INT. J. NUM. METH. HEAT FLUID FLOW, VOL. 4, 311–327 (1994)

TRANSIENT AND STEADY HEAT CONDUCTION USING AN ADAPTIVE FINITE ELEMENT CAD-BASED APPROACH

R. LÖHNER* AND J. McANALLY**

*Institute for Computational Sciences and Informatics, The George Mason University, Fairfax, VA 22030-4444, U.S.A.
**Huntsville Sciences Corp., Huntsville, AL 35806, U.S.A.

ABSTRACT

A new heat transfer simulation capability is described. Non-traditional features of this capability include: a seamless link to CAD-CAM for rapid problem specification/description, integrated automatic grid generator tools for rapid mesh generation, nonlinear and/or varying material properties, source-terms and boundary conditions, a one-element type approach for simplicity and efficiency, automatic self-adaptive mesh refinement and coarsening with accurate error estimation, heavy reliance on iterative solvers, and on-line display on workstations for immediate visualization and user feedback. These innovations are documented on several examples that demonstrate the usefulness of the developed capability.

KEY WORDS Heat transfer CAD-CAM Mesh generation

INTRODUCTION

The need to quickly simulate transient and steady heat transfer problems is common to many areas of engineering. Numerical algorithms for the simulation of heat transfer problems have been devised since the early 60's[1–4], and have found widespread acceptance in industry. However, up to very recently, these tools had to be used in batch-mode, only linear problems could be solved efficiently, and problem set-up time was a major stumbling block. In short, these tools were hard to use, requiring considerable expertise and training.

Recent advances in both numerical algorithms and computer hardware have made it possible to simulate efficiently even extremely complicated 3-D geometries that had proven intractable to date. The advances in numerical algorithms include automatic mesh generation and mesh adaptation algorithms, fast iterative solvers that exploit efficiently multiprocessor platforms, fast table look-ups for nonlinear material properties, and the widespread use of CAD-tools to define geometries. Computer hardware has seen the advent of supergraphics workstations that allow immediate visualization of results *on the fly*, and multiprocessor RISC or i860-based architectures.

The present paper describes a new set of algorithms and tools that combine in an efficient way the advances in these fields. The result is an adaptive finite element code for the simulation of transient and steady heat transfer problems that is closely coupled to CAD and on-line display tools for fast and error-free problem definition.

0961–5539/94/040311–18$2.00

Received August 1993

DESIGN CONSIDERATIONS

The following design criteria were followed for the development of the present set of tools:

(a) *Arbitrary Geometries:* an engineer designing a new product neither has the desire nor the time to go through the intellectual exercise of answering the question: will I be able to grid my new design idea? Therefore, he must be provided with tools that can quickly mesh any arbitrary domain. This naturally implies the use of unstructured grids (e.g. finite-element type grids).

(b) *Link to CAD-CAM:* the only way to quickly describe complex geometries is through the use of CAD-CAM tools. Therefore, any analysis tool has to be linked in a seamless manner to any available CAD-CAM tools. Given the variety of CAD-CAM packages used by industry, the links to establish should be as generic as possible, without sacrifice of performance.

(c) *Fast Gridding:* an engineer needs the result as fast as possible. Currently, the biggest bottleneck facing 3-D simulations is not CPU-time, but gridding time. The only automatic, fast grid generators currently available generate tetrahedral meshes[5–8]. Therefore, we will concentrate on triangles and tetrahedra for the choice of the spatial discretization. Remember that the mesh is only a vehicle to the solution, not an end in itself. Ideally, the user should not be aware of the mesh, and should not be forced to make decisions as to what element type or grid density a certain region of space requires.

(d) *Iterative Solvers:* Given the size of typical 3-D problems, and the rapid advent of parallel machines, the most rational way of solving the large coupled systems of equations that arise due to finite element discretization is by iterative solvers. Iterative solvers are also ideally suited for nonlinear applications, where no multiple right-hand side solutions via direct solvers are possible.

(e) *Simple Elements:* in order to expedite calculations for iterative solvers, and to achieve optimal vectorization, simple elements are preferred over elements of higher order.

(f) In order to reduce software complexity and maintenance costs, the method should be the same in 2-D as in 3-D.

(g) In order to be applicable to a large range of real-life problems, fast table look-up algorithms for nonlinear materials have to be provided.

LINK TO CAD-CAM

The dominant cost of 3-D simulations for complex geometries is given by the man-hours required to define the surfaces of the domain to be gridded. Once this surface has been defined, the generation of a suitable mesh using automatic grid generators is straightforward. We call this stage the pre-generation stage of a typical simulation. Every grid generator currently in use has its own way of defining surfaces and reading in data. In order to expedite the surface definition process we have developed a pre-generation tool that will take input from a variety of formats, e.g. IGES, panel-data, AutoCad, etc., and manipulate it in such a form that an error-free input file for the grid generator is created. Other options provided by the pre-generation tool are boundary condition definitions, and the merger of several individual input files, e.g. created by several co-workers in a team, to create a final detailed geometry. The idea of defining geometry and boundary conditions simultaneously is important, as it allows fully automatic mesh generation and adaptation.

Although perhaps scientifically not very relevant, this pre-generation tool has opened the possibility of simulating problems whose geometrical complexity was beyond the scope of current techniques, and which had therefore not been computed in detail before.

AUTOMATIC GRID GENERATION

A review of the literature indicates that the only truly automatic grid generators currently in use generate triangular or tetrahedral meshes. By automatic we mean that once the surface of the domain to be gridded has been defined, the surface and volume gridding proceeds without user intervention. For our purposes, we employ the advancing front technique to generate triangular, 2-D, and tetrahedral, 3-D, elements. The algorithm may be summarized as follows[7–9]:

- F.1 Define the boundaries of the domain to be gridded. This is accomplished by splines in 2-D and surface patches in 3-D.
- F.2 Define the spatial variation of element size, stretchings, and stretching directions for the elements to be created. This is accomplished with a combination of background grids and sources.
- F.3 Using the information stored on the background grid, set up faces on all these boundaries. This yields the initial front of faces. At the same time, find the generation parameters (element size, element stretchings and stretching directions) for these faces from the background grid and sources.
- F.4 Select the next face to be deleted from the front; in order to avoid large elements crossing over regions of small elements, the face forming the smallest new element is selected as the next face to be deleted from the list of faces.
- F.5 For the face to be deleted:
  - F.5.1 Select a *best point* position for the introduction of a new point **IPNEW**.
  - F.5.2 Determine whether a point exists in the already generated grid that should be used in lieu of the new point. If there is such a point, set this point to **IPNEW** and continue searching (go to F.5.2).
  - F.5.3 Determine whether the element formed with the selected point **IPNEW** does not cross any given faces. If it does, select a new point as **IPNEW** and try again (go to F.5.3).
- F.6 Add the new element, point, and faces to their respective lists.
- F.7 Find the generation parameters for the new faces from the background grid and sources.
- F.8 Delete the known faces from the list of faces.
- F.9 If there are any faces left in the front, go to F.4.

