A Discrete Event Method for Wave Simulation

JAMES NUTARO
Oak Ridge National Laboratory

This article describes a discrete event interpretation of the finite difference time domain (FDTD) and digital wave guide network (DWN) wave simulation schemes. The discrete event method is formalized using the discrete event system specification (DEVS). The scheme is shown to have errors that are proportional to the resolution of the spatial grid. A numerical example demonstrates the relative efficiency of the scheme with respect to FDTD and DWN schemes. The potential for the discrete event scheme to reduce numerical dispersion and attenuation errors is discussed.

Categories and Subject Descriptors: I.6.8 [**Simulation and Modeling**]: Types of Simulation—*Discrete event*; G.1.8 [**Numerical Analysis**]: Partial Differential Equations—*Finite difference methods*

General Terms: Algorithms,Theory

Additional Key Words and Phrases: DEVS, digital waveguide networks, wave propagation

1. INTRODUCTION

This article presents a discrete event method for simulating the linear wave equation. The method is a discrete event interpretation of the digital waveguide network described in Bilbao [2004]. The digital waveguide network has a structure and numerical properties similar to finite difference approaches, and so this work could also be seen as a discrete event interpretation of the finite difference time domain (FDTD) method (see Shlager and Schneider [1998] for a survey of the vast FDTD literature).

The discrete event interpretation retains, in any homogeneous region of a wave carrying material, the structure and numerical properties of a digital waveguide network. The most significant difference, in this case, is that undisturbed sub-regions do not schedule events. When the computational domain is large, this can reduce the computational effort needed to complete a simulation run.

Author's address: J. Nutaro, Oak Ridge National Laboratory, P.O. Box 2008, MS6085, Oak Ridge, TN 37831-6085; email: nutarojj@ornl.gov.

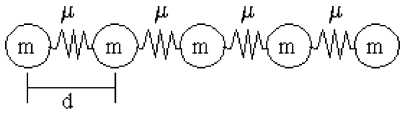


Fig. 1. A mass-spring model with masses m, separated by distance d, and connecting by springs with elasticity μ.

When a medium consists of different material types, each particular material is simulated using its preferred time step. This ensures unit gain within a homogeneous region. Differences in the time steps of adjacent regions are accounted for by event scheduling and extrapolation. Event scheduling is used to advance the simulation clock so that nonuniform time steps are easily managed. Extrapolation is used at material interfaces to approximate missing data points when the time grids of adjacent materials are not aligned.

2. WAVE PROPAGATION IN A ONE-DIMENSIONAL HOMOGENEOUS MEDIUM

A homogeneous, one-dimensional medium can be modeled with a lattice of point masses that are separated by a fixed distance, and are connected pair-wise by springs. This model is illustrated in Figure 1.

If a mass is displaced from its equilibrium position, then its left and right neighbors will move equal distances. This displacement in turn will cause the next neighbors to be displaced, and so on down the line. At the same time, the previously displaced cells return to their equilibrium positions. The macroscopic effect is a wave that propagates left and right from the initial displacement.

The velocity of a wave traversing the lattice shown in Figure 1 is given by

$$V = d\sqrt{\frac{\mu}{m}}, \tag{1}$$

assuming that the wave length is significantly larger than d. For a medium that is nearly continuous (i.e., when d is very much smaller than the wave length), the wave velocity can be approximated by

$$V = \sqrt{\frac{\epsilon}{\rho}}, \tag{2}$$

where ϵ is the bulk modulus and ρ is the density of the medium. Brillouin [1953] provides an excellent description of this model. More recent texts that describe the same model include Achenbach [1973] and Bekefi and Barrett [1977].

This model can be interpreted in two ways. Brillouin [1953] uses Figure 1 to describe the real structure of a crystalline solid. Approximate properties of a continuous material are found by assuming that d is very small relative to the wave lengths of interest. In contrast, Bilbao [2004] uses this model to describe a discrete approximation of a continuous structure. The latter interpretation is used in this article. However, the derivation of characteristic impedance, reflection, and transmission ratios, as well as other approximate properties of the continuous material, are simpler with the interpretation of Brillouin [1953].

Because these properties are used, but not derived, in Bilbao [2004], Brillouin [1953] is used here as a primary reference for wave physics.

3. AN OVERVIEW OF DEVS

The digital waveguide network described in Bilbao [2004] provides a discrete time simulation method for the lattice model. Given a homogeneous, one-dimensional medium, a numerically equivalent simulation method can be constructed as a discrete event system. The discrete event system specification (DEVS) is used to formalize this discrete event system.

DEVS is a mathematical formalism for describing discrete event systems (see Zeigler et al. [2000]). DEVS uses two types of structures to describe a discrete event system. Atomic models describe the behavior of elementary components. Coupled models describe collections of interacting components where components can be atomic and coupled models.

An atomic model is described by a set of inputs, a set of outputs, a set of states, a state transition function decomposed into three parts, an output function, and a time advance function. Formally, the structure is

$$M =< X, Y, S, \delta_{int}, \delta_{ext}, \delta_{con}, \lambda, ta >$$

where

X is a set of inputs,
Y is a set of outputs,
S is a set of states,
$\delta_{int} : S \to S$ is the internal state transition function,
$\delta_{ext} : Q \times X^b \to S$ is the external state transition function
with $Q = \{(s, e) \mid s \in S \,\&\, 0 \leq e \leq ta(s)\}$
and X^b is a bag of values appearing in X,
$\delta_{con} : S \times X^b \to S$ is the confluent state transition function,
$\lambda : S \to Y$ is the output function, and
$ta : S \to \mathbb{R}$ is the time advance function.

The external transition function describes how the system changes state in response to input. When input is applied to the system, it is said that an external event has occurred. The internal transition function describes the autonomous behavior of the system. When the system changes state autonomously, an internal event is said to have occurred. The confluent transition function determines the next state of the system when an internal event and external event coincide. The output function generates output values at times that coincide with internal events. The output values are determined by the state of the system just prior to the internal event. The time advance function determines the amount of time that will elapse before the next internal event occurs, assuming that no input arrives in the interim.

Coupled models are described by a set of components and a set of component couplings. Components of a coupled model can be atomic models and other

coupled models. Just as with atomic models, coupled models have interfaces defined by input and output sets. The behavior of a coupled model is determined by its component models and their interconnections. DEVS is a modular modeling formalism, and so interactions with a coupled model must occur through its interface.

For the purposes of this article, the discussion of coupled models is restricted to a flat structure (i.e., a structure composed entirely of atomic models) without external input or output coupling (i.e., the component models can not be affected by elements outside of the network). With these restrictions, a coupled model is described by the structure

$$N =< \{M_k\}, \{z_{ij}\} >$$

where

$\{M_k\}$ is a set of atomic models, and

$\{z_{ij}\}$ is a set of output-to-input maps $z_{ij} : Y_i \rightarrow X_j \cup \{\Phi\}$

where i and j correspond to M_i and M_j in $\{M_k\}$ and Φ is the nonevent.

The output-to-input maps describe how atomic models affect one another. The output-to-input map is, in this application, somewhat overgeneralized and could be replaced with more conventional descriptions of computational stencils. The nonevent is used in this instance to represent components that are not connected. That is, if component i does not influence component j, then $z_{ij}(y_i) = \Phi$, where $y_i \in Y_i$.

