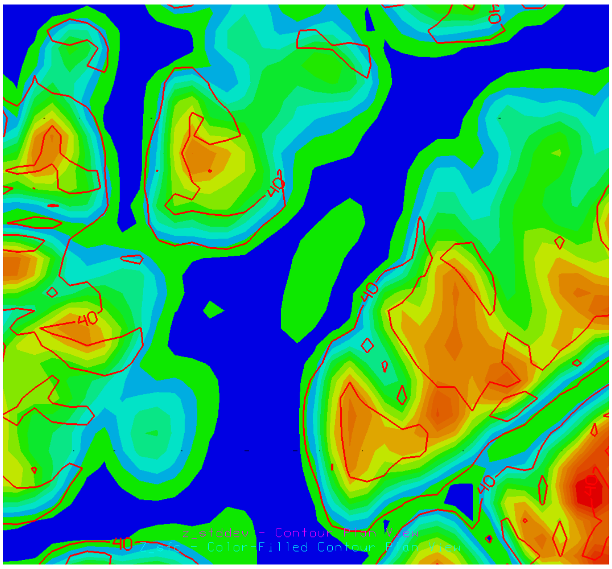

40
40
40
40
40
40
z_stddev - Contour Plan View
Z_sfc - Color-Filled Contour Plan View

Figure 5.3 *Standard deviation of model terrain height as red isolines plotted on a background colored by the terrain height. Values range from 0 to 60* m.

Table 5.1 *Windspeed forecasts statistics with symbiotic forecasts*

Location	**Mean Error,** ms^{-1}	**Correlation**	N
90450 TROMSO	-1.35	0.40**	58
90490 TROMSO LANGNES	0.92	0.60**	61

Mean error is the mean of the difference between the observed value and the mean value of all 28 forecast used. One value per day of June and July 2013. ** $p < .01$.

tainty over an area.

The plot in Figure 5.8 shows the effect of adding the statistical distribution of the many forecast, including the mean value, to a series of forecasts. It also shows the value for the initial forecast used as a star. The figure uses the only two measuring stations available in forecast region. The stations are approx. 2 km from each other. The location of the two stations is indicated by black crosses in Figure 5.6. Figure 5.7 shows an explanation of the plot for each day.

Statistics comparing observations and the mean wind speed from the symbiotic forecast is given in Table 5.1. The differences in mean error of windspeed between forecasts using the local forecast or using the mean of the symbiotic forecasts, is not statistical significant in these two cases, $t = -.34, p = .63$ for Tromso and $t = -.34, p = 0.26$ for Tromso Langnes. A user would do equally well using either the single local forecast or a mean of the symbiotic forecasts for forecasting wind speed for these two locations in this time period.

Figure 5.9 and 5.10 shows the observed windspeed plotted against the local forecast to the left, and against the mean of the symbiotic forecasts on the right. Data is from June and July 2013. The symbiotic forecasts are plotted using the mean value and

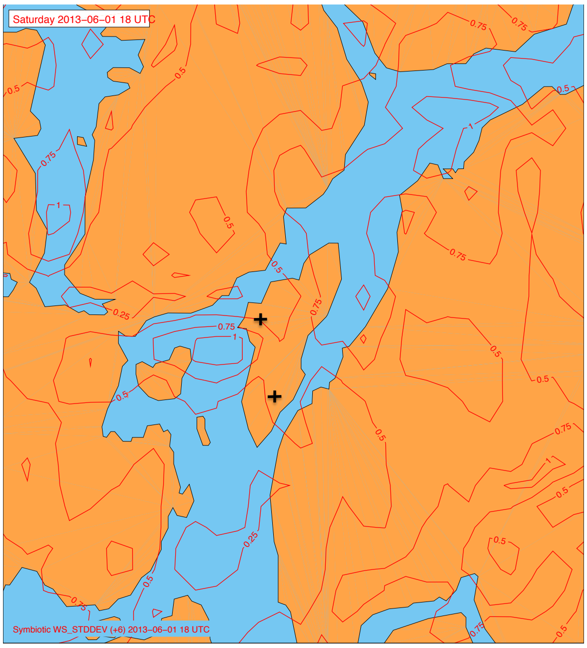

Saturday 2013–06–01 18 UTC
0.5
0.75
1
0.25
0.5
0.75
0.5
0.25
Symbiotic WS_STDDEV (+6) 2013–06–01 18 UTC

Figure 5.4 *Standard deviation of Wind Speed. Values range from 0.2 to 1.0 ms^{-1}.*

wishers for one standard deviation, one data item per day. A perfect forecast will lie on the diagonal. The figures illustrate the skill of the local forecast and of the symbiotic forecast when adding uncertainty.

5.5 Discussion

Uncertainty estimation of forecasts is by many meteorologists an essential part of the forecast. Some have stated this very strongly, one example is the 133 newsletter of the ECMWF [30, p. 1] where Alan Thorpe stated:

> *What is clear is that no forecast is scientifically credible without an estimate of its level of uncertainty.* (Alan Thorpe)

A measure of the uncertainty originating from the specific mesh placement used in the models is not currently available from any weather service. As far as can be established using web searches and private conversations with persons in this field of work, no weather service provides any uncertainty estimate of this type for their forecasts. Other measures of the quality and accuracy of forecasts are used, some originating from the original work of Allan Murphy [79].

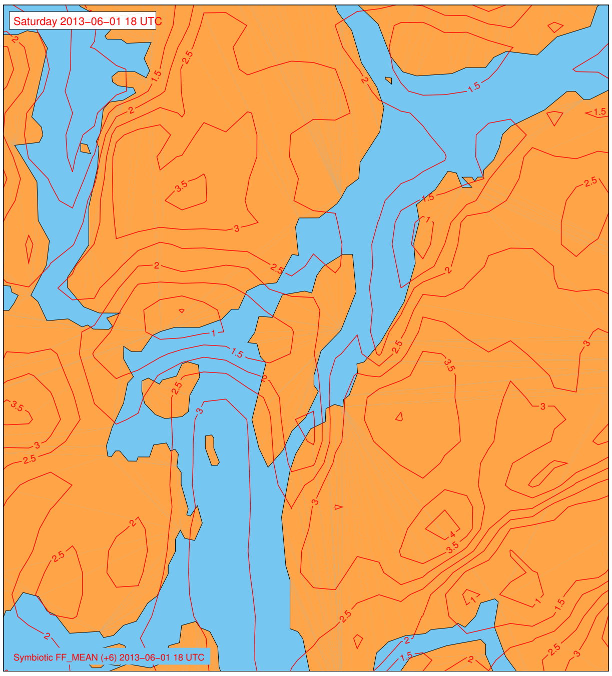

Saturday 2013–06–01 18 UTC
Symbiotic FF_MEAN (+6) 2013–06–01 18 UTC

Figure 5.5 *Mean value of Wind Speed. Values range from 0.5 to 3.0* ms^{-1}.

The benefit of having uncertainty estimations for the forecasted value is that the user will be more capable of doing a risk assessment based on the forecast. One scenario where this might be useful is for heavy lifting situations. The local symbiotic forecast provides wind speed forecasts with uncertainty estimations for both ground level and at several levels in the vertical.

5.6 Conclusions

Combining several forecast executed using the same model and background data, but with slightly different center location leads to small differences in the forecasts. These differences are related to the minor differences in the background geophysical data used in the model, generated by the pre-processing programs.

Based on the research by Berner et. al. [10] it is highly likely that the approach of combining multiple local overlapping forecasts will add some measurement of the model uncertainty. A detailed analysis of this type of uncertainty would require extensive research by experts in the field of numerical atmospheric modeling. One of the problems that needs to be established is if computing models using random offsets in grid placement would be similar to adding random variations to the topography and executing the model with the same mesh location. The latter variant is shown in Berner et. al. to represent model errors.

The levels of variation is of the same order as the variation observed in operational EPS forecasts from the ECMWF, although the numbers from the period covered in the experiments is not publicly available. A comparison between the method used here and some EPS system is possible future work.

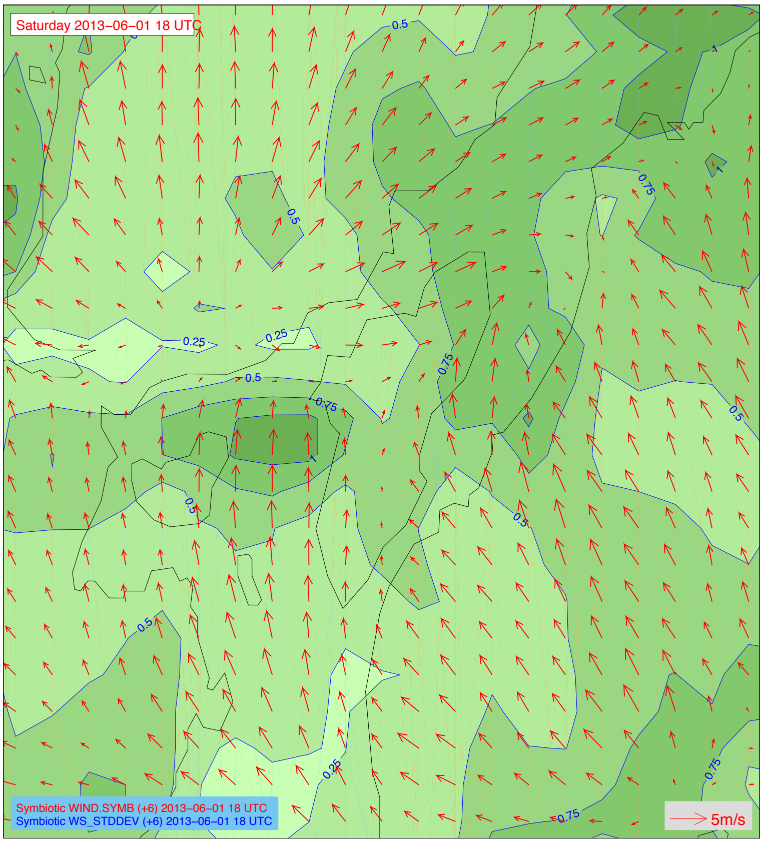

Saturday 2013–06–01 18 UTC
Symbiotic WIND.SYMB (+6) 2013–06–01 18 UTC
Symbiotic WS_STDDEV (+6) 2013–06–01 18 UTC
5m/s
0.25
0.5
0.75
1

Figure 5.6 *An illustration of standard deviation of wind speed plotted as iso-lines with a transparent color filling and single forecast wind arrows. Darker colors indicate regions of higher standard deviation. Maximum values of the standard deviation are slightly more than 2 m/s, strongest wind-speeds are in the range of 3 to 5 m/s.*

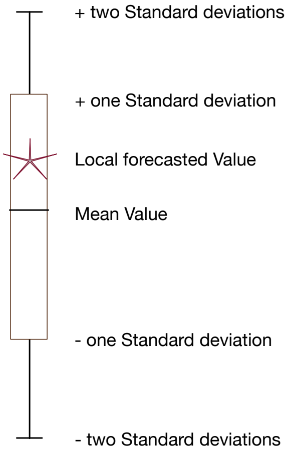


Figure 5.7 *Explanation of time-series plot with symbiotic forecasts. From the top is indicated +two standard deviations, +one standard deviation, the local forecast, the mean value of all symbiotic forecast, -one standard deviation and -two standard deviations.*

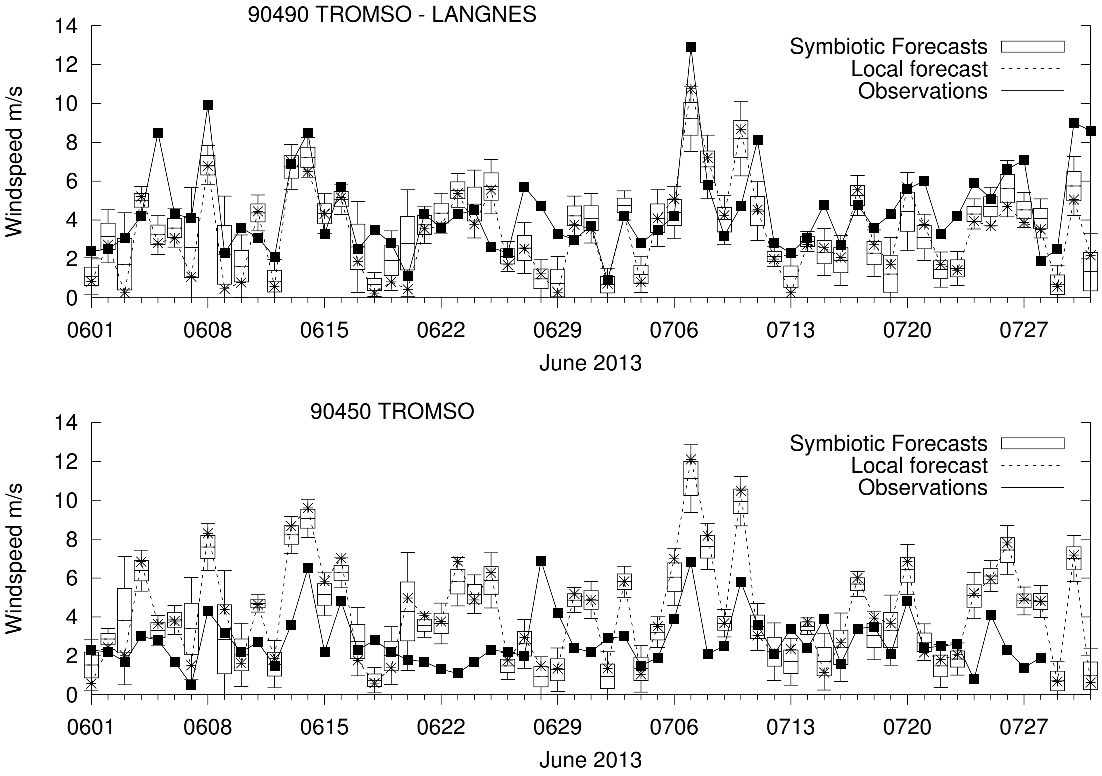


Figure 5.8 *Time-series of symbiotic forecasts. The candlestick with whiskers shows the statistical distribution of the forecasts. The star shows the originating forecast. The solid blocks and lines show the observed values.*

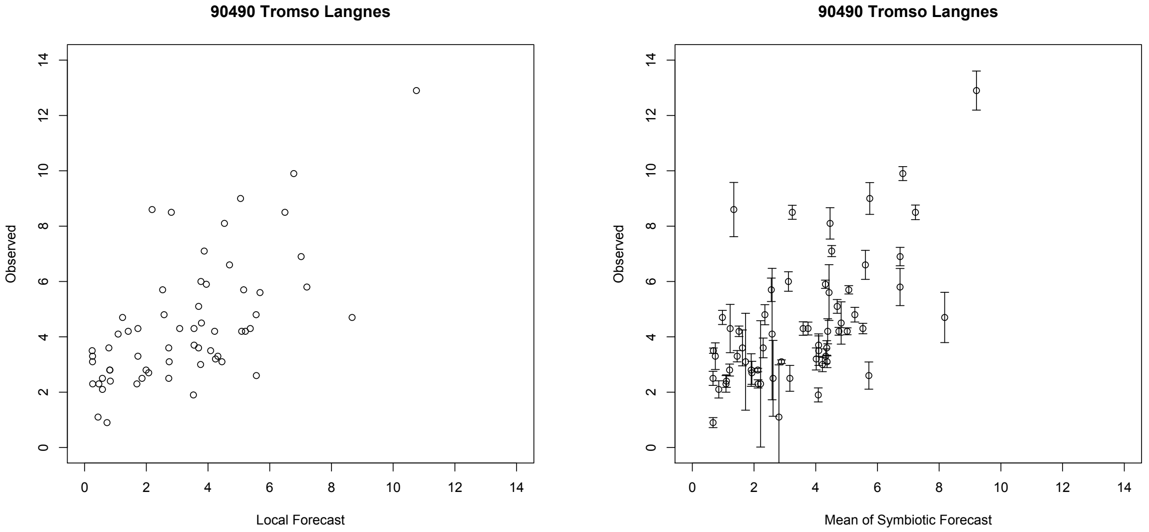


Figure 5.9 *Scatterplot of observed and forecasted windspeed for station 90490 Tromso Langnes. Right side illustrates one standard deviation uncertainty.*

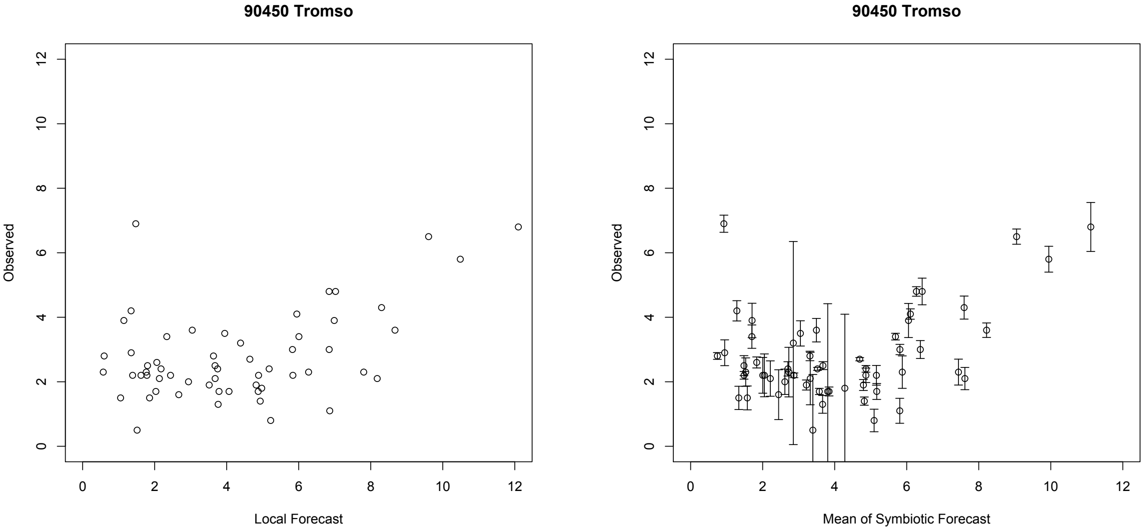


Figure 5.10 *Scatterplot of observed and forecasted windspeed for station 90450 Tromso. Right side illustrates one standard deviation uncertainty.*

6

On-Demand Small Region Very High-Resolution Forecasts

This chapter presents the development of a system for *on-demand* creation of weather forecasts using a numerical atmospheric model on a typical desktop computer. This system is a response to the research statement: *Build a system for computing on-demand very high-resolution interactively fast meteorologically sound forecasts with the purpose of identifying the characteristics and performance of an architecture, design and implementation done for a commodity platform.*.

6.1 Idea

Is it possible to make a very high-resolution numerical forecast for small areas available to a user on-demand and what are the characteristics of such a system?

6.2 Architecture

Chapter 4 presented a system for executing the WRF model on a desktop computer. The model implementation will use all of the CPU resources available. The architecture for the execution system describes a single process and entry point that will execute the model. The only input needed is the geographical center of the forecast area requested. Additional information may be given and used. The architecture is illustrated in Figure 6.1.

The architecture is based on two abstractions; the forecast presentation and the forecast production. The forecast presentation abstraction encompasses all necessary pre- and post-processing in addition to the actual production of the forecasts.

Any visualization should be focused on the small area covered by the model output and should focus on the users actual location. This is assumed possible with openly

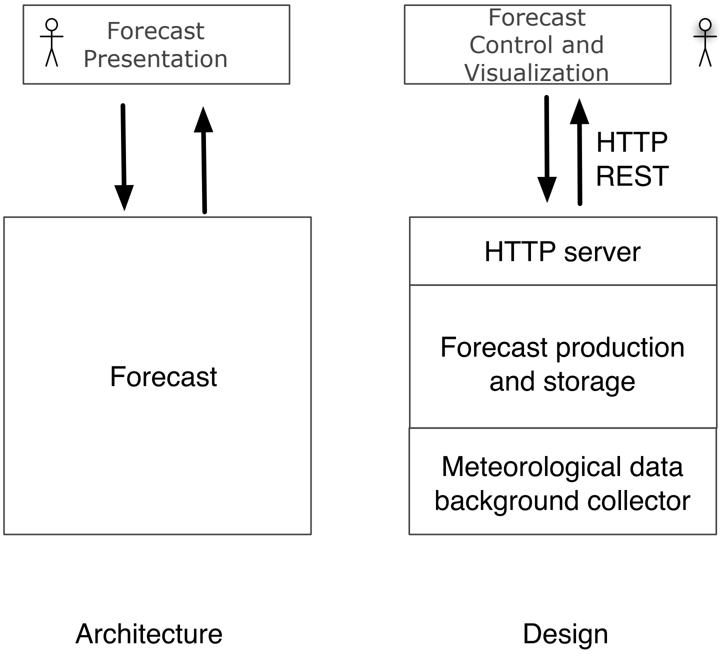


Figure 6.1 *Overview of the architecture and design of an application for producing numerical weather forecasts on a desktop computer.*

available software. The idea of the architecture is to separate the production of forecasts from the visualization or other use.

6.3 Design

The design is illustrated in the right hand side of Figure 6.1. The design limits the forecast production abstraction to a HTTP server executing requests for new forecasts using a singleton pattern to ensure that only one request are being served at any time. One effect of this is to ensure that multiple simultaneous request from different front-ends do not interfere with each other. The use of HTTP for communication is decided at design level limiting and narrowing the implementation space. The design further specifies that the implementation must use a HTTP REST [36] style for serving requests. The design splits the forecast production into three tightly coordinated services. The HTTP server acts as a front-end handling HTTP requests, effectively acting as an API for the visualization side. The forecast production executes the numerical atmospheric model at requests and stores the results for later retrieval. The meteorological background data collector is scheduled for times when new data are known to be available. The set of collected background meteorological data must match requirements of the possible forecast time and duration used by the forecast production service.

The forecast visualization does not have specific design level requirements other than using the HTTP REST API for requests.

6.4 Implementation

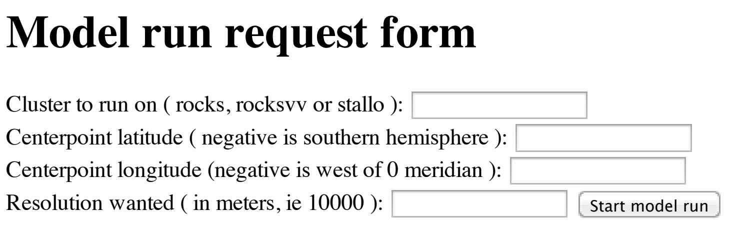
The implementation of the forecasts production system consists of a set of applications and scripts. The visualization side uses a web browser and industry standard visualization tools.

6.4.1 Hardware Used for Development

Executing the WRF model implementation on-demand was done on one node of the Tromsø Display Wall cluster [6] (Additional paper 1), using the hardware and software specified in Section 4.4.1

6.4.2 HTTP front-end

The system for on-demand weather forecasts was set up as a web page, illustrated in Figure 6.2. The user can manually enter which compute cluster to execute the model on, using one node only, which geographical location to use as the center location and the wanted resolution in meters. Available resolutions are 10 000, 3 000 and 1 000 m. The resolutions are somewhat arbitrarily selected, but the highest resolution is at the limit of this type of model use, as discussed in Section 6.4.3.



Model run request form

Cluster to run on (rocks, rocksvv or stallo):

Centerpoint latitude (negative is southern hemisphere):

Centerpoint longitude (negative is west of 0 meridian):

Resolution wanted (in meters, ie 10000):

Figure 6.2 *A web form for on-demand weather forecasts. The page will reload when the execution of the model is complete with a list of available data files.*

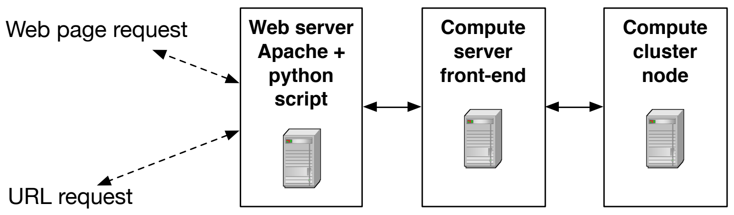


Figure 6.3 *Illustration of the orchestration of services and host for on-demand weather forecasting. All services may be executed on the same computer.*

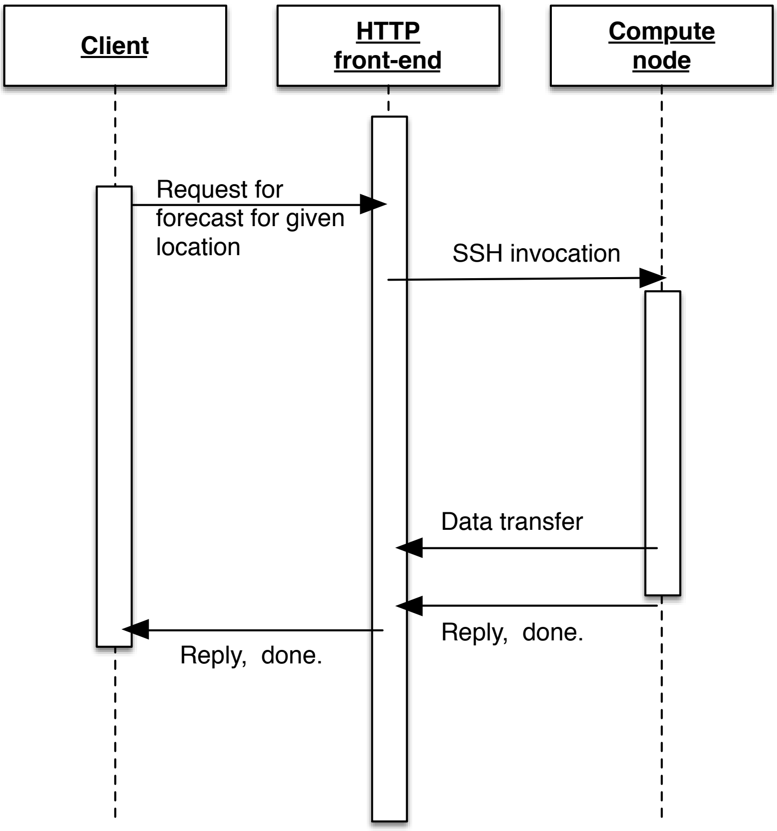


Figure 6.4 *An illustration of the messages and processing between participating entities for producing an On-demand localized very high-resolution forecast.*

The web page is hosted on an older personal computer running an Apache web server with the mod_ python module. The page is only available to authenticated users, using basic HTTP authentication. The request generated by the page is serviced by a python script that extracts the input parameters and executes a script on the front-end of the wanted cluster using SSH. One option not shown in Figure 6.2 is to execute a script on "localhost" if the server is running in a virtualized environment or on a sufficiently powerful computer capable of executing all the services including the WRF model implementation.

