

**LAPORAN TUGAS PEMROGRAMAN II**  
**PEMODELAN DAN SIMULASI KEBAKARAN HUTAN**



DISUSUN OLEH:

- |                            |          |
|----------------------------|----------|
| 1. Yohanes Yordan Tjahjono | 10120023 |
| 2. Samuel Christo          | 10120029 |
| 3. Muhammad Dira Kurnia    | 10120039 |
| 4. Pamella Cathryn         | 10820033 |
| 5. Christian August        | 10820047 |

**MA2151 Simulasi dan Komputasi Matematika**  
**Fakultas Matematika dan Ilmu Pengetahuan Alam**  
**Institut Teknologi Bandung**  
Jl. Ganesha no. 10 Bandung, Indonesia, 40132

## **A. ANALISA MASALAH**

Kebakaran ialah nyala api baik kecil maupun besar pada tempat, situasi dan waktu yang tidak dikehendaki yang bersifat merugikan dan pada umumnya sulit untuk dikendalikan. Kebakaran dapat terjadi dimana saja, baik di daerah perkotaan ataupun daerah hutan. Kebakaran hutan kerap terjadi tidak hanya di Indonesia tetapi di negara lain juga. Kebakaran hutan dapat terjadi oleh banyak hal, seperti cuaca yang panas, ulah manusia, kecelakaan, atau faktor alam lainnya. Namun dalam kasus kebakaran hutan yang menjadi menarik untuk dianalisis merupakan penyebaran api dari satu pohon ke pohon lainnya. Kita dapat memodelkan penyebaran api dari pohon ke pohon hingga mendapatkan waktu yang dibutuhkan jika hutan terbakar dan persentase hutan yang sudah terbakar.

### **1. Analisis Masalah**

Ingin diketahui rata-rata persentase hutan terbakar dan waktu yang dibutuhkan jika hutan terbakar seluruhnya melalui simulasi mengenai model penyebaran api di suatu daerah hutan yang dilambangkan dengan matriks  $17 \times 17$  dengan *burnProbability* dari 10%, 20%, ..., 90% yang masing-masing dilakukan sebanyak 10 kali pengulangan.

### **2. Objektif Masalah**

Memperoleh hasil data, rata-rata yang diinginkan, dan kurva yang menggambarkan data dari simulasi yang dibuat serta menentukan interpretasinya.

### **3. Klasifikasi Masalah**

Berdasarkan masalah yang diberikan maka secara nalar masalah ini termasuk ke dalam masalah dinamis karena masalah berubah seiring sebuah parameter berubah, yaitu waktu.

## B. RANCANGAN MODEL

### 1. Pengumpulan Data

Jika kita melihat kasus, maka daerah dibatasi dalam matriks  $17 \times 17$ .

Misalkan daerah ini kita sebut dengan matriks hutan, maka input awalnya adalah

- Hutan  $\{[17][17], [9][9], [1][1]\}$  = Pohon terbakar
- $Burnprobability = \{[0.1], [0.2], [0.3], [0.4], [0.5], [0.6], [0.7], [0.8], [0.9]\}$
- $Iteration = 10$  ;  $\forall burnprobability$

### 2. Penyederhanaan Asumsi

Data yang diasumsikan meliputi,

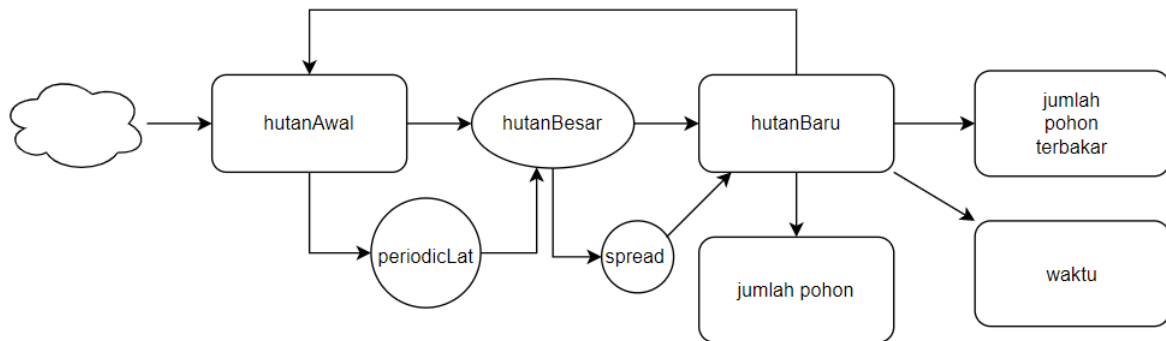
- Probability terjadi petir dan pohon tumbuh = 0
- Ketetanggaan menggunakan von Neumann
- Kondisi batas menggunakan *Periodic Boundary Condition*




