WIP: Towards Cloud-based Wildland Fire Simulation Service

Xiaolin Hu
Department of Computer Science
Georgia State University
Atlanta, GA, USA
xhu@gsu.edu

Mingxi Yan
Department of Computer Science
Georgia State University
Atlanta, GA, USA
myan6@student.gsu.edu

Tony Derado
Department of Computer Science
Georgia State University
Atlanta, GA, USA
tderado1@student.gsu.edu

Wei Zhao
Department of Computer Science
Georgia State University
Atlanta, GA, USA
wzhao7@student.gsu.edu

Bernard Zeigler
RTSync Corp.
Chandler, AZ, USA
zeigler@rtsync.com

Doohwan Kim
RTSync Corp.
Chandler, AZ, USA
dhkim@rtsync.com

Chungman Seo
RTSync Corp.
Chandler, AZ, USA
cseo@rtsync.com

***Abstract*—Wildland fire simulation is a useful tool for studying wildland fires and for developing new technologies to help wildland fire management. This paper presents an effort of developing wildland fire simulation as a service that is accessible to broader users. The developed simulation service supports fire spread simulation, fire suppression simulation, and prescribed fire simulation with dynamic ignitions, and is accessible through a graphic user interface and an application programming interface (API). We present the underlying DEVS-FIRE simulation model, the design of the cloud-based wildland fire simulation service, and some preliminary results.**

Keywords—wildfire, wildland fire simulation, simulation as a service, DEVS-FIRE, application programming interface

I. Introduction

Wildfires are natural disasters that damage natural resources, destroy homes and properties, and threaten human lives and safety. The occurrences of catastrophic wildfires have increased in recent years in the United States and around the world [1]. For example, the 2023 Hawaii wildfires on the island of Maui burned 17,000+ acres, destroyed 2,207 structures, and caused 101 deaths. Besides wildfires, there is also a need to study prescribed fires (also known as prescribed burns or controlled burns), which refer to the controlled application of fire by a team of fire experts under specified conditions to restore health to ecosystems that depend on fire [2]. Prescribed fires can serve multiple purposes, including removing hazardous fuels to reduce wildfire risk, and helping farming and grazing by replenishing soil and protecting prairies from invasive overgrowth. Prescribed fires are often differentiated from wildfires as they are planned fires ignited intentionally to achieve specific management results while wildfires generally refer to unplanned fires. This paper refers to both wildfire and prescribed fires as wildland fires.

The growing risk of catastrophic wildfires prompts increasing demands for new technologies, tools, and strategies for wildland fire management. Simulation of wildland fire is considered a key technology to help modernize wildland firefighting [3]. Simulations of wildfires can be used to analyze/predict wildfire spread to support decision makings of fire suppression and evacuation. Simulations of prescribed fires can provide useful information for planning prescribed burn operations. Wildland fire simulation is also a valuable tool for learning and understanding fire behavior and for training firefighters and fire operators. Furthermore, it is an important technology to support development of new technologies in wildland fire management. For example, when developing Unmanned Aircraft Systems (UAS) for monitoring wildfires, wildfire spread simulation allows researchers/developers to test UAS' path planning algorithms in a simulated environment in a cost-effective and safe manner.

Despite the many usages of wildland fire simulation, there is a lack of wildland fire simulation service on the market. Existing wildfire simulation tools and models that are publicly available can be grouped into two categories: stand-alone software and open-source project. Examples of stand-alone software includes FlamMap [4], which integrates the wildfire spread simulation software FARSITE [5] and is a fire analysis tool that can simulate fire behavior characteristics (spread rate, flame length, fireline intensity, etc.) and fire growth and spread. Another example is BehavePlus [6], which supports simulation of wildfire behavior under various terrain, fuels, and weather conditions. These stand-alone software tools are desktop applications, which are not easy to be integrated into other projects. For example, it would be difficult to program these tools to support a research project that needs to find optimal ways of creating fire breaks to minimize fire spread risk. Besides stand-alone software, there also exist several open source projects for wildfire spread simulations. For example, Fire Dynamics Simulator [7] is an open source project that supports large-eddy simulation (LES) for low-speed flows, with an emphasis on smoke and heat transport from fires. The ForeFire [8] is an open source fire spread model for large scale fire simulation that allows for integration of new model formulations. These open source projects can be integrated into other projects. Nevertheless, they require extensive knowledge and programming skills in order to use their code. Both the stand-alone software tools and open source projects have