Since the result from filling out the web-page form can be duplicated with an URL request to the same Apache web server, the python script also forms a REST API for executing the model from any remote application. An illustration of this orchestration is given in Figure 6.3 with the messages, commands and processing illustrated in Figure 6.4.

6.4.3 Controlling the WRF Model

The pre-processing and WRF model is controlled by two large setup files, *namelist.wps* and *namelist.input*. Both files contain a long list of options controlling the model execution. The main options used here are the period wanted, the geographical location of the center of the model, and the resolution and time-step length wanted. All these elements can be generated on the fly, using input from the URL request. Templates for generating the input files indicating the variable portions are shown in Listings 6.1 and 6.2. Various parameters that are set on-demand are indicated with arrows and a short text to the right.

Listing 6.1 Template for generating the namelist.wps file

```
 wrf_core = 'ARW',
 max_dom = %d,                      <-- If executing in nested conf.
 start_date = '%s_12:00:00',        <-- Starting time
 end_date   = '%s_18:00:00',        <-- End time
 interval_seconds = 21600,
 io_form_geogrid = 2,
 opt_output_from_geogrid_path = '/home/baardf/WRF/DAGLIG/',
 debug_level = 0,
/

&geogrid
 parent_id         = 1,1,2,
 parent_grid_ratio = 1,3,3,
 i_parent_start    = 1,14,15,
 j_parent_start    = 1,15,15,
 e_we          = 39,43,49,
 e_sn          = 41,43,49,
 geog_data_res = '30s','30s','30s',
 dx = %s,                     <------ Resolution
 dy = %s,                    <------ Resolution
 map_proj =  'polar',
 ref_lat   = %2f,            <------ Geographical location
 ref_lon   = %2f,            <------ Geographical location
 truelat1  = %2f,            <------ Geographical location
 truelat2  = 90,
 stand_lon = %2f,            <------ Geographical location
 geog_data_path = '/home/baardf/WRF/GEO',
..
..
..
```

Listing 6.2 Template for generating the namelist.input file

```
&time_control
run_days                = 0,
run_hours               = 6,
run_minutes             = 0,
run_seconds             = 0,
start_year              = %s,      %s,      %s, <— Starting time
start_month             = %s,      %s,      %s, <— Starting time
start_day               = %s,      %s,      %s, <— Starting time
start_hour              = 12,      12,      12,
start_minute            = 00,      00,      00,
start_second            = 00,      00,      00,
end_year                = %s,      %s,      %s, <— Ending time
end_month               = %s,      %s,      %s, <— Ending time
end_day                 = %s,      %s,      %s, <— Ending time
end_hour                = 18,      18,      18,
end_minute              = 00,      00,      00,
end_second              = 00,      00,      00,
interval_seconds        = 21600,
input_from_file         = .true., .false., .false.,
history_interval        = 180,      60,      60,
frames_per_outfile      = 1000,    1000,    1000,
restart                 = .false.,
restart_interval        = 5000,
io_form_history         = 2,
io_form_restart         = 2,
io_form_input           = 2,
io_form_boundary        = 2,
debug_level             = 0,
/

&domains
time_step               = %d,           <— Lengt of time-step
time_step_fract_num     = 0,            <— Fraction of time-step
time_step_fract_den     = 1,            <— Fraction of time-step
max_dom                 = %d,
s_we                    = 1,       1,       1,
e_we                    = 39,      43,      49,
s_sn                    = 1,       1,       1,
e_sn                    = 41,      43,      49,
s_vert                  = 1,       1,       1,
e_vert                  = 28,      28,      28,
num_metgrid_levels      = 27,
..
..
..
..
```

Most of these large configuration file templates are constant between each execution of the model. Many parameters could be made into controllable parameters. For example, the forecast length is specified to six hours, and could be made optional and available as a parameter for the HTTP request. In addition, the starting time is a parameter suitable for user control. The forecast length have been kept fixed in this work, only automatically adjusting the start and stop times. Other parameters would require a thorough knowledge of the model and detailed insight in the implications of choosing one or another option.

The scripts on the compute clusters front-ends are responsible for creating the correct input files for the WRF model using the geographical location as center location for the model, and executing the necessary pre-processing before executing the WRF model on one of the nodes. Some of the compute clusters used have shared file-systems, thus avoiding the need for moving files around. The scripts executing on a compute cluster front-end do not return before the execution of the WRF model is complete and the resulting file have been transferred to a location available to the webserver.

All these processes may be executed on the same host. The nodes on the compute cluster used (see Section 4.4.1) are an example of sufficient compute power for executing all processes on a single host. Except for the WRF-model related processes, most scripts and applications are lightweight and do not require a powerful personal computer.

6.4.4 Background Meteorological Data

As part of the WRF system, a static set of background geographical data including topography, for the whole globe is available. For initial disk-space reasons, a set covering Scandinavia was extracted and available. The whole dataset is around 10Gb in size, easily within the available space on modern desktop computers.

A system for retrieval of background meteorological data from a weather service is also needed. GFS data from NOAA[1] is the preferred source for day-to-day forecasts. The GFS data is available with forecasts periods of up to +120 hours and is updated twice per day with forecasts starting at 00 and 12 UTC. The background data must cover the period for the intended on-demand forecast. A static set of files covering the period 12 UTC to 24 UTC on the same day were retrieved, covering the local afternoon and evening of the same day. With the GFS background meteorological data, a forecast can be computed for any location on the globe. The GFS data have a resolution of around 50 km.

The GFS data is produced and disseminated at fixed times each day. The background data from the 00 UTC GFS forecast is retrieved using a script executed using the Linux *crontab* system at 06:12 UTC each morning. This is some time after the initial availability to avoid the heaviest traffic on the NOAA web site, and still early enough for practical use in Norway.

[1] http://nomads.ncep.noaa.gov/

Table 6.1 *Some post-processed data formats*

Format	Purpose	End device or application
NetCDF	Default output	Open source application
NetCDF CF	CF convention metadata	DIANA on the Display Wall
KML [82] arrows	Mobile device	AR app or Google Earth
KML Streamlines	Mobile device	AR app or Google Earth
PNG	Mobile or tiled use	Google maps or Display Wall
JSON	Smart watch	Pebble smart watch

The DIANA and AR applications are discussed in Chapter 8.

6.4.5 Post-processing of WRF Model Output

The WRF model produces NetCDF files as standard. The post-processing generates the formats listed in Table 6.1 for various visualization purposes. For simplicity, most formats are generated every time a new model is executed.

NetCDF is a standardized self-describing binary data-format developed by Unidata[2] and is readable by most open source visualization applications. Some systems expect NetCDF files following the Climate and Forecasts, CF[3], metadata conventions and a conversion of selected parameters have been implemented, storing the data in a new NetCDF file.

6.5 Experiments

Several tests were conducted, verifying that the system successfully could execute the models on all available clusters, one node only.

An experiment was conducted using a node in the Display Wall cluster as computational node. The hardware and software setup is listed in Section 4.4.1. The experiment used a controlling Python script for executing and timing a number of request with different resolution and for logging the results. This experiment is intended to validate the workflow using the configuration illustrated in Figure 6.3 and 6.4. The successful execution using several different parameter values was the goal of the experiment.

6.6 Results

The time it takes to compute one 6-hour forecast is directly related to the resolution via the length of the time-step needed. The WRF model computes an index that is checked during execution reported if the computation are deemed numerical unstable and has been stopped. Table 6.2 illustrates a rule of thumbs relationship between the spatial resolutions, the length of each time-step and the number of time-steps one 6-hour forecast need. The time each time-step in the model takes to compute does not

[2] http://www.unidata.ucar.edu/

[3] http://cf-pcmdi.llnl.gov/documents/cf-conventions/1.4/cf-conventions.html

vary much depending on the length of the time-step, making the relationship between the spatial resolution and the computation time nearly linear.

Table 6.2 *Spatial resolution vs. time-steps*

Resolution	Time step in seconds	# time steps
4 km	24	900
1 km	6	3 600
100 m	0.12	180 000

A rule of thumb comparison between model resolution, length of time-step and number of time-steps in a 6-hour forecast. The numbers are valid for the 3.1.1 version of the WRF model used in this dissertation.

Table 6.3 shows typical measured execution times for complete forecasts using the hardware specified in Section 4.4.1 and the default build of the model described in Chapter 4. The times include all pre-processing and data movement, and represent the waiting time from when a forecast is requested until the finished results are ready for download. The mean values and standard deviation is reported for a 6-hour forecast using a 39 x 40 mesh size with the computer under otherwise light load and low network usage.

6.7 Discussion

By limiting the model to a small region and short duration, it can be executed on commodity hardware within a few minutes. This is an example of the locality principle applied to weather forecasts by computing the numerical model close to the user of the forecasts. The availability of background geographical and meteorological data enables execution for any area.

Access to a locally produced very high-resolution numerical forecast has a number of benefits compared with the available forecasts on the Internet today. The following list illustrates a few of these.

- The available spatial resolution is higher and will not be matched by the publicly available data from the national weather services for several years.

Table 6.3 *Typical execution times for forecasts*

Spatial resolution	**Mean execution time**	**Standard deviation**
10 000 m	158 seconds	1.8 seconds
3 000 m	328 seconds	3.2 seconds
1 000 m	874 seconds	3.7 seconds

Execution times for a 6 hour forecast with mesh size 39 x 40 on a single computer, under otherwise light load and low network usage. $N = 10$.

- The available parameters from the model are highly configurable. Users with particular interests may use parameters that are available from the model but are not publicly available today. Two examples are winds at specific heights above the ground, i.e. 100 m or 800 m, and the temperature a few centimeters into the ground. Users for these parameters may include paraglider pilots and farmers.

- Parameters may be made available in formats suitable for custom applications. Controlling both the supplier and consumer of the data, any format suitable may be used.

- A set of forecasts covering a larger region or a route to be traveled can either be generated before moving, or on-demand during travel. All that is needed is an Internet connection to the home computer.

The main difference with the described system compared with what is used and publicly available from weather services, is the novel idea of producing very high-resolution forecasts using a fully-fledged atmospheric model on a home computer.

6.8 Conclusions

The on-demand production of very high-resolution meteorological forecasts on commodity hardware is possible and have several strong advantages compared to the publicly available forecasts. The executing time is governed by the forecast length and size of region; a 6-hour forecast takes around 15 minutes on a two-year-old desktop computer. Improvements in hardware on a desktop computer is expected to follow the improvements in super computers, making the production system likely to continue to be viable with newer versions of the WRF model and newer hardware.

The user maintains anonymity at all times as the background data retrieved for the model covers the whole globe and the forecast can be executed for any location. The forecast can be personalized for a users specific needs both by changing the models characteristics with what options are enabled or selected, and by changing which parameters are available for later visualization. The possible customization may be viewed as a part of an extended interpretation of *On-Demand*. The resulting forecasts are stored on the users own computer and can be utilized on whatever system the user wants. No external communication is needed after the initial download of the background meteorological data.

7 Interactive Forecasts

This chapter describes a system developed related to the research statement: *Build a system for computing on-demand very high-resolution interactively fast meteorologically sound forecasts with the purpose of identifying the characteristics and performance of an architecture, design and implementation done for a commodity platform*.

This chapter introduces *interactively* on-demand produced weather forecasts for use on a very large display, a Display Wall. *Interactively* is here in italics because the term is used slightly different from everyday use. Paper 1 [41] presents the results from a survey of operational weather forecasters at the Norwegian Meteorological Institutes forecasting division for Northern Norway, see [41, Table 2], on how long they would be willing to wait for an on-demand forecast. The median value was 2,5 minutes with a large spread from 5 seconds to more than 10 minutes. The response from the forecasters indicated that even experienced weather forecasters did not have a common expectation on how long they would be willing to wait. In the survey, a few minutes was for most forecasters considered worth waiting, and a few minutes is therefore considered *interactive* in this dissertation.

Interactivity is here understood as the ability to set or indicate a location and have something computed for that location, decided on-demand. Interactivity can also be interpreted as the possibility of choosing which of a set of models or a set of model options, to execute on-demand.

7.1 Idea

The main idea is that a user selects a location to use as the center for a new forecast interactively. One possible use demonstrated in Paper 1 [41] is to select a location using a visualization tool, and have the model executed and visualized for that area. Another possibility is to have a mobile device with built-in GPS. It is possible to use the GPS for requesting a forecast centered on the current location. The forecast needs in both cases to be available within in a time acceptable for the user.

Since the time to wait for a forecast is related to the resolution wanted, the user should make this trade-off. This implies that the resolution has to be one of the elements that the user can control in addition to the location.

7.2 Architecture

Paper 1 [41] describes using the Live Dataset (LDS)[51, Figure 4] architecture for initiating a new forecast and for visualization.

The architecture for the interactive forecasts consists of three elements; The display, the forecast visualization and the forecast production. The display initiates a forecast centered on a user-given location, the forecast is sent for visualization and the visualization returned to the display for user viewing.

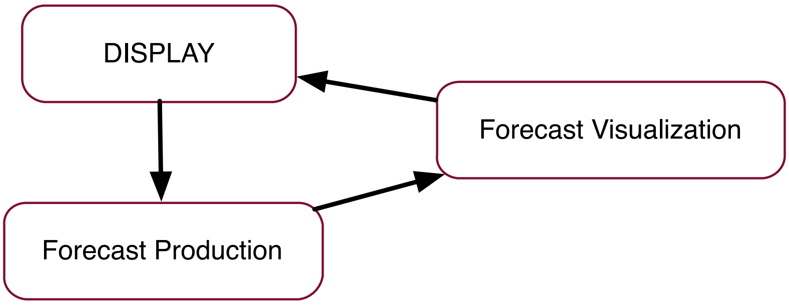


Figure 7.1 *The architecture of the interactive forecast system.*

The architecture divides the display from the visualization of a forecast, and divides the visualization of a forecast from the production of the forecast.

7.3 Design

The design specifies the use of the WallGlobe and LiveDataset, LDS, in the display component. The design also specifies that the system must use the protocols used by the LiveDataset. The LDS request a forecast from the forecast production. The forecast production will store the forecast before returning the synchronous HTTP request. The LDS will wait until the forecast production is finished before requesting the visualization used in the WallGlobe viewers, tiled images. Since the LDS waits for the production to finish before requesting a visualization, the synchronization of the forecast production and forecast visualization is maintained. The forecast visualization will not access the storage before the forecast production is finished storing the forecast.

The forecast production does not relate to the WallGlobe application directly, only to the LiveDataset. The forecast visualizer does not relate to the WallGlobe application or the display, only to the LiveDataset that requests visualized elements for a given area.

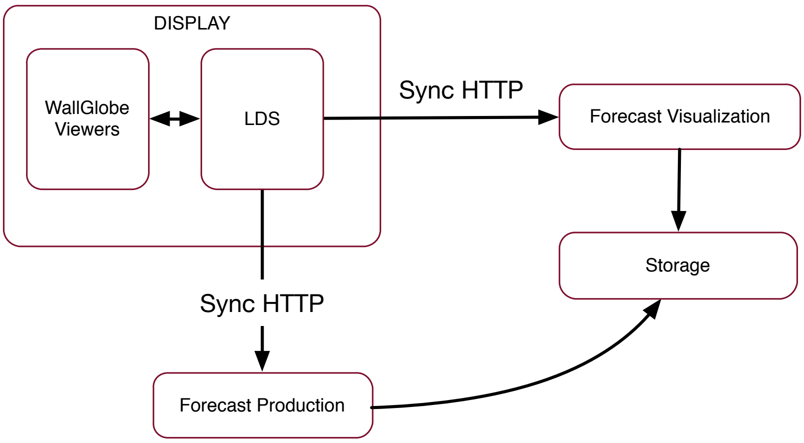


Figure 7.2 *The design of the interactive forecast.*

7.4 Implementation

The implementation maps the design of the interactive forecast systems onto a set of computers. The Display side consists of the display wall cluster with the viewers executing on the nodes, and a LiveDataset instance executing on a standalone computer. The forecast production is executing on a single node on one of the two clusters possible. The forecast is stored on a shared filesystem accessible by both the forecast production and forecast visualization. The forecast visualization is executing on a standard workstation in the HPDS lab.

The execution of the model is to be triggered by the user in the WallGlobe application via the Live DataSet that orchestrates the execution of the forecast and loading of the needed visualized data. The forecast visualizer acts as a data source for the Live DataSet. Figure 7.3 illustrates the implemented system and communication paths.

The *interactive* execution of the model and subsequent visualization of the forecast was added to the WallGlobe application using a designated trigger, snapping the fingers three times, which would use the latest touch-point on the current view on the wall as the center location for the forecast requested.

The forecast could be executed on two different compute clusters, the local old "Rocks" compute cluster at the Department of Computer Science, or the "Stallo" supercomputer[1]. The main difference between these two is that the "Rocks" cluster could be accessed interactively, and the "Stallo" cluster only through the batch oriented queuing system. The return times from executing on the "Rocks" cluster were therefore known and could be depended on. The return times from the "Stallo" cluster could not be known in advance.

The Forecast Visualization was implemented as a web service on an older workstation in the HPDS lab, returning tiled images matching the need of the LDS and the WallGlobe application. The tiled images had specific geographical coverage dependent

[1] https://www.notur.no/hardware/stallo

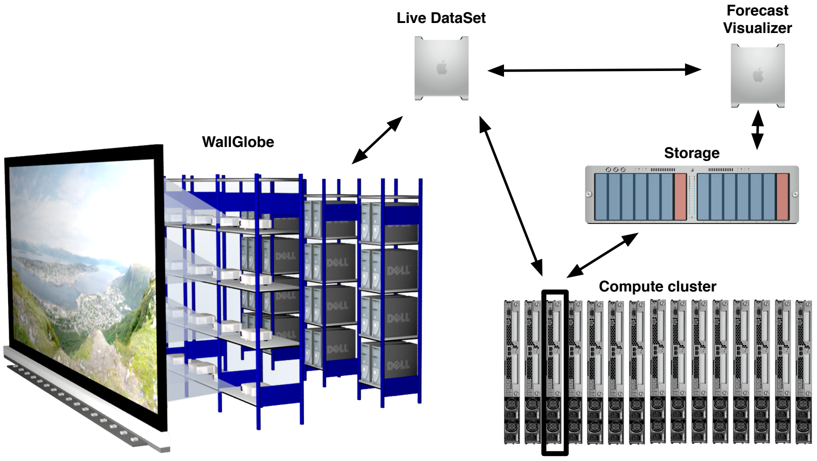


Figure 7.3 *The Tromsø Display Wall with the interactive forecast implementation.*

on the current zoom level used in the WallGlobe application.

Figure 7.4 illustrates the use of the WallScope [52] for selecting a location and visualizing forecasts. The location is selected by indicating the center location on the display wall and then triggering an execution of the model using a specific gesture.

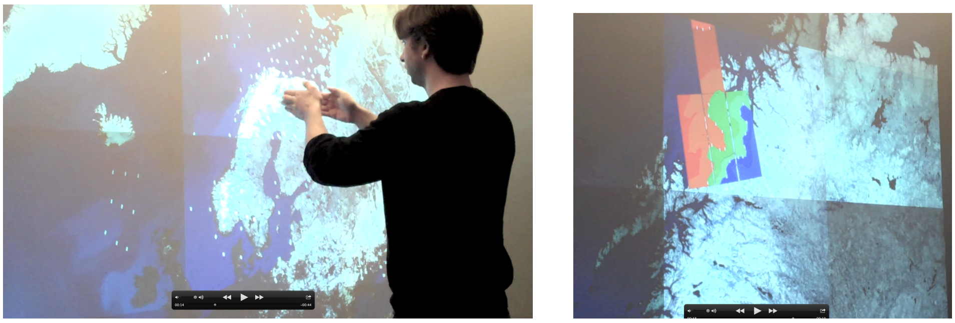


Figure 7.4 *Screenshots from a video presented at PARA2010. The video is available as listed in Appendix A.*

7.5 Experiments and Results

The goal of the experiment was to validate the correct execution and orchestration of the system. The result of the experiment was documented in the video presented at the PARA 2010 conference in Reykjavik, Iceland. See Appendix A for access to the video. The hardware used for this experiment was an older version of the Display Wall and an older compute cluster, together with the then current Stallo supercomputer.

- Old Rocks Compute Cluster
  - 3.2 GHz Intel Pentium 4.

  - Rocks Linux distribution version 3.
  - default gcc suite with fortran compilers.

- Stallo compute cluster (2010)
  - Intel 53xx Quad Core 2.66GHz. HyperThreading enabled
  - Rocks Linux cluster distribution version 5.
  - Intel ifort compiler and suite.
  - Support for shared memory and distributed memory when building the WRF model.

- Display Wall cluster (2010)
  - Intel Pentium 4 EM64T 3.2 GHz processor, 2 GB RAM, HyperThreading enabled
  - Rocks Linux cluster distribution 4.0.

- Forecast Visualization host
  - 3.20GHz Intel Pentium 4
  - Ubuntu 10.4 (LTS) Linux distribution

The production system produces a forecast given a location and wanted resolution within 3 to 17 minutes for respectively 3000 and 1000 m resolution models

The visualization system utilizing the WallScope application on the Tromsø Display Wall was demonstrated and presented as a video at the presentation of Paper 1 [41]. The video demonstrates the system as seen by the user, and the produced visualizations. The video is available as described in Appendix A. The forecast visualization was an obvious bottleneck in this system.

7.6 Discussion

The use of *On-demand* and *Interactively* in connection with weather forecasts have so far had the meaning of being able to retrieve pre-computed forecasts for any location. On example of this is the "Weather on Demand (WOD)" service from one of the largest commercial weather services in Norway, StormGeo, explained at `http://www.stormgeo.com/weather-on-demand-wod`. The described service returns a forecast from available models for any location. The service does include additional processing made on the model results to provide a higher quality forecast.

The terms *On-Demand* and *Interactively* is in this dissertation used as a label of the capability of being able to execute a new model using currently available background data, generating a very high-resolution forecast for any location, within an short time period. Visualization of the forecasts is enabled both for use on the Tromsø Display Wall and on mobile devices.

7.7 Conclusion

A prototype for personalized on-demand weather forecasts using a prototype of localized on-demand production and visualization of very high-resolution numerical weather forecasts has been presented.

The three levels of the architecture:

- Forecast production. A personal computer or node on cluster. In Paper 1 a single node on a supercomputer was used. Previous chapters have demonstrated that any modern personal computer can compute the forecasts.
- Forecast visualization. A personal computer. Previous chapter have demonstrated several possible visualizations that can be used. All can be produced on a personal computer. The display system can use many visualization systems.
- Display. WallGlobe uses the LiveDataset for combining visualizations from many sources in a high performance interactive system.

The WallGlobe high performance visualization on the display wall can use a personal computer for forecast visualization. The LiveData set architecture can use multiple systems for visualization. The architecture allows for individual devices to be providers of visualizations, for the display wall. The architecture allows for interactively triggering new computations, new forecasts, on-demand.

A set of personal computers can independently produce visualization for use. The set of computers represents a very robust system, one or more computers can fail and any other computer can take over the failed computation.

The display wall system and WallGlobe scales because the system can utilizes many computers for producing visualizations. The experiments showed that a single users forecast visualization, does not scale for using on a huge display wall. The single producer was a notable bottleneck in the visualization.

8

Visualization of Forecasts

This chapter is a response to all research statements as visualization is an important aspect of most use of weather forecasts.

8.1 Idea

Visualizing personalized on-demand very high-resolution forecasts should be centered on the location of the user. For some applications, this is the obvious angle; one example is the Augmented Reality application described in this Chapter. For standard map based, the user have to move the visualization to cover the correct area. The visualization should also ideally be un-coupled from the production side of the forecast.

8.2 Architecture and Design

The architecture and design is illustrated in Figure 8.1. The architecture is based on the two abstractions; the forecast and the forecast presentation. The design specifies the communication protocol to allow independent development of the forecast production side and the forecast visualization side. Both derived data and raw data-files are available from the forecast production side.

Design level specification of a REST API using HTTP/HTTPS ensures the loose coupling between services and applications, making the server side applications device agnostic. Specifying the file formats used narrows the implementation space. The formats must be synced with what is currently usable by standard applications. Applications developed as part of this dissertation could have used any proprietary data format, but it is beneficial to use the standard formats supported by other applications. The KML format used by the Google Earth application is one such example.

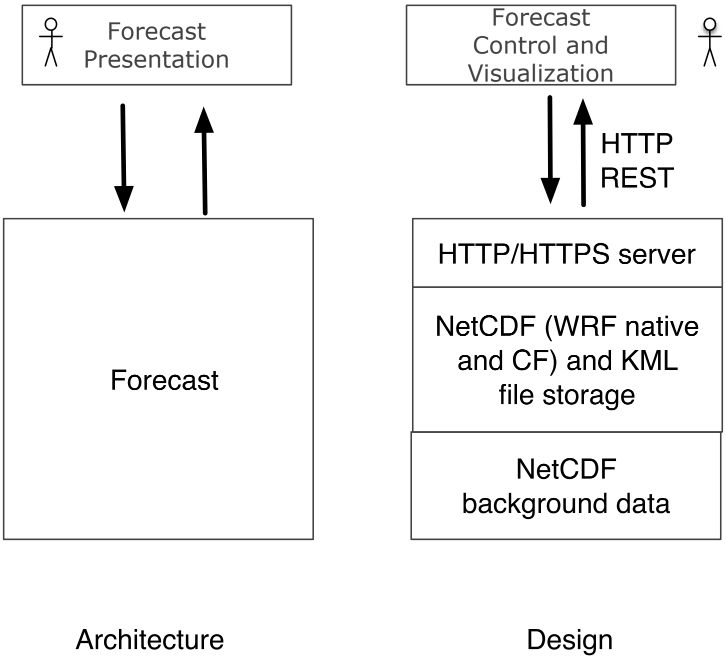


Figure 8.1 *Visualization architecture to the left, and design to the right.*

8.3 Implementation

Visualizations for mobile devices were created for use on iPhones and iPads from Apple, using 2011 models and newer. The applications for the iOS devices were developed on two Macbook Pro computers.