These structures describe what a model can do. A canonical simulation algorithm, given here as Algorithm 1, is used to generate dynamic behavior from the description. Algorithm 1 assumes a coupled model with components $\{M_1, M_2, \ldots, M_n\}$ and a suitable set of output-to-input maps. For every component model M_i, there are time of last event and time of next event variables, denoted by tL_i and tN_i, respectively. There are also state, input, and output variables s_i, x_i, and y_i in addition to the basic structural elements (i.e., state transition functions, output function, and time advance function). The variable x_i is a bag with elements taken from the input set X_i, and the variable y_i has a value taken from the output set Y_i. The simulation time is kept in variable t.

Algorithm 1 DEVS simulation algorithm.

```
t ← 0
for all i ∈ [1, n] do
   tL_i ← 0
   set s_i to the initial state of M_i
end for
while terminating condition not met do
   for all i ∈ [1, n] do
      tN_i ← tL_i + ta_i(s_i)
      Empty the bag x_i
   end for
   t ← min{tN_i}
   for all i ∈ [1, n] do
      if tN_i = t then
```

```
            y_i ← λ_i(s_i)
            for all j ∈ [1, n] & j ≠ i & z_ij(y_i) ≠ Φ do
               Add z_ij(y_i) to the bag x_j
            end for
         end if
      end for
      for all i ∈ [1, n] do
         if tN_i = t & x_i is empty then
            s_i ← δ_int,i(s_i)
            tL_i ← t
         else if tN_i = t & x_i is not empty then
            s_i ← δ_con,i(s_i, x_i)
            tL_i ← t
         else if tN_i ≠ t & x_i is not empty then
            s_i ← δ_ext,i(s_i, t − tL_i, x_i)
            tL_i ← t
         end if
      end for
   end while
```

4. A DEVS MODEL FOR ONE-DIMENSIONAL WAVE SIMULATION

A propagating wave can be simulated with an array of discrete event cells. Each cell describes a single mass in Figure 1, and the cells are separated by a distance d. A disturbance propagates through a cell with a speed V which is given by Equation 2.

A cell can change state in response to two types of events. An external event occurs when the mass at a neighboring cell is displaced from and returns to its equilibrium position. The result of an external event is to cause the cell to become displaced by a distance equal to its neighbor's displacement. An internal event occurs when the cell returns to its equilibrium position. Internal events occur d/V units of time after an external event, corresponding to the time required for the wave to propagate through the cell.

The discrete event system is constructed as a network of DEVS atomic models. Each atomic model describes a single cell, and the atomic models are connected to their left and right neighbors. A cell has three state variables. The variable U describes the displacement of the cell. The variables U_l and U_r describe the amplitude of the last displacement events received from the left and right neighbors, respectively. At the start, U is equal to the initial displacement of the mass, and U_l and U_r are equal to zero.

The input set X of a cell consists of pairs of real numbers that describe the displacement of the left and right neighbors. These values are denoted by X_l and X_r, respectively. Similarly, the output set Y consists of pairs of real numbers denoted by Y_l and Y_r. The output Y_l at cell i is mapped to the input X_r at cell $i-1$. Similarly, Y_r at cell i is mapped to X_l at cell $i+1$. This output-to-input mapping is shown in Figure 2. If a cell receives an event from the right without receiving a simultaneous event from the left, then the value of X_l is denoted by the nonevent Φ. Nonevents from the right are similarly treated.

A cell operates in the following way. If the cell and its neighbors are not displaced, then the cell does not schedule any events. In this case, the time

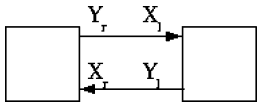


Fig. 2. Coupling between scattering models.

advance of the cell is ∞. A cell can become displaced when it receives input from its left or right neighbor. On receiving an input event, the cell computes its total displacement as the sum of the displacements of its left and right neighbors, and it remembers the observed left and right displacement values. Missing inputs (i.e., nonevents) are treated as zeros.

A displaced cell schedules an internal event to occur d/V units of time in the future. When this event occurs, the cell displaces its neighbors by producing left- and right-going output events. The value of the left-going event is equal to the total displacement of the cell minus the remembered left displacement value. The right-going event is determined in a similar way. Events with a value of zero do not need to be propagated because missing events are interpreted as zeros by the neighboring cells. After producing its left and right displacement events, the cell returns to its equilibrium position and sets its time advance to ∞.

The behavior of a cell is described concisely by its DEVS state transition, output, and time advance functions

$$\delta_{int}((U, U_l, U_r)) = (0, 0, 0) \tag{3}$$

$$\delta_{ext}((U, U_l, U_r), e, \{X_l, X_r\}) = \begin{cases} (X_l + U_r, X_l, U_r) & \text{if } X_l \neq \Phi \ \& \ X_r = \Phi \\ (U_l + X_r, U_l, X_r) & \text{if } X_l = \Phi \ \& \ X_r \neq \Phi \\ (X_l + X_r, X_l, X_r) & \text{otherwise} \end{cases} \tag{4}$$

$$\delta_{con}((U, U_l, U_r), e, \{X_l, X_r\}) = \delta_{ext}(\delta_{int}((U, U_l, U_r)), e, \{X_l, X_r\}) \tag{5}$$

$$\lambda((U, U_l, U_r)) = \{Y_l, Y_r\} = \{U - U_l, U - U_r\} \tag{6}$$

$$ta((U, U_l, U_r)) = \begin{cases} \infty & \text{if } U = U_l = U_r = 0 \\ d/V & \text{otherwise} \end{cases}. \tag{7}$$

State, input, and output trajectories of the discrete time model described in Bilbao [2004] can be recovered from this discrete event model. If the time advance function is defined to be constant with value d/V, then the discrete event model becomes a discrete time model with time step d/V. The state, input, and output trajectories of this discrete time model are the same as in the discrete event model, except that nonevents in the discrete event model appear as zero-valued inputs and outputs in the discrete time model. Similarly, if the time advance for this discrete time model is redefined as in Equation 7, then nonevents appear in place of zero-valued inputs and outputs.

These similarities can be formalized as a system isomorphism (see Zeigler et al. [2000]). Because the discrete event and discrete time models are isomorphic, numerical properties of the discrete time model also hold for the discrete event model. The discrete time waveguide model is equivalent to a second-order finite difference approximation of the wave equation (see Bilbao [2004]). By

selecting the time step of the discrete time model to be d/V, the stability of the scheme is guaranteed. Moreover, the model exactly replicates the propagating wave so long as the transmission medium is one-dimensional and homogeneous. These properties also hold for the discrete event model.

5. WAVE ATTENUATION IN A ONE-DIMENSIONAL HOMOGENEOUS MEDIUM

Wave attenuation due to real resistive effects in a material can be accounted for in the discrete event model. The strength of the attenuation is determined by the wave length and the transmission medium (e.g., Brillouin [1953]). For an attenuation coefficient α, the amplitude $A(x)$ of a wave decays exponentially with distance x as

$$A(x) = A(0)e^{-\alpha x} \ . \tag{8}$$

If the wave amplitude at position $x - d$ (for a right-traveling wave) is known, then the amplitude at position x is given by

$$A(x) = A(x-d)e^{-\alpha d} = A(x-d)K \ . \tag{9}$$

The term $K = e^{-\alpha d}$ is the loss coefficient that describes the amplitude attenuation over a distance d.