### 3. Penentuan Variabel dan Satuannya

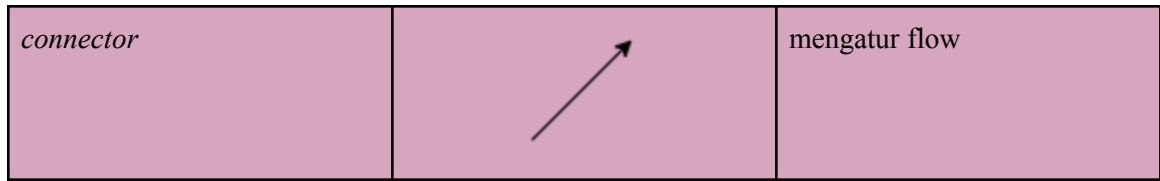
- EMPTY = variabel yang digunakan untuk mewakilkan sel kosong
- TREE = variabel yang digunakan untuk mewakilkan sel yang berisi pohon
- BURNING = variabel yang digunakan untuk mewakilkan sel yang berisi pohon yang terbakar
- burnProbability = variabel yang menampung peluang pohon terbakar
- time = variabel banyaknya iterasi
- hutan = Matriks yang menampung sel sel yang melambangkan daerah hutan awal
- site = Variabel sel yang belum dimodifikasi/mengalami perubahan
- newSite = Variabel sel yang telah dimodifikasi/mengalami perubahan
- N, E, S, W = variabel yang menampung sel tetangga sesuai arah angin
- X = variabel yang menampung bilangan random untuk menilai peluang kebakaran

- latNS = Matriks yang menambahkan ghost cell ke samping kanan dan kiri matriks
- newLat = Matriks yang menambahkan ghost cell ke sisi atas dan sisi bawah
- Grids = Matriks tiga dimensi yang menampung parameter waktu (time steps) dan matriks hutan
- hutanBesar = Matriks hutan yang sudah diperluas menggunakan Periodic Boundary Condition

#### 4. Menentukan Relasi Antar Variabel dan Submodel



Nama	Diagram	Makna
<i>stock/box variable</i>		sesuatu yang dapat berkurang/bertambah
<i>converter/formula</i>		sesuatu yang mengubah input menjadi output
<i>flow/rate</i>		aktivitas yang mengubah besar stock dan merepresentasikan turunan



## 5. Menentukan Formula dan Fungsi

Formula dan Fungsi :

- $x = np.random.rand(1)$
- $matrix = np.zeros((17,17))$
- $presentase = (100 * variabel \text{'counter'}) / 17 * 17 (\%)$

## C. SOLUSI

Model yang sebelumnya dirumuskan akan diimplementasikan ke dalam pseudocode dengan menggunakan bahasa Python.

```
#import library
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.animation as animation
import seaborn as sns
```

Pertama-tama, kita import dahulu library-library yang akan digunakan, yaitu:

1. NumPy. Library NumPy adalah library Python yang fokus pada scientific computing. NumPy memiliki kemampuan untuk membentuk objek N-dimensional array, yang mirip dengan list pada Python. Library ini kami gunakan untuk merandom suatu angka dan membuat matriks yang menyimpan informasi data.
2. Matplotlib. Library ini adalah library Python yang berfokus pada visualisasi data seperti membuat plot grafik. Pada pseudocode ini, kami menggunakan matplotlib.pyplot untuk membuat frame gambar dan matplotlib.animation untuk membuat animasi simulasi.
3. Seaborn. Library Seaborn adalah library yang biasanya digunakan untuk membuat grafik dan statistik. Pada pseudocode ini, kami menggunakannya untuk membuat heatmap untuk keperluan animasi simulasi.

```
#inisialisasi variabel
EMPTY = 0
TREE = 1
BURNING = 2
burnProbability = #10%-90%
time = 30
```

Lalu, kami melakukan inisialisasi untuk variabel-variabel yang akan digunakan nantinya. Variabel-variabel tersebut meliputi:

