Universitat Autònoma de Barcelona

Dipòsit digital de documents de la UAB

This is the **published version** of the bachelor thesis:

Costas Mateu, Carles; Cortés Fité, Ana, dir. Validació d'un sistema d'aprovisionament automàtic de simulació de propagacions d'incendis forestals. 2023. (958 Enginyeria Informàtica)

This version is available at https://ddd.uab.cat/record/272787

under the terms of the BY-NC-ND license

Validació d'un sistema d'aprovisionament automàtic de simulació de propagacions d'incendis forestals

Carles Costas Mateu

Resum — Utilitzant el simulador de propagació d'incendis forestals Farsite i el planificador de tasques Luigi podem realitzar simulacions d'incendis en temps real i per poder predir la direcció i la magnitud de propagació d'incendis. Per tal de garantir el correcte funcionament del planificador i ajudar a la generació dels inputs necessaris del simulador, aquest projecte intenta millorar i validar aquest procés, per tal de garantir el mínim d'errades en la preparació de totes les entrades necessàries per Farsite.

Aquestes millores són necessàries, ja que, actualment, l'entorn utilitzat presenta una problemàtica a l'hora de generar les dades d'entrada: la generació dels *Landscapes* és molt costosa i no és trivial. En casos on es volen simular incendis reals, la rapidesa i facilitat per poder simular-los és crucial per aportar valor i ajudar a mitigar incendis amb l'eficiència més gran possible.

En aquest treball s'exposa com generar aquests fitxers *lcps* utilitzant diferents perímetres d'ignició i es valida que aquests funcionin correctament fins a culminar el procés de simulació.

Paraules clau — Planificador, Simulador, Farsite, Verificació, Millora del procés

Abstract — Using the Farsite simulator and the Luigi orchestrator we can perform real time fire simulations and predict the direction and magnitude of fire spread. In order to guarantee the correct work of the orchestrator and help to generate the necessary inputs for the simulator. This project tries to improve and validate this process, in order to guarantee the minimum of errors in the preparation of all the necessary inputs for Farsite.

These improvements are necessary, since the current Farsite fire simulation has a problem when generating the input data, where the generation of the Landscapes is very expensive and not trivial. In cases where real fires are to be simulated, the speed and ease of simulating them is crucial to add value and help mitigate fires as efficiently as possible.

In this paper we show how to generate these lcps fitxers using different ignition periods and validate that they work correctly until the simulation process is completed.

Index Terms — Orchestrator, Simulator, Farsite, Verification ,Process improvement.

1 INTRODUCCIÓ - CONTEXT DEL TREBALL

En l'actualitat un dels simuladors més potents i més precisos de propagació d'incendis forestals és Farsite. Un simulador que a partir del model de combustibles de Rothermel i conjuntament amb una sèrie d'inputs, aconsegueix simular la propagació d'incendis en una certa regió donada.
Per tal d'aconseguir que Farsite funcioni correctament se li han de brindar una sèrie d'inputs molt concrets, amb una estructura molt precisa, entre els quals els més importants serien els següents:

- El *Landscape* proporciona informació del terreny.
- El *Weather* proporciona informació de les temperatures i humitats segons els dies.
- El *Wind* proporciona informació del vent segons les hores i els dies.
- L'*Initial Fuel Moisture* proporciona els possibles combustibles del que pot estar format el terreny
- *L'Adjustment* on es pot indicar un augment de la propagació dels incendis.

Cada un d'ells s'ha de donar en un cert format molt específic i està compost amb unes qualitats molt concretes. La motivació de simular la propagació d'un incendi forestal és predir la propagació abans que el mateix incendi i, per tant, és necessari poder generar aquests inputs de la forma més precisa i ràpida possible.

Per tal de satisfer aquest objectiu tan evident i essencial, he utilitzat de base el prototipus d'una eina elaborada en el departament ACSO en la qual s'implementa un pipeline, fent servir un planificador de tasques d'una llibreria de Python anomenada Luigi. Per tal de poder realitzar tot el tractament de les dades i poder generar els inputs era necessari fer servir conjuntament eines com *pygdal* o altres llibres Python amb el propòsit de generar els inputs esmentats anteriorment. Aquest planificador, tot i ja estàr implementat, tenia diversos problemes de versions i *corner cases* que no s'havien tingut en compte, pel qual era necessari dur a terme millores i una validació formal per garantir el correcte desplegament i funcionament de l'entorn de treball. D'aquí va sorgir la principal motivació d'aquest treball, on els principals objectius inicials es van desglossar en els següents quatre subobjectius:

- Entendre i aprendre l'estructura i la creació dels inputs

necessaris per executar el simulador.
- Assolir coneixements del funcionament i utilització de la llibreria Luigi.
- Realitzar millores en la implementació actual del planificador de tasques per tal de reduir el nombre d'execucions fallides a causa d'aquest.
- Dur a terme una verificació funcional del planificador de tasques per tal d'assegurar la correctesa i qualitat del flux d'execució.

En l'actual món dels simuladors d'incendis forestals hi ha diversos programes, entre ells Farsite4.[6] Actualment, aquest ja no té suport i Farsite5, el seu successor, està integrat en el programa FlamMap6 (una aplicació d'escriptori per Windows la qual es fa servir per simular característiques potencials del comportament dels incendis). L'interès del departament ACSO, en concret el grup de recerca de GFIRE amb el que he realitzat el treball, és intentar millorar la rapidesa amb què es realitzen les simulacions. El departament va demanar el codi font de Farsite per tal de poder treballar directament amb ell i poder dur a terme millores i optimitzacions, des de poder executar el simulador en GPU i paralitzar-lo fins a refer parts de l'algoritme per tal de millorar tot el procés. El meu projecte va partir de la base d'un *pipeline* d'execució fet pel departament, el qual tenia problemes en fer el desplegament de l'entorn i no s'havia validat correctament per tal de garantir la qualitat.

Com que la simulació d'incendis era un camp completament nou per a mi, vaig decidir utilitzar una metodologia pausada que seguís un ordre i pogués avançar pas a pas. Per aquest motiu vaig triar usar un model en cascada o també conegut com a cicle de vida. El qual és un procés lineal on el treball es fa de manera esglaonada i en un ordre seqüencial. Això m'assegurava que la feina es mantenia encaminada i es tenia una bona comunicació del punt en el qual es trobava el treball. De la mateixa manera, distribuirem l'estructura de l'article, on començarem exposant la planificació que es va plantejar inicialment i seguirem pas a pas exposant les diferents fases del projecte. Començant per entendre com funciona Farsite i els seus inputs, les diferents problemàtiques vistes durant la realització del treball, les solucions trobades i les validacions dutes a terme sobre el planificador.