Generating KML files were done on various older personal computers under the Linux operating system. Visualizations for the Display Wall were executed on older personal computers using the DIANA [73] system and VNC [69] for visualizing on the Tromsø Display Wall. See Anshus et. al. [6], Additional Paper 1, for more details. The version of the DIANA system used, only reads CF convention NetCDF files. A conversion of a limited set of parameters from the original WRF NetCDF files was created.

8.3.1 Augmented Reality Application

Visualization of the forecast should be centered on where the user is located. This can be illustrated with Figure 8.2. The user is located within the forecast mesh and has a location known by using the built-in GPS of an iPad or iPhone. The forecast is requested and computed with the current location as the center of the forecast. The back facing camera has a viewing direction that is known and provided by the built-in compass of the tablet, and a tilt and skew given by the accelerometers. This enables a correct location of the view within the forecast, and the forecast may be overlaid the actual camera view. The prototype allows for manually adjusting several scaling factors, like the apparent viewing height, the apparent grid size and the movement scaling. The larger screen size of an iPad makes this the best choice for viewing meteorological data overlaid images in the developed prototype. The developed application was created as a proof of concept. It has not been subject to further studies on the usability and so forth. Examples of prototype use are illustrated in Figure 8.3 and 8.4.

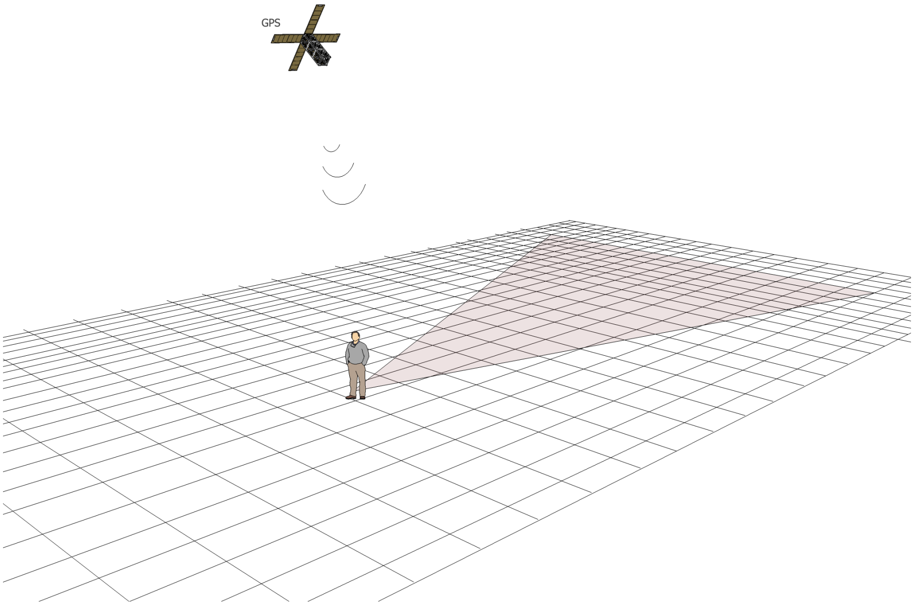


Figure 8.2 *Using the GPS signal, the user has requested and received a forecasts centered on the current location. Seeing through a device the users sees the forecast in one direction. To see other direction the user simply turns around.*

The Augmented Reality, AR, application uses data in the KML format, and an application creating KML versions of forecasts has been developed. This makes the forecast also available for visualization using Google Earth. A web service application was also developed for rendering the forecasts in a tiled manner usable in a web page using Google Maps. Figure 8.5 illustrates using Google Maps with pre-rendered tiles, or Google Earth using the KML data and rendering locally on the device.

8.4 Experiments and Results

8.4.1 Openly available visualization applications

Examples of widely used open source visualization applications are Paraview[1], IDV[2] and DIANA [73].

DIANA is an open source application developed by the Norwegian Meteorological Institute and documented at `https://diana.wiki.met.no/doku.php`. DIANA expects

[1] `http://www.paraview.org/`

[2] `http://www.unidata.ucar.edu/software/idv/`

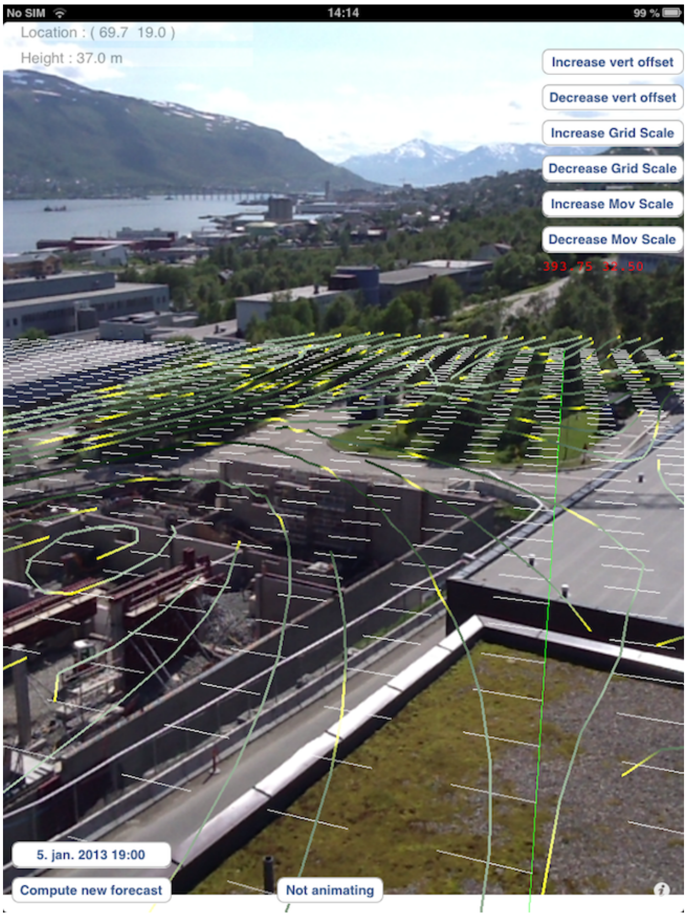


Figure 8.3 *An illustration of the Augmented Reality type application using a forecast centered at the University of Tromsø. The iPad is looking southwards.*

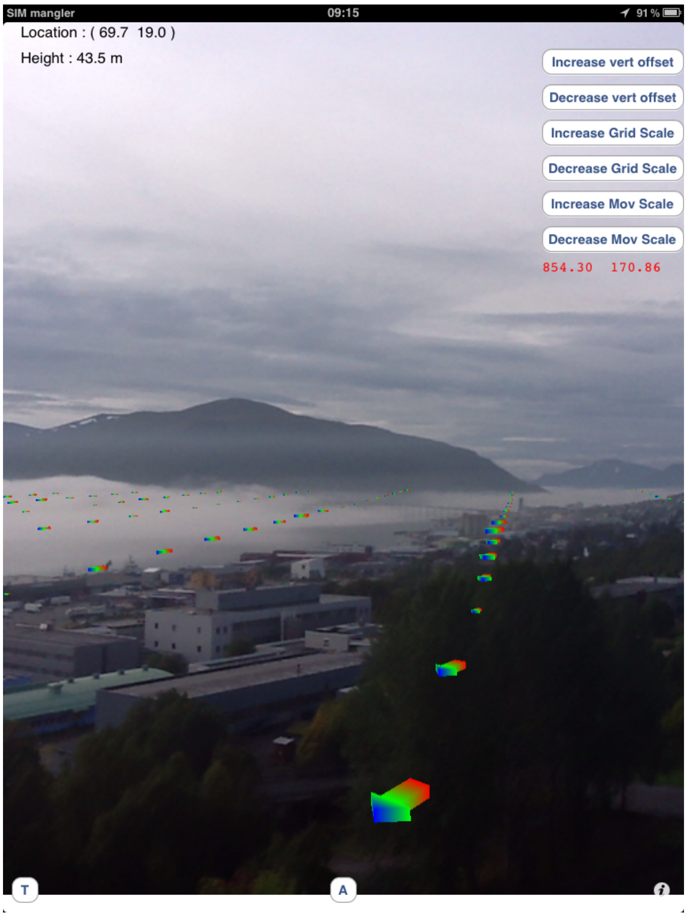
NetCDF files following the Climate and Forecasts, CF[3], metadata conventions. The DIANA application may be used for visualizing the forecasts on the Tromsø Display Wall. The DIANA system will execute on a standalone computer using a VNC server as a rendering display. A modified version of the MESA[4] libraries is needed for utilizing the very large size of the display wall, 7168 x 3072 pixels. The DIANA system is a strictly 2D mapping application. One example of using the DIANA system on the Tromsø Display Wall is given in Figure 8.6.

The personalized weather forecast is made available on devices with display sizes covering two orders of magnitude. From the iPhone to the Tromsø Display Wall. Figure 8.7 is illustrating using the DIANA system on the Tromsø Display Wall visualizing the Symbiotic Forecasts described in the previous chapter.

The ParaView application can also use the display wall by executing individual viewers on each tile of the Display Wall, using the wall front-end for coordination and a laptop for the user interaction. ParaView may visualize the forecast in 2D or 3D. ParaView

[3] `http://cf-pcmdi.llnl.gov/documents/cf-conventions/1.4/cf-conventions.html`

[4] `http://www.mesa3d.org/`



Location : (69.7 19.0)

Height : 43.5 m

854.30 170.86

Figure 8.4 *An illustration of the Augmented Reality type application using a forecast centered on the University of Tromsø and an early morning view from the University.*

and IDV use the native output from the WRF model or CF convention NetCDF files.

Uncertainty in wind forecasts for single locations can also utilize wind-rose type visualization. Figure 8.8 illustrates multiple forecasts for a single location. It should be noted that this is the forecasted direction and speed at approximate 60 m height, a parameter not publicly available from any weather service at this resolution. This example is taken form an area with a large wind farm and the height corresponds roughly to the height of the hub on the wind turbines and therefore more useful for predicting wind power production than the regular surface ie. 10 m height, wind forecast. A noteworthy element in Figure 8.8 is the large variation between the forecasts.

8.5 Discussion

Using industry standard applications for visualization of forecasts ensures the option of standardized presentation of forecast in familiar way to a professional and experienced user. Within the operational meteorological community, several tools are available un-

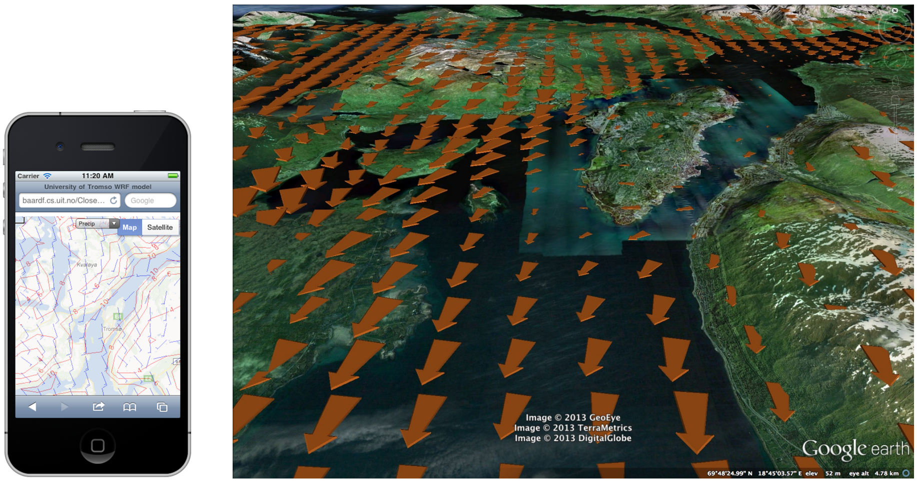

Carrier
11:20 AM
University of Tromso WRF model
baardf.cs.uit.no/Close...
Google
Precip
Map
Satellite
Image © 2013 GeoEye
Image © 2013 TerraMetrics
Image © 2013 DigitalGlobe
Google earth

Figure 8.5 *An illustration of visualization of forecasts using Google Maps to the left and Google Earth to the right.*

der open source or equivalent licenses, and therefor easily accessible. The usage of some of these applications has a steep learning curve for non-professionals. Most also require a good knowledge of semi-standard meteorological parameter names, i.e, DD and FF for wind direction and windspeed. Some tools like the DIANA system requires an expert user for editing setup-files before data can be made accessible in the standard user interface.

The Augmented Reality puts the user in the center of the forecasts, or more correctly, the forecast is centered on the user. This is a promising avenue for further research. In this dissertation, only prototype and demonstration applications were developed without further investigation of what types of visualization would be most effective. A presentation of the AR application on the Norwegian Broadcasting (NRK) science program "Schroedinger's Cat"[5], was quite popular and lead to many requests to the local NRK office. New devices like the Google Glass[6] may make this kind of visualization useful and popular.

Tailored visualization like the wind-rose in Figure 8.8 is also possible because of the access to the complete model output. Such visualization may be locally produced based on the specific local requirements.

8.6 Conclusions

Several options and prototypes for visualization of personalized on-demand weather forecasts on a number of devices and screen sizes has been presented. The augmented reality application is ideally suited for localized forecasts generated using a device's actual location.

[5] http://tv.nrk.no/serie/schrodingers-katt/dmpv73000813/07-03-2013
[6] http://www.google.com/glass/start/

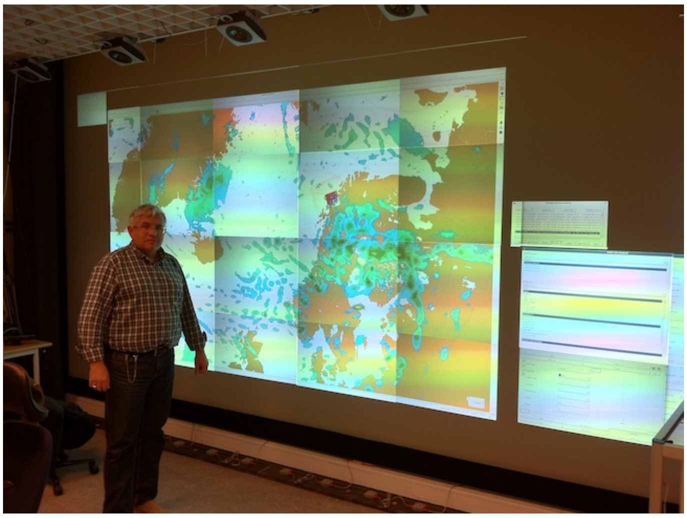


Figure 8.6 *An illustration of using the DIANA system on the Tromsø Display Wall.*

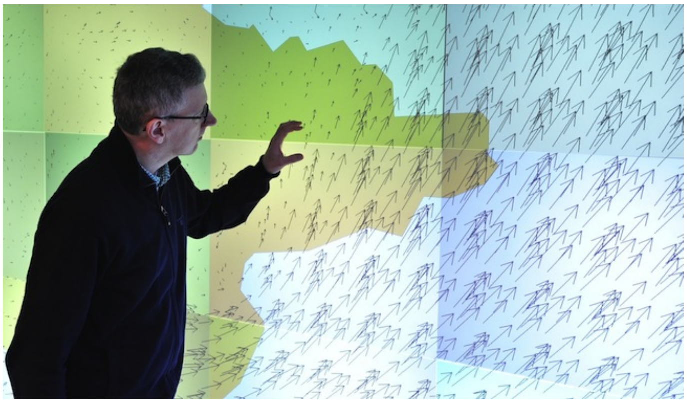


Figure 8.7 *An illustration of using the DIANA system for visualizing many forecasts at one time on the Tromsø Display Wall. Prof. Otto Anshus is studying details of several forecasts in the same plot.*

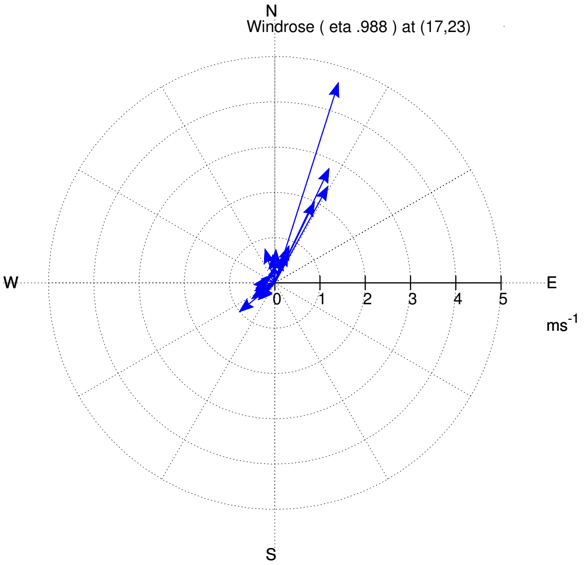


Figure 8.8 *Wind rose plot at approximate 60* m *height for a location, mesh point 17,23, within a large windmill farm area. 28 forecasts are shown. All forecasts use the same meteorological background data, but have slightly different mesh locations resulting in the variations in wind speed and direction seen.*

9

Symbiotic Collaboration

This chapter is a response to the research statement: *Build a system for combining local weather forecasts with the purpose of identifying the characteristics and performance of an architecture, design and implementation done for a commodity platform.*

One example of the ways collaboration has been used for producing high-resolution weather forecasts is found within the GLOBUS [44] system. The key feature is to share computational resources with runtime configuration for optimal utilization for a given task. This is a system mostly used for sharing between larger organizations with local compute clusters and for research purposes.

The end goal for this Chapter is to describe a service that provides localized on-demand very high-resolution weather forecast with uncertainty estimation from a distributed collaborative system, produced on commodity hardware.

9.1 Idea

The idea of symbiotic collaboration is to share locally computed forecasts between peers for mutual, symbiotic, benefit. This idea builds on the systems and prototypes created in the previous chapters.

Collaboration is optional; a complete forecast can be made without any collaboration. This implies no lower limit on how many peers that are needed. Collaboration is lazy; a user collaborates whenever needed and do not need to be connected at all times. Every user do not have to collaborate with all peers the same time. This implies that the system can be made independent on a centralized server or always-online systems. The system will use Internet resources for background data and therefore needs Internet access, but not necessarily at all times.

The system is named *Collaborative Symbiotic Weather Forecasts,* ***CSWF***.

9.2 Architecture

The architecture is presented in Paper 3 [39] and is based on two abstractions. The Forecast Presentation and the Forecast. The user interacts with the forecast presentation and the forecast abstraction contains all functionality related to the production and collection of forecasts. The architecture is illustrated in Figure 9.1.

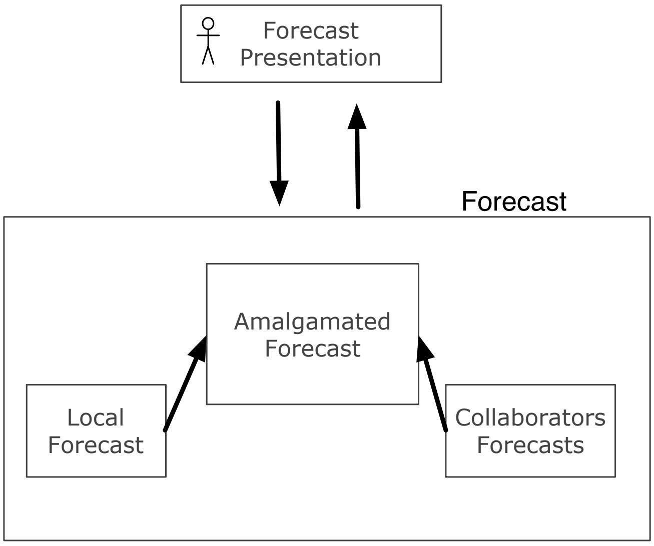


Figure 9.1 *Collaborative Symbiotic Weather Forecast architecture.*

The architecture describes a system where the collaboration is optional and is not necessary directly controlled by the user. The architecture calls for the collection of collaborators forecast amalgamated together with the local forecast into a forecast that is presented to the user. The collaborators forecasts have to be found in some not specified way. The user would also require presentation of the local forecast at all times.

9.3 Design

A design is given in Paper 3 [39] and illustrated in Figure 9.2. This design calls for a front-end service handling of all communication between the forecast production and the forecast presentation. The presentation is a set of client applications used on various devices. A collaboration system handles all communication with peers; a productions system produces the local forecast. All forecasts are stored in a common system accessible by all sub-systems.

The design specifies that requests for forecasts are handled synchronous by the front-end, and that the front-end may reply with a redirect to an URL where the forecast are or will be available. The front-end must use a HTTP REST API. The Collaboration System has to locate and collect forecasts from geographically nearby collaborators. It has been designed so that it does not need to scale, as only a few peers are needed.

9.3.1 Locating Geographically Close Peers

Bootstrapping a peer-to-peer network is a well-known problem. Several solutions have been suggested; either using static addresses of a bootstrapping node, or having bootstrapping nodes identified using a DNS service where a domain name resolves to a bootstrapping node address.

A completely decentralized service that does not depend on any known address or domain name is difficult to implement. One option is to scan a network for hosts making available forecasts on a known port. This implies that a front-end keeps a TCP/IP port accessible from the Internet. This is made very difficult by the use of NAT solutions in most home networks. The port have to made available by the first system reached from the outside, often an Internet modem or router, and forwarded to the intended computer within the home network.

A network scan was selected as a method for bootstrapping the forecast exchange. The scan of the network utilizes the high probability that IP addresses in the same sub-domain will be geographically close together. Since Internet service providers also are likely to also start using NAT strategies for mitigating problems with IPv4 address contention, it is important to obtain the local or regional network address used by the ISP in the area, not necessarily the public visible Internet address. A move to using IPv6 will possibly make the process of finding nearby peers easier.

9.4 Implementation

A CSWF system implementation use a set of multi-threaded processes communicating using a HTTP REST API. Each sub-system comprises of one or several processes and threads. All applications are compiled for executing on either Linux or OS X operative systems. The WRF model is only executed on Linux.

The sub-systems are programmed using C with scripts of Python for executing remote and/or local processes on-demand. The prototype uses HTTP servers implemented using the Mongoose[1] system for a lightweight HTTP REST API.

The implementation does not require that all processes are located at the same host although this is the "normal" configuration. A distributed configuration is also possible. This would for example allow the front-end serving the client visualization applications to be executed on a different system than the system executing the numerical model. This also allows parts to be executed on different operating systems, even if the model has to be executed on a Linux based host.

No actual multi-peer network is established in the collaboration part of the system. Communication is at all times limited to the two communicating peers. The prototype system does not implement any security or trust related features, but computes most of the statistics needed for trust evaluation. Chapter 10 expands on computing trust.

[1] https://code.google.com/p/mongoose/

9.5 Experiments and Results

All processes in the CSWF system can be executed on a single node in the Display Wall cluster, including executing the WRF model on the same node. The hardware and software setup for one node was described in section 4.4.1. The computers in the cluster have gigabit Ethernet connections and are on a private network. The cluster uses the Rocks Linux distribution, which allows for simultaneous execution of applications on all nodes. All nodes have access to a shared file-system and a local file-system on each node. The background meteorological data is stored on the shared filesystem to avoid flooding the data supplier with simultaneous requests for the same data from all nodes. All nodes use the daily downloaded data.

The CSWF system is started on all nodes and is given a primary location. All nodes except one will randomly shift their location within 10 km from this primary location to emulate different users. A request for a forecast is sent to all nodes. This will trigger a local execution of the WRF model on each node, and parallel to this, all nodes will start probing the network for other CSWF systems to exchange the forecast with. This generates a huge spike in network traffic and CPU load, but will eventually result in every node having all forecasts. Every node will have generated an amalgamated forecast based on that nodes local forecast. Every amalgamated forecast will therefore be slightly different.

An experiment for locating other computers on public networks was conducted on a local ISP in Tromsø. A class D IP range with 255 possible hosts was scanned either from an address within the range, or from another network. The results are given in Table 9.1.

Table 9.1 *Scan times*

Setting	Seconds	Hosts located
From same network, all ports	146	6
From same network, only port 25	3.1	8
From other network, all ports	107	23
From other network, only port 25	3.7	23

Scan times using *nmap*. Commands were "*nmap -p "*" 109.189.234.**" and "*nmap -p 25 109.189.234.**"

The scan was done within a few minutes of each other, and were not expected to provide the same results. The results indicated that the ISP have some network configuration in place that effects the possibility of nodes on the same network to reach each other.

The scan times for scanning for a single port are quite fast, indicating that the CSWF system have sufficient time to do scanning and transfer of forecast in parallel with computing a local forecast.

An experiment was conducted using a primary location close to the Fakken Wind Mill area. The area is illustrated in Figure 9.3 and show differences in forecasts over a short distance. A regular forecast would generate one single value in an area. The areas covered by the forecasts are illustrated in Figure 9.4. The computed standard deviation for one forecast node is illustrated in Figure 9.5. The goal of the experiment is to validate that the CSWF systems is capable of finding nearby CSWF systems and exchange forecasts with these. A successful result is the production of a forecast containing uncertainty estimates of wind speed.

Figure 9.4 and 9.5 demonstrates a successful execution of the CSWF system with production of uncertainty measurements not publicly available elsewhere.

The entire CSWF system except the WRF model implementation was built and executed on a Raspberry PI[2] computer as an experiment for validating the small size requirement of the CSWF system. All parts except the WRF model builds and executes without any changes to the code or setup on a Raspberry Pi model B. No further experiments for establishing limits on such a setups was done. Some visualization tools are not expected to be executable on the Raspberry platform because of the memory requirements. A practical limit is most probably to only execute the front-end, forecast amalgamation, collaboration and forecast storage parts of the CSWF system on the Pi.

