Procedia Computer Science

Volume 29, 2014, Pages 1535–1545

ICCS 2014. 14th International Conference on Computational Science

Wind field uncertainty in forest fire propagation prediction*

Gemma Sanjuan[1], Carlos Brun[1], Tomàs Margalef[1], and Ana Cortés[1]

Computer Architecture and Operating Systems department, Universitat Autònoma de Barcelona, Cerdanyola del Vallès, Spain
email:gemma.sanjuan@caos.uab.es, carlos.brun@caos.uab.es, tomas.margalef@uab.es, ana.cortes@uab.es

Abstract

Forests fires are a significant problem especially in countries of the Mediterranean basin. To fight against these disasters, the accurate prediction of forest fire propagation is a crucial issue. Propagation models try to describe the future evolution of the forest fire given an initial scenario and certain input parameters. However, the data describing the real fire scenario are usually subject to high levels of uncertainty. Moreover, there are input parameters that present spatial and temporal variation that make the prediction less accurate. Therefore, to overcome such uncertainty and improve accuracy it is necessary to couple complementary models such as the case of the wind field model. Such models use the meteorological forecasted wind to provide the wind direction and speed depending on the topography of the terrain. We use WindNinja as wind field simulator. This simulator takes a lot of time to deliver the predictions and it is a serious problem because fire propagation prediction must accomplish strict time constraints. To solve this problem, we propose map partitioning and solving independently for each one of the parts. However, the model has problems concerning boundary effects which is an additional source of uncertainty. Therefore, it is necessary to apply certain degree of overlapping among parts to reach a stable wind field without inconsistencies and a minimum uncertainty.

Keywords: Forest fire, Wind field, Simulation, Map Partitioning, Overlapping, Uncertainties, Prediction

1 Introduction

Forest fires are a significant problem around the world, especially in places with hot and dry summer seasons such as Mediterranean countries, California or Australia. To fight against these hazards and use the available resources in the best possible way, it is mandatory to have an accurate prediction of their evolution beforehand. So, propagation models have been developed to determine the expected propagation of a forest fire[10][2][9]. Such propagation models require several input parameters representing the scenario where the fire is taking place

*This work has been supported by Ministerio de Ciencia e Innovacion under contract TIN-2011-28689-C02-01

Selection and peer-review under responsibility of the Scientific Programme Committee of ICCS 2014

doi:10.1016/j.procs.2014.05.139

to produce the predictions of the propagation. Some parameters are well known, but in many cases the information concerning the values of the input parameter is obtained or estimated from indirect measurements. Such indirect estimations imply an uncertainty degree concerning the values of the parameters that are translated to uncertainty in forest fire propagation prediction.

Typically, a given terrain is defined as a mesh of cells (raster space) with a certain resolution. In the case of forest fire spread simulations, each cell has associated the corresponding values of every input parameter, such as slope, vegetation type, moisture contents, wind speed and direction, and so on. So, the fire behavior is defined by the values of the set of parameters corresponding to each cell. As it has been commented, the uncertainty in the values of the parameters can induce either poor or wrong prediction results that should be minimized as much as possible to reproduce the real fire behavior. An approach to overcome this problem is to apply parameter calibration strategies by means of stochastic frameworks. A Two-Stages prediction strategy was defined in [1]. In a first stage the values of the input parameters are calibrated taking into account the observed behaviour of the fire between two time instants. The calibrated values are then used to predict the behaviour of the fire in the consequent/next time interval. This approach is very successful to predict the behaviour of prescribed burnings or short fires that take few minutes and burn some few hectares.

However, when the fires last longer and covers larger terrains the values of the parameters cannot be considered constant and uniform. Wind speed and direction are critical parameters because they significantly affect fire propagation. The wind parameters (speed and direction) provided by a global weather forecast model or measured at a meteorological station in some particular point are just single values that do not represent the wind at each point of the terrain, since the wind is modified by the topography of the terrain and has a different value at every point of the terrain. To estimate the wind speed and direction at each point of the terrain it is necessary to apply a wind field model that determines those values at each point taking into account the terrain topography. In this way, the wind field model is coupled to the forest fire propagation model in order to improve the accuracy of propagation predictions [4][3]. However, coupling a wind field model with the forest fire propagation implies a significant increase in the execution time that is not affordable since the propagation prediction has strict real time constraints in order to be operational. Therefore, it is necessary to apply computational methods to reduce execution time of both, the forest fire propagation model and the wind field model. In this paper, the parallelisation of the wind field model calculation is considered.