This research is supported in part by the US National Science Foundation (NSF) under award number 2306603.

another major limitation: they require users to download and install the software/code on their local computers. This can be a significant barrier for users due to the computer hardware/software requirement and runtime environment constraints. Currently, there is a lack of robust wildland fire simulation service that is easily accessible to broader users.

This paper presents an effort of developing a cloud-based service-oriented solution for providing wildland fire simulation service to broader users. The proposed simulation service is based on the DEVS-FIRE simulation model [9, 10, 11, 12]. It supports cloud-based simulation that covers wildfire spread simulation, wildfire suppression simulation, and prescribed fire simulation with dynamic ignitions. The developed simulation service is accessible through two types of interfaces: 1) a Graphic User Interface (GUI) that allows general users to use the service through a map-based web application; and 2) an Application Programming Interface (API) that allows developers and researchers to integrate wildland fire simulation into their own applications. The cloud-based simulation service will be implemented using the microservice architecture that allows different instances of simulation runs to be set up in containers. This paper presents the DEVS-FIRE model that supports the simulation service and the design of the cloud-based simulation service, as well as some preliminary results.

II. The DEVS-FIRE Model

A cloud-based wildland fire simulation service needs the support of an underlying wildland fire simulation model. In this work, we develop the wildland fire simulation service based on the DEVS-FIRE model [9, 10, 11, 12] that has been developed in previous research. DEVS-FIRE is a discrete event wildland fire simulation model built on the Discrete Event System Specification (DEVS) formalism [13]. It has several sub-models to support different types of wildland fire simulations, including fire spread simulation, fire suppression simulation, and prescribed fire simulation with dynamic ignitions. The fire spread model [9, 10] is the core of DEVS-FIRE, which uses a cellular space to model a wildland area and employs Rothermel's Behave model [14] to compute the fire rate of spread. Built on top of the fire spread model, the fire suppression model supports fire suppression simulation with different tactic, e.g., direct attack, parallel attack, and indirect attack [11]. The prescribed fire simulation model supports prescribed fire simulation with different ignition techniques such as backfire, head fire, spot head fire, and ring fire [12]. Fig. 1 shows an overview of the DEVS-FIRE model, where the components supporting fire suppression simulation are colored in blue; the components supporting prescribed fire simulation are colored in red; and the rest of the components are for fire spread simulation.

In DEVS-FIRE, the fire area is modeled as a two-dimensional cell space, which is divided into rectangular cells whose dimensions depend on the resolution of the GIS fuel and terrain data. Each cell has a coordinate denoting its location in the cell space, and uses fuel and terrain data corresponding to its location. Cells are coupled with their neighbors according to the Moore neighborhood (except for the boundary cells), in which a central cell has eight surrounding neighbors. All cells are coupled to a weather model to receive weather data (wind speed and wind direction) that can change over time. In DEVS-FIRE, the fire spread is modeled as a propagation process as burning cells ignite their unburned neighbors. A cell, once ignited, calculates its rate of spread and spread direction using Rothermel's fire behavior model based on its fuel, slope, aspect, and weather data. The rate of spread is then decomposed into eight directions corresponding to its eight neighbor cells based on an elliptical shape, which is computed by the mid flame wind speed and the fire spread rate. A screenshot of a fire spread simulation scenario is given in Fig. 2(a). In the figure, red cells are burning cells; black cells are burned cells; and cells in other colors represent unburned cells with different fuel types.