9.6 Discussion

If the CSWF system used a persistent peer-to-peer network, the address information of network could have been kept constantly updated. Using the geographical location as a key, the network could have been implemented using a Content-Addressable Network, CAN [88]. This would have made an efficient and scalable solution. The downside is that nodes would have to re-enter the network after a period of inaccessibility, and would need to listen for periodic routing updates. Ratnasamy [88] does not specify a specific bootstrapping method, but relies on a method described in Francis et. al. [45] using a known domain name and DNS resolving. Other systems using some aspects of the stored information are discussed in Androutsellis-Theotokis and Spinellis [5, Sec. 10] in the topic of *Semantic Grouping of Information*.

See Chapter 5 for details on exchanging forecasts to enable uncertainty estimation. In the very few experiments conducted, the standard deviation of wind speed was in the order of 10-50 % of the mean wind speed. This is significant for many uses.

9.7 Conclusions

A system for localized collaborative weather forecasting and using these collaborative forecast for uncertainty and error estimation has been implemented. The chapter has demonstrated a model of collaboration that does not require previous knowledge of other participating users. Even if no other nodes are found, the user still has a valid very high-resolution weather forecast. Reciprocity was found by Elevant [32] to be of importance in sharing weather information. The developed collaborative system exchanges forecasts between pairs of peers on a peer-to-peer basis, ensuring good reciprocity.

[2] `http://www.raspberrypi.org/`

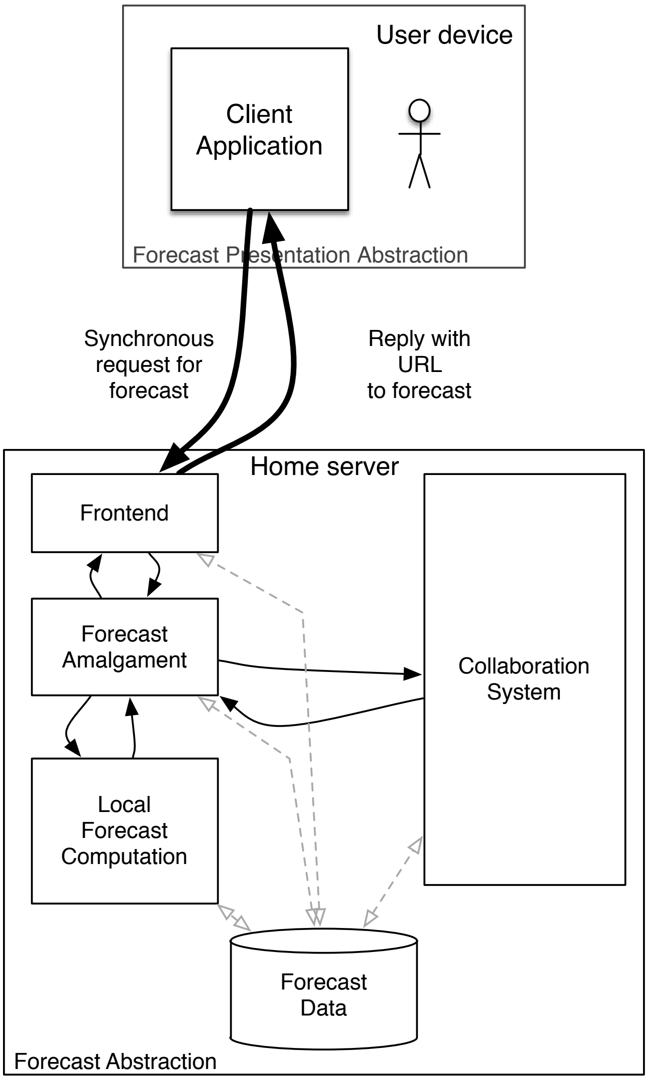


Figure 9.2 *Collaborative Symbiotic Weather Forecast design. The dashed lines indicate access to a shared forecast storage.*

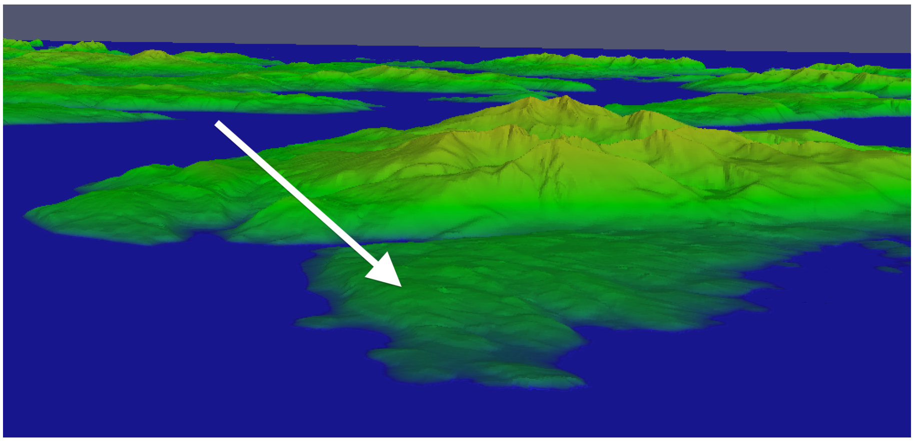


Figure 9.3 *The Fakken Wind Farm area is indicated by the arrow. The figure is a rendering of 10* m *resolution DEM data from the Norwegian Mapping Authorities.*

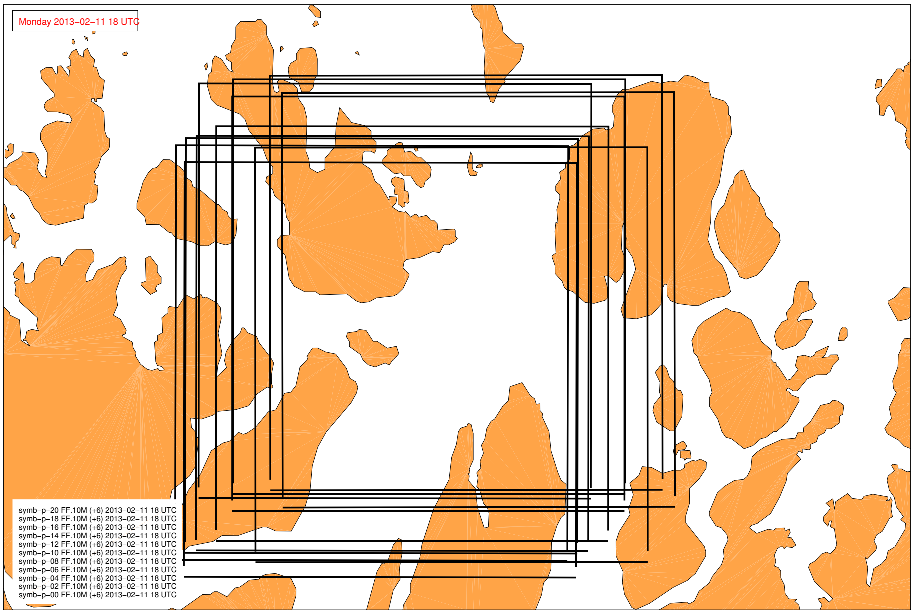

Monday 2013−02−11 18 UTC
symb−p−20 FF.10M (+6) 2013−02−11 18 UTC
symb−p−18 FF.10M (+6) 2013−02−11 18 UTC
symb−p−16 FF.10M (+6) 2013−02−11 18 UTC
symb−p−14 FF.10M (+6) 2013−02−11 18 UTC
symb−p−12 FF.10M (+6) 2013−02−11 18 UTC
symb−p−10 FF.10M (+6) 2013−02−11 18 UTC
symb−p−08 FF.10M (+6) 2013−02−11 18 UTC
symb−p−06 FF.10M (+6) 2013−02−11 18 UTC
symb−p−04 FF.10M (+6) 2013−02−11 18 UTC
symb−p−02 FF.10M (+6) 2013−02−11 18 UTC
symb−p−00 FF.10M (+6) 2013−02−11 18 UTC

Figure 9.4 *Overlapping areas of forecasts. Figure shows 10 of 28 forecast areas. The Fakken wind farm area is slightly above the center of the figure. Colored areas are land.*

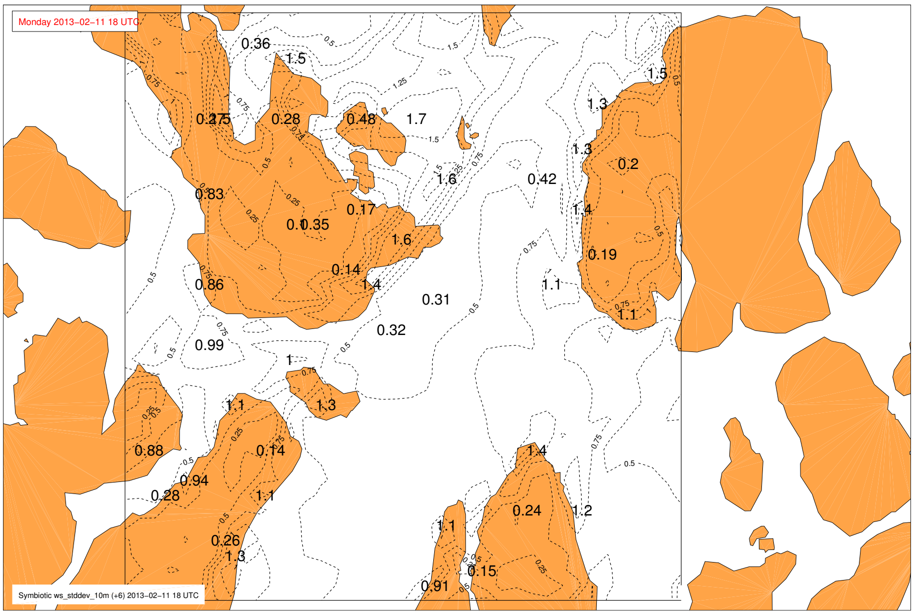


Figure 9.5 *Standard deviation of wind speed. Values within this figure range from 0.2 to 1.6 ms^{-1}, high and low values are shown.*

10

Trust

This chapter explores a part of the research statement: *Build a system for combining local weather forecasts with the purpose of identifying the characteristics and performance of an architecture, design and implementation done for a commodity platform.*

This chapter presents a ways of establishing trust between unknown parties that are exchanging data, in this chapter exemplified with weather forecasts.

Trust can be defined in several ways. In Victor et. al. [102] trust is modeled as either probabilistic or gradual. In Artz et. al. [8] trust often refers to mechanisms to verify that the source of information is really who the source claims to be. Trust is needed when exchanging data on both sides. You need to trust your peer before sending information that contains information about you, area of interest and possible location. You also need to trust the data received from a peer.

10.1 Idea

The main idea is to use the information available in the form of locally computed weather forecasts for establishing trust in peers. Trust should be inferred both with the first time contact and may be inferred from gradual building of a history between peers.

The users own numerical weather forecasts can be used for calculating a level of trust after receiving a forecast from a peer. There are at least three possible ways to do this. 1) Calculate the difference between the users own forecast and the incoming forecast. 2) Use forecasts from several peers to calculate a statistical measure. 3) Re-do the work of peers by computing the forecast locally.

10.2 Architecture

Figure 10.1 illustrates the architecture of the trust validation system. The architecture is based on the Forecast abstraction that handles all issues regarding collaboration and trust. Peer forecasts are collected and amalgamated into a symbiotic forecast, also containing statistical distribution of selected parameters.

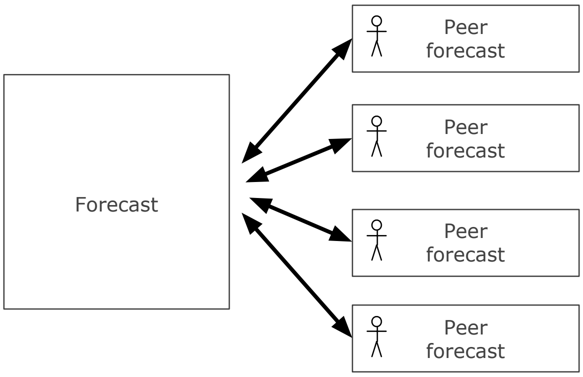


Figure 10.1 *An illustration of a trust computing architecture using the Forecast abstraction from Chapter 6.*

The issue of trust may be handled in the forecast abstraction without user intervention. Limits may be placed on the selected parameters for evaluating trust, and only trust-worthy forecast presented to the user.

10.3 Design

Figure 10.2 illustrates the design of the trust validation system. Forecasts are exchanged with peers. All collected forecasts are stored together with the local forecast. A statistical trust index is computed by a separate statistical engine and the symbiotic forecast is created using only the trust-worthy forecasts collected. Both the local forecast, the symbiotic forecast and the individual thrust-worthy forecasts can be served to the user for visualization.

For each peer forecast some index of trustworthiness is computed based on a "distance" from the local forecast. If there are forecasts from several peers available, some statistical confidence can be assigned to this "distance" index. What actual index is used will depend on the parameter used, as statistical properties of the various parameters have large variations. Different parameters can be used in different ways depending on what is the important aspect of a forecast for the individual user. The design does not specify what index to use or limits. Such issues have to be decided after more research.

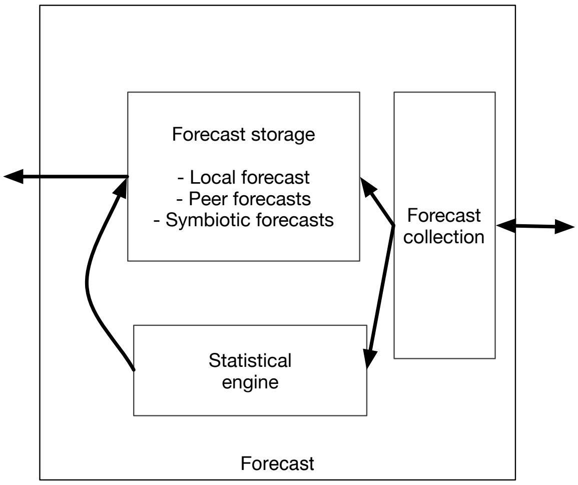


Figure 10.2 *An illustration of a trust computing design using the Forecast abstraction from Chapter 6.*

10.4 Implementation

The Root Mean Squared Error, RMSE, is an example of an index that can be used for evaluating a "distance" between an incoming forecasts and the current set of peer forecasts.

RMSE is defined as,

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\theta_i')^2}$$

where $\theta_i' = \theta_i - \theta_i^{ref}$ and θ_i is the mesh point value in any forecasts, and θ_i^{ref} is the mesh point value in the reference forecast. The deviation between each forecast and the reference forecast is accumulated across all mesh point in the reference forecast. N is the total number of forecast found with mesh-points close enough to the mesh-points in the reference forecast. Mesh points inside the relaxation zone of both forecast are excluded. The method allows for using forecasts with different mesh sizes and resolutions.

10.5 Experiments and Results

The experiments conducted in Chapter 9 produces forecasts that are can be used for evaluating trust. The purpose of using these data is to validate the idea that trust can be computed using the exchanged forecasts.

The RMSE of a parameter may depend on the distance between the center location of the local forecast and the center location of the forecast retrieved from a peer. RMSE of several potential parameters for computing trust were correlated with the distance between forecasts centers. Figure 10.3 illustrates a situation where there is a statistical significant correlation between distances between forecast center distances and RMSE of winds speed for a set of forecasts, $r = 0.94$, $p < 0.05$. The line represents a regression between the RMSE and distance. Such systematic correlation between a proposed index and the distance between forecasts-centers has to be taken into account when evaluating trust based on RMSE. Correlations between RMSE of temperature or terrain height was not found to be statistical significant. Figure 10.4 illustrates a situation where a parameter, temperature at 2 m height is not correlated with distance, $r = 0.32$, $p = 0.105$. Both Figure 10.3 and 10.4 are from the same dataset and same situation, illustrating that different parameters may be used differently depending on what part of a forecast is most critical to a user.

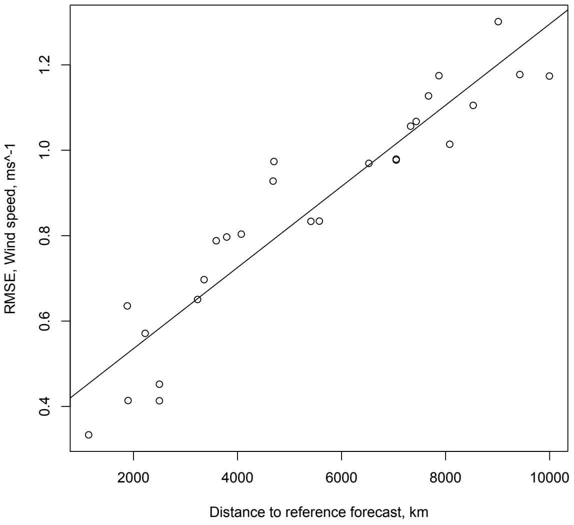


Figure 10.3 *RMSE Wind speed vs. Distance. There is a statistical significant relationship between the RMSE of the wind speed and the distance between center of the local forecast and the center of the peer forecasts.*

Before using RMSE or any other index for computing trust, the statistical properties and possible correlations with distance and other differences in the underlying forecasts have to be evaluated.

Trust can be computed using a distance to a known value or line. Using this method on situations like the one illustrated in Figure 10.3 is comparable to identifying outliers in regular least-squares regression.

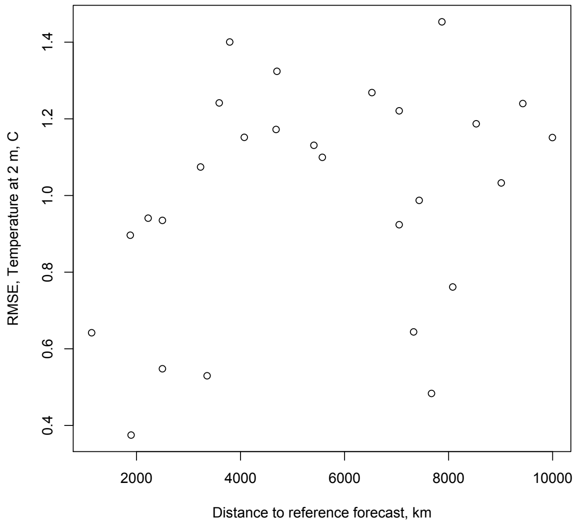


Figure 10.4 *RMSE Temperature at 2 m vs. Distance. There is no statistical significant relationship between the RMSE of the 2 m temperature and the distance between center of the local forecast and the center of the peer forecasts.*

10.6 Discussion

The Berkeley Open Infrastructure for Network Computing, BOINC [1] systems have similar trust issues, and solves this by having two computers do the same work and then comparing the results before rewarding the participants[2].

Using statistical measures for evaluating trust is described by Victor et. al. [102]. The index used for computing the trust can be made user specific and an adversary cannot know what index a user chooses to compute. Different indexes can be made for different parameters and can be computed for specific sub-regions of the forecast.

Establishing trust in peer networks is also discussed by Alvisi and Wong in [2], but in the setting of having byzantine, rational and acquiescent nodes. Current research is on implemented a system where each node follows different strategies depending on whom the node is talking to. This can be used for implementing a dependable distributed system for content delivery. Compared to the CSWF system has the Alvisi and Wong system much less information for evaluating the other peers. The CSWF system can use the local forecast for classifying other peers and enabling trust.

In the extreme case, any incoming forecasts can be re-computed by the recipient for comparison and trust evaluation. This implies that what atmospherical model, the model setup and what background that are data used, should be part of the exchange between peers. Re-computing a forecast takes the same time as computing the local forecast.

[1] http://boinc.berkeley.edu/

[2] http://boinc.berkeley.edu/wiki/How_BOINC_works

10.7 Conclusions

This chapter has presented a way to evaluate trust between peers based on the exchanged forecasts. Trust can be established without prior knowledge of the peer and can be re-evaluated at any time. Since the evaluation is local to each user, an adversary may not know enough for tailoring an attack.

Trust is not necessary a question of adversaries, but is also related to the quality of the forecasts exchanged. If a peer uses an older atmospheric model or uses older background data, the quality is likely to suffer. The same techniques used for computing trust will also be applicable for assessing quality.

11
Case Studies

January 2014, there was a situation of unprecedented wintertime wild-fire hazard in parts of Norway. Several large fires destroyed a large number of houses. This case study investigates two of these fires to see if the developed system would have been helpful in these situations.

Using the WRF model for wild fire forecasting is already an established procedure in several countries. One system is described by Mandel et. al. [72], and an online community at `http://www.openwfm.org` provides both guidelines and access to the software. The WRF model is used for creating background data for very detailed fire-spreading models.

11.1 Lærdalsøyri January 18 2014

In the evening of January 18 2014, a fire started in a house in the small township of Lærdalsøyri, Norway. Due to strong winds, fire spread rapidly downwind and would eventually claim 44 houses. Luckily, no one was seriously hurt. One of the houses claimed were the local telecom facility. The fires lead to the shutdown of regular phones, mobile phones and Internet in the area. Sporadic mobile phone coverage from remote stations was possible in some areas. The lack of mobile phone coverage meant increasing difficulties in the communication between police, fire brigades and the Norwegian defense. Helicopters had to make technical stops and had problems when trying to contact the police in Lærdalsøyri regarding the possibility of UAVs flying over the area. Homemade films from UAVs were at that point in time available from several news media. A home made video posted at YouTube (`http://www.youtube.com/watch?v=xhnHTnxFj7E`) may also illustrate the extent of the fire. The strong wind also meant that the fire spread to the surrounding terrain. The fire was not contained until late the next day.

A short forecast was created using the CSWF system, using 28 nodes on a cluster to emulate 28 users. Figures 11.1 and 11.2 illustrate the location and the computed winds in the area. The wind arrows in Figure 11.1, and the white arrows in Figure 11.2 represent the same wind.

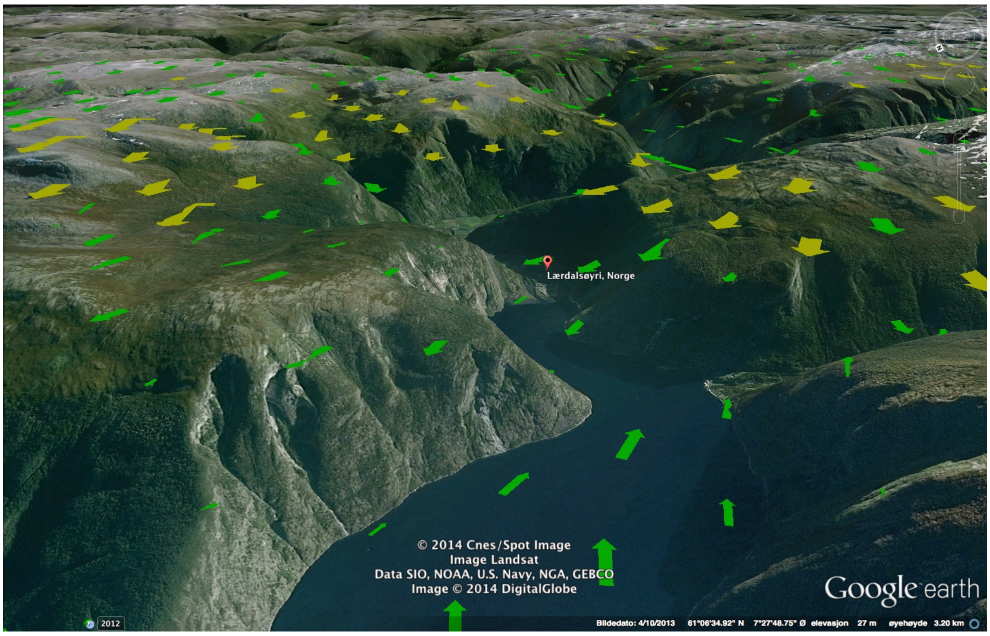

Lærdalsøyri, Norge
© 2014 Cnes/Spot Image
Image Landsat
Data SIO, NOAA, U.S. Navy, NGA, GEBCO
Image © 2014 DigitalGlobe
Google earth
2012
Bildedato: 4/10/2013 61°06'34.92" N 7°27'48.75" Ø elevasjon 27 m øyehøyde 3.20 km

Figure 11.1 *An illustration of Lærdalsøyri using Google Earth with wind arrows. Yellow arrows for* $> 10ms^{-1}$, *Green* $< 10ms^{-1}$. *The Figure is looking eastward.*

Producing these forecasts takes in total 19 minutes, including downloading background meteorological data, moving data within the compute cluster and triggering the execution of a localized model on each node.

11.2 Flatanger January 27th 2014

On the evening of January 27 2014, a small heath fire started in an area of Flatanger, Norway. The area is mostly a barren area with pockets of small-bush areas, farmland, and marshland. Strong wind spreads the fire rapidly towards the west and all people had to be evacuated from the area. Around 60 buildings were consumed by the fire, quite randomly as fires would skip one building because of variations in wind and land properties. In the affected areas, the power was deliberately cut for safety reasons and cellular communication suffered as soon as the batteries in the local cellular towers emptied. The landline telephones worked during the fire. Figures 11.3 and 11.4 shows helicopters fighting the fires and houses consumed by fire.

A short forecast for a short time period before the fire started was created using the CSWF system, using 28 nodes to emulate 28 users. Figure 11.5 illustrates the fire-affected area together with one of the wind forecasts. Figure 11.6 presents the almost same area as in Figure 11.5 and has arrows for the same wind speeds.