The specific data structures used for search operations, as well as further algorithmic details can be found in References 7, 9. The particular implementation used in the present work generates grids at a rate of 12,000 tetra/min on the IBM RISC-6000/550 workstation. The triangular or tetrahedral grid obtained from the advancing front technique can also be combined with material data to produce a SINDA[10] compatible input file.

HEAT TRANSFER SOLVER

The transient heat transfer equation given by:

$$\rho c_p T_{,t} = \nabla \cdot \mathbf{k} \nabla T + S \tag{1}$$

$$T = T_0 \quad \text{on } \Gamma_D, \qquad q_n := \mathbf{n} \cdot \mathbf{k} \nabla T = q_0 + \alpha(T - T_1) + \beta(T^4 - T_2^4) \quad \text{on } \Gamma_N \tag{2}$$

where $\rho, c_p, T, \mathbf{k}, S, T_0, \alpha, \beta, T_1, T_2$ denote the density, heat capacitance, temperature, conductivity tensor, sources, prescribed temperature, prescribed fluxes, film coefficient, radiation coefficient and external temperatures respectively, is solved using standard finite element procedures. This

implies making the assumption:

$$T \approx \sum N^i(\mathbf{x}) T_i(t) \tag{3}$$

where N^i, T_i denote the shape-function and temperature associated with node i. Note that the shape-function is assumed independent of time, i.e. no mesh movement in time is considered. The Galerkin weighted residual statement:

$$\int_\Omega N^j \rho c_p N^i \, d\Omega T_{i,t} = -\int_\Omega \nabla N^j k \nabla N^i \, d\Omega T_i + \int_\Omega N^j S \, d\Omega + \int_{\Gamma_N} N^j q_n \, d\Gamma \tag{4}$$

leads to a matrix system for the vector of unknown temperatures $\mathbf{T}$ of the form:

$$\mathbf{M} \cdot \mathbf{T}_{,t} = \mathbf{K} \cdot \mathbf{T} + \mathbf{s} \tag{5}$$

The temporal discretization of this coupled system of equations is accomplished using a standard finite difference algorithm, e.g.:

$$\left[\frac{1}{\Delta t} \mathbf{M} - \Theta \mathbf{K} \right] \cdot \Delta \mathbf{T} = \mathbf{K} \cdot \mathbf{T} + \mathbf{s} \tag{6}$$

Here, Δt denotes the time-step taken, $\Delta \mathbf{T}$ the vector of nodal temperature increments, $\mathbf{M}$ the mass or capacitance matrix, $\mathbf{K}$ the conductivity matrix, $\mathbf{s}$ the assembled nodal source-vector, and Θ an implicitness parameter. For $\Theta > 0.5$, an unconditionally stable implicit timestepping scheme is obtained, whereas $\Theta = 0$ and a lumped mass-matrix $\mathbf{M}$ yields an explicit scheme. The large coupled system of linear equations obtained on the left in (6) is never solved directly: 3-D problems are simply too large for direct solvers, and nonlinear applications make iterative solvers a much better choice. Moreover, for nonlinear applications, the advantage of being able to solve for multiple right-hand sides with a direct solver disappears, making iterative solvers the method of choice. In the present case, a conjugate gradient solver[11] with diagonal preconditioning is employed. Iterative solvers require many right-hand side like evaluations. In order to expedite these, all integrals are evaluated in closed form.

Great care was taken to achieve near-optimal performance throughout the computer platforms envisoned, i.e. from RISC workstations through vector-supercomputers to MIMD machines. For the workstation applications, the points and elements are renumbered in order to reduce cache-misses[12]. For vector-supercomputers the elements are renumbered or *coloured*[12], in order to achieve optimal vectorization during the assembly stage. Finally, for MIMD machines the computational domain is split up and allocated to the different processors[13].

NONLINEAR MATERIALS, SOURCES AND BOUNDARY CONDITIONS

In order to be applicable to a wide range of problems, the heat transfer algorithm was designed *ab initio* for nonlinear materials, as well as nonlinear and time-dependent sources and boundary conditions. The most general way to input nonlinear material properties is through a table look-up. Given the temperature, the corresponding density, heat capacitance and conductivity (ρ, c_p, k in (1), (2)) are obtained from linear interpolation. As the changes between time-steps can be assumed to be limited, the following nearest neighbour marching algorithm yields a fast table look-up scheme:

T.0 Assume given: temperature T and the table entry for the last time-step **ITABL**
T.1 Obtain the temperatures bounding the current table entry interval: T_0, T_1, with $T_0 < T_1$
T.2 If $T < T_0$: set **ITABL** = **ITABL** − 1 and goto T.1
T.3 If $T > T_1$: set **ITABL** = **ITABL** + 1 and goto T.1

T.4 Interpolate linearly the material properties:

$$\xi = (T - T_0)/(T_1 - T_0)$$
$$\rho = (1 - \xi)\rho_0 + \xi\rho_1$$
$$c_p = (1 - \xi)c_{p_0} + \xi c_{p_1}$$
$$k = (1 - \xi)k_0 + \xi k_1$$

Typically, convergence to the correct interval in steps T.2, T.3 is attained in one to two passes over the elements. This table look-up algorithm is fully vectorizable. For multimaterial applications, each element in the domain is given a material number. The table look-up then works on several materials simultaneously. The penalty in CPU time incurred by switching from constant material properties to general, nonlinear material properties is less than 10%.