In this paper the coupling of wind field model to reduce wind uncertainty and the parallelization of such model are considered. So, the paper is organised as follows. Section 2 describes the coupling of wind field model and forest fire propagation model. Section 3 presents the main limitations of WindNinja as wind field simulator and introduces the map partitioning approach to overcome the execution time and memory limitations of WindNinja. Section 4 summarises the results of the experimental study carried out. Finally, section 5 shows the main conclusion of this work.

2 Coupling wind field and forest fire propagation models

Two of the parameters that suffer from temporal and spatial variation are wind speed and direction. Such variation introduces a high degree of uncertainty in the results of forest fire propagation prediction. As it is well known the wind is not constant and certain weather forecast model must be introduced to predict the future evolution of the wind. On the other hand, the meteorological wind is modified by the topography of the terrain under consideration. So, in each point of the terrain there is a different value of the wind in speed and direction. So,

it is necessary to introduce a wind field model that evaluates the wind speed and direction for each point of the terrain. So, the values of wind speed and direction are calculated by a wind field simulator for each cell of the terrain.

WindNinja [7][6][8] is a wind field simulator that provides an estimated wind direction and wind speed at each point of the terrain given a meteorological wind. This wind field simulator can be coupled to a forest fire simulator. FARSITE [5] is one of the most widely used forest fire propagation simulators in the forestry community. It has been designed to accept a wind field map as input data. WindNinja is a wind field model that was originally developed to be directly coupled to FARSITE, so, this is the reason why we decided to use WindNinja as wind field model. Figure 1 shows the coupling of WindNinja and FARSITE.

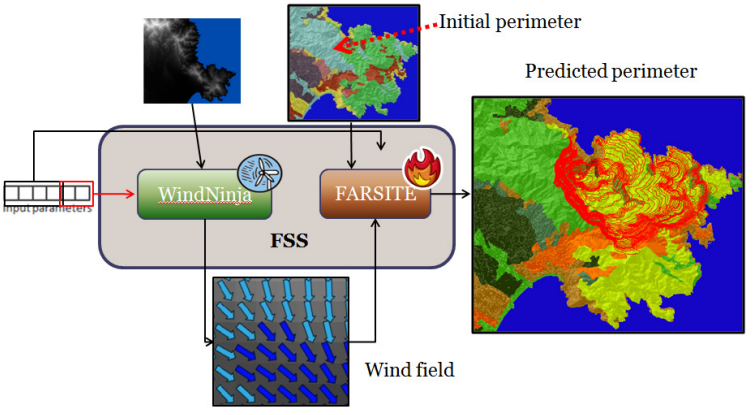

Figure 1: Coupling wind field and forest fire propagation models

WindNinja is a wind field simulator that calculates the effect of topography on wind flow. Unlike common weather forecasting models, WindNinja does not predict wind fields for future times, but computes the spatially varying wind field on the surface for one instant time. WindNinja requires as basic input parameters the elevation map of the underlying terrain (Digital Elevation Model -DEM- file), the meteorological wind speed and wind direction and the required output resolution. As output, WindNinja delivers wind speed and wind direction at each cell of the terrain at the specified resolution. In the experiments reported in this paper, the resolution delivered in the output wind field has been set to be the same resolution as the input elevation map. Using this one-to-one relationship, each map cell will have its own wind components. It is worth mentioning that the resolution of this output map has a direct impact in WindNinja execution time. Figure 2 shows the internal structure of WindNinja and the steps that it carry out to calculate the wind field.

The calculation of the wind field takes some time when the map has a considerable size (30x30 Km) and the resolution is high (30x30meters). This time penalizes the prediction of forest fire spread and may eventually make impractical the effective prediction of fire spread with wind field. Actually, calculating the wind field for such a map could take around 3200 seconds in a single node and such time is unaffordable for forest fire propagation prediction. But, execution time is not the only problem of WindNinja. Memory is another issue that significantly constrains the viability and performance of such model. It must be outlined that the data structures needed to calculate the wind field of a large map requires a large amount of memory that may not be available on a single node of a current system.