- EMPTY: Variabel yang menunjukkan sel tidak ditempati pohon. Variabel ini bernilai 0.
- TREE: Variabel yang menunjukkan sel ditempati pohon yang tidak terbakar. Variabel ini bernilai 1.
- BURNING: Variabel yang menunjukkan sel ditempati pohon yang terbakar. Variabel ini bernilai 2.
- burnProbability: Variabel yang menyatakan kemungkinan pohon yang bertetangga dengan pohon yang terbakar akan terbakar. Pengambilan data akan dilakukan dengan memvariasikan variabel ini dari 10%, 20%, 30%, ..., 90%.
- time: Variabel ini menyatakan banyak iterasi (time steps) untuk simulasi kebakaran ini.

```
#Inisialisasi kondisi hutan
def hutanAwal():
    hutan = np.zeros((17, 17))
    for i in range(17):
        for j in range(17):
            if (i==0 and j==0) or (i==16 and j==16) or (i==8 and j==8):
                hutan[i][j] = BURNING
            else:
                hutan[i][j] = TREE
    return hutan
```

Selanjutnya, kami membuat fungsi yang mengembalikan keadaan awal hutan, yaitu fungsi hutanAwal. Hutan ini terdiri dari sel berukuran 17x17 dengan setiap selnya mengandung 1 buah pohon. Pada sel paling tengah, sel ujung kiri atas, dan sel ujung kanan bawah, pohon sedang terbakar.

```
#spreading function (ketetanggaan von Neumann)
def spread(site, N, E, S, W):
    if (site == EMPTY):
        newSite = EMPTY
    else:
        if (site == BURNING):
```

```

        newSite = EMPTY
    else:
        if (N == BURNING or E == BURNING or S == BURNING or W ==
BURNING):
            x=np.random.rand(1)
            if x < burnProbability:
                newSite = BURNING
            else:
                newSite = TREE
        else:
            newSite = TREE
    return newSite

```

Lalu, kami membuat fungsi yang menunjukkan proses penyebaran api, yaitu fungsi spread. Masukan dari fungsi ini adalah suatu sel, sebelah atas dari sel tersebut, sebelah kanan dari sel tersebut, sebelah bawah dari sel tersebut, dan sebelah kiri dari sel tersebut. Jika sel tersebut tidak ditempati pohon atau kosong, maka pada sel yang baru akan tetap kosong. Jika sel tersebut ditempati oleh pohon dan sedang terbakar, maka pada sel yang baru menjadi kosong. Jika sel tersebut ditempati oleh pohon, sedang tidak terbakar, dan memiliki tetangga yang sedang terbakar (entah sebelah atas, sebelah kanan, sebelah kiri, atau sebelah bawah), maka akan dirandom sebuah angka pada interval [0,1]. Jika didapatkan angka kurang dari burnProbability, maka pada sel yang baru akan terbakar dan jika didapatkan angka lebih dari atau sama dengan burnProbability, maka pada sel yang baru, pohon tetap tidak terbakar. Terakhir, jika sel tersebut ditempati oleh pohon yang sedang tidak terbakar dan tidak mempunyai tetangga pohon yang sedang terbakar, maka pada sel yang baru akan tetap menjadi pohon yang tidak terbakar. Keluaran dari fungsi ini adalah sel baru setelah melewati kemungkinan-kemungkinan kondisi di atas.

```

# menambah 1 baris/kolom ke setiap arah pada grid (periodic boundary
condition)
def periodicLat(lat):
    latNS = np.row_stack((lat[-1], lat, lat[0]))
    return np.column_stack((latNS[:, -1], latNS, latNS[:, 0]))

```



Setelah itu, kami membuat fungsi `periodicLat`. Masukan dari fungsi ini adalah sebuah matriks berukuran  $n \times n$ . Fungsi ini dibuat untuk menambahkan sebuah baris atau kolom **sementara** pada semua sisi pada matriks dengan **periodic boundary condition**. Periodic boundary condition berarti baris atau kolom sementara yang dibuat pada suatu sisi merupakan salinan dari baris atau kolom pada sisi seberangnya. Sehingga, keluaran dari fungsi ini adalah sebuah matriks yang berukuran  $(n+2) \times (n+2)$ .

```
# hutan baru
def hutanBaru(lat):
    n = 17
    newLat = np.zeros((n, n))

    for i in range(1, n + 1):
        for j in range(1, n + 1):
            site = lat[i][j]
            N = lat[i - 1][j]
            E = lat[i][j + 1]
            S = lat[i + 1][j]
            W = lat[i][j - 1]
            newLat[i - 1][j - 1] = spread(site, N, E, S, W)
    return newLat
```