We have found that practitioners in the field tend to have complicated source terms, making it impossible to standardize them. Therefore, we allow the user to specify in a subroutine the particular source term desired. The subroutine allows access to all relevant parameters (temperatures, coordinates, material properties, etc.), but does not allow to alter these. This subroutine is linked to the heat transfer object code at run-time.

For the boundary conditions, we follow the same route as for the source-terms. Several surface patches can be lumped together into a so-called 'environment'. The appropriate imposed or external temperatures, fluxes, film and radiation coefficients are coded by the user for the particular application. As before, this subroutine is linked to the heat transfer object code at run-time.

ADAPTIVE REFINEMENT

For the adaptation of the mesh, the classic *h*-refinement and coarsening is employed. We prefer *h*-refinement[12–17] over remeshing[18–20] as:

(a) one of our main aims is the accurate tracking of transient heat transfer phenomena,
(b) *h*-refinement leads to a very fast adaptation procedure both in 2-D and 3-D, and
(c) *h*-refinement can be used as a user-friendly *black box*.

The method is described in References 16, 17. The basic idea is to subdivide elements wherever the error exceeds a desired value. Conversely, the mesh is coarsened if the measured error becomes too low. The error estimation is divided into a steady-state portion that governs the spatial resolution, and a transient portion that governs the temporal resolution. For the spatial resolution, an interpolation error indicator in the elements of the form:

$$\varepsilon_{ss} = \frac{c_1}{k_d} \max_{\text{edges}} |(\mathbf{x}^1 - \mathbf{x}^0)\cdot(\mathbf{q}^1 - \mathbf{q}^0)|, \qquad k_d = \mathbf{k}\cdot\frac{(\mathbf{x}^1 - \mathbf{x}^0)}{|\mathbf{x}^1 - \mathbf{x}^0|} \tag{7}$$

is employed. Here the max-operation refers to the edges of the element, c_1 denotes an empirical constant (experience indicates that $c_1 = 0.23$ gives reliable results), k_d the side-projected conductivity, $\mathbf{x}^{0,1}$ the coordinates of the edge-endpoints and $\mathbf{q}^{0,1}$ the heat-fluxes at the edge-endpoints. The 1-D, constant k rationale for this error indicator may be inferred from *Figure 1*. To first order, the higher order components of the solution which cannot be represented by the linear finite element solution are given by the difference of the derivatives at the nodes $u^{1,0}_{,x}$ times the element length. The point-wise heat-fluxes for the general, multidimensional and multimaterial problem are recovered using the following averaging procedure:

$$\int_\Omega N^i N^j \, d\Omega \mathbf{q}_j = \int_\Omega N^i \mathbf{k} \nabla N^j \, d\Omega T_j \tag{8}$$

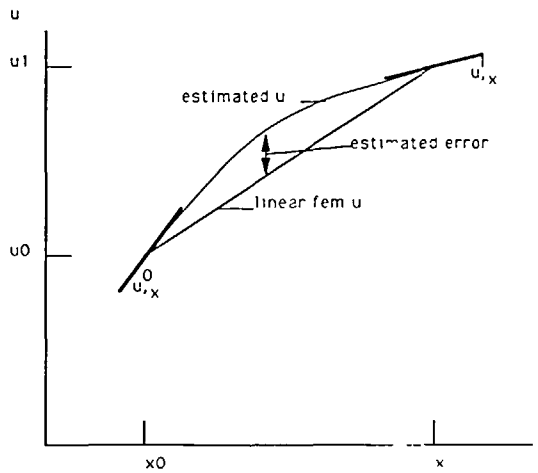


Figure 1 Error estimation for linear finite elements

One can easily see that this error estimator is of the form:

$$\varepsilon_{ss} = ch^2|T|^2 \tag{9}$$

which is to be expected for linear elements, and that it can also be related to the Zienkiewicz–Zhu[21] error estimator. In order to test this error indicator, we used the following model problem with exact solutions (see *Figure 2*):

$$\nabla^2 T + S = 0, \qquad T = 0 \quad \text{on } \Gamma_D, \qquad \mathbf{n}\cdot\nabla T = 0 \quad \text{on } \Gamma_N \tag{10}$$

This problem has the exact solutions:

$$T = 3x + 5x^3 + 3x^5 - x^6 \quad \text{for } S = 30x(1-x)(1+x(1-x)) \tag{11a}$$

$$T = \frac{1}{\omega^2}\sin(\omega x) \qquad \text{for } S = \sin(\omega x), \omega = 2\pi \tag{11b}$$

For each of these two problems, a coarse and a fine mesh were run respectively. The results obtained, as well as the exact solutions, are shown in *Figures 2a, b*. The error estimated by (7) was compared to the error ε_{ex} obtained from subtracting the exact from the numerical solution. The discrepancy between ε_{ss} and ε_{ex} was quantified by the following sum:

$$r = \sqrt{\frac{\sum_{el} (\varepsilon_{ss} - \varepsilon_{ex})^2}{\sum_{el} \varepsilon_{ex}^2}} \tag{12}$$

which gives a measure of the average discrepancy. The results obtained for the two different source-terms given by (11a, b) on the two meshes are recorded in *Table 1*. As one can see, the solution error is estimated by (7) to within 5% for all cases, which we find remarkable.

For the temporal resolution, we simply use:

$$\varepsilon_{tr} = c_2|T^{n+1} - T^n| \tag{13}$$

i.e. the pointwise variation of the temperature in time (we set $c_2 = 1.0$ for all the cases described below).