A simulation of an attenuating wave can be obtained by modifying the external transition function (i.e., Equation 4). The modified function applies the loss coefficient K from Equation 9 to incoming events. This causes a cell to attenuate any wave passing through it. The modified external transition function is

$$\begin{aligned}&\delta_{ext}((U, U_l, U_r), e, \{X_l, X_r\}) \\ &\quad = \begin{cases} (KX_l + U_r, KX_l, U_r) & \text{if } X_l \neq \Phi \ \& \ X_r = \Phi \\ (U_l + KX_r, U_l, KX_r) & \text{if } X_l = \Phi \ \& \ X_r \neq \Phi \\ (K(X_l + X_r), KX_l, KX_r) & \text{otherwise} \end{cases} .\end{aligned} \tag{10}$$

Using Equation 10 in place of Equation 4 yields an exact simulation of an attenuating wave in a one-dimensional homogeneous medium. To see this, it is sufficient to observe that the attenuation over a distance nd is given by

$$A(x) = A(x-d)K = (A(x-2d)K)K = \cdots = A(x-nd)K^n \ . \tag{11}$$

If a displacement is introduced at position $x - nd$ in the discrete event model, it will propagate left and right via application of Equation 10. Without loss of generality, consider only the right-traveling wave. Applying Equation 10 at the next cell to the right gives a wave amplitude of

$$A(x-(n-1)d) = A(x-nd)K \ . \tag{12}$$

Repeated application gives an amplitude at $x - (n-k)d$, $k \leq n$, of

$$\begin{aligned}A(x-(n-k)d) &= A(x-(n-k+1)d)K \\ &= A(x-(n-k+2)d)K^2 = \cdots = A(x-nd)K^k\end{aligned} \tag{13}$$

Letting $k = n$ gives Equation 11 and completes the argument.

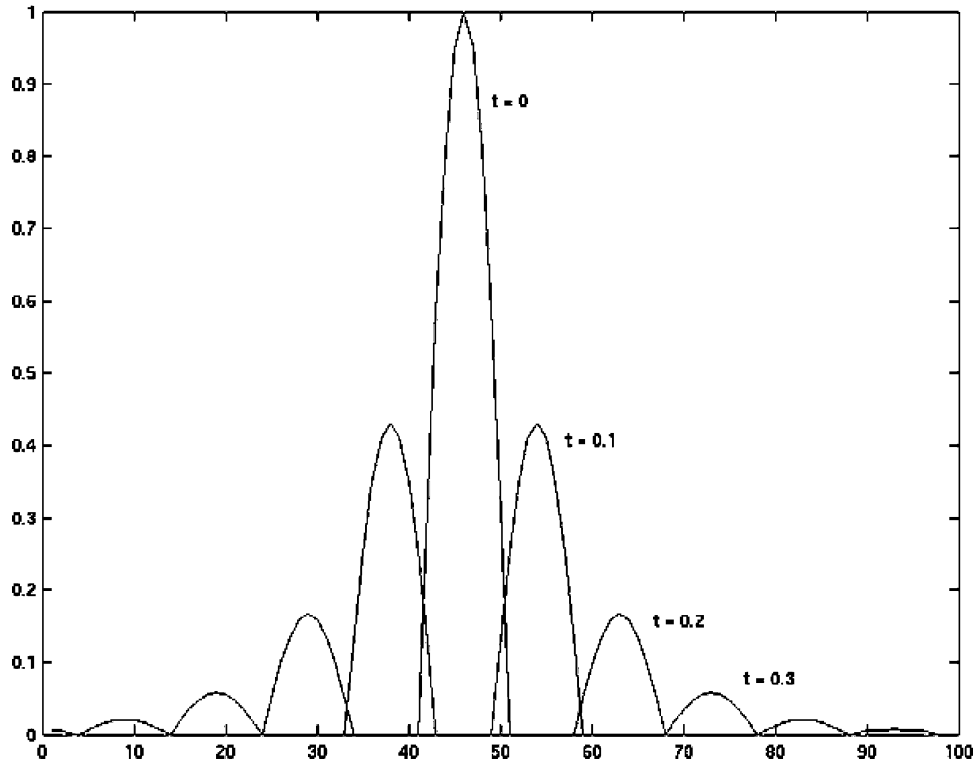


Fig. 3. Wave attenuation in a one-dimensional homogeneous medium.

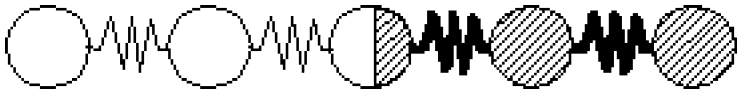


Fig. 4. A medium with two distinct materials, each with a different mass and elasticity.

Figure 3 shows a simulation of an attenuating wave by the use of this discrete event model. The example uses $K = 0.9$, $d = 0.01$, and $V = 1.0$. The wave is shown at times 0, 0.1, 0.2, 0.3, 0.4, and 0.5. The x-axis shows the indices of the computational grid points. The y-axis shows the amplitude of the wave as a function of x.

6. WAVE PROPAGATION IN A ONE-DIMENSIONAL HETEROGENEOUS MEDIUM

A heterogeneous medium can be modeled just as in Figure 1 except that mass and elasticity will vary with position. Figure 4 illustrates a medium consisting of two different materials.

Where the medium is homogeneous, propagating waves can be modeled as before. When a wave encounters an interface, the model must account for two effects. First, the wave is split into reflected and transmitted parts. The

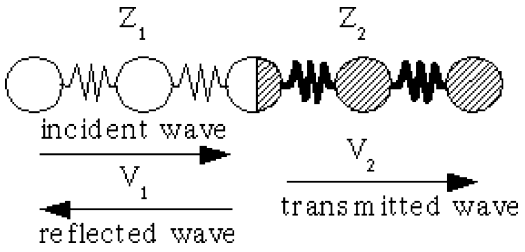


Fig. 5. A wave traveling from left to right. The reflected wave goes left, and the transmitted wave continues to the right.

transmitted part continues in the same direction as the original wave. The reflected part travels in the opposite direction. Both the transmitted and reflected waves will have the same frequency as the original wave. However, the velocities and amplitudes of the two waves will be different.

The second effect is a consequence of the differing wave velocities. The wave speed is the product of the wave length and wave frequency (i.e., the distance per unit time). The wave frequency is unchanged, thus, the wave length of the transmitted part must change to match the new wave velocity.

The amplitudes of the reflected and transmitted waves are related to the amplitude of the incident wave by the reflected amplitude ratio R and the transmitted amplitude ratio T. If A_I is the amplitude of the incident wave, A_T is the amplitude of the transmitted wave, and A_R is the amplitude of the reflected wave, then

$$A_T = TA_I, \text{ and} \tag{14}$$

$$A_R = RA_I\ . \tag{15}$$

The transmitted and reflected amplitude ratios are determined by the characteristic impedance of the two materials. In general, the characteristic impedance is dependent on the wave length. However, when the wave length is large compared to d, the characteristic impedance Z can be approximated by

$$Z = \sqrt{\rho\epsilon}, \tag{16}$$

where ϵ is the bulk modulus and ρ is the density of the medium.