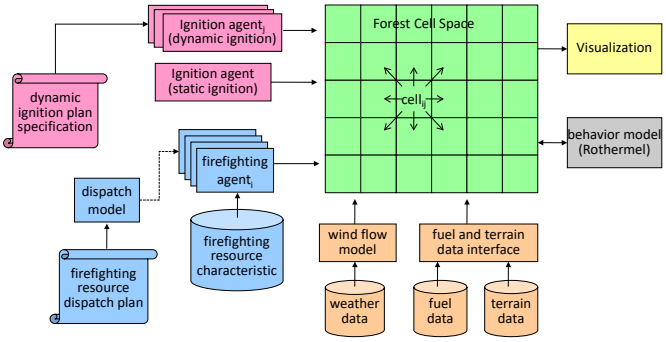
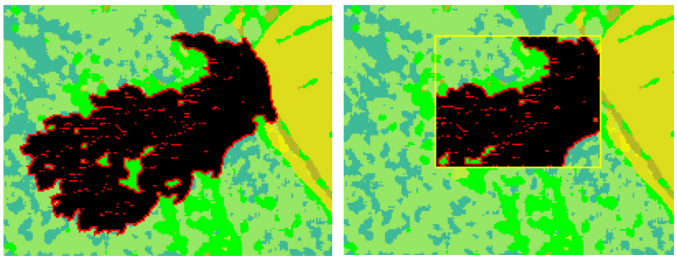


Fig. 1. Overview of the DEVS-FIRE model

DEVS-FIRE was validated by comparing with other wildland fire simulation tools [15, 10] and was used to simulate historical wildfires [16]. Based on the DEVS-FIRE model, data assimilation algorithms have been developed to assimilate real-time observation data collected from active wildfires to support real-time simulation-based predictions of fire spread [17]. A post-frontal combustion heat model was also developed to model the heat released from a burning wildfire [18], which is then used to support coupled fire-atmosphere modeling of wildland fire spread [16]. These works demonstrate the validity, robustness, and the various utilities of the fire spread model of DEVS-FIRE.



(a) Fire spread simulation (b) Fire suppression simulation

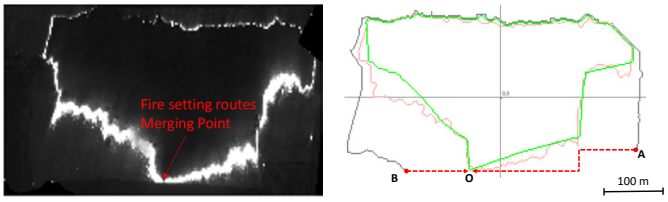
Fig. 2. Wildfire spread and suppression simulations using the DEVS-FIRE model

The fire suppression model [11, 10] of DEVS-FIRE is built on top of the fire spread model. An agent-based modeling approach is used to model firefighting resources such as bulldozers and firefighters. Fire suppression is modeled as a process for firefighting agents to construct suppression firelines to contain a burning fire. Three firefighting tactics [19] are modeled, including: 1) direct attack that constructs fireline directly on the flaming fire front; 2) parallel (indirect) attack that constructs fireline parallel to, but at a safe distance (offset) away

from, the fire front; and 3) indirect attack that constructs fireline according to a predetermined route. Each tactic can be further configured using multiple teams and different team settings such as deployment time, production rate, and suppression route. The fire suppression model of DEVS-FIRE has been used to evaluate firefighting resource deployment plans and to show the impact of different firefighting tactics on suppression results [20]. Fig. 2(b) illustrates a scenario of fire suppression simulation using indirect attack for the same fire shown in Fig. 2(a). In this scenario, a team of firefighters (modeled by a firefighting agent) builds a fireline (depicted in yellow) in a rectangle shape. The production rate of the firefighting agent is large enough in this scenario so that it fully contains the spreading fire.

The prescribed fire simulation model [12] is also built on top of the fire spread model of DEVS-FIRE. A unique characteristic of prescribed fires is that they are ignited dynamically by fire setting teams while fires are spreading. This compares to wildfires whose growth is mainly driven by the spread of fire fronts. DEVS-FIRE models all the six basic ignition techniques summarized in [21] that guide how a prescribed fire may be ignited, including head fire, backfire, strip head fire, spot head fire, flank fire, and center & ring fire. These ignition techniques capture the different patterns regarding how to ignite a prescribed fire. For example, backfire is lighted on the downwind side and often used with other ignition techniques to create a safe line to prevent the fire from jumping or spotting across the line. On the other hand, head fire generally produces the largest area burned per unit of time and is often used in clear cut areas where wide firelines have been established. Besides the patterns of ignition, other setting such as number of ignition teams, teams' ignition speeds, start and end locations and timing of different ignition segments are also modeled. An ignition plan specification is developed to systematically capture all the ignition information, which is then carried out by fire-setting agents to dynamically ignite a fire.