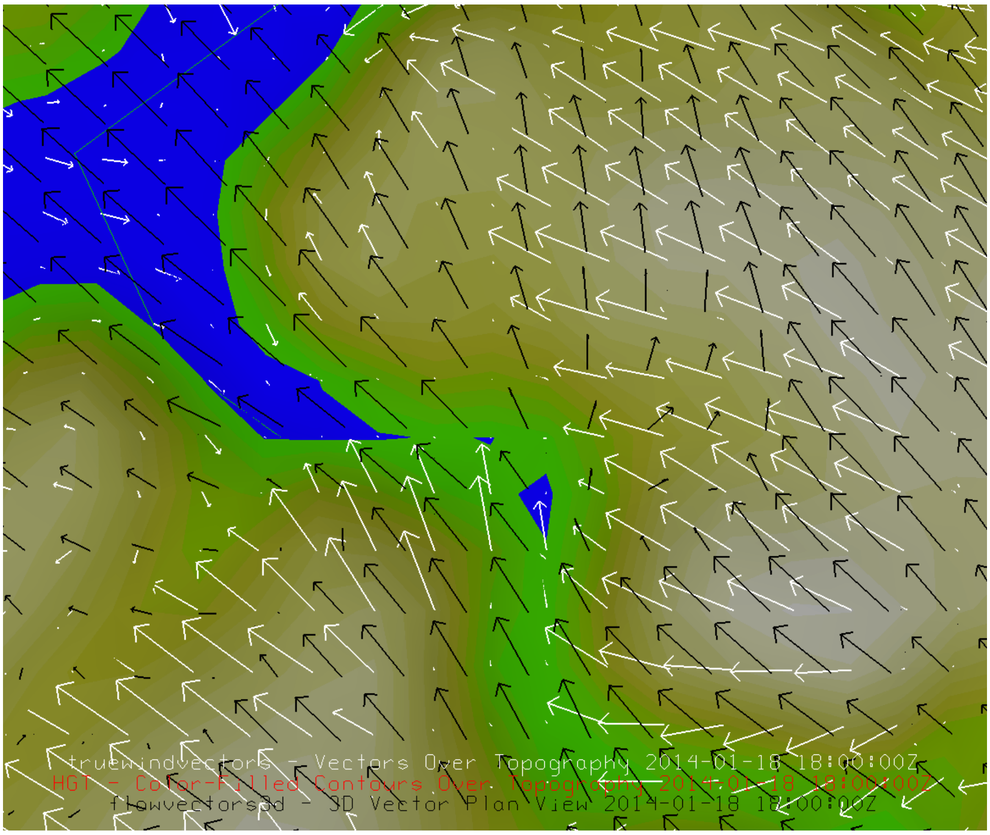

truewindvectors - Vectors Over Topography 2014-01-18 18:00:00Z
HGT - Color-Filled Contours Over Topography 2014-01-18 18:00:00Z
flowvectors3d - 3D Vector Plan View 2014-01-18 18:00:00Z

Figure 11.2 *Winds around Lærdalsøyri at 10m height (white) and at approx. 2400m height (black). North is up in the Figure.*

11.3 Lessons Learned

A few lessons can be learned from using data from these two situations. No local measurements were available within these areas, so a meteorological validation not possible.

- In Lærdalsøyri
  - A more detailed forecast than what is publicly available could have been produced in a few minutes. Whether the example is detailed enough is debatable, as it does not recreate the strong winds in parts of the area. This may be a consequence of the background meteorological data used.
  - If the initial download of background data is made each day, the system is not dependent on being connected. A single usable forecast can still be locally produced. The issue of un-connected operations has not been focused on previously in this dissertation.
  - With some connectivity, a user may utilize forecasts from neighbors, providing added value to the local forecast.
  - Some parameters not available elsewhere like wind speeds at different height, are easily available with the local forecast. Figure 11.2 illustrates the winds

at approx. 2400 m. Notable is the difference in the wind direction in the fjords in the upper left corner and at height. This would indicate turbulence as a possibility for aviation activities in the area.

- In Flatanger
  - The one locally generated forecast have more details than what is publicly available at this time, and in this situations it adds significant differences in wind speed within this small area that may be important information for the firefighters.
  - Adding the symbiotic forecasts provides uncertainty estimation for the wind, providing guidance for forecasts use.
  - From the initial fire-reports, until evacuation took several hours. This is sufficient time for producing and using new forecasts.
- Both areas
  - Both areas are in remote parts on Norway where very detailed forecasts are not publicly available.
  - Strong winds were a major contributing factor for the rapidly and uncontrolled spread of the fires.
  - The situation lasted many hours, suggesting that producing forecasts in under 20 minutes would still be beneficial.

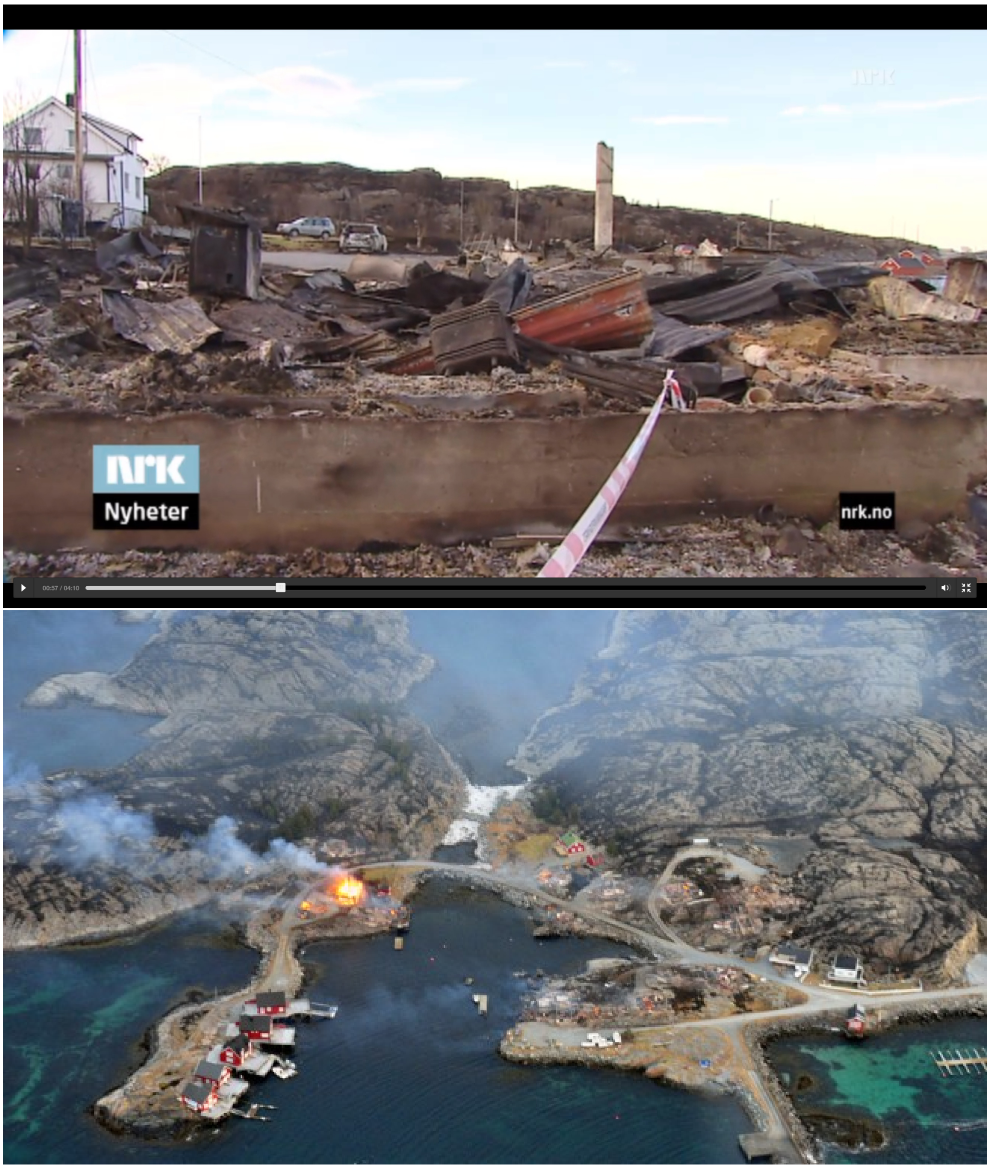


Figure 11.3 *During and after the fire in Flatanger, Norway. (Top Copyright NRK), (Bottom Photo: Morten Hegdal/Politiet).*

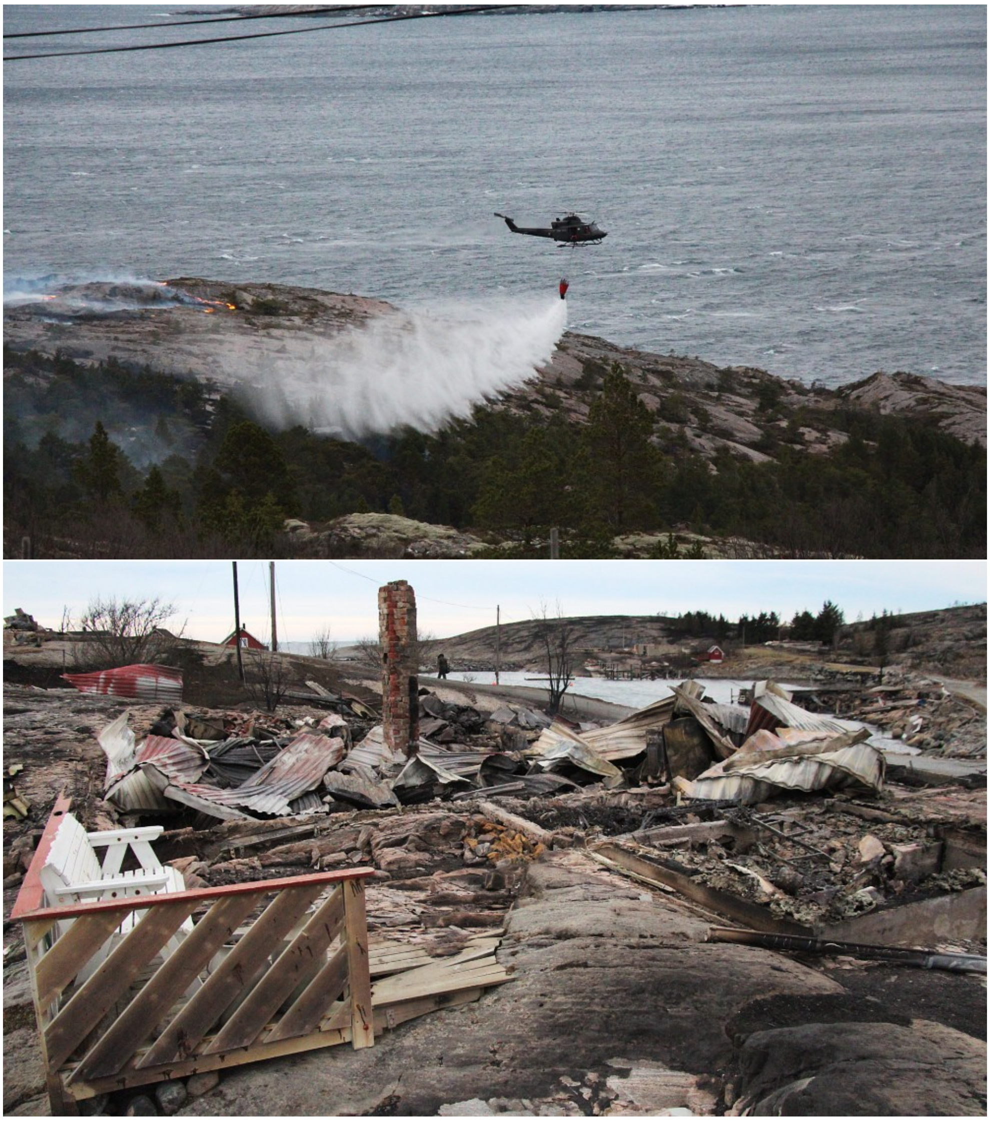


Figure 11.4 *During and after the fire in Flatanger, Norway. Photo: Joar Elgåen/NRK.*

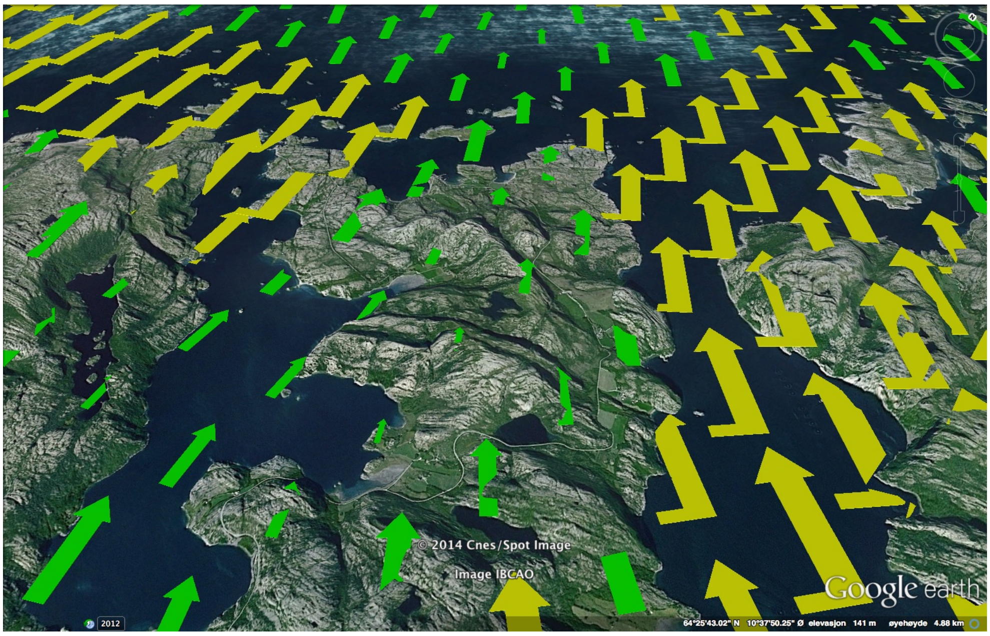

© 2014 Cnes/Spot Image
Image IBCAO
Google earth
2012
64°25'43.02" N 10°37'50.25" Ø elevasjon 141 m øyehøyde 4.88 km

Figure 11.5 *An illustration of the fire effected area of Flatanger, Norway, using Google Earth with wind arrows. Yellow arrows for* $>$ *10* $\mathrm{m/s}$, *Green* $<$ *10* $\mathrm{m/s}$. *The Figure is looking westward.*

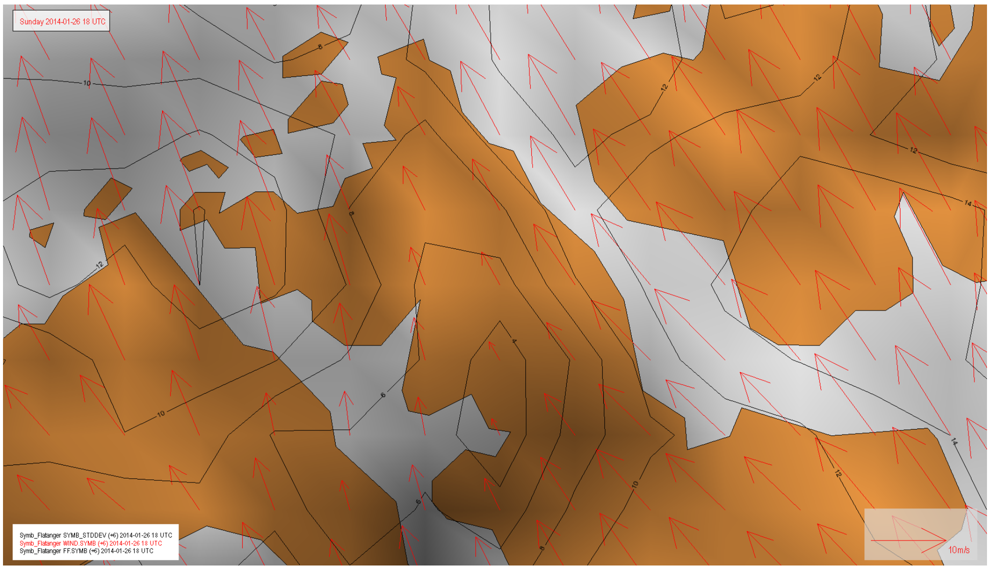

Sunday 2014-01-26 18 UTC
Symb_Flatanger SYMB_STDDEV (+6) 2014-01-26 18 UTC
Symb_Flatanger WIND.SYMB (+6) 2014-01-26 18 UTC
Symb_Flatanger FF.SYMB (+6) 2014-01-26 18 UTC
10m/s
10
12
14
8
6

Figure 11.6 *Winds around Flatanger at 10m height (red arrows), wind speed at 10m (black contours) and standard deviation of wind speed (black shading). Wind speed varies from 4 - 14* $\mathrm{m/s}$. *Standard deviation varies from 0-3* $\mathrm{m/s}$. *North is up.*

12 Contributions

This chapter describes the contributions of this dissertation and divides the contributions into sets of *principles*, *models*, *artifacts*, *facts* and *lessons learned*.

12.1 Principles

12.1.1 Principle 1

A numerical forecast computed on a super computer can be moved to a personal computer by reducing the problem size by O(C_{SC}/C_{PC})

C_{SC} Number of cores used on a supercomputer for computing a forecast.

C_{PC} Number of cores on a personal computer

We name this principle the *Super-computer to Personal-computer* principle, or in its short form *sc-to-pc*.

Numerical models used in weather forecasting are massive parallel and computationally intensive. The computational work in the numerical models is often divided into regions and sub regions, patches, to fit the available compute nodes, number of CPU cores on each node, and number of threads possible on each core.

The computations that must be executed for each mesh point is the nucleus of a mesh (grid) based model, and is often iterating over all mesh point within the assigned patch in an inner loop of code. At least a large part of this inner loop is non-parallelizable and must be executed sequentially. The sequential part limits the speedup as the number of cores increase as described by Amdahls law [3]. The law can be represented by the following equations:

$$\begin{aligned} W_s &= fT_1 \\ W_p &= (1-f)T_1 \\ S_p <&= \frac{1}{f + \frac{1}{N}(1-f)} \end{aligned}$$

where f is the sequential fraction of the program using T_1 seconds to execute on one processor, executed on N processors, W_s is the serial work, W_p is the parallel work and S_p the speedup. Amdahl's law shows speedup using implementations on several cores and a single core with the same computing capability [84, sec. 4.2.1]. The speedup is limited by the sequential fraction of an application.

Amdahl's law applied to numerical weather forecasting tells us that there is a limited speedup when the number of mesh points get close to the number of processors and/or threads available. Amdahl's law implicitly assumes that the size of the problem, or at least the part that can be executed in parallel, is static and does not depend on the number of processors. Gustafson [50] assumes that the problem size scales with the number of processors and developed a scaled speedup:

$$\begin{aligned} S_{scaled} &= \frac{s^* + p^*N}{s^* + p^*} \\ &= s^* + p^*N \\ &= N + (1-N)s^* \end{aligned}$$

where s^* is the sequential part of a program executed on a parallel system, p^* is parallel part, and N is the number of processors.

Both for Amdahl's and Gustafson's laws is the sequential part of any program the limiting parts for scaling. Hill and Marty [55] augments Amdahl's law for multicore hardware and points out the need for utilizing more parallelism in programs and the need for also making sequential cores faster, thus making programs faster even with Amdahl's assumptions.

For a numerical atmospherical models, a refinement to Amdahl's law can be stated as: *The limiting factor for model speedup will be the time needed to compute a single time-step for the smallest patch of mesh points on a single core.* That is; the sequential execution of the smallest set of mesh points on a single core.

The major factors influencing the length of the computational time for the WRF model used in this dissertation, can be investigated using a set of equations expressing relationships between various estimates. The equations do not represent an exact relationship between variable, only that one variable is dependent on others. The equations show the relationship in sufficient detail to be able to discuss the effect of changing some of the variables.

The assumption is that the work can be parallelized for execution on any number of cores, and that the time it takes for executing a single time-step is only a function on how many mesh points a single core must compute for.

T Duration of computation. Time to compute one forecast.
P Total size of computation. Rough estimate only.
D Duration of forecast, in hours or seconds. Typical value 21 600 seconds.
R Resolution of forecast, in m. Typical value 1000 m.
G Grid size. Size in north/south or east/west direction. Typical value 40.
C Number of computer cores to execute on. Typical value 8.
K Representing the mean length of time for one time-step in the model. Typical value 5 ms.
β a scaling factor for the length of the time-step. $\frac{30}{10000}$ in the prototype.
t_l Length of time-step. 3 seconds for a 1000 m forecast.

$$T \simeq \frac{P}{C}\,K \tag{12.1}$$

$$P = G^2\,\frac{D}{t_l} \tag{12.2}$$

$$t_l = \beta\,R \tag{12.3}$$

$$T \simeq \frac{K}{\beta}\,\frac{D\,G^2}{C\,R} \tag{12.4}$$

The general shape of the function for the time it takes to execute a forecast $\frac{n}{X}$ where X is the number of cores. This shape is similar to the results illustrated in Figure 4.2, with diminishing returns as the number of cores is increased.

Equation 12.1 expresses that the total time for a forecast is the total size of the computation divided by the number of CPU cores used, times a constant factor representing the length of one single time-step. This is a rough estimate neglecting pre- and post-processing and the varying numbers of writes to files etc. The equation includes the effect of Amdahl's law and is valid for workloads on computers with a limited set of cores.

Equation 12.2 expresses the total workload of a model with a mesh size of G x G points, for a forecast with duration D and length of time-step t_l. The number of mesh points increases quadratically with G and the total number of time-steps needed is the duration of the forecast D divided by the length of each time-step.

Equation 12.3 express the relationship between the length of each time-step and the spatial resolution used. In the developed prototype the β was set to $\frac{30}{10000}$. This number originates from the relationship that a 10 000 m forecast needed a time-step of 30 seconds and a 1000 m forecast needs a time-step of 3 seconds. These numbers are found by experimenting with various numbers and choosing the smallest number that kept the specific implementation for the model numerically stable in areas with complex terrain.

Equation 12.4 shows how the total time for a forecast will increase as the resolution R is increased ie. using a lower numerical value for R. The time will decrease as the number of cores used are increased, and increase quadratically with the size of the mesh sides. This equation does not take into consideration several aspects. On aspect not included is what happens if the computation is divided over several nodes and communication cost increases. The equation is only representative for computations using a single computer. The equation can be used for discussing the effect of lowering the time it takes to compute one time-step, the effect on increasing the number of cores on the computer and the effect of the size of the region used in the forecast.

Equation 12.4 illustrates that the effect of lowering the time it takes to compute a singe time-step will allow a similar increase in the model resolution and still maintaining the same computational time. For example could a 20% reduction in the time it takes to compute a single time-step could be matched with a 20% increase in the resolution. Equation 12.4 illustrates the five major parameters that must be tuned by any meteorological service using numerical models in a compute-resource constrained environment. The effect of increasing model complexity is represented by the K parameter. A more complex model may lead to longer computing time for each time-step.

The effect of the combination of Amdahl's and Gustafson's laws applied on the computation of the WRF model is that a model traditionally executed on a supercomputer can be reduced in size and still execute with a similar speed on a comparable personal computer. The very good scaling, in Gustafson's view, of numerical models allows for large supercomputers to efficiently execute large models as long as the size of the mesh used is large enough compared to the number of cores used. And the opposite is also valid; a model with a small mesh size can be computed on a personal computer using few cores with acceptable speed as long as the sequential parts of the model can be computed on the personal computer with sufficient speed.

Figure 12.1 shows the variation of the computational time of each time-step during a forecast. The Figure shows the times it takes to write out results to files at 12, 15 and 18 hours, and also the effect of computing the radiation physics every 10 minute of the forecast. Writing results to files increases the time by a 4-6 factor, and the radiation physics almost doubles the computational time for each time-step. How often the radiation physics is computed is a user controllable parameter. Figure 12.1 also illustrates two periods of unexplained increases in computational time, perhaps resulting from other processes executing on the same node.

Modern supercomputers are increasingly heterogeneous computers with a mix of traditional multi core CPUs and coprocessors like the Intel Xenon Phi or NVIDIA Tesla. Although the implementations of many numerical weather forecasting models do not make use of such coprocessors, this is an expected direction for research and development. Johnson et. al [62] is an example of using numerical model on a modern heterogeneous platform.

Johnsen et. al. [62] provides a discussion of the weak scaling shown by the WRF model. Johnson shows that the average time for computing a time-step in the model is not related to the resolution of the model. The comparison in [62, section 5.2] shows that the single node 32 core, single OMP thread per core, only performed slightly worse than the 209952 core case. The 209952 case having a vastly larger mesh size, but the part of the mesh computed at each node, the patch size, was very similar. The patch size used is of the same order as used in the prototypes developed in this dissertation. This is also an example of Gustafson's law where good scaling is achieved through increases

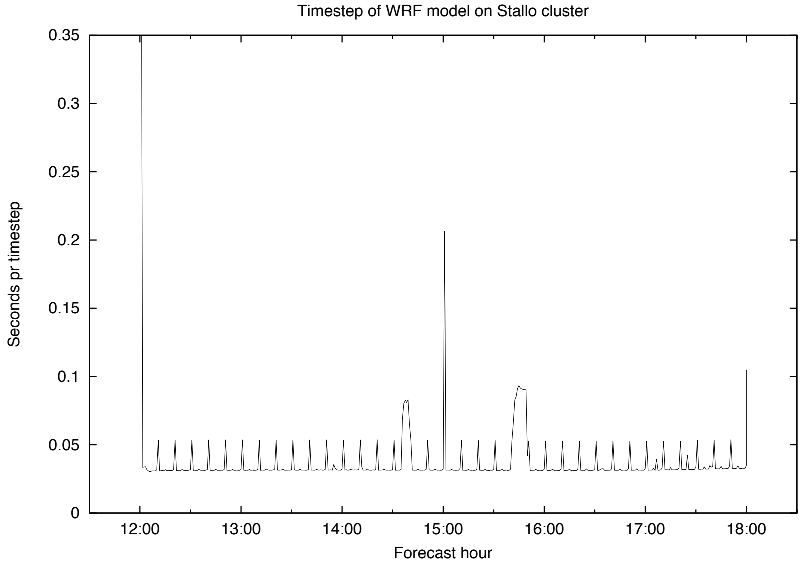


Figure 12.1 *The computational time for one time-step. The figure show a six hour forecast starting at 12 UTC. The computation is executed on a single one quad core node with hyper-threading enabled and 8 MPI processes on the Stallo supercomputer. The figure shows the additional time it takes to write out results to a file at 12,15 and 18 hours, and the effect of radiation physics computed every 10 minute of the forecast. Two episodes around forecast time 15:30 and 15:45, of unexplained longer time for several time-steps can also be seen.*

in the problem size.

Johnson et. at. is also an example of the order of reduction needed in the problem size using fewer cores. Going from 209952 cores to 32 cores reduces the number of cores with a factor of 6561. The model area was 6075x6075 kmwith resolution from 1 km to 81 km. The mesh size is reduced by a factor of 81x81 going from the mesh with 1 km resolution to the 81 km resolution, also a factor of 6561. Johnson et. al. uses this reduction to illustrate the weak scaling of the WRF model, and the size of the reduction is tailored to the specific hardware used. In this dissertation, we make use of the weak scaling properties of the WRF model for using personal computers for computing a numerical forecast.

