Shortest-path breakdown in varistors

Aleksandar Donev

July 2001

1 Main program

This program uses SSCNO and a scheme similar to the one in the IV program to study the onset of current flow in dielectric-like materials. It invokes the F77 Dijkstra shortest-path routine from Graph-Algorithms_F77.

This documentation is not finished.

"WEAVE.f90" 1.1

PROGRAM Breakdown
  USE Precision  // Kind parameters
  USE Error_Handling
  USE System_Monitors
  USE Initialization_Termination
  USE Network_Data_Structures
  USE Lattice_Geometry
  USE Network_Geometry
  USE Simple_Graphics
  USE Network_Graphics
  USE Network_Spanning_Trees
  USE Graph_Algorithms
  USE Graph_Algorithms_F77
  USE COST_TYPE@&_Cost_Parameters
  USE COST_TYPE@&_Cost_Functions
  USE Lattice_Network_Optimization
  USE Conjugate_Gradient
  USE Dual_Network_Solvers
  USE Dual_Line_Minimizers
  USE Dual_Newton_SSCNO
  USE SSCNO_Interface

IMPLICIT NONE

INTEGER (KIND = i_byte), DIMENSION (:), ALLOCATABLE :: shortest_paths
REAL (KIND = r_wp), DIMENSION (:), ALLOCATABLE :: arcs_path_lengths, paths_lengths
INTEGER (KIND = i_byte), DIMENSION (:), ALLOCATABLE :: arcs_paths
INTEGER (KIND = i_byte), :: head_path, tail_path, n_paths

INTEGER :: arc, node, sample, point, instance
CHARACTER (LEN = 60) :: title, plot_title(3) = ""
CHARACTER (LEN = 50) :: file_name = "", file_type = "CONS", file_extension = "dummy"
REAL (KIND = r_wp) :: applied_voltages_sum, min_arc_value, max_arc_value
REAL (KIND = r_wp), DIMENSION (2) :: current_interval = (/ 0.0, 1.0_wp /),
voltage_interval = (/ 0.0, 1.0_wp /)
INTEGER :: n_samples = 1, n_points = 2, n_instances = 1
INTEGER :: alloc_status, source_type
REAL (KIND = r_wp), DIMENSION (:) :: axis
REAL (KIND = r_wp), DIMENSION (:), ALLOCATABLE :: total_currents, total_voltages,
averaged_currents, averaged_voltages
REAL (KIND = r_wp), DIMENSION (:,:), ALLOCATABLE :: applied_voltages, injected_currents
LOGICAL :: plot_samples = T, change_current = T, change_voltage = F, save_plot

@#if 1
WRITE(*, *) "Enter change_current and change_voltage (T or F)"
READ(*, *) change_current, change_voltage
IF (change_current) THEN
   WRITE(*, *) "Enter the interval for the total injected current:"
   READ(*, *) current_interval
END IF
IF (change_voltage) THEN
   WRITE(*, *) "Enter the interval for the total applied voltage:"
   READ(*, *) voltage_interval
END IF
@#endif

