Pseudo-code implementation of lattice model.

```
count is array of (position, molecule)
rate is array of (position, process, molecule)

do loop

  # accumulate a total rate for all processes (in the bulk
  #   and all lattice sites)
  total_rate = 0

  # reset the bulk counts of resources according to the
  #  current "universe" state; these are generally constant
  #  or pulsed functions of E_i
  for each i in resources
    count(bulk, i) = universe(E_i, time)
  end

  # AB complex in the bulk decays
  rate(bulk, decay_AB) = k_decay * count(bulk, AB)

  # increment the total rate with the decay rate in the bulk
  total_rate += rate(bulk, decay_AB)

  for each s in lattice sites

    for each i in molecules
      rate(s, diffusion, i) = D_lattice * count(s, i)
      rate(s, bind, i) = k_on * count(bulk, i) / area
      rate(s, unbind, i) = k_off * count(s, i)
    end

    # increment the binding rates of agents with the input
    #   resource mediated cooperative binding
    rate(s, bind, A) += k_coop * count(s, 1) * count(bulk, A) / area
    rate(s, bind, B) += k_coop * count(s, 2) * count(bulk, B) / area
    rate(s, bind, AB) += k_coop * count(s, 1) * count(bulk, AB) / area

    # the resource conversion reactions
    rate(s, reaction, A) = k_1_2 * count(s, A) * count(s, 1)
    rate(s, reaction, B) = k_2_3 * count(s, B) * count(s, 2)
    rate(s, reaction, AB) = k_1_3 * count(s, AB) * count(s, 1)

    # formation and decay of the AB complex
    rate(s, form, AB) = k_form * count(s, A) * count(s, B)
    rate(s, decay, AB) = k_decay * count(s, AB)

    # increment the total rate with the summed rate of all
    #   processes at this lattice site
    total_rate += sum(rate(s, *))
  end

  # Gillespie algorithm to choose when and what event happens next

  # the time increment for this event
  time += (1 / total-rate) * log(1.0 / random(0, 1))
  # what the event is.
  # position: bulk or specific lattice site
  # process: what happens at that position (binding, unbinding, etc)
  # molecule: what agent or resource binds, unbinds, etc

  choose (position, process, molecule)
      with probability = rate(position, process, molecule) /
total_rate

  # now perform the chosen event:

  if process = diffusion:
    count(position, molecule) -= 1
    count(random_adjacent_position(position), molecule) += 1
  end

  if process = bind:
    count(bulk, molecule) -= 1
    count(position, molecule) += 1
  end

  if process = unbind:
    count(position, molecule) -= 1
    count(bulk, molecule) += 1
  end

  if process = reaction:
    if molecule = A:
      count(position, 1) -= 1
      count(position, 2) += 1
    end

    if molecule = B:
      count(position, 2) -= 1
      count(position, 3) += 1
    end

    if molecule = AB:
      count(position, 1) -= 1
      count(position, 3) += 1
    end
  end

  if process = form:
    count(position, A) -= 1
    count(position, B) -= 1
    count(position, AB) += 1
  end

  if process = decay:
    count(position, AB) -= 1
    count(position, A) += 1
    count(position, B) += 1
  end
end
```