12.1.2 Principle 2

Short traveled data and computations.

The term "Farmers Market" is often used to describe places where local farmers are selling directly to customers. Compared to traditional products in supermarkets will products sold at farmers markets often need:

- Less transport
- Less handling
- Less refrigeration
- Less time in storage

Other related examples are garage sales and neighborhood markets, all selling locally sourced items.

For locally produced weather forecasts stored on local systems and visualized on user devices, the list above translates roughly into:

- Short distance between producing system and user devices
- Only desired data are produced
- Data is produced on-demand
- Only the desired sub-set of data are presented and visualized

The locality principle is here generalized to multiple dimensions. This has obvious benefits relating to the stress on the large region or national infrastructure. Only local data networks are used, most communications happen within the private network of a user. Even when sharing data between users in the CSWF system, mostly local data networks are used. The implemented method of finding other CSWF systems ensures that the traffic is never widely routed, as only network addresses within the same segment is probed.

4G and 5G mobile networks also have multidimensional locality with network densification is one of the proposed technological approaches [12, 4]. One of the proposed new communication patterns is Device-to-device, D2D. This pattern may improve latency, bandwidth and power consumption, mostly for communication within single households or buildings. Allowing D2D relay may also introduce cooperative communications that may be utilized by a CSWF type of system. In contrast to a CSWF system, most examples of D2D scenarios in Andreev et. al. [4] involves external third party content providers where two users both wants to access the same information. D2D will in effect move content sharing from the situation where two parties are accessing content from a remote third party provider, to a situation with one user downloading the content and then sharing this between spatially close peers. Local sharing will introduce issues regarding caches and updates.

The developed CSWF system confirms that the extension of the principle of short traveled data and computations into numerical weather forecasting is viable and has some valuable benefits. The locality principle is reflected in the following properties and characteristics of the developed systems:

- Forecasts are produced locally by the user on the users own hardware.
- The user can customize the forecasts by changing parameters like resolution at run-time.
- More parameters and levels are available from the model and can be customized for the user specific requirements.
- The forecast are visualized locally by the user using any application capable of reading the standardized files produced by the model.
- A forecast can be visualized placing the user at a current geographical location within the locally produced forecast using an augmented reality style applications without depending on external data sources.
- Collaboration between peers are based on their geographical locality or geographical region of interest. A user will most probable be most interested in a nearby region.
- The establishment of trust in peers can be locally assessed by each user. The same data can also be used for quality control of others users forecasts before including the forecasts into an amalgamated forecast.
- All data are stored close to the user and can not be manipulated remotely, and can even be used in case of external communication failures. A new forecasts can for some time, be initiated using previously downloaded background data at any time without the need for communication outside the users own systems.

The difference between the CSWF system and systems like Facebook[1], Wolfram Alpha[2], Yr.no[3] and YouTube[4] in terms of locality relative to the users can be illustrated by Figure 12.2. "Yr.no" is the weather service provided by the Norwegian Meteorological Institute and the Norwegian Broadcasting. YouTube is a service where users are uploading home produced videos. Facebook stores user generated data like text and images and shares this between users. Wolfram Alpha computes answers to questions asked and presents this to the user; no user data is stored by Wolfram Alpha. The circle labeled "neighborhood" illustrates where multiple CSWF systems within a small region would be found. MS Office and Apple iWorks are both a set of applications providing text editing, presentation tools and spreadsheets. Both packages can store data locally or in the cloud and have either a local application for editing or cloud-based systems for editing in a web browser.

The x-axis describes the degree of computational locality in relation to an individual user, and the degree is a measure on where the computation related to production of content takes place. The y-axis describes the degree of storage locality in relation to the control with the stored data. Only data related to an individual user is considered.

[1] www.facebook.com

[2] www.wolframalpha.com

[3] yr.no

[4] www.youtube.com

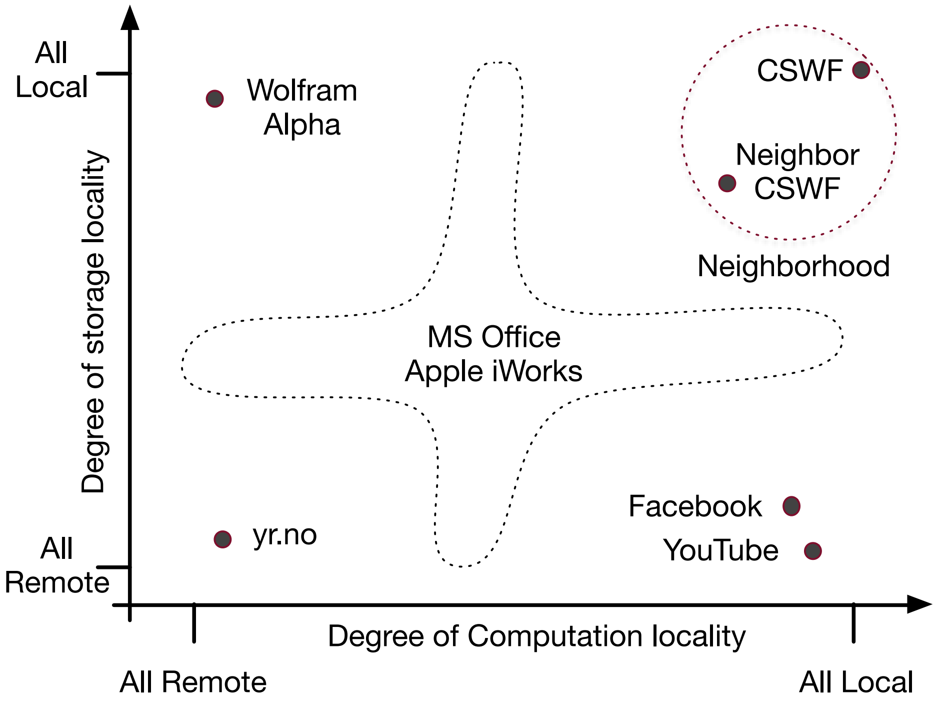


Figure 12.2 *Comparison of where different systems are located on a 2D locality coordinate system with regards to compute and data storage locality. Degree of computational locality on the horizontal axis, degree of storage locality on the vertical.*

The yr.no service computes all forecast remote for the users, and stores all forecasts on systems under their control. All visualizations are created remote to the user. Youtube and Facebook stores all data on systems under their control. An explanation of Facebook's storage and distributed caching system for images is given by Huang et. al. [57], even though much content is served from local browser caches, this is not considered storage here. All content are produced locally by the users, and this is the computational part focused on here. Wolfram Alpha does all computation on their systems but does not store any data produced by the user, all visualization and answers are produced and shipped to the users browser. The user may or may not store the results. The office packages have options for local and remote storage and can use local or remote computing when working with data. The CSWF systems produce and store all data locally. A collaborating CSWF system will be located geographically close to each other, preserving the short distance between the user of the forecast and the computation and storage.

The differences between the developed CSWF system, and well-known systems may be viewed in a multidimensional way, Figure 12.3 illustrates this. The axes are all with increasing values or weight from the center. The axes are: Degree of Computational locality, degree of Storage locality, number of cores used to compute the results, computational time, data size, number of clients, degree of On-demand processing and external communication demands. Figure 12.3 represents the view as seen from a single user. The axis are not to scale.

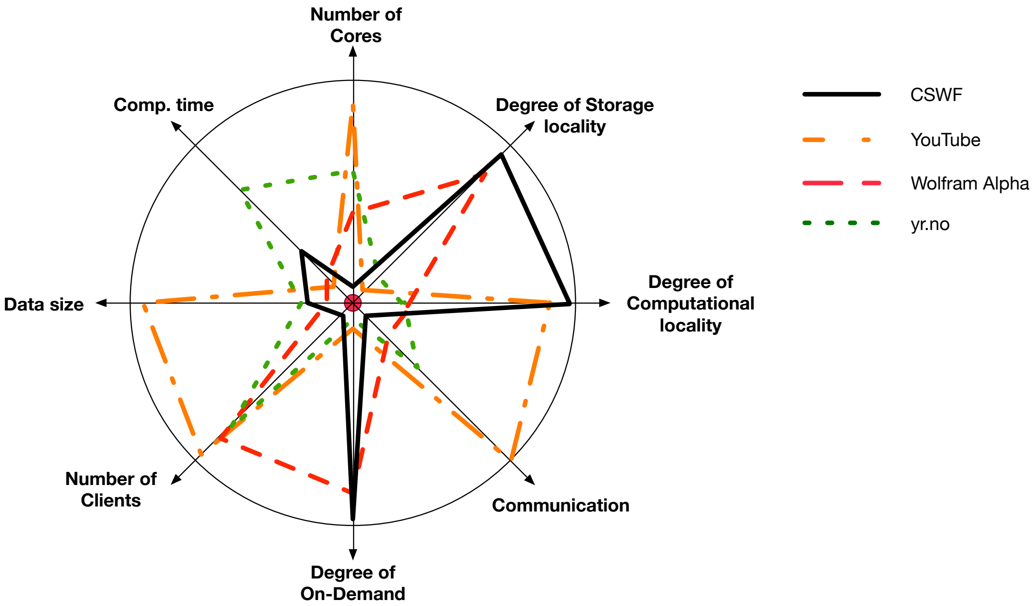


Figure 12.3 *Comparison of different aspects of known systems compared to the CSWF system as seen from a single users. The axis are not to scale.*

Some points to Figure 12.3:

- Most of the compared systems do their computations remote, the exceptions being the CSWF systems and YouTube where all users related data is produced locally.
- All systems store data remotely except for the CSWF system that stores locally, and the Wolfram Alpha system that, to our knowledge, does not permanently store results of user requests.
- The number of cores used is large or very large in all systems except the CSWF systems.
- Computational time is in Figure 12.3 taken as the interactive waiting time. The CSWF has longer waiting time compared to most of the other systems because data is not produced before it is being requested. The other systems deliver preprocessed data. YouTube is assumed to have uploaded the user provided video on a previous occasion, if not YouTube would have significantly longer computational time than the other systems.
- The CSWF system also produces more data than most systems in regard to the single user, but not as much as the videos produced for YouTube. CSWF is in this Figure illustrated as producing more data than yr.no. This is in regards to the forecast data for a single location or small region made available by yr.no. Yr.no

has data covering the whole globe and has a larger total amount of forecast data available.

- The number of clients in all the other systems is much larger than the single user of a CSWF system.

- The degree of on-demand computational is high in the CSWF system and in the Wolfram Alpha system. Both systems produce answers and data only on-demand, and very little is preprocessed. The rest of the systems use pre-computed data, although the visualizations may be customized for the actual user.

- The external communication requirements in relation to a single user is only large for YouTube, a user has to either upload a large video for viewing and sharing or viewing large volumes of data in form of videos. The CSWF system only communicates locally between a users own devices and computers, and if sharing forecasts also is included this is between a limited set of local users.

12.2 Models

This section describes the system models developed in this dissertation.

12.2.1 Embarrassingly distributed computations

An embarrassingly distributed computation is a computation that is spread out over many computers with no or very little communication between the computers. This is analogous to embarrassingly parallel computations where each node in a parallel computation executes with little communication with other nodes.

Since all users in the CSWF systems described in this dissertation is executing their own forecast independent of other users activities, the CSWF system is an realization of an embarrassingly distributed computation.

The CSWF system can be executed in total isolation from other users. The system will still produce very high-resolution weather forecasts for the region of interest. The user will has more data available for visualization than elsewhere publicly available.

If a CSWF user collaborates with neighbors, this introduces a small amount of communication, but no coordination. The forecasts are collected if they already are produced and are within the period and region of interest.

12.2.2 Symbiotic Collaborative Weather Forecasting

The CSWF system demonstrates a novel model of collaborating in weather forecasting. End users will compute local forecasts and share these for additional value. The level of cooperation is fully user controlled and will scale as a user only collaborates with as many other users as deemed necessary. A user may also throttle request from other user by simply not making a forecast available or the computer visible to other CSWF systems.

Collaboration in weather forecasting have until now been either collaboration within the meteorological community on programs, observation collection and models, or collaborative systems used by weather forecasters and users for sharing the interpretation of a set of given numerical forecast into specific graphical or text forecasts. The latter can be illustrated by the Collaborative Weather Forecast System, CWFS, system from Info-Electronics Systems Inc. [58], that allows interactive chats between forecasters combined with visualization of meteorological and aeronautical information.

Collaboration between end-users producing numerical forecasts using numerical models has not been found in the literature. The system developed in this dissertation represents therefore to our knowledge, a novel model within numerical weather forecasting. Two approaches are new to this model. 1) The Users are producing localized personal forecast based on lower resolution large region meteorological background forecast 2) users with interest in the same geographical region are collaborating by sharing forecasts and generating uncertainty estimates.

12.2.3 Scalable Distributed Weather Forecasting

The CSWF system provides a scalable distributed system for doing weather forecasts. A single CSWF system will collaborate with a small number of neighbors found by probing the local network. The number if collaborators may be limited by the user and a user may configure the CSWF system to only respond to a limited number of requests for forecast.

The restrictions placed on a single CSWF system does not limit the total number of CSWF systems that can produce forecasts, only the workload on each of the CSWF systems. CSWF systems without overlapping regions of interest would not want to collaborate.

It is also notable that each CSWF system ads a new computer to the collaboration and each new CSWF system also produces its own forecasts. The effect of adding more systems to the collaboration would be a small increase in the communication volume. Producing a symbiotic forecast from forecasts collected from collaborating systems is a lightweight process and only takes a few seconds.

12.2.4 Uncertainty and Error Estimations of Weather Forecasts based on Collaborative Exchange of Forecasts

Uncertainty estimations in modern weather forecasting are traditionally made using an Ensemble Prediction System, EPS[5]. The EPS uses the same model with same mesh location but using slightly different meteorological starting conditions and/or background data. The ECMWF uses 51 models in its ensemble product, all with a spatial resolution of around 30 km. The resolution is lower than the operational high-resolution model and is limited by the huge computer resources needed for executing that many models.

One example of EPS use is the situation when Hurricane Sandy hit the east coast of the United States. Even eight days ahead, a significant number of the 51 forecast in the EPS had a possibility for the hurricane to strike the New York area. The best EPS system

[5] http://en.wikipedia.org/wiki/Ensemble_forecasting

in that situation was the ECMWF EPS forecast according to some reports [7]. Using the uncertainty estimates from the EPS system allowed for making informed decisions on what actions to take and preparations for a potential serious situation could be started early.

The uncertainty estimation produced by the CSWF system originates from a different, but related, problem in numerical weather forecasting; What is the best description of the background geography and state of the land and sea in the model. Before executing the main WRF model in the CSWF system, several pre-processing tasks must be computed, the WRF Preprocessing System, WPS.

One of these tasks is the *geogrid* application that re-samples the various very high-resolution terrestrial data sets to the actual model mesh (grid) used. If started with slightly different center location, the *geogrid* application will generate slightly different versions of parameters like terrain height and land use. These differences lead to small differences in the subsequent numerical forecasts. These differences represent one of the uncertainties of the model. Estimates of uncertainties related to effects of the mesh location in model implementations are not today made available in public forecasts.

Using collaborative forecasts for estimating the uncertainty is novel in two ways. 1) Uncertainty originating from mesh locations in the model implementations are estimated using several forecasts with the same model. The meteorological background conditions are the same but the center locations is slightly different. 2) Forecasts used for estimating uncertainties are computed by end-users and exchanged within a small region.

12.2.5 Forecast Visualization using Augmented Reality

Traditional visualization tools for weather forecasts all place the user as a remote spectator to the forecasts. The augmented reality visualization application described in section 8.3.1 is intended for use when the user is physically located within the forecast region. The intended use is for the user to look around and have meteorological parameters overlaid the view as seen through a tablet. This illustrates a new approach by producing forecasts on-demand centered on the users actual location and visualizing the forecasts seen from this location as an overlay on the real world.

One of the main advantages is that information is limited to what is relevant, in this case, to the direction of view. Additional research will show how to visualize 3-dimensional meteorological data using AR techniques in an efficient and user-friendly manner. The work in this dissertation should be considered a proof-of-concept. It is likely that this approach may be suitable for devices like the Google Glass or head-up displays in cars.

Previous systems, for example as presented in Gliet et. al. [47] or Make Magazine [71], have added emulation of simulated weather to AR systems. This dissertation describes a system using actual on-demand very high-resolution weather forecasts, not simulated weather or weather information from external sources.

The approach used in this dissertation differs from traditional techniques like Caves and VR helmets and other immersive virtual reality environments by displaying a forecast from the users actual geographical location. The CSWF system can produce forecasts for other locations on-demand; these forecasts would be useful in Caves and VR helmets where the view from the actual location is not used.

12.2.6 Computing Trust in Locality Based Collaborative Weather Forecasting Systems

The options the user has for validating exchanged forecasts has previously been discussed. The locally produced forecasts can be used for establishing a index of how different an exchanged forecast is from the locally produced forecast. This index may be used for evaluating trust between peers. Trust can also be computed or estimated by monitoring the difference of peer forecast from the local forecast over time. A forecast from any peer can also be validated by re-computing the same forecast locally.

12.3 Artifacts

Artifacts are prototypes and software created during the work with the dissertation. Products produced by this software are also examples of artifacts. Artifacts are tangible objects even if they exist mostly in digital form.

12.3.1 Prototypes

Prototypes produced in the included papers are software, applications and scripts, that implements a system. The prototypes are implemented to investigate issues with the proposed system and as instruments for experiments and measurements. Complex interactions and issues between system parts are often only visible when executing the system.

A System for On-Demand Production of Very High-Resolution Numerical Weather Forecasts

The prototypes are several scripts and applications described in Chapter 6. The prototypes handles requesting a forecast for any geographical location with specified resolution, producing NetCDF files that can be visualized using commonly available software. The prototypes also produces forecasts in KML formats that can be visualized using an AR type application, and point forecast for a given location for use on smart watches, see Section 12.5.7 for an illustration.

A System for Collaborative Weather Forecasting using Peer-To-Peer Exchange of Forecasts Between Unknown Neighbors

The prototype is the CSWF system described in Chapter 9. A CSWF node can produce a local forecast on-demand either requested using a web page, or requested from a users mobile device utilizing the location of the device. A CSWF node will probe the local network for neighbor CSWF systems and may exchange forecast with these. The exchanged forecasts are amalgamated into a symbiotic forecast and made available for visualization on a users device.

Augmented Reality System for Visualization of Forecasts using a User-Location Centric View

The Augmented Reality type of application is described in Chapter 8. The application uses the built-in GPS when requesting a forecast. The forecast is visualized as an overlay of the camera input. Both accelerometers and the compass are used for matching the view of the camera with the forecast data. The application uses data in the KML format, and these data can also be visualized using other applications like Google Earth and Google maps. The application is implemented on iOS devices, but available only on registered development devices within the HPDS[6] lab.

12.3.2 Output from the Prototypes

This section describes some of the products produced by the prototypes described in the previous section. Examples of produced files are available electronically from `http://hpds.cs.uit.no/people/bfj002/index.html`.

Local Forecasts with Three Predetermined Resolutions

The local forecast is produced for any given location. A NetCDF file is retrievable after the execution of the model. The CSWF system uses a model setup that may be altered by changing the execution script. The size of the region covered by the forecast is a function of the resolution and mesh size. A one km resolution forecast using a 39x41 mesh size will produce meteorological sound forecasts for a region of approx. 840 $\mathrm{k^2m}$. Appendix A lists an example of a forecast.

Amalgamated Forecasts from Collaborative Weather Forecasts

Amalgamated forecasts from several users create a novel type of forecast. The forecast is represented using the mesh of the local forecast for each user. The forecast contains mean values of requested parameters, number of forecasts used in each mesh point, and the standard deviation of the parameter at each mesh point. Appendix A lists an example of an amalgamated forecast.

Estimation of Uncertainty in Wind Speed Forecasts

One example of this is the standard deviation of wind speed together with the mean wind speed forecasted. Standard deviation is illustrated in Figure 9.5. The standard deviation will give the user a measure of the local variation of the forecasted wind speed. The variation may be used for assessing the risk for the wind speed to exceed certain limits. Appendix A lists an example of a forecast with standard deviation as one of the parameters.

[6] `hpds.cs.uit.no`

Visualization of Weather Forecasts

Several types of visualization are produced by using the prototype applications. Since the model produces standardized NetCDF files, consequently several industry standard applications can be used for visualizing the forecasts, in 2D and in 3D. All visualizations have been illustrated with figures earlier in this dissertation.

12.4 Facts

The following facts have been established in this dissertation:

- The WRF model can be executed on a commodity 2012 era computer producing very high-resolution numerical forecasts for a small region and a short duration within a few minutes depending on the resolution and forecast duration selected. A 39x41 1km 6 hour forecast executes in approx. 874 seconds on a Intel Xenon 3.06 GHz Quad core desktop personal computer.
- Weather forecasting using higher spatial resolution than used for publicly available data reproduces known an possibly important features not observable in current public available weather forecasts.

12.5 Lessons Learned

This section lists some lessons learned during the work with this dissertation.

12.5.1 Suitable Trade-Offs for Sound Forecasts

The computational time for producing a weather forecast using a numerical atmospheric model is dependent on at least three parameters that can be user specified; Size of region, spatial resolution of model and time duration of the forecast. One of the effects of changing the spatial resolution is that the time-step in the model has to be adjusted. To ensure numerical stability, higher resolution models must use a shorter time-step and thereby increasing the number of time-steps needed for a forecast of a given duration. The length of the time-step needed for a numerical stable model is strongly dependent on the gradient in the background meteorological and physical phenomena. E.g. steep mountains and other high gradient features require shorter time-steps.

Johnsen et. al. [62] used the WRF model with a very large mesh size; 9120x9216 points, 48 vertical levels and a total of 1.4 TB of input. The experiment simulated the landfall of hurricane Sandy. The implementation used a patch size of 37x10 points for their single node, 32-core experiment.

In the prototypes developed in this dissertation the WRF model is executed with a total mesh size of 39x41 independent of spatial resolution used. The WRF model is in the prototype executed using eight processes on four cores with hyper-threading enabled. The mesh size was chosen because of the problem with relaxation, see Chapter 3 for details on relaxation, which remove up to five mesh point on all sides of the region from the fully resolved region of the forecast. The chosen mesh size results in an effective region of at least 29x31 points. At one km resolution, this covers a region of approx. 840 km^2. This is clearly a useful size. The one km resolution is at least 2-3 times better that what is publicly available at this time.

12.5.2 Better Weather Forecasts

One way of evaluating new methods and systems for numerical weather forecasting is by evaluating how the new system addresses known problems or challenges. Some of the problems and challenges with todays weather forecasting from the national weather services, can be stated in a few points. Can the systems developed in this dissertation do some of these challenges better? The challenges are presented and the response to each is listed as sub-points.

- Forecasts are only available at the best resolution possible within the constraints created by the region that must be covered, the timeframe forecasts must be delivered in and the computational resources available to the weather services.

  ⟶ Forecasts can be produced for any location on the globe. The forecasts are produced on-demand within a few minutes. Forecast resolution is decided by the user. This dissertation presents results using up to 1 km resolution, but the WRF models is reported capable of higher resolutions. See [46, 98] for examples. 1 km resolution is higher that what is currently publicly available in Norway.

- Currently the resolution publicly available does not resolve meso- and micro-scale phenomena. Examples are sea breeze, valley-winds and circulations resulting from non-homogenous land use, see [91] for R. Rotunno's presentation of mesoscale modeling at high resolution.

  ⟶ The 1 km resolution version resolves more, but not all of the meso- and micro-scale phenomena.

- Forecasts are updated at fixed intervals using batch systems regardless of the actual use at the time.

  ⟶ The prototype produces forecast updates on-demand and for requested areas and time period within a few minutes.

- Forecasting for a large region implicates a long cut-off time waiting for observations to reach the national weather services before the processing can begin.

  ⟶ No observations are used in the prototypes forecasts for very small regions, and therefore no waiting period is necessary. If a local observation within the small region were available, it would have been instantly usable in the model.

- Only selected parameters, levels and times are made publicly available.

  ⟶ All parameters, levels and times produced by the WRF model are accessible and what is included in the output is user configurable.

- Forecasts are usually computed using a fixed mesh. No information of the possible uncertainties resulting from the location of the mesh is publicly available. Uncertainties resulting from differences in meteorological starting conditions are now regularly disseminated.

  ⟶ Uncertainty estimation related to mesh locations is a major benefit from the symbiotic collaborative weather forecast.

- Visualizations are only available in pre-computed forms and only in the formats decided by the weather services.

  ⟶ Any visualization tool that can use NetCDF files can be used to visualize the forecasts. All forecast and background data is available and can be visualized.

Some of the features in the list above are available as paid services from commercial organizations. Even these paid services have not to our knowledge, reported making forecasts using different mesh placements for error estimations.

Would a national weather service benefit from providing good background data for local forecasts and collecting and re-distributing user contributed forecasts? The forecasts would have higher resolution, would cover regions and time periods with known interests in. This is probably more a question of politics than actual problems. Providing good background data would, in Norway, be in line with the current policy of better access to public data. Collecting and redistributing local forecasts would require connectivity and storage and there exists many technological solutions for this. The weather service could for example serve as a bit-torrent tracker of available forecasts, and thus minimizing network loads for the service.

The basis for the local forecast is always the available background meteorological data. This system is therefore an extension of existing centralized forecasting systems and extends the centralized forecast by adding higher levels of details at a local scale. Local forecasts will therefore not replace current systems.

12.5.3 Scalable Distributed Weather Forecasting

A distributed system for weather forecasting is realized using many CSWF systems within a region. Every CSWF system computes a forecast for their own region of interest, and the regions may overlap. The forecasts are computed independent of each other and exchanged afterwards. A single CSWF system communicates with a limited number of neighbors that shares interest in parts of the same region. The set of neighbors will be different for each CSWF system. The CSWF system does not support a large number of users communicating with every CSWF system. The total set of CSWF systems is scalable and may include a very large number of systems because each local CSWF system only communicates with a very limited set of other systems. Every CSWF system also includes its own local communication bandwidth and its computing and storage resources.