Let Z_1 be the characteristic impedance of the material carrying the incident wave, and Z_2 the characteristic impedance of the adjacent material. This arrangement is shown in Figure 5. The transmitted amplitude ratio is

$$T = \frac{2Z_1}{Z_1 + Z_2}\ . \tag{17}$$

The reflected amplitude ratio is

$$R = \frac{Z_1 - Z_2}{Z_1 + Z_2}\ . \tag{18}$$

These ratios are required for energy conservation at the interface; notice that $A_I = A_T - A_R = TA_I - RA_I$. Moreover, when $Z_1 = Z_2$, energy conservation

requires that the transmitted wave have the same amplitude as the incident wave. In a homogeneous medium, where the characteristic impedance is constant, this can only be satisfied when the optimal time step is used (i.e., d/V). The discrete event model uses an optimal time step for each region, and so it is energy conserving within homogeneous sections. Derivations of R and T can be found in Brillouin [1953], Achenbach [1973], and Bekefi and Barrett [1977]. Brillouin [1953], in particular, derives them directly from energy conservation laws at the interface.

When the wave speeds on the two sides of an interface are different, the transmitted wave will have a different wave length than the reflected wave. The wave number a of a wave is the inverse of its wave length. Let w be the frequency of the incident wave, $A_I(x)$ be the position-dependent wave amplitude, and a_I be the wave number. Let the time-and space-dependent amplitude of the incident wave be described by

$$\hat{A}_I(t,x) = A_I(x)sin(wt - a_I x)\ . \tag{19}$$

Then the reflected wave $\hat{A}_R(t,x)$ is described by

$$\hat{A}_R(t,x) = R\hat{A}_I(t,x), \tag{20}$$

where R is the reflected amplitude ratio given by Equation 18. The reflected wave has the same speed and wave number as the incident wave.

The transmitted wave, however, has a different speed and wave number. The wave number of the transmitted wave, denoted a_T, is

$$a_T = \frac{V_1}{V_2}a_I\ . \tag{21}$$

In Equation 21, V_1 is the wave speed in the material carrying the incident wave, and V_2 is the wave speed in the material carrying the transmitted wave (see Figure 5). The transmitted wave $\hat{A}_T(t,x)$ is described by

$$\hat{A}_T = TA_I(x)sin(wt - a_T x)\ . \tag{22}$$

If $V_1 = V_2$, then the wave numbers of the transmitted and reflected/incident waves are the same. It is also true that the time grids of the two materials are the same. However, if $V_1 \neq V_2$, then the wave numbers and time grids will be different. In this case, information will be missing when a wave moves across the interface boundary.

If $V_1 > V_2$, then the wave is moving from a high speed to a low speed medium. On a fixed spatial grid, this means that the incident wave is sampled with a higher frequency that the transmitted wave. In this case, there is too much information, and the high speed wave must be down sampled as it moves into the slower material. When $V_1 < V_2$, the continuity of the high speed wave can only be maintained by introducing information that is not available on the low speed grid. These cases are instances of a general time alignment problem,

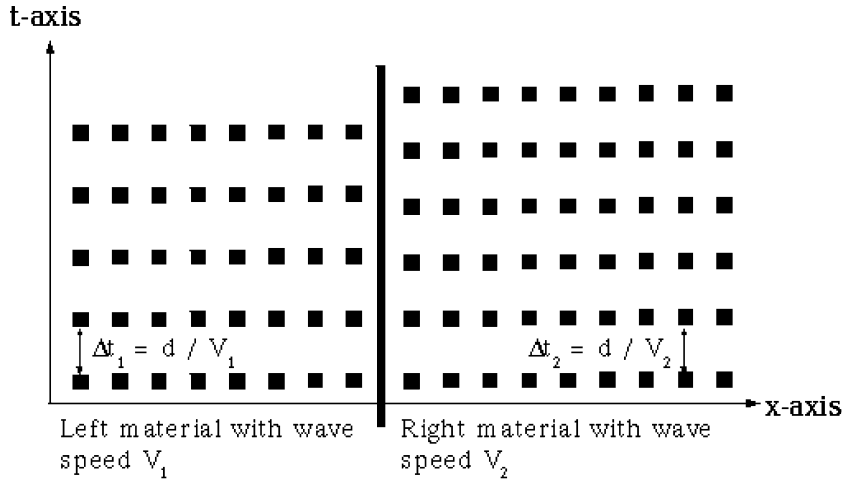


Fig. 6. A string consisting of two materials. The left material has wave speed V_1 and the right material has wave speed V_2. In this figure $V_1 < V_2$, and so the time grid points for the right material are closer together than those for the left material. In general, only the grid points at time zero are aligned.

where information required at a time grid point of one material is not available on the time grid of the adjacent material. The time grid alignment problem is illustrated in Figure 6.

Without loss of generality, suppose that the incident wave approaches the interface from the left. There will be at least one, but possibly more, missing time points on the right. Let t be the time at which the incident wave strikes the interface, n the number of missing time points, and t_i, with $i \in [1, n]$, the missing time points. The time t_1 is the first time point on the right, following time t. Subsequent t_is are found by adding the time step of the right material to t_1. The t_i can be written concisely as

$$t_i = \left\lceil \frac{t\,V_2}{d} \right\rceil \frac{d}{V_2} + \frac{(i-1)d}{V_2}\,. \tag{23}$$

The number of approximating points n is found by counting the number of right time grid points that occur in the interval between t and the next time grid point on the left. This number can computed using Algorithm 2.

Algorithm 2 Compute the number of approximated points.

$n \leftarrow 0$
$t_{start} \leftarrow \lfloor t\,V_2/d \rfloor d/V_2$
$t_{end} \leftarrow t + d/V_1$
while $t_{start} < t_{end}$ **do**
  $n \leftarrow n + 1$
  $t_{start} \leftarrow t_{start} + d/V_2$
end while

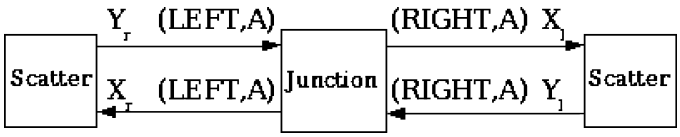


Fig. 7. Couplings between a junction model and its associated scattering models.

Note that Algorithm 2 constructs an approximation to Equation 21, computing n as, approximately, $(t_{end} - t)(V_2/d)$. Without loss of generality, let $t = 0$. Then $t_{end} = d/V_1$ and $n \approx V_2/V_1$, so the wave length of the transmitted wave is roughly V_2/V_1 times the wave length of the incident wave. The wave number is the inverse of the wave length, and so we have arrived, approximately, at Equation 21.

Note that if the incident wave strikes the interface on the right, then V_1 and V_2 in Equation 23 and Algorithm 2 are exchanged. That is, the t_is in Equation 23 are computed using V_1. Also, t_{start} in Algorithm 2 is determined with V_1, and V_2 is used to find t_{end}.