2 PLANIFICACIÓ INICIAL

La planificació que es va plantejar inicialment es va dividir en diferents fases per tal de seguir un ordre coherent en el treball i avançar sobre unes bases sòlides. La primera fase es va centrar en la investigació i presa de contacte amb el simulador per tal d'entendre el funcionament. Principalment, em vaig enfocar en aprendre a executar el simulador i entendre com es generaven els diferents inputs que aquest requeria, ja que posteriorment necessitaria tenir un coneixement ampli sobre quins inputs necessitava i també com era necessari preparar-los per a consumir-los. Aquesta fase estava planificada per dur-se a terme entre tres i quatre setmanes.
Un cop assimilat aquest coneixement, començaria una fase per tractar amb el planificador de tasques i entendre com estava implementat i on podien estar les mancances d'aquest. Aquesta segona fase es va plantejar originalment per ocupar entre cinc i sis setmanes i l'objectiu principal era aconseguir executar el planificador fins la última tasca. D'aquesta manera no només vaig entendre com estava implementat sinó que també es començava un procés d'anàlisis de debilitats i un procés de validació.
Un cop arribat aquest punt, ja podíem començar amb una fase d'implementació on la idea era aportar millores en el planificador original, extretes de l'anàlisi de debilitats que hauríem realitzat en la fase anterior. Per tal d'implementar aquestes millores teníem un marge d'unes quatre setmanes, ja que finalment només ens quedarien cinc setmanes per fer una verificació formal, documentar com usar el planificador i concloure el projecte en aquesta última fase de validació.

Per tant, disposava d'unes dinou setmanes per tal d'entendre, analitzar, implementar i verificar el planificador. També cal remarcar que durant el treball, s'ha realitzat cada setmana una sessió amb l'equip de GFIRE per tal de posar en comú els avenços, problemàtiques i dubtes tinguts durant la setmana anterior.

3 FARSITE FUNCIONAMENT

[1] Farsite és un simulador d'incendis forestals el qual divideix el front de propagació de l'incendi en diferents punts i, basant-se en el model de propagació de Rothermel, calcula la propagació d'aquests en un pas de temps determinat. Per tal de calcular la propagació de cada punt del front es tenen en compte 3 factors diferents:

- Les condicions locals com el tipus de vegetació, la humitat i la temperatura, que són les que defineixen la direcció normal de propagació de l'incendi, representat a la figura 1 com ha R_0.
- El vent afectant a la velocitat i la direcció de prepagament, representat a la figura 1 com ha ϕ_w.
- El pendent que també influeix en el grau de difusió i direcció de propagació, representat a la figura 1 com ha ϕ_s.

Utilitzant aquests tres paràmetres es realitza un càlcul basat en la següent fórmula, tal com es pot veure en la figura 1, que determina la direcció i la velocitat final del punt que es vol propagar.

$$R = R_0 \cdot (\vec{n} + \vec{\emptyset}_w + \vec{\emptyset}_s)$$

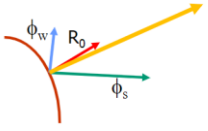

Figura 1. Factors de propagació

Un cop Farsite ha propagat tots els punts del perímetre, es realitza una reconstrucció del perímetre per tal d'obtenir el nou front de l'incendi en aquell salt de temps determinat. Aquest procés es repetirà fins a arribar al temps de simulació indicat. Durant aquest procediment poden succeir anomalies en la propagació, on els punts es creuen i apareixen conflictes a l'hora de reconstruir el perímetre el qual pot fer augmentar el temps de simulació i el cost d'execució de la simulació, tot i que no entrarem en explicar ni estudiar aquesta part cal tenir present que és un algoritme complex.

4 FARSITE I INPUTS

Tal com es pot suposar, el simulador requereix uns mínims fitxers per poder funcionar, entre ells es troben el *Landscape* amb tota la informació del terreny, el *Wind* i el *Weather* els quals proporcionen informació sobre la meteorologia, i altres fitxers com la configuració de l'execució o el tipus de combustibles que pot haver-hi sobre el terreny i finalment, el perímetre del foc inicial del incendi.

- *E-mail de contacte: 1491578@uab.cat*
- *Menció realitzada: Enginyeria de Computadors*
- *Treball tutoritzat per: Ana Cortés Fité*
- *Curs 2022/23*

4.1 LANDSCAPE

[5] El *Landscape* (.lcp) és un dels fitxers més importants i complexes de generar. Per tal d'entendre la seva estructura primer hauríem de fer una breu explicació dels fitxers de tipus *Raster*. Tal com es pot veure a la figura 2, els *Rasters* són models de dades espacials que es defineixen com un conjunt de cel·les de la mateixa mida. Aquests fitxers contenent fotografies digitals o imatges de satèl·lits, i poden contenir diferents capes d'informació tenint així una cel·la amb diferents propietats definides per cada capa que conforma el *Raster*.

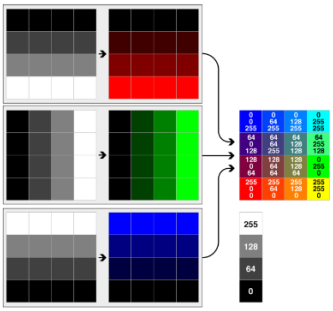

Figura 2. Capes del Raster.

D'aquesta manera podem generar *lcps* de fins a un màxim de deu capes de les quals només són necessària cinc per poder executar el simulador. Tal com es mostra a la taula 1, les capes necessàries per funcionar són l'elevació, el pendent, l'orientació, el combustible terreny i la cobertura del terreny. On cada un es representa amb les unitats mostrades a la taula. Per tal de poder crear un *lcp* correcte cal que totes les capes que conformen el *lcp* estiguin en la mateixa projecció de la terra, continguin la mateixa mida de resolució de les cel·les i no falti cap de les capes requerides.