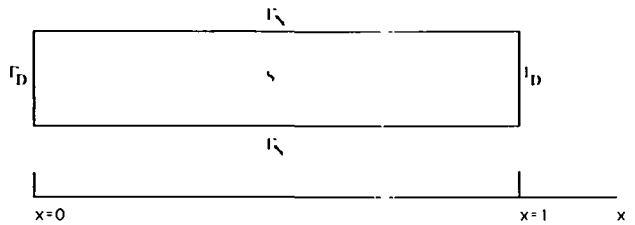


Figure 2 Model problem definition

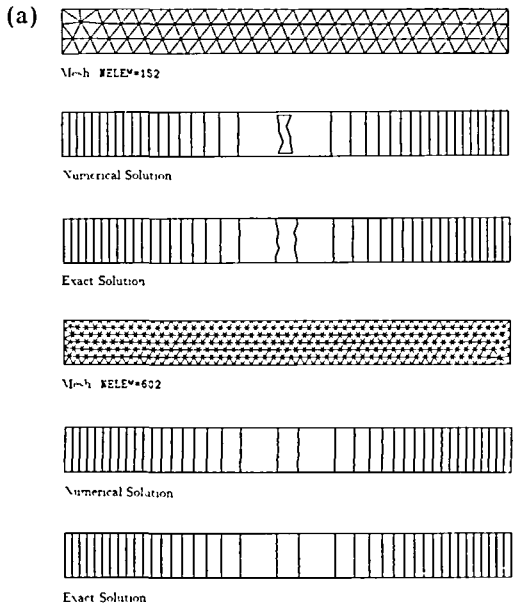


Figure 2a Error estimation test: $\nabla^2 T = 30x(1-x)(1+x(1-x))$. Temperature: $T_{min}=0$, $T_{max}=1$, $\delta T_c = 0\,05$

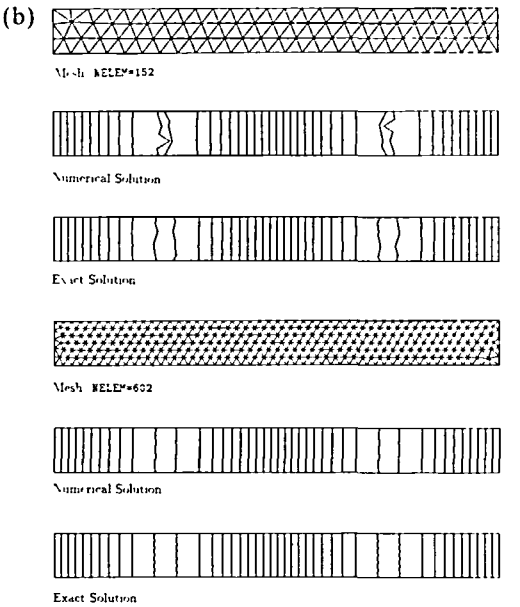


Figure 2b Error estimation test: $\nabla^2 T = \sin(2\pi x)$ Temperature: $T_{min} = -0.025$, $T_{max} = 0\,025$, $\delta T_c = 0.0025$

Table 1 Actual and estimated errors for model problems

Source	NELEM	r
Eqn. (11a)	152	0.0151
Eqn. (11a)	602	0.0112
Eqn. (11b)	152	0.0369
Eqn. (11b)	602	0.0145

Table 2 Run-times for 3-D cylinder problem

Mesh	NELEM	CPU (secs)
Coarse	3,327	159
Adapted	11,000	5,116
Fine	138,985	53,000

These error indicators are used to refine the mesh in space (local h-refinement) or time (increase/decrease of the timestep Δt employed). Once the user sets a desired temperature tolerance ΔT, the algorithm will refine/coarsen both the mesh and the timestep used automatically. One of the main hidden advantages accrued from adaptive meshing procedures as the one used for the present research is user-friendliness: the procedure automatically steers towards a mesh that is optimally suited to the problem at hand, freeing the user from an ill-defined (in particular for transient problems) and arduous task.

ON-LINE DISPLAY

The advent of supergraphics workstations has allowed the solution of large 3-D problems in combination with immediate, on-line display of results. This combination has proven extremely powerful in practice, enhancing understanding of physical processes and avoiding input and problem set-up errors. Therefore, the capability to display immediately the results as they are being computed constitutes an important ingredient of any modern analysis tool.

RESULTS

Joint cooling

The problem geometry, as well as the material parameters used, are displayed in *Figure 3a*. The initially hot material ($T_i = 1200°C$) is immersed at time $t = 0$ into an oilbath at $T_0 = 300°C$. All the external surfaces are assumed to be subjected to a convection condition with a film coefficient $\alpha = 1$ Kcal/m^2 s °C. The accuracy required for the temperature field during all the stages of this transient calculated was set to $T_{acc} = +/-10°C$, corresponding to about 1% of the initial temperature range. *Figures 3b, c* show the solutions and the grids corresponding to times $t = 0.1805$, $t = 0.5338$, $t = 1.379$ and $t = 150.0$ respectively. The last result is the expected steady-state in which the whole domain has reached the outside temperature. Observe the different levels of adaptation which change as the solution evolves in time.

Casting solidification

In order to demonstrate the use of several materials with temperature-dependent material properties, a casting solidification problem was simulated. The geometry, as well as the material

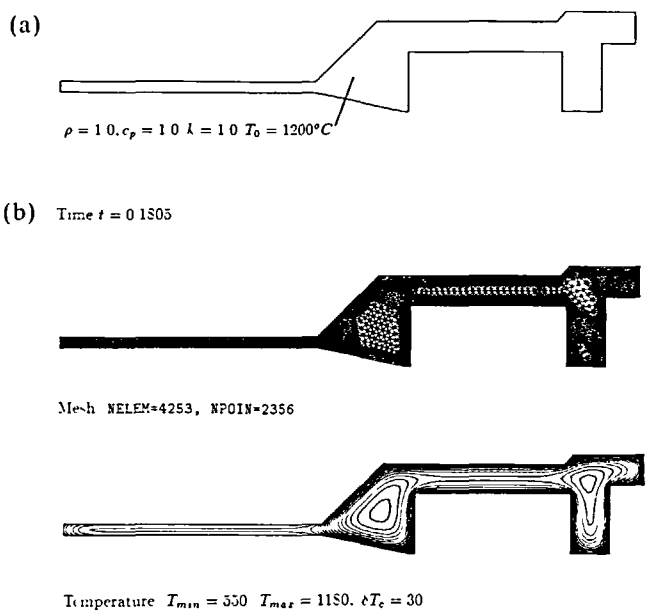
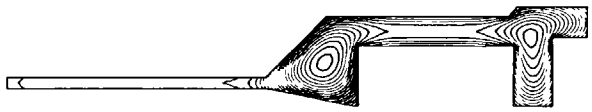