Selanjutnya, kami membuat fungsi bernama `hutanBaru`. Fungsi ini menerima masukan berupa sebuah matriks berukuran  $(n+2) \times (n+2)$ . Pada fungsi ini, dibuat sebuah matriks baru berukuran  $n \times n$ . Sel-sel pada matriks baru merupakan sel-sel pada matriks masukan yang telah melewati fungsi `spread` dan keluaran dari fungsi ini adalah matriks baru tersebut.

```
# memulai simulasi
def play(time):
    hutan = hutanAwal()

    grids = np.zeros((time+1, 17, 17))
    grids[0][:][:] = hutan #untuk t=0, hutan=hutanAwal()
    for i in range(1, time+1): #untuk t>0, hutan=hutanBaru
        hutanBesar = periodicLat(hutan)
        hutan = hutanBaru(hutanBesar)
```

```
    grids[i][:][:] = hutan
    return grids
```

Lalu, kami membuat sebuah fungsi untuk mempersiapkan animasi dan menamakannya fungsi play. Fungsi ini menerima masukan berupa variabel time sebagai iterasi pada simulasi. Pada fungsi ini, dibuat sebuah array tiga dimensi. Parameter pertama pada array menunjukkan iterasi pada simulasi dan parameter lainnya menunjukkan matriks nxn yang akan ditampilkan per iterasi. Pada iterasi pertama, yaitu time=0, hutan merupakan kondisi awal hutan yang sudah diinisialisasi pada fungsi hutanAwal. Untuk iterasi selanjutnya, hutan merupakan keadaan pada iterasi sebelumnya yang sudah melewati fungsi periodicLat dan fungsi hutanBaru. Sehingga, keluaran dari fungsi play ini adalah array tiga dimensi berisikan frame matriks yang akan ditampilkan per iterasinya.

```
final=play(time)

# Menyiapkan figure, heatmap
fig = plt.figure()

# Fungsi init
def init():
    plt.clf()
    return None

# Fungsi iterasi animasi
def animate(i):
    plt.clf()
    ax = sns.heatmap(final[i-1],
                      vmin=0,
                      center=1,
                      vmax=2,
                      cmap="Reds",
                      square=True,
                      xticklabels=False,
                      yticklabels=False,
                      )
```

```

        ax.set_title(f"burnProbability={int(100*burnProbability)}% pada
t={i-1}")
        return None

# Fungsi init
anim = animation.FuncAnimation(fig,
                                animate,
                                frames=range(1, time+1, 1),
                                blit=False,
                                interval=250,
                                init_func=init)

# Kode tambahan jika dijalankan di colab/jupyter
from matplotlib import rc
from IPython.display import HTML
rc('animation', html='jshtml')
anim

```

Selanjutnya, kami membuat sebuah variabel baru bernama final yang memuat fungsi play dengan masukan variabel time. Setelah itu, kami mulai menganimasikannya dengan heatmap menggunakan library Seaborn dan menjalankan animasi tersebut menggunakan library matplotlib. Sehingga, diperoleh sebuah animasi penyebaran kebakaran hutan yang berukuran 17x17 dengan iterasi sebanyak 30 kali.

Keterangan warna:

- Merah tua : Sel yang ditempati pohon yang sedang terbakar
- Jingga : Sel yang ditempati pohon yang tidak terbakar
- Putih : Sel yang tidak ditempati pohon

```

#kesimpulan hasil simulasi

counter=0 #menampung jumlah cell yang empty pada t=time
for i in range(17):
    for j in range(17):
        if final[-1][i][j]==0:
            counter+=1

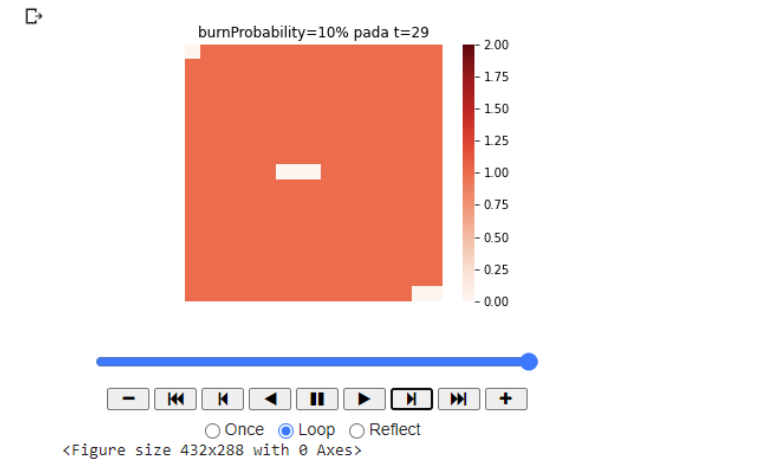
```

```
print(f"Persentase hutan yang terbakar =  
{round(100*counter/(17*17),2)}%")  
  
checker=0  
for k in range(time):  
    for i in range(17):  
        for j in range(17):  
            if final[k][i][j]==2:  
                checker+=1  
        if checker==counter:  
            break  
print(f"Waktu yang dibutuhkan sampai hutan habis terbakar adalah  
{k+1}")
```