Fig. 3 shows a prescribed fire simulation using DEVS-FIRE for a real prescribed fire event [22]. Fig. 3(a) shows the thermal image of the prescribed fire. Fig. 3(b) shows the prescribed fire simulation using two dynamic ignition routes (red lines AO and BO) carried out by two teams of fire operators. In the figure, the gray line is the initial fire perimeter line in the beginning of the simulation; the green line is the simulated fire perimeter line; the pink line is the actual fire perimeter extracted from the thermal image shown in Fig. 3(a). As can be seen, the simulated fire perimeter matches well with the actual fire perimeter.



(a) Thermal image of a prescribed fire (b) Prescribed fire simulation

Fig. 3. Prescribed fire simulation using the DEVS-FIRE model

To summarize, the DEVS-FIRE model is a robust wildland fire simulation model supporting fire spread simulation, fire suppression simulation, and prescribed fire simulation with dynamic ignitions. It has been used to simulate historical wildfires and prescribed fires. This work uses the DEVS-FIRE model to develop cloud-based wildland fire simulation service for broader users.

III. Cloud-based Service-Oriented Wildland Fire Simulation

The cloud-based service-oriented wildland fire simulation will be developed following a client-server architecture, as shown in Fig. 4. The server side includes the three sub-models of DEVS-FIRE for fire spread simulation, fire suppression simulation, and prescribed fire simulation. These models use a shared raster database hosting fuel, terrain, and weather data. A microservice architecture is used to implement and deploy the DEVS-FIRE-based simulations to provide simulation services. The client side runs user applications using the simulation service from the server side. We are developing two types of interface for accessing the simulation services: a Graphic User Interface (GUI) and an Application Programming Interface (API). The GUI allows users to access the simulation services through a map-based web application. The API allows developers and researchers to integrate the simulation services into their projects by programming using the API. Below we describe the design of each component in detail.

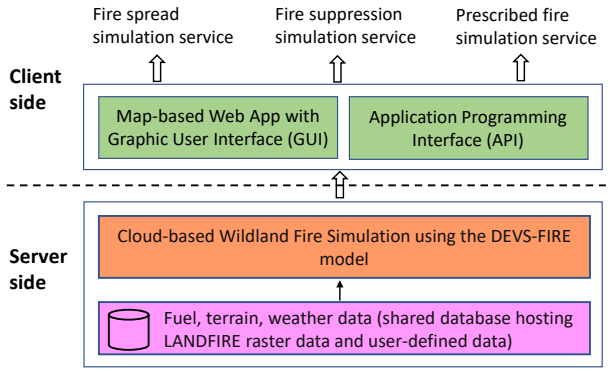


Fig. 4. Overall system architecture

A. Fuel, Terrain, and Weather Data

An essential component for wildland fire simulation is the data that characterize the vegetation, terrain, and weather conditions of a fire area. In wildland fire literature, vegetation is described by fuels, which refer to the composite of variables related to the vegetation through which the fire spreads. Terrain is described by slope and aspect, where slope is the inclination of a land surface and aspect is the direction the surface faces. Compared with fuel and terrain, weather has a dynamic influence on fire behaviors. The two components of weather that greatly influence fire spread are wind speed and wind direction. Corresponding to these factors that influence fire behavior, a DEVS-FIRE simulation needs four types of data: 1) fuel data; 2) slope data; 3) aspect data; and 4) weather data. The first three are raster data that represent geo-spatial information as a matrix of grids with attribute values. The weather data is temporal data that is either provided by users or obtained from local weather stations or online weather forecast services.