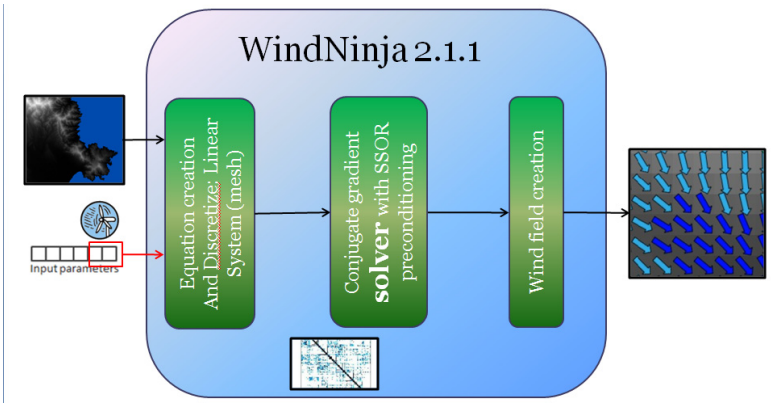

Figure 2: WindNinja System

The operational requirements of emergency services indicate that the total forest fire prediction time must be below 15 minutes, and the wind field calculation should last less than 2 minutes. So we must fix a boundary value of 100 seconds to calculate the wind field.

To reduce the computation time of the wind field a data partitioning method has been applied. In this method the wind field is calculated in parallel on each part of the map and then the wind fields of the different parts are joined to form the global wind field. Furthermore, by partitioning the terrain map, the data structures necessary to solve the wind field in each part are reduced significantly and can be stored in the memory of a single node in a current parallel system. Therefore, the existing nodes can perform computation in parallel with data that fit the capacity of the memory on each node.

3 WindNinja parallelisation using map partitioning

The initial approach that has been considered to overcome WindNinja limitations is to use map partitioning. In this way, a global map is partitioned in a certain number of square parts and the wind field is calculated simultaneously on each part. In this way the execution time should reduce significantly and the amount of memory to calculate the wind field corresponding to each part is also reduced.However, far from being an easy approach, this map partitioning scheme involves new issues that must be tackled. WindNinja is based on the equations that describe air flow variation in the atmosphere. Specifically, it is based on a mass conservation model initialized by boundary conditions. The function to minimize is constructed using the square of the difference between the adjusted and observed values as is shown in equation 1, where u, v, w are the velocity components in the x (positive to East), y (positive to North), and z (positive upward) directions, respectively. The initial values of velocity are u^0, v^0, w^0. Furthermore, $\lambda(x, y, z)$ is a Lagrange multiplier and α_1 is the Gauss precision moduli.

$$E(u,v,w,\lambda) = \int [(\alpha_1)^2(u-u^0)^2 - (\alpha_1)^2(v-v^0)^2 - (\alpha_1)^2(w-w^0)^2 + \lambda(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z})]dvol \quad (1)$$

This implies that terrain slope variation generates wind changes and, because of boundary conditions, the obtained results in regions close to the borders of the map will not be correct until the system is stabilized. Consequently, many external map cells have a non reliable value

and, therefore, a set of cells around the evaluated map must be dismissed as a final result. When the main map is partitioned into parts, this problem is extrapolated to all parts and a direct combination of output wind fields results in an aggregation of boundary errors introducing additional uncertainty in wind values.

To solve this problem, it is necessary to include a certain degree of overlapping among the map parts. So, there is a margin from the beginning of the part and the part cells itself. The overall wind field aggregation is obtained by discarding the calculated overlapping margin of each part. The inclusion of an overlapping to each part increases the execution time, but the variation and uncertainty in the wind field is significantly reduced. An example of this partitioning and overlapping approach can be seen in figure 3 where the result of applying overlapping in a AxB parts partitioning is shown. So, we propose a map partitioning with overlapping scheme for wind field evaluation as follows:

1. partition the input DEM map into X parts with a given overlapping,
2. run in parallel as many executions of the wind field model as parts have been generated at the partitioning process and,
3. combine the outputs of the X parts discarding the overlapping cells to obtain the global map.