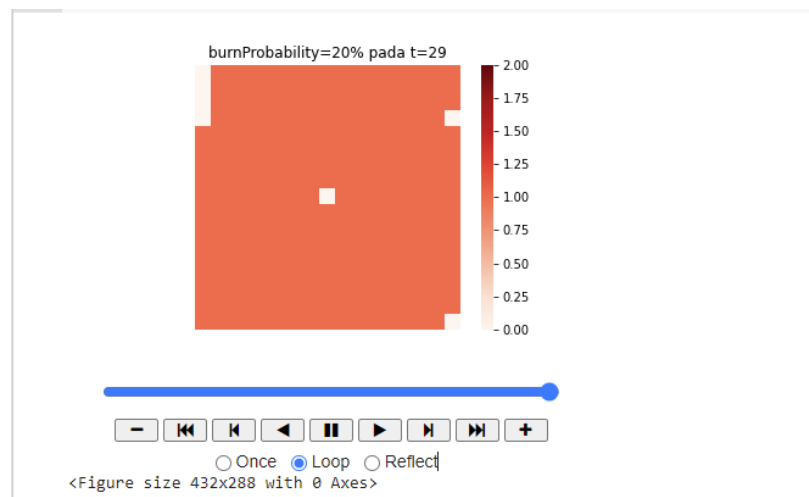
Terakhir, kami membuat kode tambahan untuk mempermudah pengambilan data. Output dari kode ini adalah informasi persentase hutan yang terbakar dan waktu yang dibutuhkan sampai hutan tersebut habis terbakar.

## D. HASIL DAN KESIMPULAN

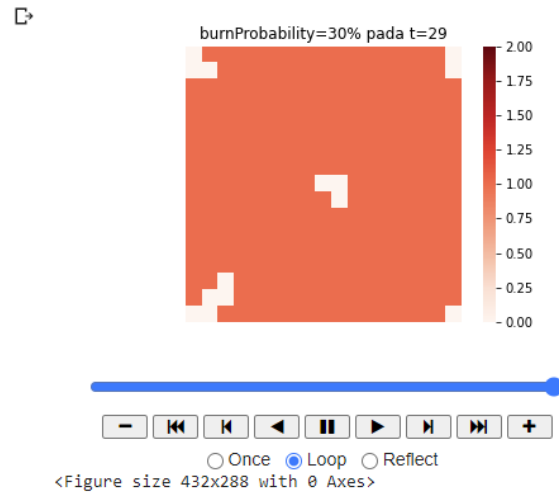
Dari pemodelan kebakaran hutan yang sudah diperoleh, diperoleh output dari program dalam bentuk Heat-Map, seperti berikut:



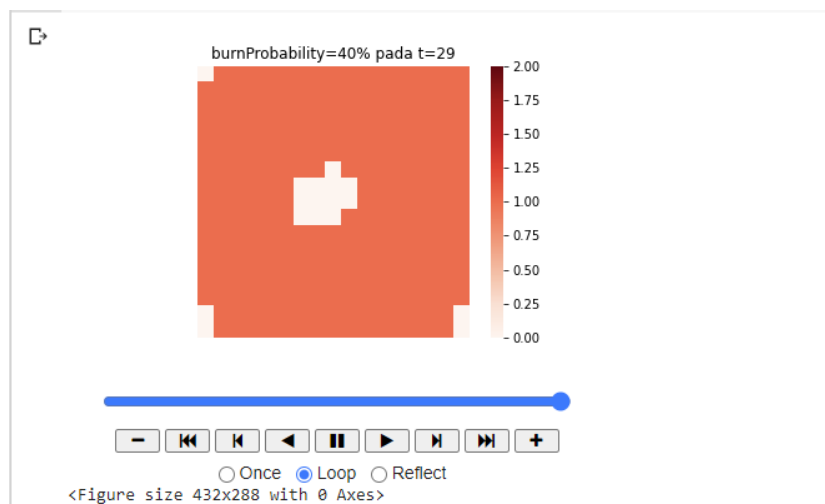
Gambar 1 Heatmap dari hutan dengan burn probability 10 %