Figure 3 Joint cooling

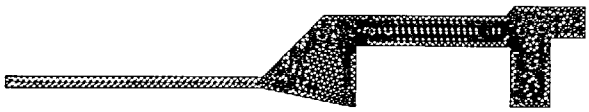
(c) Time $t = 0.5338$

Mesh NELEM=2977, NPOIN=1628

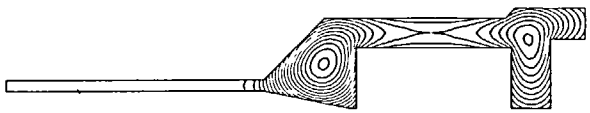


Temperature $T_{min} = 340$, $T_{max} = 1060$, $\delta T_c = 30$

(d) Time $t = 1.379$

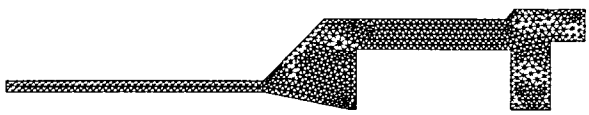


Mesh NELEM=1696, NPOIN=972

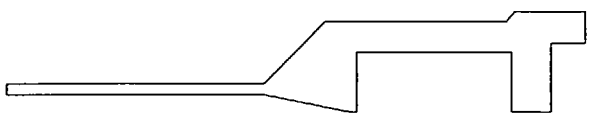


Temperature $T_{min} = 300$, $T_{max} = 700$, $\delta T_c = 20$

(e) Time $t = 150.0$



Mesh NELEM=1160, NPOIN=703



Temperature. $T_{min} = 300$ $T_{max} = 300$, $\delta T_c = 0$

Figure 3 continued

parameters and boundary conditions are shown in *Figure 4a*. Observe that the heat capacitance of the melt is temperature-dependent, and exhibits a jump at the point of solidification. The accuracy required for the temperature was set to $T_{acc} = +/-5°C$. *Figures 4b–e* show the solutions and the grids corresponding to times $t = 0.041$, $t = 1.394$, $t = 8.027$ and $t = 18.0$ respectively. As before, observe the different levels of adaptation which change as the solution evolves in time.

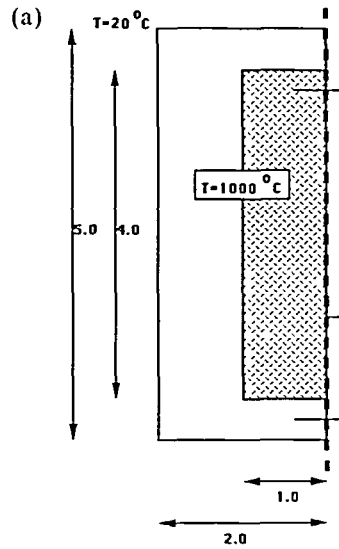


Table 1 Material Properties for Region 1

Temperature	ρ	c_p	k
0	1	1	2
690	1	1	2
700	1	1000	2
710	1	1	2
1000	1	1	2

Table 2 Material Properties for Region 2

Temperature	ρ	c_p	k
1	1	1	1

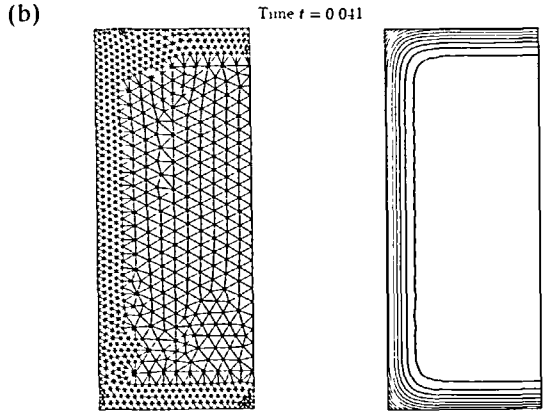


Figure 4 Casting solidification problem definition

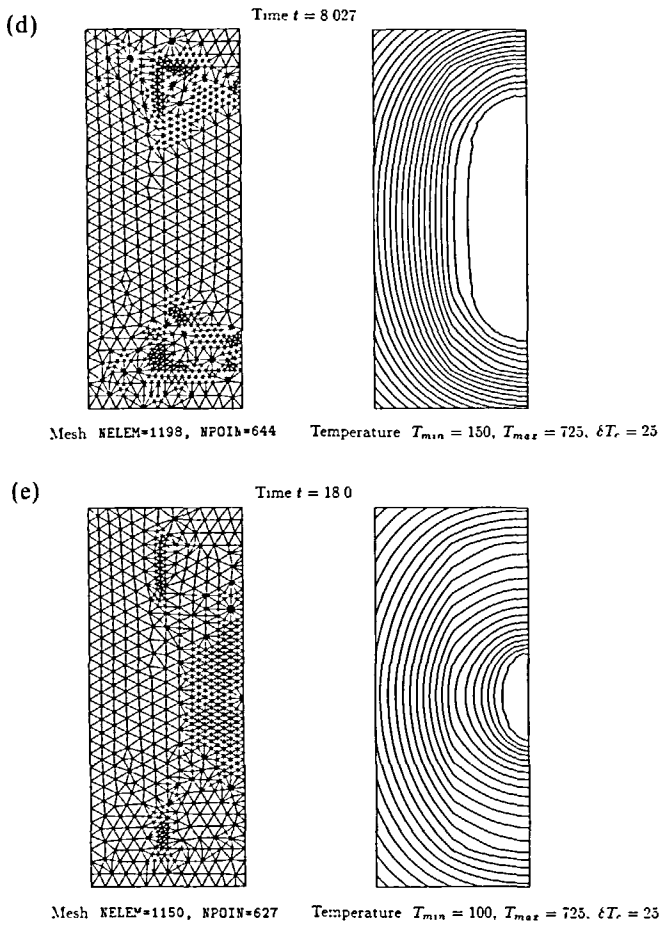
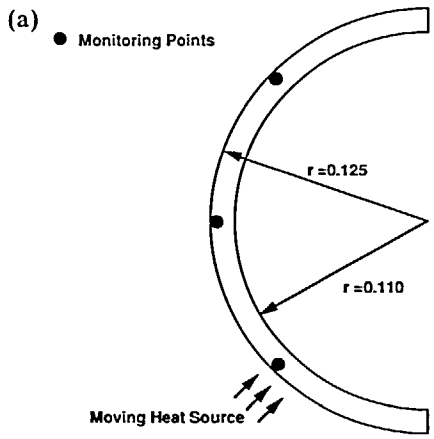