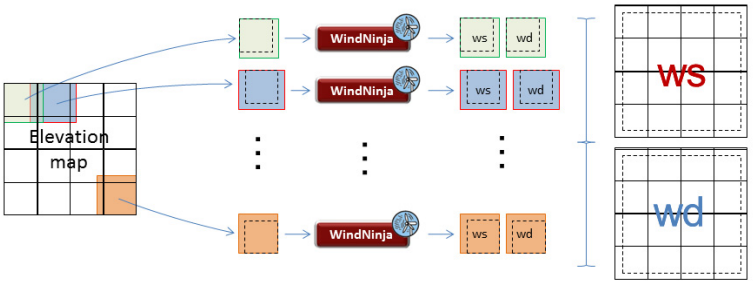

Figure 3: AxB partitioning with overlapping

Finally, the resulting wind field map, once the map partitioning scheme has been applied, has the same dimensions as the original one. In the following section this methodology is studied using different map partitioning, many overlap values and different wind speeds and wind directions. The execution times and the error obtained will be compared to the values obtained when executing WindNinja with a non-partitioned map.

To determine the adequate map partitioning and part overlapping it is necessary to reach a trade-off between memory requirements, execution time and wind field differences.

The memory size is defined by the 3 vectors that are created to form the mesh. The expression to estimate the amount of memory in MBytes provided by WindNinja developers is the following one (2):

$$Memory(bits) = 163840 + 122880NRows + 122880NCols + 92160NRowsNCols \quad (2)$$

This equation shows that the amount of memory is related to the number of columns of the map (NColumn) and the number of rows of the map (Nrows). So, it can be concluded that the time depends on the map size.

From the results obtained using maps of different size it is determined that the relationship between execution time and the number of cells can be approximated to a straight applying linear regression. Figure 4 shows the execution time for an IBM cluster depending on the number of cells of the map.

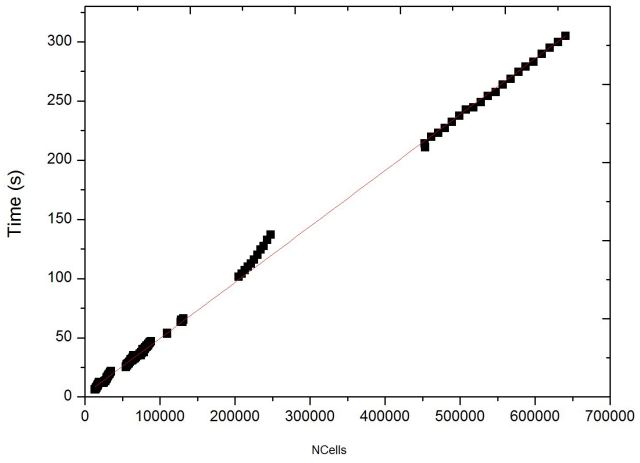

Figure 4: Execution time depending on the number of cells

The expression obtained from linear regression is the following one (3):

$$t = 4,73 * 10^{-4} NCells + 2,31 \tag{3}$$

Since it has been determined that the maximum time is 100s, from expression (3), it can be deduced that the size of each map part should be around 160000. The results of the wind direction and speed of a cell depend on the cells that are around. Therefore, the squarer the part the fewer exposed cells were found in the partition, as the square is the geometric figure that has the minimum perimeter. So, the parts must be of a maximum size of $400x400$, where the overlapping for each part is included within these $400x400$ cells.

With this part of $400x400$ different possibilities can be considered as shown in figure 5. It is possible to consider a part map of $350x350$ cells with an overlapping of 25 cells for each side ($4x4$), a part map of $300x300$ with an overlapping of 50 cells for each side ($5x5$) and a part map of $250x250$ with an overlapping of 75 cells for each side ($6x6$).

Since all the parts have an actual size of $400x400$ the execution time and memory requirements are approximately the same for each case. However, since in case ($4x4$) the overlapping in smaller the difference in the wind field could be larger. In case ($6x6$) with the largest overlapping the difference in wind field should be the smaller one, but in this case, the effective map is so small that WindNinja could not reach stabilisation.

So, it is necessary to carry out an experimental study to validate the memory requirements model, the execution time model and determine the effect of overlapping and part size on wind field difference. This study is carried out in next section.