12.5.4 Localized Collaborative Weather Forecasting

The collaboration within the CSWF system is designed for simplicity. The collaboration only includes providing others with a forecast when they ask for it. This simplifies the collaborative part of the system. Peers can be discovered on-demand using a network probe within the local IP address range.

The collaborative part of the system is not necessary an aspect the user needs to be active aware of, once the user have specified the personal preferences on whether to enable collaboration.

In addition, this is a non-essential collaboration. If no peers are found, the user still has a fully functional very high-resolution weather forecast that can be used. Features like uncertainty estimations will not be available without forecast from collaborating peers.

12.5.5 On-Demand Weather Forecasts for Achieving Safety

In additional Paper 2 [38], we presented a model for using on-demand weather forecasts as a tool for managing safety concerns. The case studies presented in Chapter 11 illustrate two situations where this model would be of some benefit.

- A very detailed forecast for the affected region can be produced both locally and at any emergency operations room within a few minutes.

- The forecast can be used for planning at both a local and regional scale.

- The forecast and consequences of the forecast can be communicated to external users via radio or other means.

- The forecast can be shared with all devices within the local (in-house) network. For a centralized emergency organization, this may include different types of users.

- Using older background data for some period, a forecast can still be produced after more than 24 hours without external communication.

Since the computation, storage and visualization is not dependent on access to external sources, the multiple locality model represented by the CSWF system, allows for continued independent operations with access to very detailed forecasts for some time after an emergency occurs even with the loss of external communication.

12.5.6 Visualization of Meteorological Data on a Display Wall

Visualizing very high-resolution weather forecast on a large display wall allows for studying forecast for large regions with very high level of details. This is an direct consequence of the number of pixels available on the display. Very high level of details can be visualized using the Tromsø Display Wall [6](Additional Paper 1) with 22 megapixels with the DIANA application. The GUI of the DIANA applications is not suitable for use on the large Tromsø Display Wall. Window menus are only just visible in the top left side of the display, some icons are in the bottom left side and a toolbar is along the right side. These GUI elements are relatively small but easily viewable standing close to the screen, but are situated several meters apart. The distance makes finding and hitting the correct menu or icon very difficult. An application using a very large display should therefore use a specialized GUI, and ideally should the GUI change with the users distance to the display.

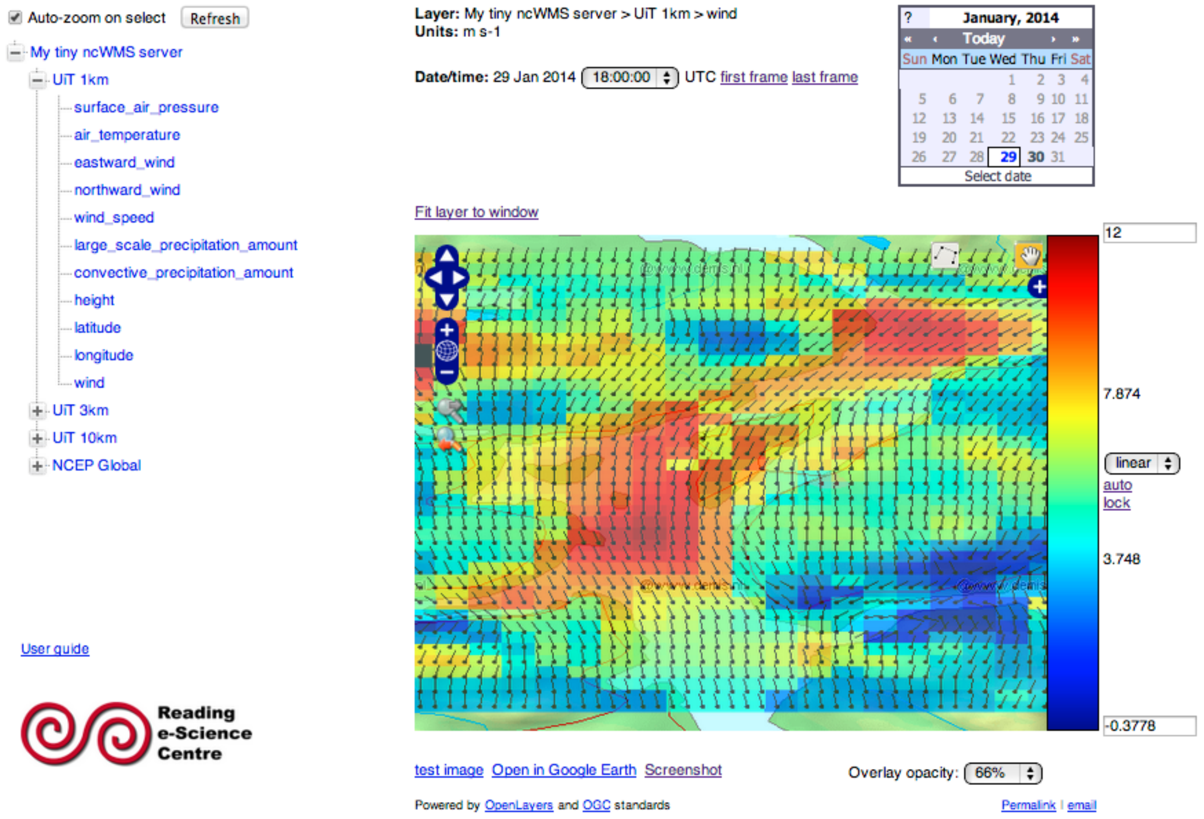
12.5.7 Miscellaneous Applications

A number of applications and scripts have been developed and used for this dissertation and the papers included here.

> **Conversion of WRF produced NetCDF files to a CF convention version**. A python script that takes one standard WRF output NetCDF file and writes a selected set of parameters to a new file using NCL[7] (NCAR Command Language). Only very few parameters and levels are converted. This generates files that are used with the DIANA visualization on the Tromsø Display Wall. Examples of use include Figures 8.6 and 9.4. The produced forecasts are also available using a local open source WMS server, ncWMS [15]. One example of the available output is given in Figure 12.4. An example file is listed in Appendix A.
>
> The lessons learned from this application are that having the full set of parameters and levels produced by the WRF model locally available, other standardized formats may be locally produced and the forecast used in a multitude of available visualization applications.
>
> **Point forecasts for a Pebble**. An application executing on the Pebble Smart Watch[8] utilizes the GPS based location on the connected smart phone and requests a forecast for the current location. A script is executed on the web server and serves a local weather forecast by executing a small application that extracts temperature, wind speed and wind direction only for the requested location. The values are interpolated using bilinear interpolation between the mesh points in

[7] http://www.ncl.ucar.edu/

[8] https://getpebble.com/



Layer: My tiny ncWMS server > UiT 1km > wind
Units: m s-1

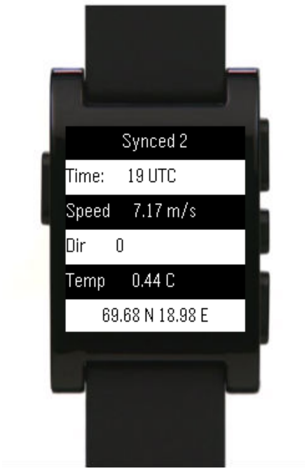
Date/time: 29 Jan 2014 18:00:00 UTC

Overlay opacity: 66%

Figure 12.4 *One example of output using the Godiva2 [16] interface to a local ncWMS server and WRF 1km data.*

the forecast. The resulting data are visualized on the Pebble. One example of output is given in Figure 12.5.

One of the lessons learned from this application is that having the whole dataset from the WRF model, interpolation and extrapolation between mesh points and levels can be used with some confidence in the validity of the methods used, and any additional parameter could be included if wanted.



Synced 2

Time: 19 UTC

Speed 7.17 m/s

Dir 0

Temp 0.44 C

69.68 N 18.98 E

Figure 12.5 *Example of point-forecasts on a Pebble smart watch. The actual location of the smart watch/cellphone is used for requesting the forecast.*

13

Related Work

13.1 The Locality Principle

Locality in distributed systems is often connected to caching and system speedup by minimizing remote access. A cache is a local copy of some remote data. If the remote data is infrequent updated, accessing the local cache may speed up the system, even if a check back to the originating system must be performed. One early example of literature on this topic is the Distributed Systems book [78] by Sape Mullender et. al. Locality is also promoted as a good design principle for distributed file systems, see for example Satyanaryanan [94].

In Section 12.1.2, locality was noted as one of the motivating aspects of new research into mobile communication networks. The IEEE Communication Magazine has recently published the first of two parts on smart-device-to-smart-device communication (April 2014, Vol 52, issue 4). The main motivating factor is the need to offload mobile network data traffic to local peer-to-peer communication because current technology are struggling to accommodate the rapid increase in bandwidth and data usage by mobile users.

13.2 Peer To Peer

Locality based on interest was demonstrated by Sripanidkulchai et. at. [97] as an important and efficient way to resolving queries in a peer-to-peer network using a minimum of hops. Interest was used for an overlay network on top of the established Gnutella network. Establishing a P2P network based on interest is similar to the geographical focus in this dissertation. A relevant comparison would be if peers did not use the geographical location as a search criterion, but another interest. For example could peers interested in powers production from wind farms be organized in a peer-to-peer network. A different way of finding peers than what is used in the CSWF system would be necessary as peers would not necessary be geographically close.

Recent research by Castellà [18] analyses using a locality based overlay on regular peer-to-peer network. The proposed architecture uses a two level approach with the top level organized according to some locality classification and the bottom level a set of trees where each tree gathers nodes with similar classification values. If this structure was used in the geographical setting in this dissertation with a semi-permanent peer-to-peer network, the classification could be done using the geographical area as the top level classifier, and all nodes within an area gathered in a single sub-tree. This structure would restrict searches for neighbors to a single tree, and make searches for neighboring areas also efficient.

13.3 Secure Computations and Data Security

The CSWF system does not make any assumptions on the security of data produced by neighboring peers. The trust can be computed after data is received. The system does make assumptions about the trustworthiness of other systems it exchanges local forecasts with. Local forecasts exchanged with others will leak information of geographical location or area of interest.

One example of current research on secure distributed computations and data security is the Fabric programming language [67], which provides programmers with a high level abstraction of security and distributed details of a system. Fabric programmers access objects in a uniform way, regardless of location of the object. Fabric supports both data and function shipping allowing for a secure flow of information within a system. Notably a node may only execute fabric code if signed by a trusted source. Fabric also includes a peer-to-peer dissemination layer for replication of objects across a system.

13.4 Social Networks

The CSWF system provides additional value to a forecast if multiple geographically close neighbors share their forecasts. This resembles a limited social network where you only share with a select group of people. This is similar to the decentralized peer-to-peer social network sketched by Cutillo et. al.[25], where a small group of trusted peers participate in sharing personal data of all, and other peers can find this information though lookups and traversing possibly several layers of peers. This scheme requires that an identity is established and trust is individually evaluated between peers based on this identity.

The CSWF system does not require a priori establishment of identities and trust-relationships, as trust can be computed when data are shared. The CSWF system does not require establishment of any peer-to-peer network prior to node discovery, even if this would speed up node discovery and would allow for pushing forecasts between users instead of pulling forecasts on demand.

13.5 On-Demand Weather Forecasting

One example of a semi-operational on-demand weather forecasting service is the `sarweather.com` system. A prototype version of this system was also presented at the same session in the conference as Paper 1. The current work [1] of this group is geared towards the wind energy sector. The work includes improvements to forecasts using observational nudging from UAVs. This service uses the WRF model, but is geared towards larger areas and is using compute clusters or the Amazon Elastic cloud computing, EC2, service for computing the model. A typical region is somewhat larger than used in this dissertation.

On-demand weather forecasting is elsewhere taken as on-demand access to pre-computed high-resolution forecasts. One example of this is the "Weather on Demand", WOD[1], discussed in Section 7.6.

A slightly different approach used for the specific problem of computing wind over terrain for wild fire estimation in WindNinja [42]. This is a tool suitable for end users like firefighters. WindNinja computes wind along the ground in complex terrain with limited meteorological background input and needs only a terrain description and a description of vegetation type and moisture. This model is also specifically created to be able to be executed on personal computers. WindNinja is specific only focused on ground level winds.

13.6 Distributed Weather Forecasting

Most of the research found when searching for terms like *Distributed* and *Weather Forecasting* involves the distribution of execution of computational tasks and numerical models on several clusters, like the work of Yalec et. al. [103] . This works distributes the execution of large-scale hydrological models on a GRID infrastructure. Or the work of Fernández-Quiruelas et. al. [34] which studied the use of the community atmospheric model, CAM, on a GRID platform.

Except for the papers included in this dissertation, to our knowledge no research on systems that explicitly distributes the execution of a numerical atmospheric model on geographically separated computers controlled by individual users has been published.

13.7 Collaborative Weather Forecasting

The CAM model [23, 22, 80] from the previous section is an example of the extensive cooperation that is typical within the meteorological research community.

The focus on collaboration within geophysics and computer science has been related to environmental monitoring and sensor networks, and issues concerning collaboration within a sensor network, and collaboration between multiple sensor networks. This work is mostly related to handling observational data, pre-processing these, and sharing data between interested parties. One example is the work of Liu [68] that uses a cloud based GIS approach to sharing observations from sensor networks.

[1] `http://www.stormgeo.com/weather-on-demand-wod`

Other approaches have studied collaboration using social media, like Demirbas [26]. This work introduces three application domains, using smart phones. 1) Participatory Sensing, 2) Crowd-Sourcing and 3) Social Collaboration. The paper suggests that Twitter[2] provides a suitable platform for publish-subscribe infrastructure for smartphone applications. Both applications are transmitting current observations in the background, and have created applications where the users in an area are actively asked to describe the current weather.

One area of collaboration that has been extensively researched is within Collaborative Intrusion Detection. One example can be found in Berger et. al. [9] where a lightweight framework for sharing notions of suspiciousness among network operators are presented. The basic idea for such systems is that when aggregating signals from many ISPs, malicious traffic can be identified very early and handled before major disruption happens. In Berger [9] there is also an element of reciprocity as the system requires the ISP to submit a report before gaining the right to receive notification about other reports concerning the same suspected malicious host.

13.8 Trust

The basis for computing trust in this dissertation is using the local forecast and the forecasts from peers. Statistical measurements of differences between the local and the exchanged forecasts can be calculated and a level of trust can be established. Having this kind of a priori knowledge for trust establishment validates the *data*, not the user. Similar work has been done on other ad hoc networks. Raya et. al. [89] develops several models for evaluating trust in Vehicular Ad hoc Networks (VANETs) based on the reported statements, as a contrast to trust based on the identity of the reporter. A major difference between the VANET system and the system described in this dissertation is that the reported data from peers can be locally reproduced and the trust computed with very high degree of certainty.

13.9 Visualization of Forecasts using a User-Centric View

Adding current weather information to images has been done in several systems. Gliet et. al [47] is one example using web cameras and public weather information.

In a recent web article from DailyTech[3] a nice overview of various augmented reality technologies is presented. Citing research at MIT [56] the focus is on heads-up displays in cars. These displays may also be able to incorporate warnings about up-coming weather, like strong winds and fog.

The application presented in this dissertation incorporates the option of producing a new numerical weather forecast based on the current location, and then visualizing the result using an augmented reality type view. To our knowledge no research on the

[2] `twitter.com`

[3] `http://www.dailytech.com/MIT+Seeks+to+Enable+the+Ultimate+HeadsUp+Display+for+War+and+Peace/article34181.htm`

combination of using the device for both localization, on-demand weather forecasting and AR type display is found in the literature.

14

Conclusion

In this conclusion, the four problem statements are presented as a basis for concluding on the developed systems and the research findings.

14.1 Research Statement 1

Increase resolution, decrease area, and decrease duration of a numerical atmospherical model using available background data with the purpose of finding the highest resolution being meteorologically sound.

The work presented in this dissertation has demonstrated that a modern numerical model can be executed with 1 km resolution for a limited area providing forecasts with details not available in other forecasts today. A forecast at this resolution is very difficult to evaluate because of the very few observation points within the limited area. No publicly available high-resolution observational network exists in Norway. The WRF models has been extensively validated for use in complex terrain and with very high resolutions, see for example [64, 100, 101]. Validation of the WRF model presented in other studies gives confidence in assessing that these forecasts are meteorological sound.

The computation time is controlled by at least three variables, the size of the forecast area, the resolution of the forecast and the period, or length of, the forecast, if the mesh size is kept constant. The workload is proportional to the resolution because higher resolution demands shorter time-steps in the model in order to keep the model numerically stable.

14.2 Research Statement 2

Combine multiple local very high-resolution forecasts with the purpose of determining the meteorological benefits.

The benefits of combining multiple local very high-resolution forecasts was presented when introducing the symbiotic forecasts, and originates from the basic difference between forecast when producing forecasts with slightly different center locations. The location of each forecast is decided by the individual user requesting the forecast, possibly from the GPS location provided by a mobile device. During pre-processing of the background data, this location is used for specifying the mesh location for the forecast, and the high-resolution background parameters like terrain height, are interpolated and possibly smoothed to create a geophysical background for each specific mesh.

Varying the forecast mesh locations is in effect very similar to randomly varying some of the background data used for computing numerical weather forecasts. Berner et. al. [10] show that random variation of one parameter in the background data will provide significant improvements to probabilistic forecasting. The method used in this dissertation is similar to randomly modifying all surface parameters at the same time and provides sound forecasts and information usable for error or uncertainty estimation.

14.3 Research Statement 3

Build a system for computing on-demand very high-resolution interactively fast meteorologically sound forecasts with the purpose of identifying the characteristics and performance of an architecture, design and implementation done for a commodity platform.

The architecture for the on-demand very high-resolution interactive forecasts separates the visualization of forecasts from forecast production. The system for producing forecasts is built according to the *sc-to-pc* principle. By only producing the wanted forecasts for a limited region, with a limited duration and on-demand, the forecasts can be produced on commodity hardware. And extending the trade-off between size of mesh, duration of forecast and resolution, a very high-resolution forecast can be produced. The forecast is also meteorologically sound.

The separation of visualization of forecasts from production allow for the visualization to be tailored to any device. The developed systems can visualize the forecasts on devices spanning several orders of magnitude in size, from mobile phones to a display wall.

The design specifications on using a HTTP REST style of communication between the forecast production and forecast presentation provide a technological and implementation independence on both sides. The implemented systems produce forecasts in an acceptable time, although the numerical atmospheric model is not fully optimized for the actual hardware present. This points to further improvements being possible.

Experiments show that the typical execution times range from 2 minutes for a 10 km forecast to 15 minutes for a 1 km forecast. The available hardware using a quad-core Intel processor with hyper-threading enabled were confirmed having the shortest execution time when using the model with 8 processes.

14.4 Research Statement 4

Build a system for combining local weather forecasts with the purpose of identifying the characteristics and performance of an architecture, design and implementation done for a commodity platform.

A system for combining local forecasts collected in a collaborative setting between local CSWF systems within a small region has been developed. The system is in accordance with the principle of *short traveled computations and data*.

The architecture places the collaboration as a function of the forecasts production. The collaborative symbiotic weather forecast systems does not require prior-knowledge of peers. All peers are found on-demand, and the system does not need to scale because of the limited number of peers needed. Information on persistent groups are not needed. Trust in peers is computable using only local available information and any incoming forecast can be recreated locally for absolute verification if needed. A user may not actually need to be aware of the collaborative system in the daily use of the forecasts.

The design separates the production of forecasts into several parts that will operate in parallel. A front-end handles requests from the forecasts visualization and presentation applications while collaboration, local forecast production and forecasts amalgamation into symbiotic forecasts can execute simultaneously. The design allows for the individual parts to be executed on individual computers, if needed. The implementation use C and the Mongoose system, with additional scripts i Python and Bash. This makes the CSWF system very portable. The implication of the modest hardware requirements is also that most of the CSWF system could be executed on tiny Raspberry-PI[1], while only the execution of the WRF model implementation and the visualization tools needs a common desktop computer.

The system also shows that exchanging forecasts can be implemented in a scalable way. The system is scalable because it does not need to scale as even in the worst case scenario, no peers to collaborate with, leaves the user with a fully functional local forecast. The system also scales because of the few neighbors needed for a successful collaboration and forecast exchange.

The system presents a way of finding peers that works well in laboratory conditions, but will require correctly configured firewalls and internet connections for peers on the Internet to connect when using this method. While the simple way of finding peers finds peers works in a laboratory setting, no experiments have been conducted outside the laboratory. The simplicity of the method also demonstrates that within

[1] `http://www.raspberrypi.org/`

these constrains, a fully decentralized exchange of forecast is viable in the sense that it will lead to forecasts that are more useful.

15

Future Work

Locality based computing is probably possible within several domains. For example can a social network be executed in a community based distributed way, like Diaspora[1]. One question is if the efficiency of this approach, on home computers or on virtualized cloud-based services, is good and secure enough for this to be practical and interesting to people.

This work is based on the 3.1.1 version of the WRF numerical atmospheric model. Current research is also studying and building the next generation of models, the non-hydrostatical icosahedral model, NIM [65]. Using GPUs for computations seems to be a good candidate in order to increase the resolution of the models. As long as desktop computers keep up with nodes on the supercomputers, systems like the CSWF will continue to make an alternative for on-demand collaborative numerical weather forecasting.

One option for the weather services is to use the coarse resolution models as guidance for where and when to execute high-resolution models. This would allow the weather services to utilize limited computing resources and still have high-resolution weather forecasts for critical regions. A related effort would be to utilize modern UAVs or other sensors for targeted observations in critical areas. This kind of effort has already been demonstrated to be efficient for enhancing wind forecasts for the energy sector, see Augustsson et. al. [1].

Ongoing research like the work of Saito et. al. [93], into ultra high-resolution models with spatial resolution in the order of a few tenths of meters's will also require research into observational systems for validating forecasts at these scales. Very few observational networks available today are capable of such resolutions. Research into wireless sensor networks often cite weather forecasting as one potential use case, and may provide useful data for validating very- and ultra high-resolution models.

[1] https://joindiaspora.com/

The CSWF system provides an API for accessing and controlling forecast from a web browser or from applications. The CSWF system prototype is composed of several distinct applications also using an API for internal communication. This has been an efficient division of labor, but may not be the optimal configuration for an operational system. One example to follow might be introduction of an API Orchestration Layer (OL) [2]. The key element is to divide the potential user devices from the underlying data structures and production idiosyncrasies. The abstractions introduced in Section 6 will provide a good starting point with this re-write.

A new technological use which was presented during the later parts of writing this dissertation is BitTorrent Sync[3], presented in WIRED [99]. This is a proprietary file synchronization tool where data is only stored on participating nodes, not on any centralized servers. A shared folder is simply a folder with at shared secret (value) 20 bytes long.

For use in the CSWF system, the users would need an agreement on this shared value, and this could for example be automatically generated using the geographical location as a key. An automatic generated value would obviously not be a secret, and anyone could share the forecasts without contributing to the system. Using boxes with the size of around 50 x 50 km, the resolution needed would be around 0.5°.

A shared *secret* could be formed using the following pattern:

```
DD.Dddd.dYYYYMMDDHH

where
DD.D    Latitude
ddd.d   Longitude
YYYY    Year
MM      Month
DD      Day
HH      Hour

example

69.5018.02014021500

for

69.5 North 18.0 East February 15th 2014 at 00 UTC.
```

The convention could be that this secret covers the area within 69.5 - 70.0°North and 18.0 - 18.5°East. The date and hour could be either the validating start time for the forecast, or the date of the originating background meteorological data, as is usual in meteorological services.

Everyone wanting to share a forecast would store it in a folder using this secret. A user close to the borders would probably also need to retrieve forecasts in neighbor

[2] http://thenextweb.com/dd/2013/12/17/future-api-design-orchestration-layer

[3] http://getsync.com/

folders. Using the scheme above this would be in at most three other folders. The API for BitTorrent Sync allows storing encrypted data in these folders, so privacy could possibly be maintained by sharing with only trusted peers even if the folder and content is publicly visible.

BitTorrent Sync uses a range of technologies for locating peers. On a local network it will broadcast for peer discovery, it may also use known hosts, a DHT or trackers. It also allows for using a traffic relay for the difficult NAT and firewall situations.

If the background meteorological data has much better resolution than what is used in this work, the forecasting period could be made much shorter and the forecast would still resolve high-resolution flow patterns. Fides and Gruber [35] provided a method for downscaling background meteorological data that may serve as a good staring point. Depending on the computational complexity of the method, the total computational cost of the downscaling and model could be lower than the existing model-only cost. The WRF regional climate model version 3.1.1 has also been used for downscaling [54], but only with 30 km and 10 km resolutions, indicating that the models own downscaling properties may be sufficient.

Using newer hardware and utilizing GPUs leads to faster computations exemplified by Mielikainen et. al. [76]. The current trend in super computers makes heterogeneous systems the standard. Desktop computers can be expected to follow the same path.

Research into porting existing and newer numerical models to GPUs show promising results. Shimokawabe et. al. [95] reports on a 80-fold speedup after porting the complete model to a GPU platform compared to a single CPU. This moves the issue of sequential parts from the CPU to the GPU. Lee et. al. [66] found that CPUs and GPUs are much closer in performance (2.5 times speedup) when using optimized code on both sides. A 2.5 speedup is still a significant improvement for large computations.

One example of possible future improvements is using local observations and doing a full 3D or 4Dvar [105] analysis. Using observational nudging as described by Agustsson et. al. [1] is also possible.

One problem facing current improvement of multi-core CPUs is the so called "Dark silicon" problem were large parts of a CPU have to be under-utilized for power consumption and heath reasons, leading to a weak growth in computational power in future CPU generations. Having to move computations between different cores may create problems for effective utilization of registers and shared storage. One example is the study of Esmaeilzadeh et. al. [33] that shows that multicore scaling is power limited. One should of course always be cautious of predictions of the demise of the popular interpretation of Moore's law. New research may come, the work of Cherepov et. al. [21] reported by [14] may be one example; A class of materials called *multi-ferroics* could make future devices much more energy-efficient.