WRITE(*, *) "Enter n_points, n_paths"
READ(*, *) n_points, n_paths
WRITE(*, *) "Plot every sample (T or F):";
READ(*, *) plot_samples
WRITE(*, *) "Enter file output type (CONS, XWIN, PNG, POST, PSCL)"
READ(*, *) file_type
SELECT CASE (file_type)
   CASE ("POST", "PSCL")
      file_extension = "ps"
   CASE ("PNG")
      file_extension = "png"
   CASE DEFAULT
      file_extension = "dummy"
ENDSELECT
CALL StartProgram
ALLOCATE (total_currents, n_points, total_voltages, n_points)
ALLOCATE (averaged_currents, n_points, averaged_voltages, n_points)
ALLOCATE (applied_voltages, n_points, injected_currents, n_points, n_samples)
CALL InitializeLatticeNetworkProblem
CALL InitializeSSCNO
CALL ResetTimer (100)
CALL StartTimer (100)
DO point = 1, n_points    // Preassign the desired set of total current flows or applied voltages
  total_currents_point = current_interval1 + REAL(point - 1) * (current_interval2 - current_interval1) / 
  REAL(n_points - 1)
  total_voltages_point = voltage_interval1 + REAL(point - 1) * (voltage_interval2 - voltage_interval1) / 
  REAL(n_points - 1)
END DO
injected_currents = 0.0_rwp    // Initialize
applied_voltages = 0.0_rwp    // Zero all averages at start!
WRITE(*, *) "Solving", n_points * n_samples * n_instances, " No problems took a total of 
  (s) ; ", ReadTimer (100)
WRITE(*, *) "-----------------------------------------------"
CALL ProfileSSCNO    // Detailed timing statistics
WRITE(*, *) "-----------------------------------------------"
@#if 1
CALL InitGraphics (file = "V_I," || TRIM(file_extension), file_type = TRIM(file_type),
  n_plots = n_samples + 1, tick_labels = "9F1", legend_position = "UL", plot_title = (/ "IV curve 
  for a 2D network" /), xlabel = "$I\$", ylabel = "$V\$"
IF (change_voltage \& change_current) THEN
  axis = (/ MIN(MINVAL(injected_currents), MINVAL(total_currents)),
  MAX(MAXVAL(injected_currents), MAXVAL(total_currents)), MIN(MINVAL(applied_voltages),
  MINVAL(total_voltages)), MAX(MAXVAL(applied_voltages), MAXVAL(total_voltages)) /)
ELSE IF (change_voltage) THEN
  axis = (/ MINVAL(injected_currents), MAXVAL(injected_currents), MINVAL(total_voltages),
  MAXVAL(total_voltages) /)
ELSE IF (change_current) THEN
  axis = (/ MINVAL(total_currents), MAXVAL(total_currents), MINVAL(applied_voltages),
  MAXVAL(applied_voltages) /)
END IF
IF (change_current) THEN
  averaged_voltages = SUM(applied_voltages, DIM = 2) / n_samples    // Final average IV curve
  CALL Plot2D (x = total_currents, y = averaged_voltages, plot_spec = "L,B", axis = axis)
DO sample = 1, n_samples
    CALL Plot2D (x = total_currents, y = applied_voltages, sample, plot_spec = "SCR", axis = axis)
END DO
END IF
IF (change_voltage) THEN
  averaged_currents = SUM(injected_currents, DIM = 2) / n_samples    // Final average IV curve
  CALL Plot2D (y = total_voltages, x = averaged_currents, plot_spec = "L:R", axis = axis)
DO sample = 1, n_samples
    CALL Plot2D (y = total_voltages, x = injected_currents, sample, plot_spec = "SSB", axis = axis)
END DO
END IF
CALL EndGraphics ()
@#endif
CALL EndProgram
DEALLOCATE (total_currents, total_voltages, averaged_currents, averaged_voltages)
DEALLOCATE (applied_voltages, injected_currents)
END PROGRAM Breakdown

"WEAVE.f90" 1.2 ≡
@m姻PlotPaths(_file_name, _file_type, _fmt, arc_array)
  arcs_mask = T_1_wp
  arcs_mask_0 = F_1_wp
  do arc = 1, n_arcs  // We do not plot the periodic boundary arcs
    arcs_mask_arc = (arcs_mask arc a (arcs_status arc ° periodic_bc arc))
  end do
  nodes_mask = F_1_wp
  nodes_mask_0 = F_1_wp
  call InitNetworkGraphics ( file = _file_name, file_type = _file_type, x_label = "Current flow\n\"Huge(\$\arrow\"\$)", & page_size = (/5000, 5000/), label_format = "NO", color_table = "RAIN", colorbar_position = "Horizontal", axis_labels_format = (/"NONE", "NONE", "NONE") )
  call PlotNetwork2D ( heads_tails = heads_tails, 1:n_arcs, node_offset = -1,
    node_coords = node_coords, 1:n_nodes, node_mask = nodes_mask_1:n_nodes,
    arc_mask = arcs_mask_1:n_arcs,
    arc_values = REAL(arcs_array 1:n_arcs, r_wp), resize_arcs = T, color_arcs = T, &
    arc_colorbar_format = _fmt, vector_type = 0, axis = (/0.0, 0.0, REAL(lengths + 1)/)
  )
  // The special arcs need to be plotted separately now
  nodes_mask_0: n_nodes = F_1_wp
  nodes_mask = -n_special_nodes : -1 =T_1_wp
  arcs_mask_1:n special_arcs = (-1) =T_1_wp arcs = -n_special_arcs, -1  // We do not plot the periodic boundary arcs
  arcs_mask arcs = (-1, r_wp), resize_arcs = T, color_arcs = F, & arc_size_range = (/0.0_wp,
  HUGE(1.0_wp) / 10/), node_colorbar_format = "5E1", arc_colorbar_format = "5E1",
  vector_type = 0, axis = (/0.0, 0.0, REAL(lengths + 1)/)
  call EndNetworkGraphics ()

"WEAVE.f90" 1.3 ≡
@m姻PlotNetwork(_file_name, _file_type, _title, arc_array, nodal_array, _nodes_mask, _arcs_mask)
  arcs_mask = _arcs_mask
  arcs_mask_0 = F_1_wp
  do arc = 1, n_arcs  // We do not plot the periodic boundary arcs
    arcs_mask_arc = (arcs_mask arc a (arcs_status arc ° periodic_bc arc))
  end do
  nodes_mask = _nodes_mask
  nodes_mask_0 = F_1_wp
  call InitNetworkGraphics ( file = _file_name, file_type = _file_type, x_label = "Current flow\n\"Huge(\$\arrow\"\$)", & page_size = (/5000, 5000/), label_format = "NO",...