Figure 4 continued

Moving heat source on cylindrical surface

This example, extended to 3-D from the 2-D example shown in Reference 20, illustrates the advantage of the adaptive refinement capability for transient heat transfer problems. A heat source of constant magnitude of $S = 26{,}000$ BTU/ft^2 s and width $\delta = 0.01$ ins is applied to the external surface of a half-cylinder of outer radius $r_0 = 0.125$ in and inner radius $r_i = 0.110$ ins. The heat source is initially positioned at the lower edge of the half-cylinder and subsequently moves around the cylinder at a speed of $v = 2$ ins/s. This type of heat loading may be found in aerospace vehicles that travel and maneuver at supersonic speeds. The internal surface of the cylinder is subjected to a convection condition with a film coefficient $\alpha = 7.8$ BTU/ft^2 s °R and a surrounding temperature of $T_0 = 50$°R. Adiabatic (no heat-flux) boundary conditions are imposed for all other surfaces. The material properties for the cylinder were assumed constant, with $\rho = 0.283$lb m/in^3, $c_p = 0.1825$ BTU/lb m^0 R, and $k = 2 \times 10^{-4}$ BTU/ins s °R. The accuracy required for the temperature was set to $T_{acc} = +/-100$°R, corresponding to about 3% of the temperature range. *Figures 5b–d* show the solutions and the grids in the x/y plane corresponding to times $t = 0.05$, $t = 0.10$ and $t = 0.15$ respectively. A maximum of two levels of refinement were specified. This yielded grids with an average number of approximately **NELEM** = 11,000 elements. In order to assess both the accuracy and the performance of the

adaptive procedure, a coarse mesh of **NELEM** = 3,237 elements and a uniformly fine mesh that had the same element size as the adaptively refined mesh, consisting of **NELEM** = 138,985 elements, were run besides the adaptive mesh. The temperature proiles obtained at the three outer monitoring points shown in *Figure 5a* are displayed in *Figure 5e*. Observe that the adaptively refined mesh captures very well the rise in temperature as the source moves around the outer cylinder, whereas the coarse mesh misses the maximum temperature by about 10%. The CPU times recorded for the different runs have been summarized in *Table 2*. One can see that a significant gain in CPU performance is achieved without sacrificing accuracy: 3-D runs in particular are unforgiving for uniformly fine meshes.