These effects are implemented in a junction model. A junction model is placed between scattering models at an interface. When a junction model receives an input event from the left, it sends an output left immediately. This output is the reflected wave, and its amplitude is given by Equation 15. The transmitted wave consists of a series of events that are sent to the right. Each event sent to the right has an amplitude given by Equation 14. There are n such events where n is determined using Algorithm 2. The times at which the events are transmitted are given by Equation 23.

A junction model is formalized as an atomic model. Its input events are pairs (D, A), where D describes the wave direction (*LEFT* or *RIGHT*) and A is the wave amplitude. An input event from the scattering model on the left will have $D = LEFT$. An input from the right scattering model will have $D = RIGHT$. Junction output events are bags of such pairs. An output event with $D = LEFT$ becomes an input X_r at the scattering model to the left. Similarly, an output with $D = RIGHT$ becomes an input X_l at the scattering model to the right. The coupling between a junction model and its associated scattering models is shown in Figure 7.

Its state is a sorted list of output events. Each event in the list is described by the time at which it will occur, a direction in which the event is to be sent, and the wave amplitude. The tuple (t, D, A) is used to denote this, where t is the event time, and D and A are as before. The events are sorted by time in increasing order. That is, the smallest element is at the front of the list and the largest at the back. The list, denoted by q, has five operations that can be performed on it. These are:

—*empty*(q), which is true if there are no events in the list, and false otherwise,
—*first*(q), which gives the first tuple stored in the list,
—*add*$(q, (t, D, A))$, which places the tuple (t, D, A) in its appropriate list position, and replaces any existing tuple with equivalent t and D (i.e., t and D

uniquely identify an event in the list, and the list always contains the last such event that was added),

—*remove_smallest*(q), which removes every event in the list with time *first*(q).t, and

—*get_smallest*(q), which returns a bag containing every event in the list whose time is *first*(q).t, or an empty bag if q is empty.

The junction model knows the current simulation time, which is denoted as t in the state transition and time advance functions.

The list is initially empty and its time advance is ∞. When the model receives an input event, it places a reflected wave event and transmitted wave events on the list q. The time advance function schedules the next output event at the time of the first tuple in the sorted list. If q is empty, then the time advance is ∞. The internal transition function removes the most recent events from q. The junction model behavior is described concisely by

$$\delta_{int}(q) = remove\_smallest(q) \tag{24}$$

$$\delta_{ext}(q, e, x^b) = q' \text{ , where } q' \text{ is computed from } q \text{ and } x^b \text{ using Algorithm 3} \tag{25}$$

$$\delta_{con}(q, x^b) = \delta_{ext}(\delta_{int}(q), 0, x^b) \tag{26}$$

$$ta(q) = \begin{cases} first(q).t - t & \text{if } empty(q) = false \\ \infty & \text{otherwise} \end{cases} \tag{27}$$

$$\lambda(q) = \{(D_i, A_i) \,|\, (D_i, A_i) \in get\_smallest(q)\} \,. \tag{28}$$

Algorithm 3 Compute the external transition function.

```
q' ← q
for each (D, A) ∈ x_b do
    q' ← add(q', (t, D, RA))
    compute n using Algorithm 2
    for all i ∈ [1, n] do
        compute t_i using Equation 23 with incident wave direction D.
        if D = LEFT then
            q' ← add(q', (t_i, RIGHT, TA))
        else
            q' ← add(q', (t_i, LEFT, TA))
        end if
    end for
end for
```

7. NUMERICAL PROPERTIES OF THE DISCRETE EVENT SCHEME IN ONE-DIMENSION

In a homogeneous region, the discrete event simulation scheme is exact. This is due to two facts. First, in any homogeneous region the discrete event scheme is isomorphic to the discrete time scheme described in Bilbao [2004]. The discrete time scheme is known to be exact (see Bilbao [2004]) for a one-dimensional homogeneous material. It follows that the discrete event scheme is exact as well.

Second, because every material within a nonhomogeneous medium is simulated using its preferred time step, this relationship holds in every homogeneous region of an inhomogeneous material.

Because the scheme is exact within any homogeneous region, error is introduced only when a wave crosses an interface. When this occurs, there are two sources of error. The first error comes from the wave amplitude approximation scheme used at the interface. The second error is in the simulated time required for the wave to traverse the interval that spans the interface.

The error introduced when a wave crosses an interface is given by the sum of the amplitude approximation error and the transit time error. Let $u(x)$ describe the incident wave at the moment it strikes the interface, and $u(0)$ be the wave amplitude at the interface.

The junction model approximates the transmitted wave amplitude with the last computed value at the interface. The approximation error, denoted E_1, is given by neglected terms in the Taylor series expansion of $u(x)$ about 0. Only the first term is used in the approximation, therefore, the error is

$$E_1 = \sum_{p=1}^{\infty} \frac{u^{(p)}(0)}{p!} h^p \ , \tag{29}$$

where h is the length of the extrapolation interval. The extrapolation interval h is bounded from above by the spatial grid spacing d. Let $u^{(p)}_{\max}$ be the largest p-th derivative of u with respect to x. Dropping the higher order terms in Equation 29, the amplitude error can be written as

$$E_1 \leq |u^{(1)}_{\max}| d \ . \tag{30}$$

Let V_1 and V_2 be the wave speeds on either side of an interface. The actual speed in the cell where the interface resides is between V_1 and V_2. Because the simulation uses either V_1 or V_2 depending on the wave direction, the error is bounded by $|V_1 - V_2|$. This speed error, multiplied by the simulated time required to cross the interface, gives a wave position error. The simulated traversal time is bounded from above by $d / V_{\min}$, where $V_{\min}$ is the smaller of V_1 and V_2. Multiplying the position error by the $u^{(1)}_{\max}(x)$ gives a bound on the error in $u(x)$ due to inaccuracy in the interface traversal time. This bound, denoted E_2, is

$$E_2 \leq \frac{d \ |(V_1 - V_2) u^{(1)}_{\max}|}{V_{\min}} \ . \tag{31}$$

Let $\Delta V_{\max}$ be the largest difference in the velocities of adjacent regions. Then, the error due to incorrect propagation times is bounded by

$$E_2 \leq \frac{d \, \Delta V_{\max} |u^{(1)}_{\max}|}{V_{\min}} \ . \tag{32}$$

The total error bound is given by

$$E_1 + E_2 \leq d \, |u^{(1)}_{\max}| \left(1 + \frac{\Delta V_{\max}}{V_{\min}} \right) \ . \tag{33}$$

A test problem is used to verify Equation 33. The test problem consists of a string with a unit length that is divided into two segments. The first segment