File Theme	Required	Default Units
elevation	yes	meters
slope	yes	degrees
aspect	yes	categories 1-25
fuel model	yes	13 NFFL models
canopy cover	yes	categories 1-4
tree height	no	meters*10
crown base height	no	meters*10
crown bulk density	no	kg/m^3*100,
duff loading	no	Mg/ha
coarse woody	no	

Taula 3. Capes necessaries per funcionar

Per tal d'extreure tota aquesta informació i poder generar els *lcps* és necessari els arxius (.tif), els quals contenent tota la informació del terreny. Per tractar aquest fitxer .tif i assegurar que es compleixen els requisits necessaris per poder tenir un *lcp* correcte es fa servir l'eina GDAL. Aquesta eina és una llibreria d'abstracció de dades geoespacials amb la qual podem reprojectar, modificar la resolució dels *Rasters*, retallar i extreure les diferents capes que necessitem per generar el *lcp*. [2] D'aquesta manera aconseguir crear cinc fitxers en el format (.asc) corresponents a cada capa necessària, on es representa el *Raster* en un format ASCII i on cada un hauria de tenir un fitxer parell amb el mateix nom i extensió .prj on s'indica la projecció en la qual està representat. Per aconseguir aquests cinc fitxers s'han de seguir els següents passos:

1. Retallar la zona del *Raster* amb la informació del terreny on es vol simular l'incendi.
2. Modificar la projecció i la resolució per tal de tenir la mateixa a tots els fitxers. Cal tenir en compte que s'ha de aplicar la resolució més gran que tinguem entre els tres mapes d'informació que hem de tractar.
3. Generar el fitxer d'Elevació, Pendent i Orientació a partir del *Raster* tractat.
4. Realitzar els punts 1,2 i 3 amb el *Raster* que conte la informació dels combustibles del terreny.
5. Dur a terme els punts 1,2 i 3 amb el *Raster* amb la informació de la cobertura dels arbres.

Finalment, utilitzant els fitxers resultants podem acabar generant el *Landscape*, el qual té un aspecte com el de la figura 4, el que visualitzarem utilitzant QGIS.

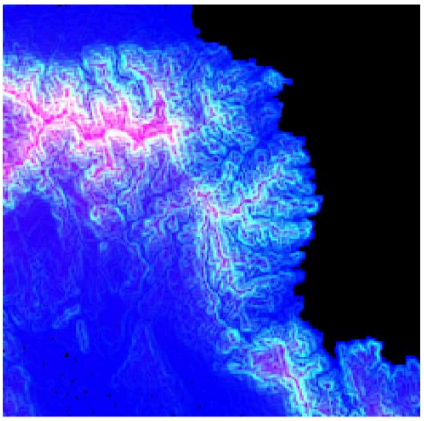

Figura 3. Lamdscape

Tots aquests passos són necessaris per crear un *lcp* correctament i aconseguir que funcioni Farsite, en el cas de tenir alguna resolució o projecció diferent en alguns dels .asc, el *lcp* no seria correcte i el simulador no sabria propagar l'incendi.

4.2 WIND I WEATHER

Per tal d'entendre completament com funciona Farsite, vaig haver de familiaritzar-me amb els fitxers que contenien la meteorologia, tot i que el meu treball es centra més en la generació del *Landscape*, crec que és important tenir una visió general sobre aquests dos fitxers, ja que són pilars bàsics sobre el que es basa el model de Rothermel.
Tal com hem dit, aquests dos fitxers tenen dades de la meteorologia.
[4] El fitxer *Wind* (.wnd) té informació sobre la direcció, velocitat i cobertura del vent per cada hora del dia i ha de seguir la següent estructura:

- Format de les dades sigui en milles o en kilòmetres per hora.
- Data i hora amb el format "mes dia hora en minuts (0-2400)".
- Velocitat del vent especificada en milles per hora o kilòmetres per hora.
- Direcció del vent especificada en graus de (0-360).
- Cobertura del vent especificada com un percentatge de (0-100).

[3] Per altra banda, tenim el fitxer *Weather* (.wtr) el qual ens proporciona les dades de la precipitació, temperatura i humitats per cada hora, la representació de la informació ha de seguir la següent estructura:

- Format de les dades sigui *Fahrenheit* o *Celsius*.
- Precipitació especificada en centèsimes de pols o mil·límetres amb un nombre enter.
- L'hora del registre de la temperatura mínima en minuts.
- L'hora del registre de la temperatura màxima en minuts.
- La temperatura mínima en *Fahrenheit* o *Celsius*.
- La temperatura màxima en *Fahrenheit* o *Celsius*.

- La humitat màxima en percentatge.
- La humitat mínima en percentatge.
- L'elevació sobre el nivell del mar.

Aquests dos fitxers són molt importants tenir-los en consideració, ja que si volem simular un cert període de temps, però no tenim les dades d'aquest període concret, el simulador no funcionarà. Per altra banda, també és molt rellevant si volem tenir una simulació més realista aconseguir les dades meteorològiques del moment real de l'incendi, tot i que per casos acadèmics, es podria simular sempre en la mateixa franja temporal per facilitar l'execució.

5 PLANIFICADOR DE TASQUES LUIGI

Un cop assimilat la generació dels inputs i el funcionament del simulador, se'm va proporcionar una imatge de la màquina virtual on hi ha la implementació del planificador de tasques utilitzant Luigi.

[7] Luigi és un paquet de Python el qual permet la construcció de treballs *batch* complexos, visualitzar l'estat del processament de les tasques i el seu encadenament entre elles. És una solució que ha estat adaptada per empreses com Spotify o Red Had. L'estructura de Luigi és una estructura *Server – Client*, on tenim un servei o dimoni de sistema "*luigid*" que dona servei a diferents clients que poden posar en cua *pipelines* de tasques i d'una manera gràfica veure l'estat d'aquests. Tal com es pot observar a la figura 4, el planificador genera un conjunt de tasques, representades com a nodes de diferents colors, els quals es van executant en l'ordre establert segons les dependències.

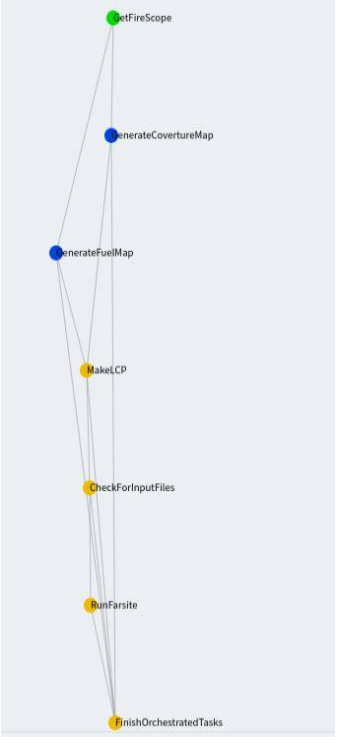

Figura 4. Tasques generades pel planificador

El color amb el qual està representat cada node indica l'estat en què es troba aquella tasca, tal com podem veure a la figura 5, el color verd vol dir que la primera tasca ja està conclosa, els nodes blaus són tasques que s'estaven executant i els nodes grocs són tasques que estaven pendents a ser executades.