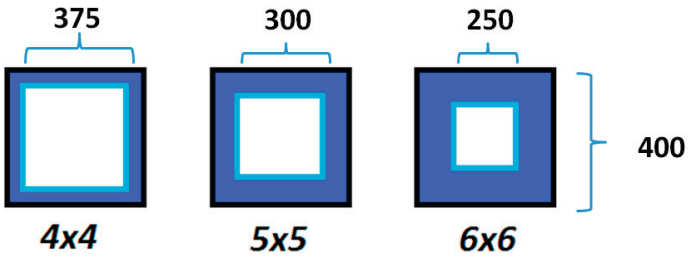

Figure 5: Map parts with different overlapping

4 Experimental results

To determine the most suitable part division and overlapping size for a given map, different kind of partitions and different overlapping have been tested. In particular, the partitions used have been $4x4$, $5x5$ and $6x6$, where AxB means that the original map has been partitioned into A divisions in the horizontal axis and B divisions in the vertical direction. For each kind of partition respectively, different overlapping has been tested ranging from 25, 50 and 75. So, for all the cases all the parts are $400x400$ cells, but the effective wind field calculated for each case are $350x350$, $300x300$ and $250x250$, respectively. Besides, wind components must be taken into account, so, 3 wind directions (45, 180 and 270 degrees) and 3 wind speeds (5, 10 and 15 mph) have been considered to cover a wide range of combinations.

The terrains used for this study are located in different points of Spain. They are a region of interest because they present very different slope from each other. The raster maps used are composed by 1501 rows and 1501 columns with 30m resolution per cell. That means that the map has a dimensions of 45km x 45km. The only point to be considered is that the studied maps do not include cliffs because so high slopes as those of cliffs provoke inconsistencies in the wind field generated by WindNinja.

The experiments have been executed in two different multi-core platforms:

1. IBM cluster x3650 composed of 32 Dual-Core Intel(R) Xeon(R) CPU 5150 2.66GHz nodes and 12 GB Fully Buffered DIMM 667 MHz per node.

2. DELL cluster based on Poweredge C6145 with a total of 8 CPUs with 16 cores and 128 GB of memory.

Every combination has been simulated and compared with the non-partitioned global wind field simulation in terms of similarity of the resulting wind field map and the time incurred in the map partitioning process. To evaluate the similarity between both wind fields three measures has been considered: the Root-Mean-Square Error (RMSE), the maximum difference between both maps and the number of cells with a difference higher to 1mph for the case of speed and 5° for the case of wind direction.

The Root-Mean-Square Error (RMSE) is shown in equation 4. More precisely, for each map cell (N), the value of wind speed in that particular cell i obtained when no partition is applied to the input DEM map ($NP(ws)_i$) is compared to the speed obtained in the same cell when applying the map partitioning strategy with a given partition scheme ($SC_{AxB}(ws)_i$). The same

procedure is also applied to wind direction just changing the corresponding terms of equation 4 to wd.

$$RMSE_{AxB}(ws) = \sqrt{\frac{\sum_{i=0}^{N} (NP(ws)_i - SC_{AxB}(ws)_i)^2}{N}} \quad (4)$$

Before discussing the results obtained, some general considerations must be taken into account for wind speed and wind direction. WindNinja implements a system of equations for each part of the map. That means, for the particular case of the wind speed, that the solution error propagation will not be too much significant if the general wind speed is low (5 mph) because the obtained wind speed at each map cell will have slight variations. However, for higher wind speeds such as 10 and 15 mph, the values obtained in the wind field will have a higher variation rank. Under this condition, the WindNinja solver propagates higher error in the solution provided. This feature is independent on the underlying elevation map.

On the other hand, analyzing what happens with wind directions, it has been determined that supplementary angles (0°-180° and 90°-270°) have identical errors for a given speed-direction-partition configuration. The equations that define WindNinja system try to find saddle points. These points are those which have the maximum elevation in a given direction and the minimum in the perpendicular direction. More precisely, a saddle point of a function is a point at which the first derivative is zero, while the sign of the second derivative (curvature) depends on the direction to be calculated.