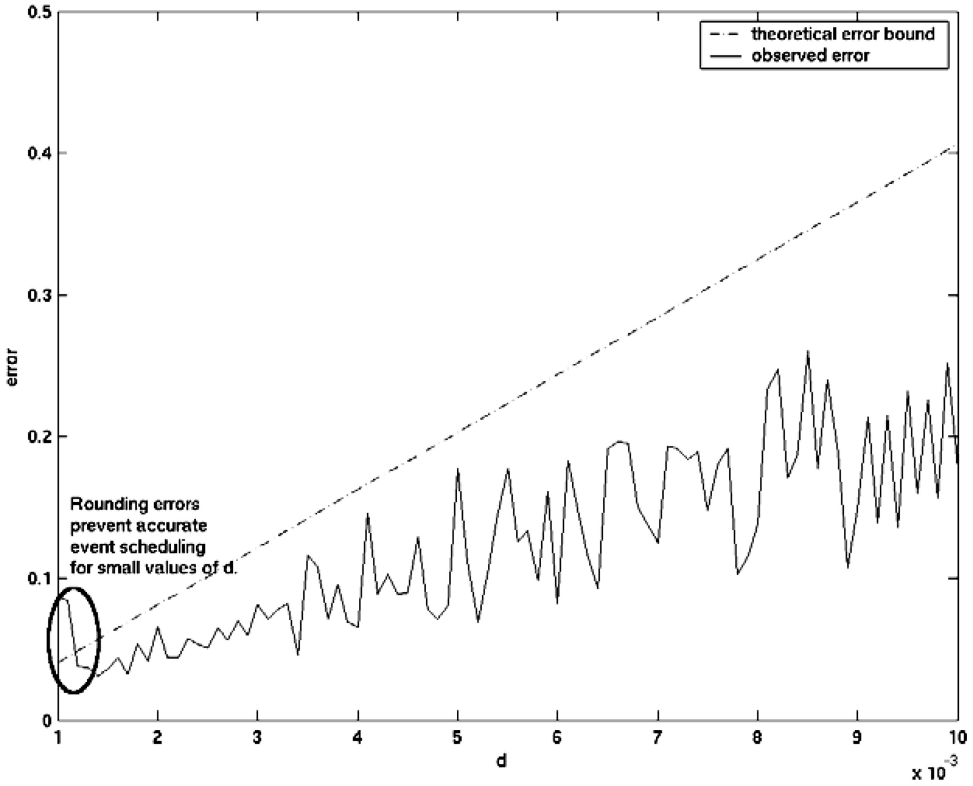


Fig. 8. Observed and theoretical error bound for the one-dimensional wave propagation scheme.

has a bulk modulus of 1 and density of 1. The second segment has a bulk modulus 0.5 and density 1. The initial disturbance of the string is described by

$$u(0,x) = \begin{cases} sin(10\pi x) & 0 \leq x \leq 0.1 \\ 0 & \text{otherwise} \end{cases}. \tag{34}$$

The simulation is run for one time unit. At then end of the simulation, the error is found by taking the difference of the known and computed solution at each grid point. Figure 8 shows a plot of error versus d, where the error is the largest observed at any grid point.

The error bound given by Equation 33 matches the observed errors very nicely for the range of d shown in Figure 8. However, for small d the error bound fails to hold. A likely culprit is the notorious sensitivity of some discrete event models to rounding errors (e.g., Nicol et al. [2000] and Wieland [1997]). In fact, all numerical methods are susceptible to rounding errors when the discretization is small enough. While small enough is typically smaller than practically useful for time stepped methods, it is possible that the discrete event method proposed here is orders of magnitude more sensitive to this kind of error.

One particular source of rounding error is the ceiling function used in Equation 23. Rounding errors here can cause an event to be scheduled one

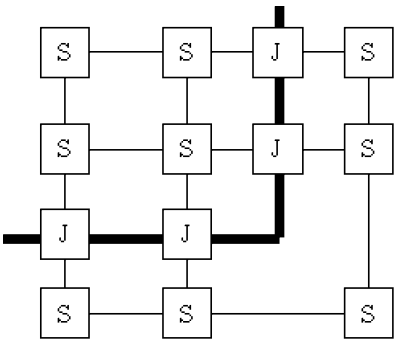


Fig. 9. A two-dimensional network of scattering (S) and junction (J) models. Couplings are shown as thin lines, and a material interface is shown with a thick line.

time step before or after it should actually occur. To illustrate this, suppose that, in Equation 23, $d/V = 0.1$ and $t = 1.0$. It should be the case that $t_1 = \lceil 1.0 \cdot 10.0 \rceil 0.1 = 1.0$. However, a small rounding error, say 0.0001, in the multiplication $1.0 \cdot 10.0$ gives $t_1 = \lceil 10.0 + 0.0001 \rceil 0.1 = 11.0 \cdot 0.1 = 1.1$, which is incorrect. In this case, an event that should have been scheduled at time 1.0 is scheduled for one time step later, at time 1.1.

8. PROPAGATION IN TWO- AND THREE-DIMENSIONS

Extending the scheme into two- and three-dimensions is straightforward. Interfaces are handled just as in the one-dimensional case. The scattering models are modified to become four input/four output (in two-dimensional) or six input/six output (in three-dimensional) models. The structure of a two-dimensional wave propagation model is shown in Figure 9.

Just as in the one-dimensional case, the generalized scattering model in Bilbao [2004] is reinterpreted as a discrete event system. In Bilbao [2004], it is shown that these types of scattering models introduce errors proportional to the square of the grid spacing d. Assuming that the scattering and junction errors are additive, the introduction of the generalized scattering models does not materially affect the linear errors that occur at the interfaces. Consequently, the k-dimensional case has errors dominated by Equation 33 when the material is inhomogeneous.

A k-dimensional scattering model has $m = 2k$ possible input and output directions and $m+1$ state variables. The state variables are denoted $U, U_1, \ldots, U_m$. The time advance function is the same as in the one-dimensional case:

$$ta((U, U_1, \ldots, U_m)) = \begin{cases} \infty & \text{if } U = U_1 = U_2 = \cdots = U_m = 0 \\ d/V & \text{otherwise} \end{cases}. \tag{35}$$

The scattering model can receive input from any of m directions. The input bag x^b contains pairs (D, A) where D indicates the direction from which the wave arrives, and A is the amplitude of the wave. The direction variable D has a range $[1, m]$ corresponding to the m possible input directions. The external

transition function is

$$\delta_{ext}((U, U_1, \ldots, U_m), e, x^b) = \left(\sum_{i \in [1,m]} U_i', U_1', \ldots, U_m' \right), \text{ where} \tag{36}$$

$$U_i' = \begin{cases} A_i & \text{if } (i, A_i) \in x^b \\ U_i & \text{otherwise} \end{cases}. \tag{37}$$

The stored value U is scattered equally in all directions except for those directions from which input was received. This is described by the scattering model output function

$$\lambda((U, U_1, \ldots, U_m)) = \{Y_1, \ldots, Y_m\} = \left\{ -U_1 + \frac{2}{m}U, \ldots, -U_m + \frac{2}{m}U \right\}, \tag{38}$$

where Y_i goes in direction i. The perhaps surprising $2/m$ factor appearing in Equation 38 is derived in Bilbao [2004]. Note, however, that the factor is necessary for Equation 38 to become Equation 6 in the one-dimensional case.