Figura 5. Simbologia de l'estat de la tasca

Tot aquest entorn, conjuntament amb el paquet d'eines de *pygdal*, conformen l'entorn de simulació que venia a la màquina virtual. Una de les principals problemàtiques que es van trobar l'equip de GFIRE en preparar la màquina va ser la incompatibilitat de *pygdal* amb la versió que requeria el sistema operatiu i Luigi. Fent d'aquesta màquina virtual un entorn molt fràgil on no es podia actualitzar la versió de Python ni dels paquets d'aquest.

Per l'altra banda, vaig tenir problemes de dos tipus: els problemes tècnics, els quals comprenien la implementació del planificador de tasques i els problemes conceptuals, on el problema no era la tecnologia sinó el procés que es feia per realitzar la simulació i tractar els fitxers d'entrada no era correcte.

En el sentit tecnològic vaig detectar que el planificador no funcionava correctament, ja que l'estructura de carpetes que necessitava no estava degudament creada i moltes de les crides que realitzava el *pipeline* no trobaven els fitxers que necessitaven. Aquesta va ser una tasca laboriosa, pel fet que vaig haver d'anar revisant el codi per tal de corregir totes les *urls*, però no va ser una tasca complicada. Un cop vaig tenir totes les *urls* modificades el *pipeline* va començar a funcionar tot i no acabar la simulació correctament.

Per altra banda, en el sentit conceptual, vaig trobar-me amb problemes a l'hora d'executar les tasques i que acabessin amb un resultat correcte. Com per exemple retallar els mapes o bé crear els *lcps*, els principals problemes eren causats per dos conceptes que podríem dir que van ser on més temps vaig invertir investigant:

1. Projecció original dels mapes.
2. Algoritme per calcular les dimensions a retallar del mapa.

Per tal de poder executar el planificador era necessari tenir els fitxers .tif en les carpetes corresponents. Aquests, en un bon principi, els vaig projectar a la projecció ESPG:4326, el que causava que els mapes es representessin en graus. Això ocasionava que els processos per calcular la mida i retallar els mapes ocasionessin diferents errors com: mapes amb amplades inferiors o iguals a zero o retalls dels mapes massa grans per la capacitat de la màquina virtual. Un cop em vaig adonar que la projecció per la qual estava preparat el codi era la projecció ESPG:3035, la qual treballa amb metres, vaig reprojectar els mapes a la projecció correcta i el procés per retallar els .tif va començar a funcionar correctament.

A partir d'aquí vaig començar a generar els meus propis punts

- *E-mail de contacte: 1491578@uab.cat*
- *Menció realitzada: Enginyeria de Computadors*
- *Treball tutoritzat per: Ana Cortés Fité*
- *Curs 2022/23*

d'ignició dels incendis per tal de verificar que funcionava. Em vaig adonar que només funcionava pel perímetre de *test* que tenia a la màquina virtual. Després de realitzar diferents proves em vaig trobar un error en el codi de l'algoritme per calcular les dimensions a retallar del mapa. El codi utilitzava el perímetre inicial de l'incendi donat, per calcular la màxima distància del polígon del perímetre inicial i a partir d'aquest valor multiplicar per una constant i obtenir un mapa deu cops més gran que el perímetre inicial. Aquest codi tenia dos problemes, els casos on el perímetre inicial fos un punt la distància màxima seria 0, per tant, esdevindria un mapa de 0x0 i els casos on el perímetre inicial fos excessivament gran el mapa resultant no podria ser tractar per Farsite, ja que no es feia cap validació. Per solucionar aquesta dificultat vaig modificar el codi, per generar mapes de 20000x20000 metres quan el perímetre inicial era un punt, i, per altra banda, vaig començar a realitzar simulacions provant quin podia ser el límit de grandària d'un *lcp* per realitzar una simulació.

Durant aquest procés vaig tenir problemes amb la màquina virtual, ja que per algun motiu no alliberava correctament la memòria i deixava la màquina sense memòria disponible. Això va resultar ser una qüestió de la configuració de la màquina virtual, on per algun motiu tenia més memòria assignada que la memòria de la que tenia disponible el *host*. Per solucionar aquesta qüestió, vaig actualitzar la versió de VirtualBox, seguidament vaig reassignar la memòria i va començar a funcionar correctament. Tot i que va ser un problema secundari sobre el projecte, cal esmentar que va ocasionar un retard en la planificació del projecte.

6 LUIGI AL DOCKER

Un cop ja tenia el planificador funcionant i ja havia vist les seves mancances, calia prendre una decisió respecte a quina millora aportar al planificador. Com que el temps era limitat, no podia entrar en totes les debilitats que havia localitzat, entre les quals vaig detectar les següents debilitats:

- L'entorn de simulació era fràgil a causa de les versions que necessitaven els diferents components.
- Desplegament de la màquina virtual era costós, ja que el .ova de la maquina amb tot el sistema funcionant pesava 19 GB.
- No acceptava mapes amb diferents projeccions i era necessari reprojectar els mapes a la projecció ESPG:3035
- Millores en l'algoritme de retall de mapes i evitar problemes en el càlcul de les dimensions.
- Difícil obtenció de les dades actualitzades per executar Farsite.

Analitzant totes les debilitats vaig decidir passar a un entorn de contenidor tota la implementació. D'aquesta manera evitaríem problemes de versions i podríem fer un entorn més robust i més fàcil de tornar a desplegar en el cas de ser necessari. Evitaríem un entorn pesat fent més àgil el desplegament i la compartició d'aquest i en un futur es podrien aplicar millores per intentar automatitzar l'obtenció de les dades i la reprojecció dels mapes.

Per tal de fer aquesta millora pretenia utilitzar Docker,[8] un projecte de codi obert que automatitza el desplegament d'aplicacions dins de contenidors de programari, proporcionant així una capa addicional d'abstracció i automatització de virtualització d'aplicacions en diferents sistemes operatius.

En primera instància, la idea era crear un únic contenidor. Per tal de poder-ho aconseguir necessitava poder executar el dimoni de Luigi en *background*. El que semblava una tasca fàcil, ja que a la documentació s'especifica que per executar en segon pla el dimoni s'ha d'executar amb la següent comanda *"luigid –background"*, no va acabar funcionant per raons relacionades amb el contenidor. Tot i que les proves que vaig realitzar sobre la màquina virtual sí que permetien aquesta opció. També vaig provar altres variants com intentar executar el procés sencer en segon pla afegint el caràcter & al final de la comanda, però tampoc va tenir èxit. Aleshores, vaig pensar que es necessitava una opció diferent.