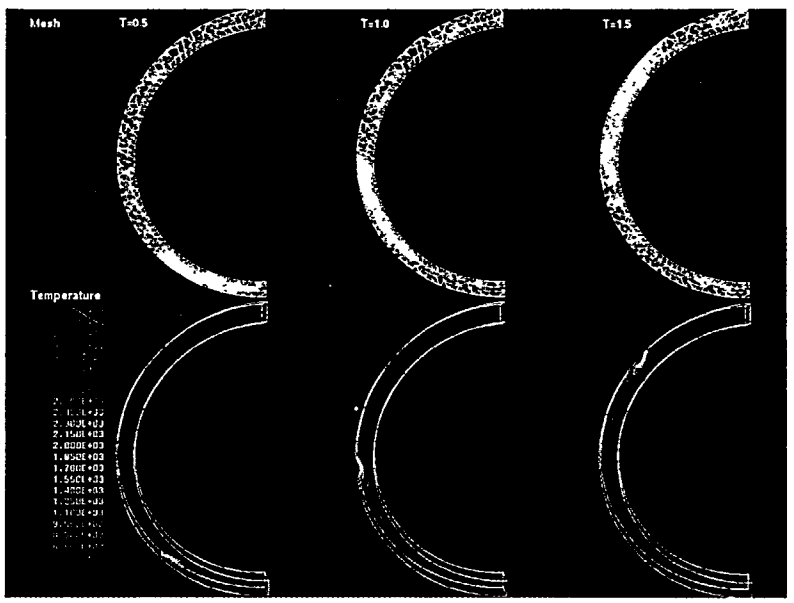


Figure 5a **Problem definition**

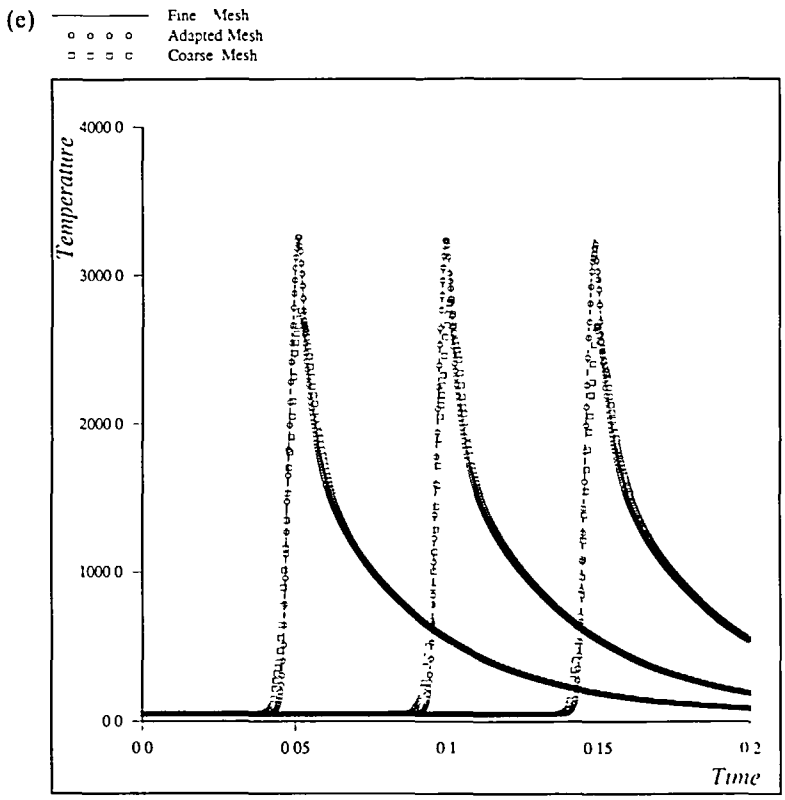


Figure 5 continued

SSME bearing

This case demonstrates the use of the techniques presented for a more realistic geometry. The part for which the temperature fields and heat fluxes are computed belongs to the Space Shuttle main engine (SSME). Starting from blueprints, we used our pre-generation tool in order to define the surfaces. A total of 700 surface patches were required to model the bearing accurately. The surface definition process took half a day. The wireframe of the surface is shown in *Figure 6a*. The boundary conditions were given as follows: The external surface was assumed to be subjected to a convection condition with a film coefficient $\alpha_{ext} = 1.0$ and a surrounding temperature of $T_{ext} = 500°$. Three liquid propellant entry ports are given at the center of the bearing, and the cold fluid is transported through the holes in the bearing (see *Figure 6a*). The heat transfer imposed by the liquid propellant was modelled using a convection condition with film coefficients and external temperatures of $\alpha_{lox} = 2.0$, $T_{lox} = -200°$, $\alpha_{Hpri} = 2.0$, $T_{Hpri} = -150°$, and $\alpha_{Hsec} = 2.0$, $T_{Hsec} = -100°$ respectively. The material properties for the bearing were assumed constant, with $\rho = 1, c_p = 1$, and $k = 1$. The surface of the mesh, which consisted of approximately **NELEM** = 280,000 elements and **NPOIN** = 50,000 points is shown in *Figure 6b*. Note that although this number of elements and points seems high, this is the minimum required for a uniform mesh to define reasonably well the geometry of the bearing. The steady-state results, obtained after approximately 10 min of run-time on an IBM-RISC-6000/550, are displayed in *Figures 6c, d*. The error estimated was $+/-10°C$, or 1% of the temperature range. This was deemed within the required accuracy for engineering purposes. For this reason, mesh adaptation was not invoked to enhance further the accuracy of the solution.

(a)

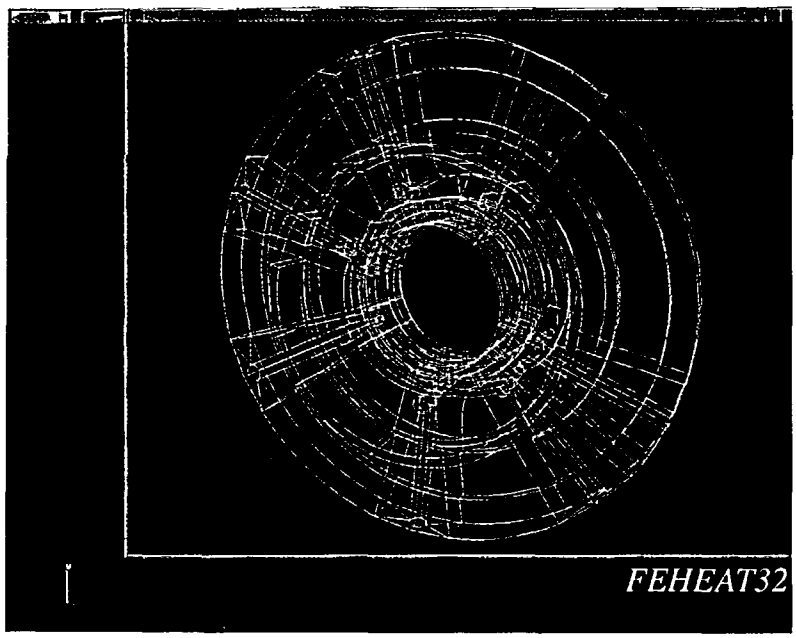


Figure 6a SSME bearing. Wireframe defining surface

(b)

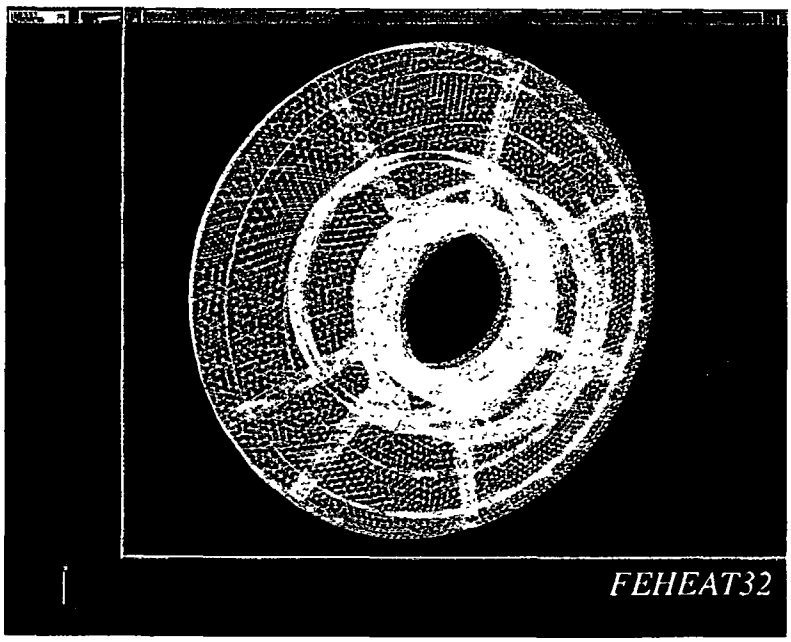


Figure 6b Surface of mesh: NELEM = 280K, NPOIN = 50K

(c)