The internal and confluent transition functions of the system remain the same as in the one-dimensional case;

$$\delta_{int}((U, U_1, \ldots, U_m)) = (0, 0, \ldots, 0) \text{ , and} \tag{39}$$

$$\delta_{con}((U, U_1, \ldots, U_m), x^b) = \delta_{ext}(\delta_{int}((U, U_1, \ldots, U_m)), 0, x^b) \text{ .} \tag{40}$$

9. BOUNDARY CONDITIONS

Boundary conditions can be applied by generating appropriate input events for scattering models at the edge of the solution domain. Periodic boundary conditions can be constructed by attaching the free outputs of the edge-scattering junctions to the free inputs of the edge-scattering junctions appearing on the opposite side of the solution domain. Perfectly reflecting boundary conditions can be implemented by adding a reflecting terminator model to the free outputs and inputs of the edge-scattering junctions. The reflecting terminator model has one input and one output. The output of the model is a copy of the input Δt units of time in the past, where Δt is equal to the noninfinite value taken by the time advance function of the adjacent scattering model.

Absorbing boundary conditions can be complicated to implement, and they continue to be an area of active research. However, it is possible in many instances to implement known boundary conditions directly as DEVS models (for a description of some frequently used absorbing boundary conditions, see Taflove [1995]). For instance, approximately absorbing boundary conditions can be implemented by attaching a one-dimensional wave simulator to the free inputs and outputs of edge scattering models. The one-dimensional wave simulator uses a single cell finite difference approximation to the wave equation. The left edge of this one-dimensional simulator is fixed at zero. The right edge is equal to the input value received from the adjacent scattering model. The model output is the value of the state variable. The time step, spatial grid spacing, and wave speed of the one-dimensional simulator are equal to that of the attached scattering junction.

A DEVS model that implements this approximately absorbing boundary condition is described in the following. The model has a single input and a single output. The three state variables are the edge-wave amplitude at the current time step, the edge-wave amplitude at the last time step, and the last input value. The model is parametrized by the wave speed V and time step Δt of the adjacent scattering model. The values of the state variables are initially zero. The model state transition functions are

$$\delta_{ext}((U_1, U_0, U), e, x) = (U_1, U_0, x)\ , \tag{41}$$

$$\delta_{int}((U_1, U_0, U)) = \left(\frac{1}{V^4}U + 2\left(1 - \frac{1}{V^4}\right)U_1 - U_0, U_1, 0\right)\ , \text{and} \tag{42}$$

$$\delta_{con}((U_1, U_0, U), x) = \delta_{ext}(\delta_{int}((U_1, U_0, U)), 0, x)\ . \tag{43}$$

The output function is given by

$$\lambda((U_1, U_0, U)) = U_1 \tag{44}$$

and the time advance function is

$$ta((U_1, U_0, U)) = \begin{cases} \infty & \text{if } U_1 = U_0 = U = 0 \\ \Delta t & \text{otherwise} \end{cases}\ . \tag{45}$$

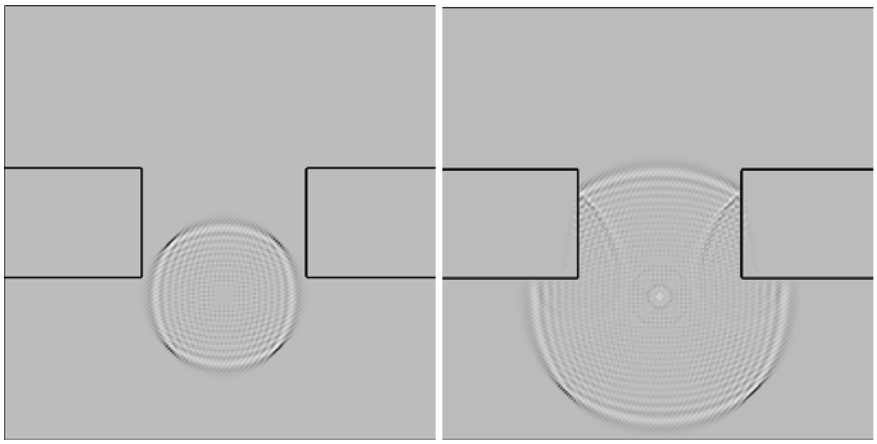
10. AN EXAMPLE

Figure 10 shows a electromagnetic wave propagating through a gap in a concrete wall. The wave is produced by an impulse. The impulse has a height of ten, and it originates at the center of the displaced area seen in the first image. The approximately absorbing boundary conditions described in Section 9 are used in this simulation. The four meter by four meter region is approximated by a grid spacing of 0.02 meters. The concrete blocks are outlined in black. The air is assumed to have a permeability (inductance per meter) of $1.26 \times 10^{-6}\ H/m$ and a permittivity (capacitance per meter) of $8.85 \times 10^{-12}\ F/m$. The concrete has a permeability of $1.26 \times 10^{-5}\ H/m$ and a permittivity of $8.82 \times 10^{-11}\ F/m$.

This simulation is optimized by placing a lower bound on the magnitude of a wave that is propagated. In some applications (e.g., wireless communications), wave amplitudes that are small enough can be treated as being effectively zero. This lower limit is applied in the boundary and scattering models. In both cases, outputs with values below the threshold are replaced by nonevents. The threshold value used for this simulation is 10^{-3}.

The computational cost of the simulation can be measured by the number of state transitions that are computed. This measure can be translated into real performance gains if an efficient discrete event simulation engine is used (e.g., Nutaro et al. [2003], Muzy et al. [2005], and Tang et al. [2005]). This example uses the adevs simulation engine (see Muzy and Nutaro [2005]).

A state transition in the discrete event scheme is any computation of an internal, external, or confluent transition function. A state transition in an FDTD scheme is a calculation of a new value at a grid point. The example problem is simulated for $3.66E \times 10^{-8}$ seconds, at which time the discrete event model becomes passive. The discrete event simulation requires 6, 790, 199 state



(a) $t = 3.27 \times 10^{-9}$ (b) $t = 5.80 \times 10^{-9}$

(c) $t = 1.09 \times 10^{-8}$ (d) $t = 1.47 \times 10^{-8}$

Fig. 10. A wave propagating through a gap in a concrete wall.

transitions. The smallest time advance used by a scattering model is 6.67×10^{-11}.

If an FDTD scheme is used to perform the same calculation, then a time step of 6.67×10^{-11} is the largest possible while maintaining the stability of the calculation (e.g., Taflove [1995] and Bilbao [2004]). The domain of the test problem is approximated with $40,000$ grid points (i.e., a four meter by four meter and grid resolution of 0.02 meters). Using a time step of 6.67×10^{-11}, 548 time steps are needed to simulate 3.66×10^{-8} seconds. A new value is computed at each grid point for each time step, and so $21,920,000$ state transitions are required. This is 3.2 times more computational effort than is required by the discrete event simulation.

When the same scenario is simulated using a 0.01 meter grid resolution (i.e., with $160,000$ grid points), the discrete event model requires $19,502,660$ state transitions. The same analysis as given before indicates that an FDTD solution

requires 142, 720, 000 state transitions, or 7.3 times more computational effort. When a 0.005 meter grid resolution is used, an FDTD scheme requires 16.3 times more state transitions than the discrete event simulation.

This trend is anticipated by the performance models constructed in Jammalamadaka [2003]. The relative performance improvement is a result of the discrete event scheme focusing computational effort only on those portions of the grid that are changing. In this particular example, the earlier frames (Figures 10a and 10b) have large numbers of inactive cells (i.e., cells with $ta(\cdot) = \infty$). The latter frames (Figures 10c and 10d), while containing more active cells, still have many inactive cells within the concrete walls.