Per tal d'evitar posar el servei de Luigi en segon pla vaig decidir utilitzar més d'un contenidor, per tal de tenir en un contenidor el servei de Luigi corrent i, per una altra banda, un entorn per executar el codi que llença les tasques contra el servei del Luigi. Per poder facilitar l'escalabilitat d'aquest entorn, vaig fer servir Docker Compose. Docker Compose és una eina que defineix i executar un entorn multi contenidor i facilita la implementació d'entorns que necessiten més d'un servei actiu per funcionar. La configuració parteix d'un fitxer base *docker-compose.yml*, en el qual per cada servei que es vol tenir es crearà un contenidor diferent. Un dels avantatges és que els contenidors entre ells es veuen sense necessitat de configurar cap xarxa, d'aquesta manera fent servir simplement el nom del contenidor, podríem configurar on ha de llençar les tasques el client del Luigi.

6.1 CONFIGURACIÓ DE L'ENTORN

Per tal de tenir un entorn on l'equip de GFIRE no hagués d'entrar a la configuració del contenidor vaig intentar desenvolupar l'entorn de la manera menys costosa d'utilitzar i menys pesada de transportar. Ja que una de les debilitats que tenia la màquia virtual original era el seu pes a l'hora de compartir-la.

Com que els contenidors necessiten una imatge pròpia vaig crear dos *Dockerfile*. El primer contindria només el servei de Luigi i el segon on tindríem tot el codi de Farsite i del planificador per llençar les tasques. Per tal de poder fer això era molt important instal·lar la versió 3.2.2.10 de *pygdal* sobre la següent imatge d'Ubuntu *osgeo/gdal:ubuntu-full-3.2.2*, ja que sinó hi havia problemes de conflictes amb el Python i la llibreria de *pygdal*. Seguidament, es defineixen tots els paquets de Python necessaris per al planificador i es creava l'estructura de directoris de treball de la màquina. Finalment, es crida un *ENTRYPOINT* en el cas del servidor l'execució del servei de Luigi i pel cas del client un *script bash* amb les crides necessàries per executar el planificador.
Un cop vaig tenir el dos *Dockerfiles* creats, vaig passar a la configuració que resideix en el fitxer *docker-compose.yml* tal com podem veure a la figura 6. En aquesta figura podem veure el que seria la configuració d'un dels serveis. Tenim un contenidor que es crearà a partir d'un dels *Dockerfiles* creats anteriorment. Per tal d'evitar imatges molt passades vaig decidir crear volums compartits entre el host i el contenidor. En aquests volums s'han de deixar tots els mapes i inputs necessaris per a l'execució. D'aquesta manera reduiríem el pes, el temps de creació de la imatge i no caldria accedir al contenidor per extreure ni modificar cap input.

```
luigi-client:
 container_name: luigi-client
 build:
  context: ./
  dockerfile: Dockerfile_luigi-client
 image:
  luigi/luigi-client
 volumes:
  - ${PATH_PERIMETER}:/opt/app/FARSITE5/luigi_execution_input_files:rw
  - ${PATH_ICGC}:/opt/app/Downloads/ICGC:rw
  - ${PATH_COPERNICUS}:/opt/app/Downloads/Copernicus:rw
  - ${PATH_USGS}:/opt/app/Downloads/USGS:rw
  - ${PATH_SpatialReference}:/opt/app/Downloads/SpatialReference:rw
 privileged: true
```

Figura 6. Configuració d'un dels serveis

Per evitar que els usuaris que consumissin aquest entorn necessitin modificar el .yml per corregir les rutes dels directoris compartits, vaig triar utilitzar variables d'entorn, tal com es pot veure a la figura 6. Per utilitzar una variable d'entorn simplement hem de definir en el fitxer .env el nom de la nostra variable i el valor. Aleshores, podrem fer

servir la variable usant la nomenclatura que es veu a la figura. Aquest fitxer *.env*, també ha d'anar acompanyat d'un fitxer *.dockerignore*, per tal d'indicar al Docker que ignori els fitxers i les carpetes compartides a l'hora de crear les imatges. Aquest últim pas és molt important, ja que si no ignorem els directoris compartits, el Docker passarà tota l'estructura del directori al dimoni de Docker incloent els mapes. Fent així que el temps de construcció de les imatges sigui molt més lent.

Finalment, per acabar de configurar el contenidor del client vaig haver de modificar la configuració del Luigi,[9] perquè en lloc d'atacar a la direcció del *localhost:8082*, fes servir el nom del contenidor del servei. Per fer això és necessari afegir un fitxer anomenat *Luigi.cfg* amb el nom del *host Schedule*, on en el nostre cas era *lugi-server:8082*, en la ruta de configuració */etc/luigi*. Aquest pas és necessari per indicar al nostre client de Luigi que el *Schedule* central no està a la mateixa màquina sinó que ha de consumir un *Schedule* extern ubicat al contenidor que està proveint el servei de Luigi.

Un cop finalitzada tota la configuració de l'entorn Docker, podia executar el planificador sense necessitat de desplegar una màquina virtual i sense la necessitat d'accedir-hi. Simplement deixant els inputs als volums compartits i aixecant l'entorn amb la comanda *docker compose up*. Automàticament es començarà a executar el servei de Luigi, el qual podem també veure des del nostre *host* accedint a la url localhost:8082 pel fet que estem fent un reenviament dels ports i es començaria a llençar les tasques del planificador. Un cop el planificador ha acabat, si tot ha anat bé i Farsite ha pogut generar els resultats, podríem veure al directori compartit */outputs* tots els fitxers de la simulació resultants.

7 VALIDACIÓ FUNCIONAL

Per tal de garantir una qualitat i correcta validesa de les execucions, després d'haver solucionats tots els errors amb el simulador i realitzades les millores migrant a l'entorn Docker la implementació, vaig començar a realitzar proves funcionals amb el planificador. L'objectiu d'aquestes proves era validar que els diferents casos d'execució funcionaven correctament i la implementació a l'entorn dels contenidors continuava retornant resultats correctes. Per tal d'aconseguir aquests objectius, vaig seleccionar tres casos els quals poden fer variar l'execució del simulador depenent del perímetre d'ignició que se li proporciona, per retallar els *Rasters*, generar els *lcps* i determinar el punt de partida de la simulació. Aquest perímetre d'ignició pot ser un punt, una línia o un polígon, per tant, les proves que vaig fer van basar-se en aquests tres tipus de perímetres.