Table 1 shows the RMSE for speed and direction, number of cells with difference greater than 1mph in the case of the speed and difference greater than 5° in the case of the angle and the maximum difference in speed and direction. These values are shown for $4x4$, $5x5$ and $6x6$ partitions considering different wind speeds and a direction of meteorological wind of 45° because it was the most unfavorable condition in the studied maps.

Speed (mph)	Partitioning	$RSME_{sp}$ (mph)	Speed ≥ 1 (mph)	Max_{sp} (mph)	$RSME_{ang}$ (°)	Angle ≥ 5 (°)	Max_{ang} (°)
5	6x6	0.191	8183	3.76	1.703	11062	20
5	5x5	0.187	7818	3.76	1.571	7216	26
5	4x4	0.188	7531	3.76	1.701	9719	24
10	6x6	0.383	60920	7.50	1.703	11062	20
10	5x5	0.373	49565	7.50	1.571	7216	26
10	4x4	0.375	49182	7.50	1.701	9719	24
15	6x6	0.574	156479	11.3	1.703	11062	20
15	5x5	0.559	140826	11.3	1.571	7216	26
15	4x4	0.563	140196	11.3	1.701	9719	24

Table 1: Similarity indexes for different partitioning

The results show that partitioning the map in $5x5$ parts of $400x400$ cells with an overlapping of 50 cells per side provide a reasonable wind speed and direction difference. As it has been mentioned above with this type of partitioning, each process solves an effective part of a map of $300x300$ cells.

From the experiments carried out it can be observed that as the wind speed is increased, the error increases proportionally to the speed. Comparing the different results it can also be observed that the $5x5$ partitioning is the one that presents the lowest RMSE in speed

and direction. The same results have been obtained for different maps. Therefore, it can be concluded that the best partitioning is $5x5$ with 50 rows and columns overlapping.

It is also interesting to represent graphically differences between the wind field obtained with the global map and the 5x5 partitioning map in the worst case (wind speed of 15mph). Figure 6 shows in dark brown the points with a difference lager than 1mph. It can can be seen that the selected boundary conditions do not affect the resolution of the map. The differences are concentrated at those points where there is a significant change in slope. This is due to the fact that the partitioning map systems of equations is different from the global map system of equations and there are always some differences that propagates through the map. However, the differences are very slight and only affect on the more extreme conditions.

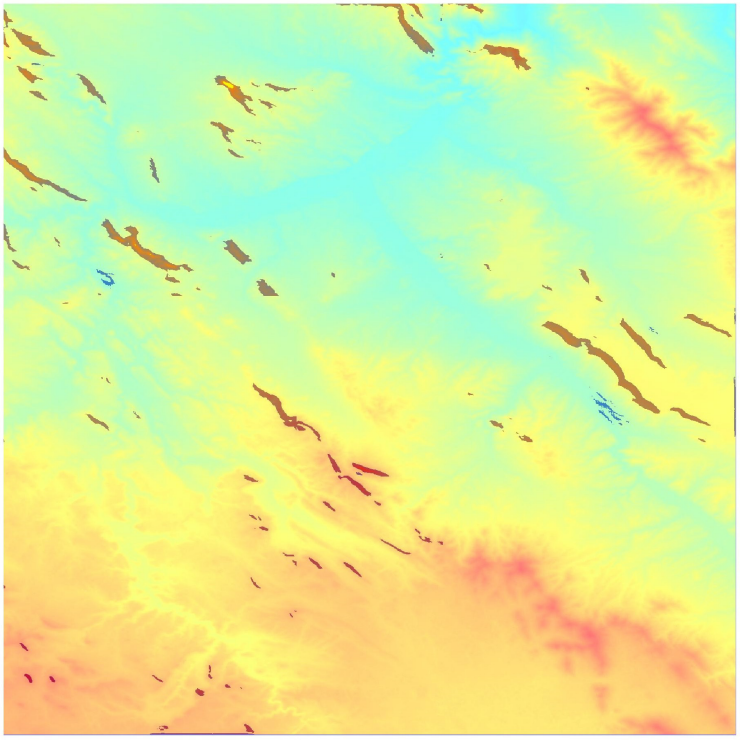

Figure 6: Points with wind speed difference higher than 1mph

Concerning execution time and memory requirements the experiments show that for map parts of equal size, the execution time and the memory used is almost exactly the same. Table 2 summarizes the results for map parts of $400x400$ with different overlapping. The execution time is for all the cases under 100 seconds and the memory required is under 2GB. These results perfectly fit the expectations. So, the map partitioning approach allows to reduce the memory requirements and the execution time of WindNinja wind field simulator by exploiting the features of current multicore architectures.