In contrast, FDTD schemes apply equal amounts of computational effort everywhere. As the grid becomes more refined, the relative number of active cells becomes smaller. Consequently, the relative performance of the discrete event scheme is improved. This performance improvement can be significant when the number of grid points is large and the extent of the disturbance is relatively small. In the most refined version of this example (i.e., with $d = 0.005$), there is an order of magnitude reduction in the computational effort.

11. CONCLUSIONS

The wave simulation scheme described in this article is closely related to the dynamic structure cellular automata (DSCA) described in Muzy et al. [2005]. In fact, homogeneous regions can be simulated using the very efficient algorithms developed for DSCA models. Within any homogeneous region, the time advance function has the effect of turning a cell on (i.e., $ta(\cdot) = d/V$) or off (i.e., $ta(\cdot) = \infty$). Moreover, inactive cells have a unique quiescent state. This allows inactive cells to be removed from the simulator, and active cells to be created on demand. This can substantially reduce the memory needed to simulate a large cell space, thereby improving performance (e.g., Muzy et al. [2003, 2005]).

The wave simulation method is only weakly related to quantized state integration schemes (e.g., Zeigler et al. [2000], Kofman [2004], Nutaro et al. [2003], and Giambiasi et al. [2000]). Quantized state methods are characterized by their continuous time base and discrete state space. In contrast, the scheme presented here retains the discrete time base and continuous state space of FDTD methods.

This difference manifests itself in two ways. First, the time advance function is restricted to d/V and ∞. Quantized state methods use a time advance function whose range is all of the positive real numbers and ∞. Second, event rescheduling is not needed by the wave simulation; once a state transition is scheduled (i.e., when $ta(\cdot) \neq \infty$), it is guaranteed to occur at the specified time. Quantized state methods, on the other hand, require event rescheduling.

The errors exhibited by the discrete event scheme could be further reduced by using second-order approximations in the junction model. If a second-order extrapolation technique is used, then the discrete event scheme will exhibit first-order errors due only to uncertainty in the wave transit time at an interface. A second-order accurate approximation to the transit time would allow

for a second-order accurate discrete event scheme (i.e., the simulation error is proportional to d^2).

The simplest FDTD schemes for wave simulation are second-order accurate. The discrete event scheme presented in this article is first-order accurate, but it allows each region in an inhomogeneous material to be simulated with its optimal time step. This suggests that the difference between simulated and actual wave velocities will be smaller with the discrete event scheme than is possible with a simple FDTD scheme. Moreover, the discrete event scheme's simulated gain within a homogeneous region will more closely match the actual gain, relative to what is possible with a simple FDTD scheme.

Taken together, this implies that numerical dispersion and attentuation errors are likely, in general, to be substantially smaller with the discrete event scheme. This could result in errors that are comparable to, and in some instances better than, the second-order accurate FDTD scheme, in spite of the discrete event scheme being only first-order accurate. To establish or disprove this notion will require further experimentation.

REFERENCES

ACHENBACH, J. 1973. *Wave Propagation in Elastic Solids*. Elsevier, New York.

BEKEFI, G. AND BARRETT, A. H. 1977. *Electromagnetic Vibrations, Waves, and Radiation*. MIT Press, Cambridge.

BILBAO, S. D. 2004. *Wave and Scattering Methods for Numerical Simulation*. Wiley, New York.

BRILLOUIN, L. 1953. *Wave Propagation in Periodic Structures: Electric Filters and Crystal Lattices*. Dover, New York.

GIAMBIASI, N., ESCUDE, B., AND GHOSH, S. 2000. GDEVS: A generalized discrete event specification for accurate modeling of dynamic systems. *Simulation 17*, 3, 120–134.

JAMMALAMADAKA, R. 2003. Activity characterization of spatial models: Application to the discrete event solution of partial differential equations. M.S. thesis, University of Arizona, Tucson, Ariz.

KOFMAN, E. 2004. Discrete event simulation of hybrid systems. *SIAM J. Sci. Comput. 25*, 5, 1771–1797.

MUZY, A., AIELLO, A., SANTONI, P.-A., ZEIGLER, B. P., NUTARO, J. J., AND JAMMALAMADAKA, R. 2005. Discrete event simulation of large-scale spatial continuous systems. In *Proceedings of the International Conference on Systems, Man and Cybernetics (SMC)*. IEEE (Hawaii, USA).

MUZY, A., INNOCENTI, E., AIELLO, A., SANTUCCI, J.-F., AND WAINER, G. 2005. Specification of discrete event models for fire spreading. *Simulation 81*, 2, 103–117.

MUZY, A., INNOCENTI, E., SANTUCCI, J. F., AND HIL, D. R. C. 2003. Optimization of cell spaces simulation for the modeling of fire spreading. In *Proceedings of the 36th Annual Simulation Symposium*. IEEE (Orlando, Fla.), 289–296.

MUZY, A. AND NUTARO, J. 2005. Algorithms for efficient implementations of the DEVS & DSDEVS abstract simulators. In *Proceedings of the 1st Open International Conference on Modeling & Simulation*. ISIMA / Blaise Pascal University (France), 401–407.

NICOL, D., LIU, J., AND COWIE, J. 2000. Safe timestamps and large-scale modeling. In *Proceedings of the 14th Workshop on Parallel and Distributed Simulation*. IEEE (Bologna, Italy), 71–80.

NUTARO, J. J., ZEIGLER, B. P., JAMMALAMADAKA, R., AND AKERKAR, S. R. 2003. Discrete event solution of gas dynamics within the DEVS framework. In *Proceedings of the International Conference on Computational Science*, P. M. A. Sloot et al., eds. Lecture Notes in Computer Science, vol. 2660. Springer, Melbourne, Australia, 319–328.

SHLAGER, K. AND SCHNEIDER, J. 1998. A survey of the finite-difference time-domain literature. In *Computational Electrodynamics: The Finite-Difference Time-Domain Method*, A. Taflove, ed. Artech House, Boston, Mass. 1–62.

TAFLOVE, A. 1995. *Computational Electrodynamics*. Artech House, Boston, Mass.

TANG, Y., PERUMALLA, K., FUJIMOTO, R., KARIMABADI, H., DRISCOLL, J., AND OMELCHENKO, Y. 2005. Parallel discrete event simulations of physical systems using reverse computation. In *Proceedings of the ACM/IEEE/SCS Workshop on Principles of Advanced and Distributed Simulation (PADS)* (Monterey, Calif.).

WIELAND, F. 1997. The threshold of event simultaneity. In *Proceedings of the 11th Workshop on Parallel and Distributed Simulation*. IEEE (Lockenhaus, Austria). 56–69.

ZEIGLER, B. P., PRAEHOFER, H., AND KIM, T. G. 2000. *Theory of Modeling and Simulation, 2nd Ed*. Academic Press, San Diego, CA.

ZEIGLER, B. P., SARJOUGHIAN, H., AND PRAEHOFER, H. 2000. Theory of quantized systems: DEVS simulation of perceiving agents. *Cybernetics Syst. 31*, 6 (Sept.), 611–647.

Received April 2005; revised January 2006; accepted February 2006