Figura 7.Possibles punts d'ignició

Per altra banda, per tal de garantir la qualitat del nou entorn de planificació i simulació, també vaig realitzar proves amb casos de simulacions d'incendis reals com podria ser el "Top 10" (recull de casos del departament utilitzats per realitzar verificacions i càlculs de mètriques) o bé simulant perímetres extrets de punts calents de mapes, extrets des del satèl·lit de la NASA.

7.1 PERÍMETRE POLÍGON

Un dels punts d'ignició més importants per tal de poder simular incendis és el polígon, ja que, normalment un incendi es simula un cop ha començat i existeix una petita regió cremada en el mapa. Per poder representar aquesta secció inicial de l'incendi, es fa servir un polígon que no té perquè ser regular, fent així un cas de prova necessari per garantir una mínima qualitat.

Per tal de garantir que els perímetres de tipus polígon funcionaven, vaig crear diferents perímetres de diverses grandàries, donat que la mida impactava directament sobre la mida del *lcp*. Com es pot veure la figura 8, la simulació sobre polígons funciona correctament tant la generació del *lcp* com la propagació de l'incendi. Durant la validació d'aquests tipus de perímetre vaig adonar-me que per polígons que generaven *lcps* de més de cinquanta kilòmetres començava a donar problemes. Es podien percebre lentituds sobre l'execució de les tasques del planificador i en els casos que superaven els *lcps* de més de cinquanta kilòmetres, Farsite donava errors a l'hora de l'execució. Tot i això, vaig donar per vàlid aquest cas, ja que els casos on el perímetre tenia una mida estàndard entre cinc sents metres i fins a dos kilòmetres funcionaven correctament.

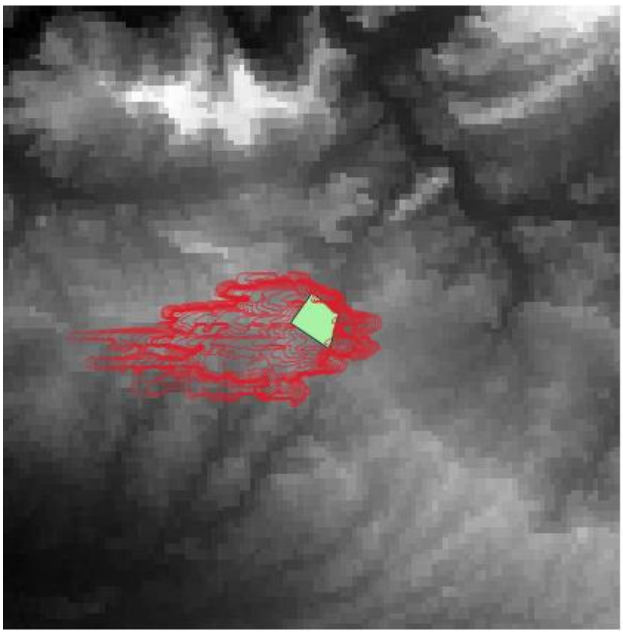

Figura 8. Simulació sobre polígons

7.2 PERÍMETRE LÍNIA

El segon cas de prova, tot i no ser un dels perímetres més utilitzats, va ser la validació d'execucions utilitzant línies. Per tal de garantir que els perímetres de tipus línia funcionaven vaig crear diferents perímetres de diferents mides de la mateixa manera que el cas anterior. Tal com es pot veure a la figura 9, la simulació sobre línies funciona correctament, tant la generació del *lcp* com la propagació de l'incendi. Igual que el cas del polígon també hi hauria problemes per línies que generessin *lcps* que superessin els cinquanta kilòmetres de llarg. Donat que el problema real no és la mida del perímetre inicial, sinó la mida del *lcp* i, per tant també es podria tenir en una execució

- *E-mail de contacte: 1491578@uab.cat*
- *Menció realitzada: Enginyeria de Computadors*
- *Treball tutoritzat per: Ana Cortés Fité*
- *Curs 2022/23*

sense el planificador, vaig donar per correcte els casos utilitzant línies com a punts d'inici.

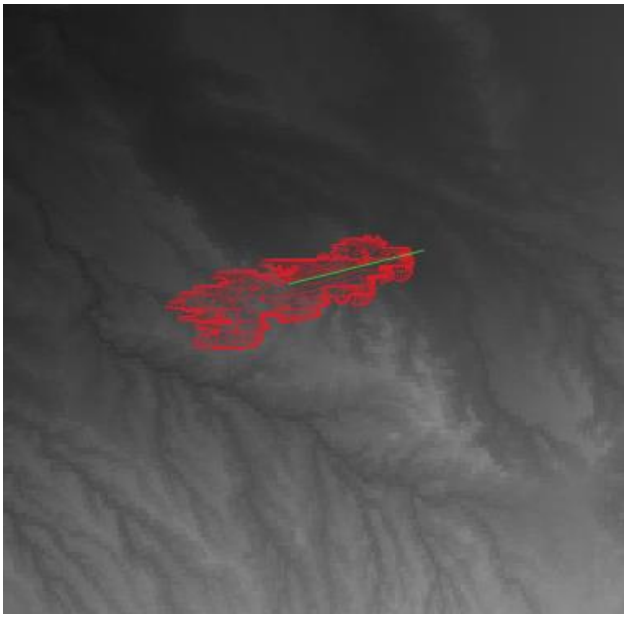

Figura 9. Simulació sobre línies

7.3 PERÍMETRE PUNT

Finalment, però no el menys important, tenim els perímetres de tipus punt. Són un inici d'incendi que rarament podrem veure en una simulació d'un cas real, ja que normalment els incendis que se simulen parteixen d'una petita regió i no d'un punt en el mapa, tot i que es pot fer servir per simular incendis amb dos punts d'ignició diferents que acaben sent un incendi únic. Aquest són els únics tipus de perímetre que no funcionen, el qual el vaig detectar en una fase prèvia a la fase de validació. Això és pel fet que inicialment ni tan sols generava el *lcp*, a causa del problema en la funció de càlcul de les dimensions per retallar els *Rasters*. Aquesta funció no tenia en compte els casos on la distància màxima era zero, el que causava un error en el procés per generar els inputs. Per tal de resoldre això, vaig modificar la funció per retallar *lcps* d'uns quince kilòmetres en els casos on la distància màxima del perímetre inicial era zero. Realitzant aquesta correcció sí que genera els *lcps* tal com es pot veure a la figura 10, però no propaga l'incendi el que em va fer donar per fallit aquest cas de prova.