To support simulation services for broader users in the US, a raster database is established to store the fuel, slope, and aspect

data for the entire US. This raster database of fuel/slope/aspect uses the datasets from LANDFIRE [23], which provides raster fuel/slope/aspect data at a 30-meter resolution for wildland fire simulation. These data are developed from field-collected plot data, remote sensing data, and ecological modeling using standardized methods. Our previous work [24] has shown the feasibility of integrating the LANDFIRE fuel/slope/aspect data with DEVS-FIRE simulation. Besides the raster database that serves as the default source of information for fuel/slope/aspect data in wildland fire simulations, users will also be able to supply their own data or modify/customize the default raster data to carry out simulation experiments. The user-customized data will be stored in a separate user database that can be reloaded and modified for future experiments.

B. Cloud-based Simulation

The cloud-based simulation services will be realized using a microservice architecture, which allows the server-side functions to be distributed across multiple microservices that work together to provide simulation services for clients. Several microservices will be developed, including a DEVS-FIRE Web Server, a DEVS-FIRE API Server, a DEVS-FIRE Simulation Server, and a DEVS-FIRE Socket-IO Server. The DEVS-FIRE Web Server interacts with the client-side map-based web applications (described later) to provide web service requests/responses and to invoke simulation runs by calling the APIs provided by the DEVS-FIRE API server. The DEVS-FIRE API Server provides restful web service interfaces to configure and run simulations. The API server sends simulation requests to the DEVS-FIRE simulation server and returns results to the DEVS-FIRE web server or user applications. The DEVS-FIRE Simulation Server creates simulation jobs and run the simulations. Each simulation job is created within a container, the resources of which is released after the simulation is finished. The DEVS-FIRE Socket-IO Server returns simulation results to client-side web applications and user applications through socket-IO communication channels. This microservice-based architecture makes the cloud-based simulation system robust because the multiple microservices run independently and are not subject to a single point of failure.

To handle simulation requests from different users, simulation management will be developed to support simulation job creation, scheduling, and simulation run control. The simulation job creation is responsible for creating simulation jobs and configuring them accordingly based on users' requests. The job scheduling is responsible for scheduling the simulation jobs based on different types of simulations and their priorities. For example, a batch-run job received from the API interface will have a lower priority than a job originated from a user web application. A job queue will be implemented to support the job scheduling. The simulation run control is responsible for carrying out the simulation runs within containers. This is needed to synchronize with external programs and to carry out batch runs.

C. User Interfaces

The wildland fire simulation services will be accessible through two types of user interfaces: a graphic user interface (GUI) and an application programming interface (API). The GUI will be developed as a map-based web application that allows users to carry out simulation exercises on a map through a web browser. This map-based web application brings several advantages, including: 1) it removes the need of downloading and installing any software because users access the simulation service through a web browser; and 2) it provides an intuitive way for users to set up simulation experiments for any selected areas on the map. For example, when a landowner wants to use prescribed fire simulation to plan a prescribed burn for her own land, she would scroll to the location of her land using the map tool and then set up simulations on the map. This improves user experience because a user can directly relate the simulation outcomes to the specific land of her interest.

We are developing the map-based web application based on the Mapbox platform [25], which is a provider of custom online maps for websites and applications. Mapbox supports multiple map styles (such as terrain map and satellite map) that can be customized based on users' preference. It provides a multi-layer structure and a range of APIs, which can be used to support displaying different data and simulation results on the map, such as fuel, terrain, ignition routes, fireline locations, as well as fire spread simulation results.

The other type of user interface is the application programming interface (API). The goal of the API is to allow developer and researchers to integrate the wildland fire simulation into their own applications. For example, when developing UAS' path planning algorithms for monitoring wildfires, it is challenging to test the algorithms on real fire scenarios due to accessibility and safety concerns. The developed API makes it possible to integrate wildfire spread simulation into the research project for evaluating different path planning algorithms. To support usability, the API will support multiple programming languages on the user side.