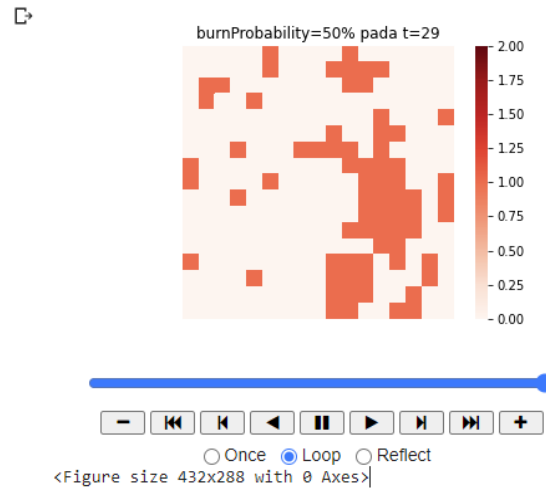
Gambar 2 Heatmap dari hutan dengan burn probability 20 %



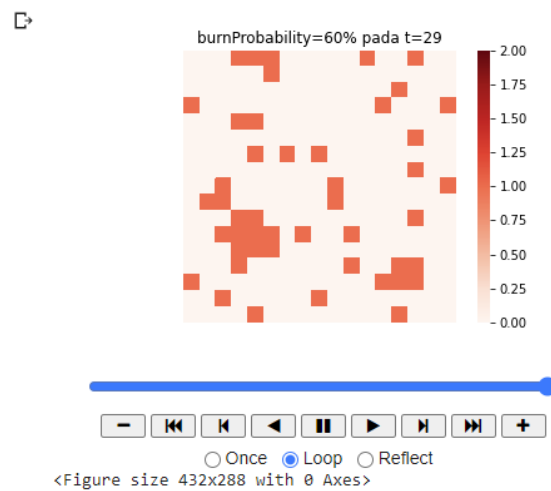
Gambar 3 Heatmap dari hutan dengan burn probability 30 %



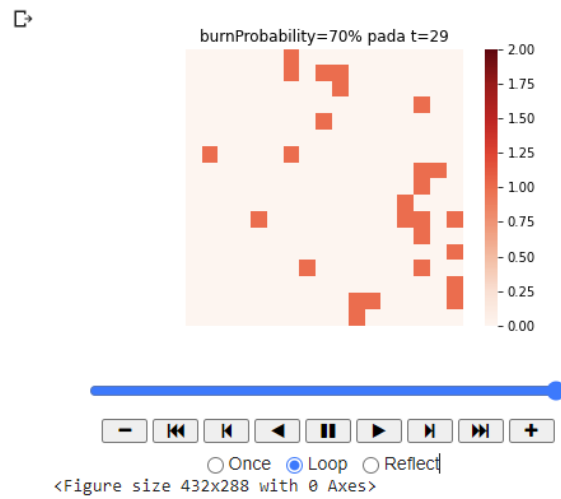
Gambar 4 Heatmap dari hutan dengan burn probability 40 %



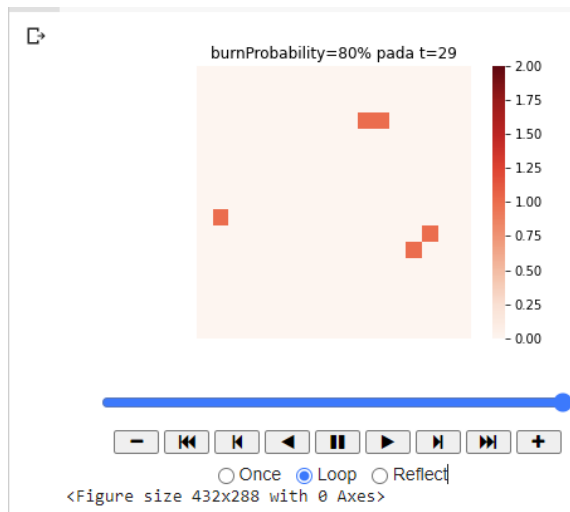
Gambar 5 Heatmap dari hutan dengan burn probability 50 %