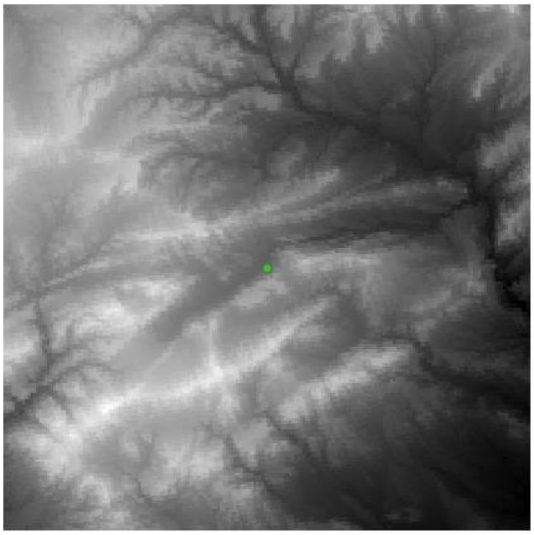

Figura 10. Simulació sobre punts

7.4 VALIDACIÓ CASOS TOP 10

Per poder garantir la qualitat dels resultats en el nou entorn, era necessari executar algun dels casos reals recopilats pel departament, ja que això ens asseguraria la correctesa del planificador i els seus resultats.[10] Entre els deu casos que té el departament, vaig triar un cas emblemàtic: l'incendi de la Jonquera. Va ser un incendi produït el vint-i-dos de juliol del 2012, el qual estava entre els deu pitjors incendis d'Espanya del segle XXI. Va ser un incendi provocat per unes burilles de tabac i va suposar gairebé deu mil cinc-centes hectàrees cremades, fent que fos necessari desallotjaments massius i provocant la mort de dues persones.

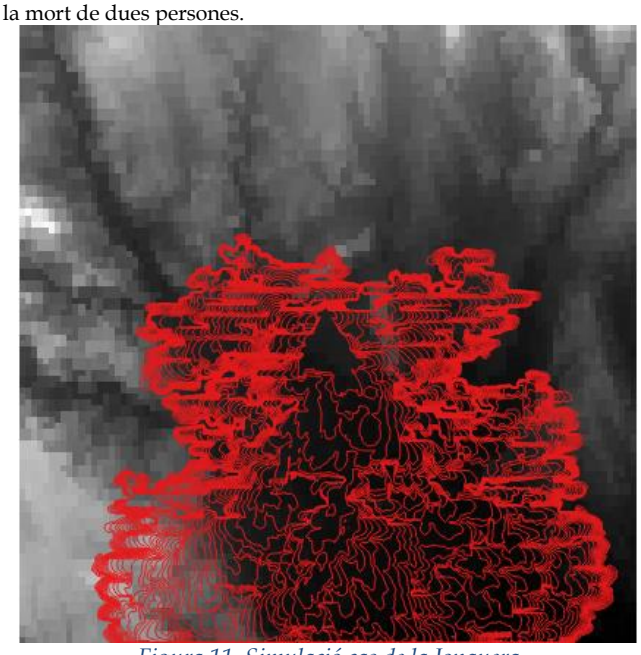

Figura 11. Simulació cas de la Jonquera

Com es pot apreciar a la figura 11, partint del perímetre inicial aproximat del cas de la Jonquera, el planificador aconsegueix generar el *lcp* i extreure el resultat de la simulació. [11] Si comparem aquest resultat amb l'informe dels bombers, tal com es veu a la figura 12, el resultat de la propagació de l'incendi es realitza en la mateixa direcció. Cal tenir en compte que en l'incendi real hi ha diferents factors que no es produeixen en la simulació, com l'impacte dels bombers, les carretes i els possibles tallafocs.

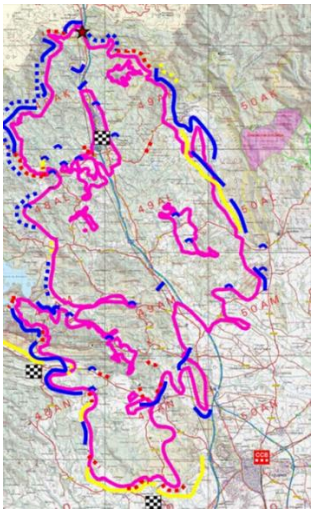

Figura 12. Informe Bombers Jonquera

8 CONCLUSIONS

Tal com marca el títol del projecte, aquest és un treball basat en la millora i la validació del planificador per l'execució d'incendis forestals del departament ACSO. En el qual he pogut localitzar diferents debilitats com:

- Dificultat a l'hora de desplegar l'entorn i compartir-lo
- Introduir i extreure de l'entorn les dades.
- Problemes de versions entre les diferents eines utilitzades en la implementació.

Tot i no poder abordar altres aspectes també importants sobre el planificador, com podria ser afegir un sistema de validacions per evitar execucions fallides abans d'hora, he aportat una millora que, des del meu punt de vista, no només ha solucionat algunes de les debilitats trobades sinó que també ha aportat una millora substancial podent fer més escalable la implementació i més fàcil aportar noves millores. El nou entorn basat en Docker fa més robusta la implementació i aporta millores substancials a l'hora de desplegar, compartir i manipular l'entorn de treball. Fent que l'usuari pugui consumir el planificador des de qualsevol *host* sense necessitat d'accedir l'entorn.

Per altra banda després de la validació formal del planificador, puc dir que és una implementació funcional, la qual pot aportar valor al món de la simulació d'incendis forestals. Cal remarcar que és necessari corregir la funció utilitzada per retallar els mapes per evitar problemes de memòria i assegurar que tots els casos possibles de simulacions obtindran un resultat correcte.

8.1 LÍNIES FUTURES

Un cop finalitzat el treball he pogut extreure moltes idees i punts de millorar que podrien aportar valor tant al nivell de qualitat dels resultats de la simulació d'incendis com al procés de planificació i desplegament de l'entorn de simulació.