Partitioning	Overlap	t (s)	Memory (Gb)
6x6	75	92	1.9
5x5	50	90	1.9
4x4	25	95	2.0

Table 2: Execution time and memory requirements for different partitioning

5 Conclusion

Wind speed and direction are parameters that affect forest fire propagation dramatically. So, an accurate estimation of such parameters is crucial to predict the fire propagation precisely. However, meteorological wind is modified by terrain topography and a different value of wind speed and direction is effectively found on each point of the terrain. To overcome such uncertainty in wind parameters it is necessary to introduce a wind field simulator and couple this simulator to forest fire propagation simulator. A wind field simulator, such as WindNinja, presents to main drawbacks to become operational coupled to FARSITE forest fire propagation simulator when the size of the map under consideration increases significantly: It takes to long to compute the wind filed and it requires to much memory. So, a map partitioning strategy has been developed to compute partial wind field maps that can be aggregated afterwards. Each map part can be computed in parallel and the amount of memory required is available in a single node. In this way the wind field calculation becomes feasible to be integrated to an operational forest fire propagation prediction framework. However, it must be considered that wind field calculation includes some border effects and the straight map partition could introduce uncertainty in the wind values in the points close to the border of each resulting part. Therefore, a certain degree of overlapping has been introduced to minimize such uncertainty in wind field parameters. The results show that the wind field obtained by map partitioning is very close to the wind field obtained with a global map, that implies that the forest fire propagation prediction will not be affected by map partitioning.

References

[1] Baker Abdalhaq, Ana Cortés, Tomàs Margalef, and Emilio Luque. Enhancing wildland fire prediction on cluster systems applying evolutionary optimization techniques. *Future Generation Computer Systems*, 21(1):61–67, 2005.

[2] F.A. Albini, Intermountain Forest, Utah) Range Experiment Station (Ogden, and United States. Forest Service. *Estimating wildfire behavior and effects*. General technical report INT. Dept. of Agriculture, Forest Service, Intermountain Forest and Range Experiment Station, 1976.

[3] Carlos Brun, Tomàs Artées, Tomàs Margalef, and Ana Cortées. Coupling wind dynamics into a dddas forest fire propagation prediction system. *Procedia Computer Science*, 9:1110–1118, 2012.

[4] Mónica Denham, Ana Cortés, Tomàs Margalef, and Emilio Luque. Applying a dynamic data driven genetic algorithm to improve forest fire spread prediction. In Marian Bubak, Geert van Albada, Jack Dongarra, and Peter Sloot, editors, *Computational Science – ICCS 2008*, volume 5103 of *Lecture Notes in Computer Science*, pages 36–45. Springer Berlin / Heidelberg, 2008. 10.1007/978-3-540-69389-5_6.

[5] M. A. Finney. *FARSITE, Fire Area Simulator–model development and evaluation*. Res. Pap. RMRS-RP-4, Ogden, UT: U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station. 1998.

[6] J. M. Forthofer, K. Shannon, and B. W. Butler. Simulating diurnally driven slope winds with windninja. In *8th Symposium on Fire and Forest Meteorological Society*, 2009.

[7] Jason Forthofer, Kyle Shannon, and Bret Butler. 4.4 simulating diurnally driven slope winds with windninja. 2009.

[8] J.M. Forthofer, K. Shannon, and B. W. Butler. Initialization of high resolution surface wind simulations using nws gridded data. In *Proceedings of 3rd Fire Behavior and Fuels Conference; 25-29 October*, 2010.

[9] Josep Jorba, Tomas Margalef, Emilio Luque, J Campos da Silva Andre, and Domingos Xavier Viegas. Parallel approach to the simulation of forest fire propagation. In *Proceedings of the 13 Internationales Symposium" Informatik fur den Umweltshutz" der Gesellshaft Fur Informatik*, pages 69–81, 1999.

[10] R.C. Rothermel. *How to predict the spread and intensity of forest and range fires.* Intermountain Forest and Range Experiment Station Ogden, 1983.