Here's a mathematical description of the provided R functions for simulating fire spread, structured like the methods section of a scientific paper.

## 1. Fire Spread Model

This section describes a cellular automaton model designed to simulate the spatial and temporal progression of fire spread across a discretized landscape. The model considers an initial ignition point and propagates fire based on intrinsic cell properties and a neighborhood influence mechanism.

### 1.1 Model Domain and Discretization

The simulation takes place on a regular grid G={g1​,g2​,…,gN​}, where each element gi​ represents a square geographical cell. Each cell gi​ is characterized by a set of attributes, including:

* A unique identifier, idi​.
* A speed attribute, Si​, representing the inherent rate at which fire can propagate through cell i. In the model, this is derived from a biomass or similar scenario-dependent variable.
* An attraction attribute, Ai​, representing the cell's susceptibility or contribution to fire spread, derived from a connectivity variable.

The spatial resolution of the grid is defined by the cell\_size, denoted as L (in meters). The temporal resolution of the simulation is defined by time\_step, denoted as Δt (in minutes).

### 1.2 Fire Advancement Level

Each cell gi​ at a given time step t possesses an advancement\_level, denoted as Bi​(t)∈[0,1]. An advancement\_level of 1 indicates the cell is fully burning. Initially, all cells are set to Bi​(0)=0, except for a designated ignition\_cell, gign​, for which Bign​(0)=1.

### 1.3 Fire Spread Mechanism

Fire spreads from a burning cell to its neighbors based on a calculated burn\_pressure. The core spread mechanism is governed by the spread\_fun function:

$$\text{spread\_fun}(B\_{neighbor}(t-1), S\_{neighbor}', A\_{neighbor}) = B\_{neighbor}(t-1) + (S\_{neighbor}' \times A\_{neighbor})$$

where:

* Bneighbor​(t−1) is the advancement\_level of the neighboring cell at the previous time step. This term implies that fire spread builds upon any pre-existing advancement in the neighbor.
* Sneighbor′​ is the *adjusted speed* of the neighboring cell, which accounts for the time\_step and the distance to the burning origin\_cell.
* Aneighbor​ is the attraction value of the neighboring cell.

The burn\_pressure for a potential neighbor cell gj​ due to an origin\_cell gk​ is calculated as:

P(gj​,gk​,t−1)=Bj​(t−1)+((D(gj​,gk​)×LSj​​)×Δt×Aj​)

where:

* D(gj​,gk​) is a dimensionless distance factor between the origin\_cell gk​ and the neighbor cell gj​.
  + For **Rook neighbors** (sharing an edge), D(gj​,gk​)=1.
  + For **Queen neighbors** (sharing a corner), D(gj​,gk​)=2​.

### 1.4 Neighborhood Definition

For each cell gi​, a pre-processed list of neighboring cells is provided, differentiated by their spatial relationship:

* NR​(gi​): Set of Rook neighbors of cell gi​.
* NQ​(gi​): Set of Queen (exclusive) neighbors of cell gi​.

### 1.5 Simulation Dynamics

The simulation progresses iteratively over time\_horizon steps. At each time step t (from 1 to time\_horizon), the following operations are performed:

1. **Identify Burning Cells:** Let U(t−1)={gk​∈G∣Bk​(t−1)≥1} be the set of cells currently burning (i.e., with an advancement\_level of 1 or greater) at the previous time step.
2. **Carry-over Burning Status:** All cells in U(t−1) automatically remain burning in the current time step: Bk​(t)=1 for all gk​∈U(t−1).
3. Calculate Accumulated Burn Pressure for Neighbors:  
   A temporary vector, Pacc​(t), is initialized to zero for all cells. For each origin\_cell gk​∈U(t−1):
   * Rook Neighbors: For each potential Rook neighbor gj​∈NR​(gk​) such that gj​∈/U(t−1) (i.e., not already burning):  
     The burn\_pressure P(gj​,gk​,t−1) is calculated using D(gj​,gk​)=1.  
     The accumulated pressure for gj​ is updated as: Pacc​(t)j​=max(Pacc​(t)j​,P(gj​,gk​,t−1)).
   * Queen Neighbors: For each potential Queen neighbor gj​∈NQ​(gk​) such that gj​∈/U(t−1):  
     The burn\_pressure P(gj​,gk​,t−1) is calculated using D(gj​,gk​)=2​.  
     The accumulated pressure for gj​ is updated as: Pacc​(t)j​=max(Pacc​(t)j​,P(gj​,gk​,t−1)).
4. Update Advancement Levels: For all cells gj​ where Pacc​(t)j​>0:  
   The advancement\_level of cell gj​ at time t is updated: Bj​(t)=max(Bj​(t),Pacc​(t)j​).  
   This ensures that cells already burning (copied in step 2) maintain their advancement\_level of 1, and newly affected cells accumulate the maximum burn\_pressure they received from any origin\_cell.
5. **Cap Advancement Level:** For completeness, the advancement\_level of cells at the *previous* time step t−1 is capped at 1: Bi​(t−1)=min(1,Bi​(t−1)). This ensures that advancement\_level values used in the next iteration's Bneighbor​(t−1) input to spread\_fun do not exceed 1 from previous calculations.

The simulation runs for $T\_{max} = \text{time\_horizon}$ steps.

### 1.6 Output

The function can return either the full matrix of advancement\_level for all cells across all time steps (if full = TRUE) or a filtered spatial data frame containing only the cells with Bi​(Tmax​+1)>0 and their final advancement\_level. The final advancement\_level column is renamed to burning.