Com que ara ja tenim el planificador i el simulador en un entorn Docker, és més escalable tota la implementació. Es poden afegir més contenidors que proporcionin millors prestacions o bé ajudin a la preparació de les dades inicials que consumeix el planificador. Com a línies futures he pogut detectar els següents punts:

- **Actualització dels Mapes:** Com que el terreny és canviant un dels punts de millora que podria aportar més qualitat a les simulacions, seria l'actualització dels mapes. On podríem tenir un contenidor que tingués com a objectiu descarregar les noves versions dels mapes quan aquests estiguessin actualitzats.
- **Actualització de la Meteorologia:** La meteorologia és diferent cada dia i aquest és un problema clar a l'hora de simular incendis reals, on la prioritat és realitzar execucions ràpidament i amb la màxima precisió. Un altre punt de millora podria ser la generació dels fitxers d'entrada de la meteorologia, fent així més precises les simulacions i més ràpida la captació de les dades.
- **Execucions paral·leles:** Com que el Farsite admet diferents configuracions a l'hora d'executar les simulacions, es podrien llençar diferents execucions de la simulació pel mateix incendi fent variar algun dels paràmetres de la configuració, com podria ser el model de vegetació o aplicar altres paràmetres a la simulació. Això ens permetria poder comparar ràpidament resultats que poguessin diferir depenent d'algun paràmetre d'entrada.
- **Millora de l'algoritme per retallar mapes:** Tal com he pogut veure, un dels punts febles del planificador, és a l'hora de retallar els mapes depenent del perímetre, aquí es podria dur a terme un control de la mida dels punts d'ignició, per tal d'evitar errors i reduir el temps d'execucions errònies.

Tots aquests punts de millora, des de la meva perspectiva, podrien ser línies futures a abordar, per tal d'aconseguir complaure l'objectiu principal del simulador que, al cap i a la fi, és ajudar a prevenir i mitigar incendis forestals.

9 AGRAÏMENTS

Expresso els meus agraïments a tot l'equip de GFIRE pel suport i l'ajuda proporcionada durant el projecte. Especials agraïments a Carles Carilló que m'ha resolt tots els dubtes pertinents al simulador Farsite i m'ha donat suport durant tot el projecte. També vull agrair a la meva tutora Ana Cortés Fité l'oportunitat per poder realitzar aquest treball amb el departament ACSO i l'equip de GFIRE

10 BIBLIOGRAFIA

[1] Overview of FARSITE process [Online]. Available: http://fire.org/downloads/farsite/WebHelp/usersguide/ug2_overview_of_farsite_process.html
[2] Creating ASCII Input Files [Online]. Available: http://fire.org/downloads/farsite/WebHelp/usersguide/ug7_building_ascii_files.html
[3] Weather File (.WTR) [Online]. Available: http://fire.org/downloads/farsite/WebHelp/referenceguide/pop_ups/pu_weather_file.html
[4] Wind File (.WND) [Online]. Available: http://fire.org/downloads/farsite/WebHelp/referenceguide/pop_ups/pu_wind_file.html
[5] Generate Landscape File [Online]. Available: http://fire.org/downloads/farsite/WebHelp/referenceguide/input_menu/landscape_utilities/lu_generate_landscape.html
[6] Review of WildFair Simulators [Online]. Available: https://www.researchgate.net/publication/224226185_A_Comparative_Review_on_Wildfire_Simulators
[7] Luigi Overview [Online]. Available: https://alvaromonsalve.com/2018/03/14/luigi/
[8] Docker Compose [Online]. Available: https://docs.docker.com/compose/
[9] Luigi Documentation [Online]. Available: https://luigi.readthedocs.io/en/stable/
[10] Diari Girona- Incendi la Jonquera [Online]. Available: https://www.diaridegirona.cat/alt-emporda/2019/05/30/l-incendi-2012-jonquera-els-48821681.html
[11] Informe Jonquera v20120823 [Online]. Available: http://www.cadiretesmontbarbat.org/documentacio/cursos-bombers/20120722_i_reg_jonquera_v201208231.pdf

---

- *E-mail de contacte: 1491578@uab.cat*
- *Menció realitzada: Enginyeria de Computadors*
- *Treball tutoritzat per: Ana Cortés Fité*
- *Curs 2022/23*

APÈNDIX

CODI

DOCKER COMPOSE

A continuació deixo el codi complert del fitxer docker-compose.ylm:

```
version: "3.3"

services:
 luigi-server:
  container_name: luigi-server
  build:
   context: ./
   dockerfile: Dockerfile_luigi-server
  image:
   luigi/luigi-server
  privileged: true
  ports:
   - "8082:8082"
  restart: always

 luigi-client:
  container_name: luigi-client
  build:
   context: ./
   dockerfile: Dockerfile_luigi-client
  image:
   luigi/luigi-client
  volumes:
   -
${PATH_PERIMETER}:/opt/app/FARSITE5/luigi_execution_input
_files:rw
   - ${PATH_ICGC}:/opt/app/Downloads/ICGC:rw
   - ${PATH_COPERNICUS}:/opt/app/Downloads/Copernicus:rw
   - ${PATH_USGS}:/opt/app/Downloads/USGS:rw
   -
${PATH_SpatialReference}:/opt/app/Downloads/SpatialReference:
rw
  privileged: true
```

DOCKER FILE

```
#Lugi docker orquestrator container
FROM osgeo/gdal:ubuntu-full-3.2.2

MAINTAINER Carles Costas <1491578@uab.cat>

# Install required packages
RUN apt-get update -q && apt-get upgrade -yq
RUN apt-get install -y build-essential libaec-dev zlib1g-dev libcurl4-
openssl-dev libboost-dev zip unzip gfortran gcc g++ python3-pip


RUN pip3 install luigi
RUN pip3 install pygdal==3.2.2.10
RUN pip3 install pandas
RUN pip3 install requests
RUN pip3 install fiona
RUN pip3 install geopandas


# crearemos un directorio donde instalaremos nuestra aplicacion
RUN mkdir -p /opt/app
# Indicaremos que el directorio de trabajo sera el de la aplicacion
WORKDIR /opt/app

COPY Luigi_env.zip /opt/app

RUN unzip Luigi_env.zip

RUN cd /opt/app/FARSITE5/firemod-master/ && make clean &&
make


RUN mkdir -p /opt/app/Downloads/
RUN mkdir -p /opt/app/Downloads/SpatialReference/
RUN mkdir -p /opt/app/Downloads/Copernicus/
RUN mkdir -p /opt/app/Downloads/USGS/
RUN mkdir -p /opt/app/Downloads/ICGC/

EXPOSE 8082

RUN mkdir -p /etc/luigi/
COPY luigi.cfg /etc/luigi

COPY execute.sh /opt/app
ENTRYPOINT ["./execute.sh"]
```