Gambar 6 Heatmap dari hutan dengan burn probability 60 %

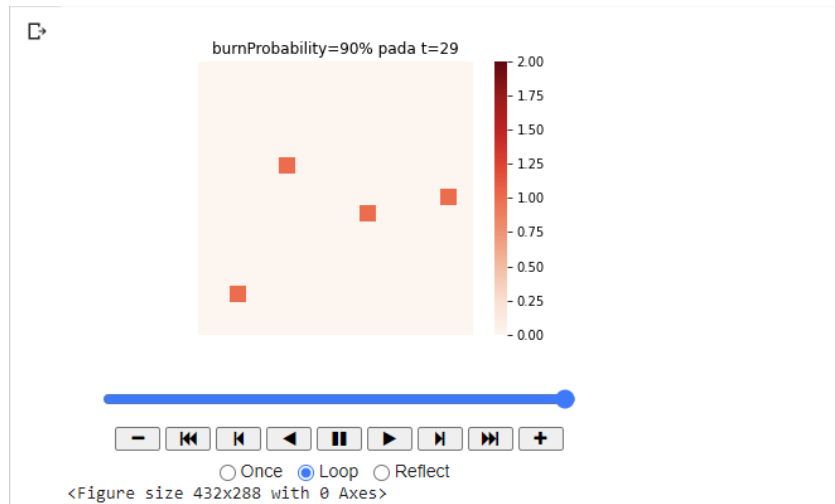


Gambar 7 Heatmap dari hutan dengan burn probability 70 %



Gambar 8 Heatmap dari hutan dengan burn probability 80 %





Gambar 9 Heatmap dari hutan dengan burn probability 90 %

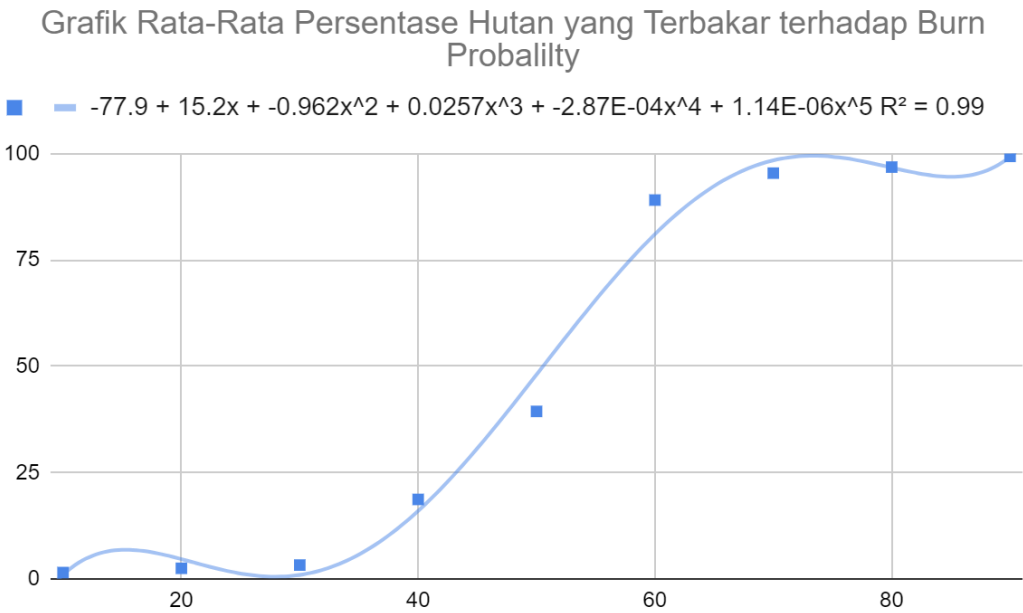
Dari simulasi kebakaran hutan tersebut, dilakukan pengambilan data dan *data fitting*. Sehingga, diperoleh data dan kesimpulan sebagai berikut:

<i>BurnProbability (%)</i>	Rata- Rata Hutan yang Terbakar (%)									
	1	2	3	4	5	6	7	8	9	10
10%	1.38	1.38	1.38	1.38	1.04	1.04	1.04	3.11	1.04	2.08
20%	3.46	2.42	2.77	1.38	2.42	2.77	1.04	1.73	2.08	4.84
30%	2.42	2.77	3.46	3.46	2.42	1.04	3.46	5.19	4.84	3.46
40%	14.19	10.03	42.91	20.07	5.19	29.41	33.22	1.38	11.42	19.03
50%	39.45	50.17	73.7	2.08	20.07	24.91	39.45	51.9	58.48	33.56
60%	90.66	90.66	80.28	92.04	88.24	87.2	90.66	92.04	92.04	86.85
70%	96.54	94.12	94.12	95.5	95.16	97.23	95.16	95.5	96.19	94.46
80%	95.85	96.54	96.54	97.23	98.62	97.58	98.27	95.16	95.5	96.89
90%	98.62	98.96	99.65	98.96	100	99.31	99.65	99.65	98.96	99.65

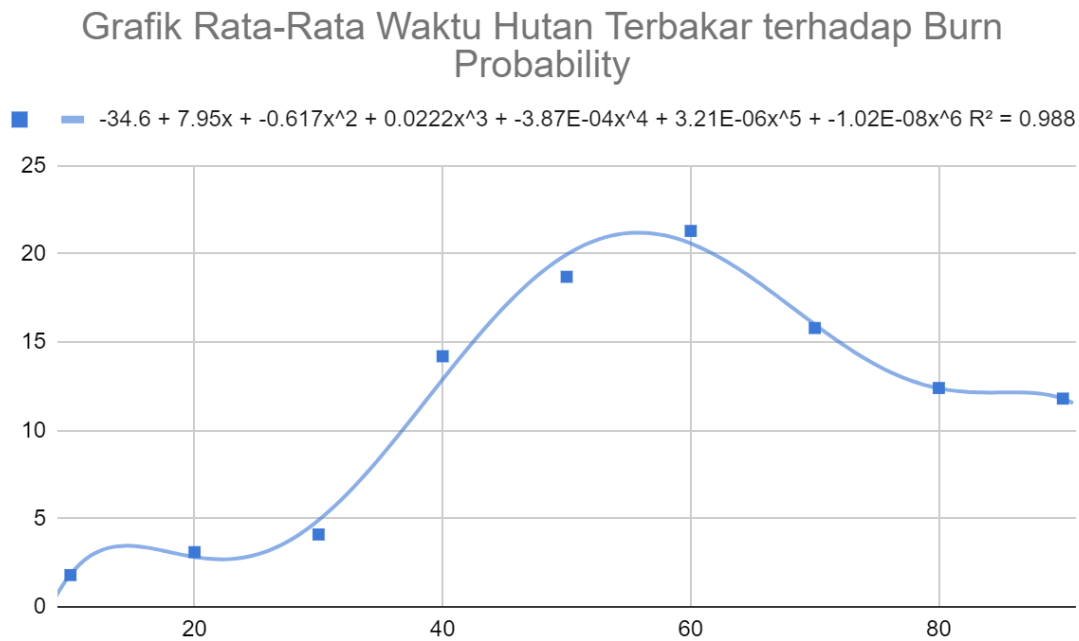
Tabel 1 Perolehan Persentase Hutan yang Terbakar

BurnProbability (%)	Waktu yang dibutuhkan untuk hutan terbakar (s)									
	1	2	3	4	5	6	7	8	9	10
10%	2	2	2	2	1	1	1	3	1	3
20%	4	4	3	2	4	4	1	2	3	4
30%	3	4	6	5	3	1	4	6	4	5
40%	11	9	19	20	6	21	23	2	12	19
50%	24	18	19	2	15	26	17	18	24	24
60%	20	24	17	22	19	20	30	22	20	19
70%	15	15	15	15	15	20	14	17	17	15
80%	12	12	12	12	12	13	13	13	12	13
90%	12	10	12	12	12	11	11	11	15	12

Tabel 2 Perolehan Waktu yang Diperlukan untuk Seluruh Hutan Habis Terbakar



Grafik 1 Grafik Rata-Rata Persentase Hutan yang Terbakar terhadap Burn Probability



Grafik 2 Grafik Rata-Rata Waktu Hutan Terbakar terhadap Burn Probability

#### Kesimpulan:

1. Semakin besar burnProbability, semakin besar persentase hutan yang terbakar.
2. Waktu yang dibutuhkan sampai hutan selesai terbakar terus meningkat sampai ke burnProbability 50%, mencapai puncaknya pada sekitar burnProbability 50%-60%, dan menurun pada burnProbability setelahnya.