A set of API methods are being developed to fully support the different functions of the simulation service. An essential group of the API methods is the simulation control. To work with a broad range of user applications that may need to run simulations in different ways, four modes of simulation control will be supported: 1) *Master-slave mode* that allows a user program to have a fine-grained control of a simulation execution by running simulation in a step-by-step fashion, where a step is defined according to the next event or the next time step; 2) *Synchronous model* that allows a user program to run a simulation to a pre-defined simulation time while waiting for the completion of the simulation; 3) *Asynchronous mode* that allows a user program to start a simulation without waiting for its completion. The user program works on other tasks while the simulation is running, and will be notified when the simulation is finished. 4) *Batch simulation mode* that supports large-scale batch simulation runs, covering Monte Carlo simulations and simulation experiments using parameter sweeping.

IV. Preliminary Results

We have implemented a web application called *FireMapSim* (http://firesim.cs.gsu.edu:3000/) to provide an initial version of the graphic user interface for accessing the simulation service. A snapshot of a fire spread simulation scenario using *FireMapSim* is shown in Fig. 5, which displays the simulation result in two different map views (to save space the GUI for setting up the simulation is not shown in the figure). In this

scenario, the fire was ignited using a line ignition shown in the middle; the outside blue rectangle is the fireline boundary that blocks fire spared. As can be seen, the fire spread inside the fireline boundary is also influenced by the road on the east side and the water area on the south side due to the unburnable fuel data associated with the road and water.

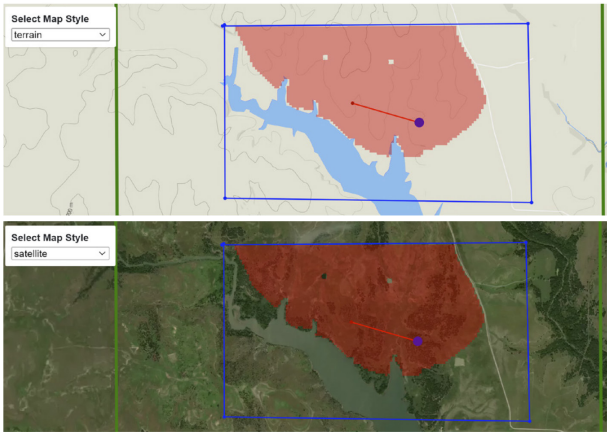


Fig. 5. Snapshot of a fire spread simulation using the *FireMapSim* web application. Top: terrain map view; bottom: satellite map view.

A preliminary version of the API server is also developed that allows users to set up simulations and obtain simulation results by programming to a small set of API methods. Information about how to use the API can be found at https://sims.cs.gsu.edu/sims/research/DEVSFIRE_API.html, which also includes a Java programming example of using the API. This example includes multiple steps: 1) Step 1: connect to the DEVS-FIRE API server and obtain a key, which is needed for future API calls. 2) Step 2: set wind condition for the simulation. 3) Step 3: set location of the fire area using the latitude and longitude info, which decides what fuel and terrain data from the LANDFIRE database will be used. 4) Step 4: set ignition point. 5) Step 5: start simulation run by providing a simulation time. The simulation result of burning cells' ignition time is returned as a string. A visualization tool (a Java class) is also provided to allow users to visualize the simulation result.

V. Conclusion

This paper presents an effort of developing a cloud-based service-oriented wildland fire simulation system that provides wildfire fire simulation services for broader users. The DEVS-FIRE model that supports the simulation services is described. A design of the cloud-based simulation, including the data (fuel, terrain, and weather), implementation of cloud-based simulation, and two types of user interfaces are presented. Future work includes fully implementing the cloud-based simulation services, evaluating their validity and performance, and extending the GUI and API for accessing the simulation services.