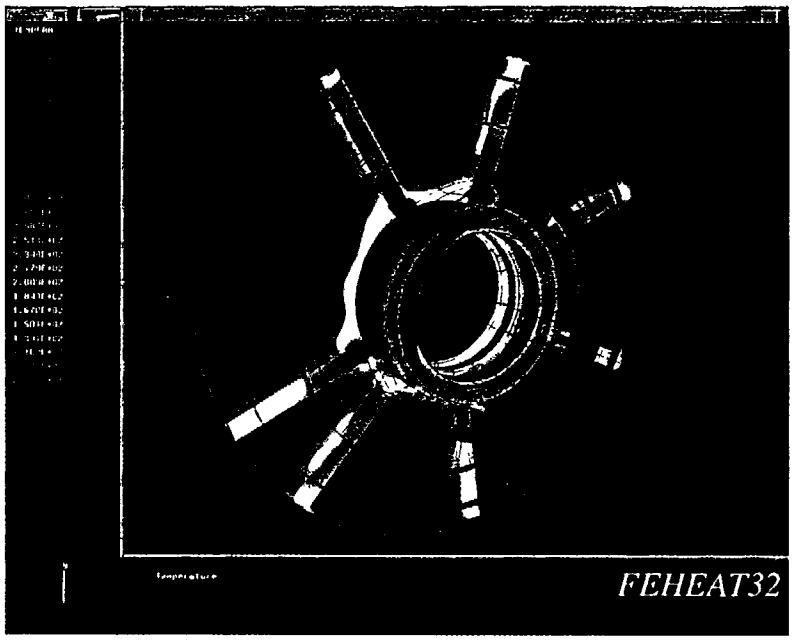


Figure 6c Temperature on the surface: $T_{min} = -300.0$, $T_{max} = 500$ (backface removal off)

(d)

Figure 6d Temperature on the surface: $T_{min} = -300.0$, $T_{max} = 500$ (backface removal on)

CONCLUSIONS

A new nonlinear heat transfer simulation capability has been described. Important innovative design features of this capability, that set it apart from traditional heat transfer capabilities are:

- A seamless link to CAD-CAM for rapid problem specification/description;
- An integrated automatic grid generator tool for rapid mesh generation;
- Heavy emphasis on nonlinear and/or varying material properties, source-terms and boundary conditions;
- A one-element type approach, i.e. only linear triangles/tetrahedra for simplicity, efficiency and low software maintenance costs;
- Automatic self-adaptive mesh refinement and coarsening with accurate error estimation;
- Preconditioned iterative solvers for the rapid solution of large 3-D problems (> 50 K nodes);
- Full vectorization for efficient use on the current generation of industrial super-computers;
- On-line display on workstations for immediate visualization and user feedback.

Future developments will center on better on-line visualization for 3-D problems, inclusion of radiation effects for absorbing media, optimum design as an option, and porting of the algorithms to a MIMD environment[13].

ACKNOWLEDGEMENTS

This work was supported by the NASA Marshall Engineering Center under an SBIR contract. The first author would also like to acknowledge the support of IBM in the form of a RISC-6000/550 workstation. Most of the results presented were carried out on it.

REFERENCES

1 Visser, W. A finite element method for the determination of non-stationary temperature distribution and thermal deformation, *Proc. Conf. on Matrix Meth. Struct. Mech.*, Air Force Int. Tech., Wright-Patterson AFB, Ohio (1965)
2 Zienkiewicz, O. C. and Cheung, Y. K. Finite elements in the solution of field problems, *The Engineer*, 507–510 (1965)
3 Collatz, L. *The Numerical Treatment of Differential Equations*, Springer Verlag (1966)
4 Zienkiewicz, O. C., Arlett, P. L. and Bahrani, A. K. Solution of three-dimensional field problems by the finite element method, *The Engineer*, **27** (1967)
5 Yerry, M. A. and Shepard, M. S. Automatic three-dimensional mesh generation by the modified-octree technique, *Int. J. Num. Meth. Eng.*, **20**, 1965–1990 (1984)
6 Baker, T. J. Developments and trends in three-dimensional mesh generation, *Appl. Num. Math.*, **5**, 275–304 (1989)
7 Löhner, R. and Parikh, P. Three-dimensional grid generation by the advancing front method, *Int. J. Num. Meth. Fluids*, **8**, 1135–1149 (1988)
8 Peraire, J., Morgan, K. and Peiro, J. Unstructured finite element mesh generation and adaptive procedures for CFD, *AGARD-CP-464*, **18** (1990)
9 Löhner, R. Some useful data structures for the generation of unstructured grids, *Comm. Appl. Num. Meth.*, **4**, 123–135 (1988)
10 Gaski, J. and Collins, R. L. *SINDA 1987-ANSI Revised User's Manual*, Network Analysis Associates, Inc. (1987)
11 Hestenes, M. and Stiefel, E. Methods of conjugate gradients for solving linear systems, *J. Res. Nat. Bur. Standards*, **49**, 409–436 (1952)
12 Löhner, R. Finite elements in CFD: grid generation, adaptivity and parallelization; Chapter 8 in *AGARD Rep. 787, Proc. Special Course on Unstructured Grid Methods for Advection Dominated Flows*, VKI, Belgium (1992)
13 Löhner, R., Ramamurti, R. and Martin, D. A parallelizable load balancing algorithm, *AIAA-93-0061* (1993)
14 Berger, M. J. and Oliger, J. Adaptive mesh refinement for hyperbolic partial differential equations, *J. Comp. Phys.*, **53**, 484–512 (1984)
15 Oden, J. T., Devloo, P. and Strouboulis, T. Adaptive finite element methods for the analysis of inviscid compressible flow: I. Fast refinement/unrefinement and moving mesh methods for unstructured meshes, *Comp. Meth. Appl. Mech. Eng.*, **59**, 327–362 (1986)
16 Löhner, R. An adaptive finite element scheme for transient problems in CFD, *Comp. Meth. Appl. Mech. Eng.*, **61**, 323–338 (1987)

17 Löhner, R. and Baum, J. D. Adaptive H-refinement on 3-D unstructured grids for transient problems, *Int. J. Num. Meth. Fluids*, **14**, 1407–1419 (1992)
18 Peraire, J., Vahdati, M., Morgan, K. and Zienkiewicz, O. C. Adaptive remeshing for compressible flow computations, *J. Comp. Phys.*, **72**, 449–466 (1987)
19 Löhner, R. Three-dimensional fluid-structure interaction using a finite element solver and adaptive remeshing, *Computer Systems in Eng.*, **1**, 2–4, 257–272 (1990)
20 Probert, E. J., Hassan, O., Morgan, K. and Peraire, J. Adaptive explicit and implicit finite element methods for transient thermal analysis, *Int. J. Num. Meth. Eng.*, **35**, 655–670 (1992)
21 Zienkiewicz, O. C. and Zhu, J. Z. A simple error estimator and adaptive procedure for practical engineering analysis, *Int. J. Num. Meth. Eng.*, **24**, 337–357 (1987)