Bibliography

[1] Halfdan Agustsson, Olafur Rognvaldsson, and Logi Ragnarsson. High-resolution wind forecasts: On-demand and operational forecasting and observational nudging , December 2013. http://www.ewea.org/events/workshops/wp-content/uploads/2013/12/PO.0123_EWEA-Forecasting-Workshop-2013.pdf.

[2] Lorenzo Alvisi and Edmund L Wong. Reasoning with MAD distributed systems. In *CONCUR'13: Proceedings of the 24th international conference on Concurrency Theory*. Springer-Verlag, August 2013.

[3] Gene M Amdahl. Validity of the single processor approach to achieving large scale computing capabilities. *The spring joint computer conference*, pages 483–485, April 1967. http://www.ece.cmu.edu/~ece740/f11/lib/exe/fetch.php?media=wiki:r1_amdahl.pdf.

[4] S Andreev, A Pyattaev, K Johnsson, and O Galinina. Cellular traffic offloading onto network-assisted device-to-device connections. *Communications Magazine, IEEE*, 52(4):20–31, April 2014.

[5] Stephanos Androutsellis-Theotokis and Diomidis Spinellis. A survey of peer-to-peer content distribution technologies. *ACM Computing Surveys (CSUR)*, 36(4):335–371, December 2004.

[6] Otto J Anshus, John Markus Bjørndalen, Daniel Stødle, Lars Ailo Bongo, Tor-Magne Stien Hagen, Yong Liu, Bård Fjukstad, and Lars Tiede. Nine Years of the Tromsø Display Wall. In Chris Rooney, Alex Endert, Jean-Daniel Fekete, Kasper Hornbæk, and Chris North, editors, *POWERWALL International Workshop on Interactive, Ultra-High-Resolution Displays, part of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1–6, April 2013.

[7] arstechnica. Why European forecasters saw Sandy's path first, 2012. http://arstechnica.com/science/2012/12/why-european-forecasters-saw-sandys-path-first/.

[8] Donovan Artz and Yolanda Gil. A survey of trust in computer science and the Semantic Web. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(2):58–71, June 2007.

[9] A Berger, J Cesareo, and A D'Alconzo. Collaborative Network Defense with Minimum Disclosure. In *IEEE Globecom*, pages 1–6. IEEE, 2011.

[10] J Berner, S Y Ha, J P Hacker, A Fournier, and C Snyder. Model Uncertainty in a Mesoscale Ensemble Prediction System: Stochastic versus Multiphysics Representations. *Monthly Weather Review*, 139(6):1972–1995, June 2011.

[11] Jost Berthold, Jonas Bardino, and Brian Vinter. A Principled Approach to Grid Middleware. In Y Xiang, editor, *Lecture Notes in Computer Science*, pages 409–418, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[12] N Bhushan, Junyi Li, D Malladi, R Gilmore, D Brenner, A Damnjanovic, R Sukhavasi, C Patel, and S Geirhofer. Network densification: the dominant theme for wireless evolution into 5G. *Communications Magazine, IEEE*, 52(2):82–89, 2014. `http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6736747`.

[13] V Bjerknes. The problem of weather forecasting as a problem in mechanics and physics. *Meteorol Z*, 1904.

[14] Kurzweil Blog, 2014. `http://www.kurzweilai.net/`.

[15] J D Blower, A L Gemmell, G H Griffiths, K Haines, A Santokhee, and X Yang. A Web Map Service implementation for the visualization of multidimensional gridded environmental data. *Environmental Modelling & Software*, 47:218–224, September 2013.

[16] J D Blower, K Haines, A Santokhee, and C L Liu. GODIVA2: interactive visualization of environmental data on the Web. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 367(1890):1035–1039, March 2009.

[17] David Carvalho, Alfredo Rocha, Moncho GÃ mez Gesteira, and Carlos Santos. A sensitivity study of the WRF model in wind simulation for an area of high wind energy. *Environmental Modelling & Software*, 33(0):23–34, 2012.

[18] Damià Castellà, Francesc Giné, Francesc Solsona, and Josep Lluis Lérida. Analyzing locality over a P2P computing architecture. *Journal of Network and Computer Applications*, 36(6):1610–1619, November 2013.

[19] J G Charney and A Eliassen. A Numerical Method for Predicting the Perturbations of the Middle Latitude Westerlies. *Tellus*, 1(2):38–54, 1949.

[20] J G Charney, R Fjörtoft, and J Neumann. Numerical Integration of the Barotropic Vorticity Equation. *Tellus*, 2(4):237–254, 1950.

[21] Sergiy Cherepov, Pedram Khalili Amiri, Juan G Alzate, Kin Wong, Mark Lewis, Pramey Upadhyaya, Jayshankar Nath, Mingqiang Bao, Alexandre Bur, Tao Wu, Gregory P Carman, Alexander Khitun, and Kang L Wang. Electric-field-induced spin wave generation using multiferroic magnetoelectric cells. *Applied Physics Letters*, 104(8):082403, February 2014.

[22] W D Collins, P J Rasch, and B A Boville. Description of the NCAR community atmosphere model (CAM 3.0). *NCAR Tech Note ...*, 2004.

[23] William D Collins, Cecilia M Bitz, Maurice L Blackmon, Gordon B Bonan, Christopher S Bretherton, James A Carton, Ping Chang, Scott C Doney, James J Hack, Thomas B Henderson, Jeffrey T Kiehl, William G Large, Daniel S McKenna, Benjamin D Santer, and Richard D Smith. The Community Climate System Model Version 3 (CCSM3). *dx.doi.org*, February 2010.

[24] NIST Computer Security Division CSD. The NIST Definition of Cloud Computing. *pecial Publication 800-145*, pages 1–7, April 2012.

[25] L A Cutillo, R Molva, and T Strufe. Privacy preserving social networking through decentralization. *2009 Sixth International Conference on Wireless On-Demand Network Systems and Services (WONS)*, pages 145–152, 2009.

[26] M Demirbas, M A Bayir, and C G Akcora. Crowd-sourced sensing and collaboration using twitter. *World of Wireless ...*, 2010.

[27] Peter J Denning. The locality principle. *Communications of the ACM*, 48(7):19, July 2005.

[28] Peter J Denning. The locality principle. Technical report, Naval Postgraduate School, Monterey, California, June 2008. `http://denninginstitute.com/pjd/PUBS/ENC/locality08.pdf`.

[29] H A Duran-Limon, L A Silva-Bañuelos, V H Tellez-Valdez, N Parlavantzas, and Ming Zhao. Using Lightweight Virtual Machines to Run High Performance Computing Applications: The Case of the Weather Research and Forecasting Model. *2011 IEEE 4th International Conference on Utility and Cloud Computing (UCC)*, pages 146–153, 2011.

[30] ECMWF. ECMWF Newsletter, October 2012. Last accessed 2013.02.26 `http://www.ecmwf.int/publications/newsletters/pdf/133.pdf`.

[31] ECMWF. ECMWF Newsletter, 2013. Last accessed 2013.02.26 `http://ecmwf.int/publications/newsletters/pdf/137.pdf`.

[32] K Elevant. Why Share Weather? Motivational Model for "Share Weather" Online Communities and Three Empirical Studies. *2013 46th Hawaii International Conference on System Sciences (HICSS)*, pages 781–790, 2013.

[33] H Esmaeilzadeh, E Blem, R St Amant, K Sankaralingam, and D Burger. Dark silicon and the end of multicore scaling. *Computer Architecture (ISCA), 2011 38th Annual International Symposium on*, pages 365–376, 2011.

[34] V Fernández-Quiruelas, J Fernández, A S Cofiño, L Fita, and J M Gutiérrez. Benefits and requirements of grid computing for climate applications. An example with the community atmospheric model. *Environmental Modelling & Software*, 26(9):1057–1069, September 2011.

[35] J Fiddes and S Gruber. TopoSCALE: deriving surface fluxes from gridded climate data. *Geoscientific Model Development*, 2013.

[36] Roy T Fielding. *Architectural styles and the design of network-based software architectures*. PhD thesis, University of California, Irvine, UNIVERSITY OF CALIFORNIA, IRVINE, 2000. PhD Thesis.

[37] B Fjukstad, John Markus Bjørndalen, and Otto Anshus. Uncertainty Estimation and Visualization of Wind in Weather Forecasts. In *IN PRESS. IVAPP. Lisbon, Portugal*, pages 1–8, 2014.

[38] Bård Fjukstad, John Markus Bjørndalen, and Otto Anshus. SAFE-WEATHER: User Specified, Rapidly Produced, On-Demand, Very High-Resolution Numerical Weather Forecasts. In *Arctic Frontiers*, pages 1–1, Tromsø, January 2012. Department of Computer Science, Faculty of Science and Technology, University of Tromsø, Norway.

[39] Bard Fjukstad, John Markus Bjorndalen, and Otto Anshus. Accurate weather forecasting through locality based collaborative computing. In *Collaborative Computing: Networking, Applications and Worksharing (Collaboratecom), 2013 9th International Conference Conference on*, pages 571–578, 2013.

[40] Bård Fjukstad, John Markus Bjørndalen, and Otto Anshus. Embarrassingly Distributed Computing for Symbiotic Weather Forecasts. In *Proceedings of the International Conference on Computational Science, ICCS*, pages 1217–1225, June 2013.

[41] Bård Fjukstad, Tor-Magne Stien Hagen, Daniel Stødle, Hoai Phuong Ha, John Markus Bjørndalen, and Otto Anshus. Interactive Weather Simulation and Visualization on a Display Wall with Many-Core Compute Nodes. In *State of the Art in Scientific and Parallel Computeing (Para 2010) - Programme Schedule and Short Abstracts (Updated version published in LNCS 7133, February 2012)*, pages 142–151. University of Iceland, 2012.

[42] Jason Forthofer, Kyle Shannon, and Bret Butler. Simulating Diurnally Driven Slope Winds with WindNinja. In *Proceedings of 8th Symposium on Fire and Forest Meteorological Society*, 2009.

[43] I Foster, C Kesselman, and S Tuecke. The anatomy of the grid: Enabling scalable virtual organizations. *International Journal of High Performance Computing Applications*, 15(3):200–222, 2001. `http://hpc.sagepub.com/content/15/3/200`.

[44] Ian Foster and Carl Kesselman. Globus: A Metacomputing Infrastructure Toolkit. *International Journal of Supercomputer Applications*, 11:115–128, 1996.

[45] Paul Francis. Yoid: Extending the Internet Multicast Architecture, April 2000. Tech report `http://www.icir.org/yoid/docs/index.html`.

[46] Brian J Gaudet, Aijun Deng, David R Stauffer, Nelson L Seaman, and Astrid Suarez. Nested Realistic Daily WRF-LES Simulations Using Eddy-Seeded Lateral Boundary Conditions . `http://www.mmm.ucar.edu/wrf/users/workshops/WS2013/posters/p83.pdf`.

[47] Jana Gliet, Antonio Krüger, Otto Klemm, and Johannes Schöning. Image geomashups: the example of an augmented reality weather camera. In *AVI '08: Proceedings of the working conference on Advanced visual interfaces*, page 287, New York, New York, USA, May 2008. ACM.

[48] G Gramelsberger. Conceiving Meteorology as the exact science of the atmosphere: Vilhelm Bjerknes's paper of 1904 as a milestone. *Meteorologische Zeitschrift*, 2009.

[49] A Gupta, D Agrawal, and A El Abbadi. Approximate Range Selection Queries in Peer-to-Peer Systems. *CIDR*, 2003.

[50] John L Gustafson. Reevaluating Amdahl's law. *Communications of the ACM*, 31(5):532–533, May 1988.

[51] Tor-Magne Stien Hagen. *Interactive Visualization on High-Resolution Tiled Display Walls with Network Accessible Compute- and Display-Resources*. PhD thesis, University of Tromsø, September 2011.

[52] Tor-Magne Stien Hagen, Daniel Stødle, and Otto J Anshus. On-demand high-performance visualization of spatial data on high-resolution tiled display walls. In *IVAPP 2010 - International Conference on Information Visualization Theory and Applications*, pages 112–119. Department of Computer Science, Faculty of Science and Technology, University of Tromsø, Norway, Conference on Information Visualization Theory, 2010.

[53] Mike Hawkins. New cray xc30 hpcf, system description and project status, 25th meeting of the computing representatives meeting, 2013. `http://nwmstest.ecmwf.int/newsevents/meetings/computing_representatives/presentations/pdf/ECMWF/Hawkins_Cray.pdf`.

[54] U Heikkilä, A Sandvik, and A Sorteberg. Dynamical downscaling of ERA-40 in complex terrain using the WRF regional climate model. *Climate Dynamics*, 37(7-8):1551–1564, October 2010.

[55] M D Hill and M R Marty. Amdahl's Law in the Multicore Era. *Computer*, 41(7):33–38, 2008.

[56] Chia Wei Hsu, Bo Zhen, Wenjun Qiu, Ofer Shapira, Brendan G DeLacy, John D Joannopoulos, and Marin Soljačić. Transparent displays enabled by resonant nanoparticle scattering. *Nature Communications*, 5, January 2014.

[57] Qi Huang, Ken Birman, Robbert van Renesse, Wyatt Lloyd, Sanjeev Kumar, and Harry C. Li. An analysis of facebook photo caching. In *Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles*, SOSP '13, pages 167–181, New York, NY, USA, 2013. ACM.

[58] Info-Electronics Systems Inc. Collaborative Weather Forecasting System (CWFS), 2013. `http://www.info-electronics.com/products/cwfs.html`.

[59] Norwegian Meteorological Institute. Metcoop presentation. Internal webcast, December 2013.

[60] Norwegian Meteorological Institute. yr.no statistics. Internal web use statistics, September 2013.

[61] A J Jakeman, R A Letcher, and J P Norton. Ten iterative steps in development and evaluation of environmental models. *Environmental Modelling & Software*, 21(5):602–614, May 2006.

[62] Peter Johnsen, Mark Straka, Melvyn Shapiro, Alan Norton, and Thomas Galarneau. Petascale WRF simulation of hurricane Sandy. *The International Conference for High Performance Computing, Networking, Storage and Analysis. SC'13*, pages 63–7, November 2013.

[63] A Jolma and J Norton. Methods of uncertainty treatment in environmental models. *Environmental Modelling & Software*, 2005.

[64] John S Kain, Michael C Coniglio, James Correia, Adam J Clark, Patrick T Marsh, Conrad L Ziegler, Valliappa Lakshmanan, Stuart D Miller Jr., Scott R Dembek, Steven J Weiss, Fanyou Kong, Ming Xue, Ryan A Sobash, Andrew R Dean, Israel L Jirak, and Christopher J Melick. A Feasibility Study for Probabilistic Convection Initiation Forecasts Based on Explicit Numerical Guidance. *Bulletin of the American Meteorological Society*, 94(8):1213–1225, August 2013.

[65] Kirk L Kroeker. Modeling chaotic storms. *Communications of the ACM*, 54(11):15–17, November 2011. `http://kroeker.net/published/modeling-chaotic-storms.pdf`.

[66] Victor W Lee, Changkyu Kim, Jatin Chhugani, Michael Deisher, Daehyun Kim, Anthony D Nguyen, Nadathur Satish, Mikhail Smelyanskiy, Srinivas Chennupaty, Per Hammarlund, Ronak Singhal, Pradeep Dubey, Victor W Lee, Changkyu Kim, Jatin Chhugani, Michael Deisher, Daehyun Kim, Anthony D Nguyen, Nadathur Satish, Mikhail Smelyanskiy, Srinivas Chennupaty, Per Hammarlund, Ronak Singhal, and Pradeep Dubey. Debunking the 100X GPU vs. CPU myth: an evaluation of throughput computing on CPU and GPU. *ACM SIGARCH Computer Architecture News*, 38(3):451–460, June 2010.

[67] Jed Liu, Michael D George, K Vikram, Xin Qi, Lucas Waye, and Andrew C Myers. Fabric: a platform for secure distributed computation and storage. In *SOSP '09: Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*. ACM Request Permissions, October 2009.

[68] Jen-Chu Liu and Kuo-Yu Chuang. WASP: An innovative sensor observation service with web-/GIS-based architecture. *2009 17th International Conference on Geoinformatics*, pages 1–6, 2009.

[69] Yong Liu, John Markus Bjorndalen, and Otto J Anshus. Using Multi-threading and Server Update Pushing to Improve the Performance of VNC for a Wall-Sized Tiled Display Wall. *Electronic Healthcare - Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 306–321, 2009.

[70] Jeff Chun Fung Lo, Zong Liang Yang, and Roger A Pielke. Assessment of three dynamical climate downscaling methods using the Weather Research and Forecasting (WRF) model. *Journal of Geophysical Research: Atmospheres (1984–2012)*, 113(D9), 2008.

[71] Make Magazine. Augmented reality gear simulates weather, 2011. `http://makezine.com/2011/03/30/augmented-reality-gear-simulates-weather/`.

[72] J Mandel, J D Beezley, and A K Kochanski. Coupled atmosphere-wildland fire modeling with WRF 3.3 and SFIRE 2011. *Geoscientific Model Development*, 4(3):591–610, 2011.

[73] E.A. Martinsen, A. Foss, L. Bergholt, A. Christoffersen, H. Korsmo, and J. Schulze. Diana: a public domain application for weather analysis, diagnostics and products., September 2005. `http://diana.met.no/ref/0279_Martinsen.pdf`.

[74] J Michalakes, J Dudhia, D Gill, and T Henderson. The weather research and forecast model: software architecture and performance. *11th ECMWF Workshop on the Use of High Performance Computing In Meteorology*, 2004. `http://www.wrf-model.org/wrfadmin/docs/ecmwf_2004.pdf`.

[75] John G Michalakes, Michael McAtee, and Jerry Wegiel. Software Infrastructure for the Weater Research and Forecast Model. *Presentations UGC_2002*, pages 1–13, 2002.

[76] Jarno Mielikainen, Bormin Huang, Hung-Lung Allen Huang, and Mitchell D Goldberg. Improved GPU/CUDA Based Parallel Weather and Research Forecast (WRF) Single Moment 5-Class (WSM5) Cloud Microphysics. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 5(4):1256–1265, 2012.

[77] G E Moore. Cramming more components onto integrated circuits. *Electronics*, pages 114–117, 1965.

[78] Sape Mullender, editor. *Distributed Systems*. ACM, New York, NY, USA, 1989.

[79] A H Murphy. What is a good forecast? An essay on the nature of goodness in weather forecasting. *Weather and forecasting*, 1993.

[80] R B Neale, C C Chen, and A Gettelman. Description of the NCAR community atmosphere model (CAM 5.0). *NCAR/TN-486+ STR*, 2010.

[81] NOTUR. VILJE-the new supercomputer at NTNU. *META*, 4(4):4–5, 2011.

[82] Open Geospatial Consortium Inc. OGC KML. Open Geospatial Consortium Inc., April 2008.

[83] M Tamer Özsu and Patrick Valduriez. *Principles of Distributed Database Systems*. Springer Science+Business Media LLC, 2011.

[84] In Kyu Park, N Singhal, Man Hee Lee, Sungdae Cho, and C W Kim. Design and Performance Evaluation of Image Processing Algorithms on GPUs. *Ieee Transactions on Parallel and Distributed Systems*, 22(1):91–104, January 2011.

[85] Anders Persson. Early operational Numerical Weather Prediction outside the USA: an historical introduction: Part II: Twenty countries around the world. *Meteorological Applications*, 12(3):269–289, 2005.

[86] Norman A Phillips. The Start of Numerical Weather Prediction in the United States. In *Symposium on the the Anniversary of Operational Numerical Weather Prediction*. U.S. Department of Commerce, National Oceanic and Atmospheric Administration, National Environmental Satellite, Data, and Information Service, National Oceanographic Data Center, September 1999.

[87] R Pielke Sr. *Mesoscale Meteorological Modeling*. Academic Press, 3rd edition, 2013.

[88] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. A scalable content-addressable network. In *ACM SIGCOMM Computer Communication Review*, pages 161–172. ACM, October 2001.

[89] M Raya, P Papadimitratos, V D Gligor, and J P Hubaux. On Data-Centric Trust Establishment in Ephemeral Ad Hoc Networks. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, 2008.

[90] G K Rutledge, J Alpert, and W Ebuisaki. NOMADS. *BAMS*, March 2006. `http://nomads.ncep.noaa.gov/txt_descriptions/marRutledge-1.pdf`.

[91] Rich Rutonno. Mesoscale Modeling at High (but not LES) Resolution, 2013. `http://video.ucar.edu/mms/mmm/r_rotunno.mp4`.

[92] N Sadashiv and S M D Kumar. Cluster, grid and cloud computing: A detailed comparison. In *Computer Science & Education (ICCSE), 2011 6th International Conference on*, pages 477–482, 2011.

[93] K Saito, T Tsuyuki, H Seko, F Kimura, T Tokioka, T Kuroda, L Duc, K Ito, T Oizumi, G Chen, J Ito, and the Spire Field 3 Mesoscale Nwp group. Super high-resolution mesoscale weather prediction. *Journal of Physics: Conference Series*, 454(1):012073, August 2013.

[94] M Satyanarayanan, John H Howard, David A Nichols, Robert N Sidebotham, Alfred Z Spector, and Michael J West. The ITC distributed file system: principles and design. *ACM SIGOPS Operating Systems Review*, 19(5):35–50, December 1985.

[95] Takashi Shimokawabe, Takayuki Aoki, Chiashi Muroi, Junichi Ishida, Kohei Kawano, Toshio Endo, Akira Nukada, Naoya Maruyama, and Satoshi Matsuoka. An 80-Fold Speedup, 15.0 TFlops Full GPU Acceleration of Non-Hydrostatic Weather Model ASUCA Production Code. In *SC '10: Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE Computer Society, November 2010.

[96] William C Skamarock, Joseph B Klemp, Jimy Dudhia, David O Gill, Dale M Barker, Michael G Duda, Xiang-Yu Huang, Wei Wang, and Jordan G Powers. A Description of the Advanced Research WRF Version 3. Technical Report NCAR/TN–475+STR, NCAR, June 2008. `http://www.mmm.ucar.edu/wrf/users/docs/arw_v3.pdf`.

[97] K Sripanidkulchai, B Maggs, and Hui Zhang. Efficient content location using interest-based locality in peer-to-peer systems. *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies*, 3:2166–2176 vol.3, 2003.

[98] Charles Talbot, Elie Bou-Zeid, and Jim Smith. Nested Mesoscale Large-Eddy Simulations with WRF: Performance in Real Test Cases. *Journal of Hydrometeorology*, 13(5):1421–1441, October 2012.

[99] Ryan Tate. The Next Big Thing You Missed: A Would-Be Dropbox Meant to Thwart the NSA. *WIRED*, February 2014. `http://www.wired.com/business/2014/02/bittorrent-sync/`.

[100] Stanley B Trier, Christopher A Davis, David A Ahijevych, and Kevin W Manning. Use of the Parcel Buoyancy Minimum (B min) to Diagnose Simulated Thermodynamic Destabilization. Part I: Methodology and Case Studies of MCS Initiation Environments. *Monthly Weather Review*, November 2013.

[101] Stanley B Trier, Christopher A Davis, David A Ahijevych, and Kevin W Manning. Use of the Parcel Buoyancy Minimum (B min) to Diagnose Simulated Thermodynamic Destabilization. Part II: Composite Analysis of Mature MCS Environments. *Monthly Weather Review*, November 2013.

[102] P Victor, C Cornelis, and M De Cock. *Trust networks for recommender systems*. Atlantis Press, 2011.

[103] S Yalew, A van Griensven, N Ray, L Kokoszkiewicz, and G D Betrie. Distributed computation of large scale SWAT models on the Grid. *Environmental Modelling & Software*, 41:223–230, March 2013.

[104] yr.no. yr.no 5 years: 2007 - 2012. Webpage, December 2013. In Norwegian, `http://om.yr.no/info/fakta/historie/`.

[105] Xin Zhang and Xiang-Yu Huang. WRF Four-dimensional variational data assimilation system Tutorial for V3.5 , July 2013. `http://www.mmm.ucar.edu/wrf/users/wrfda/Tutorials/2013_July/docs/WRFDA_4dvar.pdf`.

A

Online Available Files

Files referenced in the dissertation available at `http://hpds.cs.uit.no/people/bfj002/index.html` are described in Table A.1, Table A.2 and Table A.3.

Table A.1 *Published papers*

Filename	File type	Content	File Size
para2010_ article.pdf	PDF	Paper 1	361 KB
iccs_ 2013.pdf	PDF	Paper 2	4,3 MB
collaboratecom_ 2013.pdf	PDF	Paper 3	732 KB
ivapp_ article.pdf	PDF	Paper 4	2,9 MB
POWERWALL-Anshus.pdf	PDF	Additional Paper 1	518 KB
Poster_ standing.pdf	PDF	Additional Paper 2	5,1 MB

PDF files are all viewable using either software supplied with the operating system or a reader from Adobe. Files are available at `http://hpds.cs.uit.no/people/bfj002/index.html`.

Table A.2 *Presented videos*

Filename	File type	Content	File Size
Para2010-WallWeather.m4v	MPEG-4 file	Video presented at PARA-2010	25 MB

MPEG-4 files are viewable using VLC `http://www.videolan.org/vlc/` or other video applications. Files are available at `http://hpds.cs.uit.no/people/bfj002/index.html`.

Table A.3 *Output files from developed prototypes and the WRF model*

Filename	File type	Content	File Size
bulk_ statistics.txt	Text	Statistics for each file in a set of 28 CSWF forecasts	6,6 KB
wrfout_ amaglamated.nc	NetCDF	Amalgamated forecast from a set of 28 CSWF forecasts	480 KB
amalgamated_ stat.txt	Text	Statistics for each mesh point in the amalgamated file	1,0 MB
wrf_ wind.kml	KML	KML file for use in the AR application of Google Earth	887 KB
wrfout_d01_01000_2014-01-18_61100748.nc	NetCDF	Default WRF model output from a single CSWF forecast	20 MB

NetCDF file are viewable using IDV, DIANA og ParaView applications. KML files are viewable in Google Earth. Files are available at `http://hpds.cs.uit.no/people/bfj002/index.html`.

Part II

Collection of papers

16

Papers

16.1 Paper 1

16.2 Paper 2

16.3 Paper 3

16.4 Paper 4

16.5 Additional paper 1

16.6 Additional paper 2