References

[1] D.J. Wuebbles, D.W. Fahey, K.A. Hibbard, D.J. Dokken, B.C. Stewart, and T.K. Maycock (eds.), Climate Science Special Report: Fourth National Climate Assessment, Volume I. U.S. Global Change Research Program, Washington, DC, USA, 470 pp. 2017

[2] "Prescribed Fire". https://www.fs.usda.gov/managing-land/prescribed-fire. (accessed April. 7, 2024)

[3] "Modernizing Wildland Firefighting to Protect Our Firefighters," https://www.whitehouse.gov/wp-content/uploads/2023/02/PCAST_Wildfires-Report_Feb2023.pdf

[4] "FlamMap," https://www.firelab.org/project/flammap (accessed April. 7, 2024)

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

[6] "BehavePlus Fire Modeling System," https://www.frames.gov/behaveplus/home (accessed April. 7, 2024)

[7] "Fire Dynamics Simulator (FDS) and Smokeview (SMV)," https://pages.nist.gov/fds-smv/manuals.html (accessed April. 7, 2024)

[8] "ForeFire," https://github.com/forefireAPI/firefront https://github.com/forefireAPI/firefront

[9] L. Ntaimo, X. Hu, and Y. Sun, "DEVS-FIRE: Towards an integrated simulation environment for surface wildfire spread and containment," *Simulation*. 84(4):137-55, 2008

[10] X. Hu, Y. Sun, and L. Ntaimo, "DEVS-FIRE: design and application of formal discrete event wildfire spread and suppression models," *Simulation*. 88(3):259-79, 2012

[11] X. Hu and Y. Sun, "Agent-based Modeling and Simulation of Wildland Fire Suppression," in Proc. 2007 Winter Simulation Conference, 2007

[12] X. Hu and G. Mu, "Modeling and Simulating Prescribed Fire Ignition Techniques," in Proc. 2021 Annual Modeling and Simulation Conference (ANNSIM'21), 2021

[13] B. P. Zeigler, H. Praehofer, T. G. Kim, Theory of modeling and simulation, 2nd ed. NewYork: Academic Press, 2000

[14] R. C. Rothermel, "A mathematical model for predicting fire spread in wildland fuels," Research Paper INT-115. Ogden, UT: U.S. Department of Agriculture, Forest Service, Intermountain Forest and Range Experiment Station. 40 p, 1972.

[15] F. Gu, X. Hu, and L. Ntaimo, "Towards validation of DEVS-FIRE wildfire simulation model," in Proc. of the 2008 High Performance Computing and Simulation Symposium (HPCS'08), 2008, pp. 355–361.

[16] N. Dahl, H. Xue, X. Hu, and M. Xue, "Coupled Fire-Atmosphere Modeling of Wildfire Spread Using DEVS-FIRE and ARPS," *Natural Hazards*, Volume 77, Issue 2, pp 1013-1035, 2015

[17] H. Xue, F. Gu, and X. Hu, "Data Assimilation Using Sequential Monte Carlo Methods in Wildfire Spread Simulation," The ACM Transactions on Modeling and Computer Simulation (TOMACS), Vol. 22, No. 4, Article No. 23, 2012

[18] H. Xue, X. Hu, N. Dahl, and M. Xue, "Post-frontal combustion heat modeling in DEVSFIRE for coupled atmosphere-fire simulation," in Proc. International Conference on Computational Science (ICCS) 2012, pp. 302-311, 2012

[19] J.S. Fried and B. D. Fried, "Simulating wildfire containment with realistic tactics," Forest Sci; 42: 267–281, 1996

[20] X. Hu and L. Ntaimo, "Integrated Simulation and Optimization for Wildfire Containment," The ACM Transactions on Modeling and Computer Simulation (TOMACS), Vol. 19, No. 4, Article No. 19, 2009

[21] R. E. Martin and J. D. Dell, "Planning for prescribed burning in the inland northwest," Pacific Northwest Forest and Range Experiment Station GTR-76, 1978.

[22] X. Hu, M. Ge, S. Gowravaram, H. Chao, and M. Xin, "Prescribed Fire Simulation with Dynamic Ignitions Using Data from UAS-based Sensing," Journal of Simulation, DOI: 10.1080/17477778.2023.2217335, 2023

[23] "Landfire (LF)," https://landfire.gov/ (accessed April. 7, 2024)

[24] M. Yan and X. Hu, "Towards A Map-Based Web Application for Prescribed Fire Simulation," in Proc. IEEE SoutheastCon 2023, April 2023

[25] "Mapbox Documentation," https://docs.mapbox.com/ (accessed April. 7, 2024)