42nd AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit
9 - 12 July 2006, Sacramento, California

AIAA 2006-5043

Recent Progress Towards a Rule-Based Computational Tool for Liquid Rocket Combustion

Edward Luke[*], Siddharth Thakur[**], David Thompson[*], Jeffrey Wright[§], and Wei Shyy[§§]

[*]*Mississippi State University, Mississippi State, MS*
[**]*University of Florida, Gainesville, FL*
[§]*Streamline Numerics, Inc., Gainesville, FL*
[§§]*University of Michigan, Ann Arbor, MI*

The progress in the development of a new computational tool called Loci-STREAM is presented in this paper. Loci-STREAM is a pressure-based, Reynolds-averaged Navier-Stokes (RANS) solver for generalized unstructured grids, which is designed to handle all-speed flows (incompressible to hypersonic) and is particularly suitable for solving multi-species flow in fixed-frame combustion devices. Loci-STREAM integrates proven numerical methods for generalized grids and state-of-the-art physical models in a novel rule-based programming framework called *Loci* which allows: (a) seamless integration of multidisciplinary physics in a unified manner, and (b) automatic handling of massively parallel computing. The objective of the ongoing work is to develop a robust a robust simulation capability for combustion problems in rocket engines. As an initial step towards validating this capability, a model problem is investigated in the present study which involves a gaseous oxygen/gaseous hydrogen (GO2/GH2) shear coaxial single element injector.

Nomenclature

C_{pi}	=	specific-heat
h	=	specific enthalpy
h_{0i}	=	standard heat of formation of species *i*
H	=	total (or stagnation) enthalpy
k	=	turbulence kinetic energy
M_w	=	molecular weight
P_k	=	production of *k*
p	=	pressure
p'	=	pressure correction
Pr_L	=	laminar Prandtl number
q_j	=	heat flux vector
R	=	gas constant
T	=	temperature
u_i	=	velocity vector
u', v', w'	=	velocity corrections
x,y,z	=	Cartesian coordinates
μ	=	molecular viscosity
μ_t	=	turbulent (eddy) viscosity
ρ	=	density
Y_i	=	mass fraction of species *i*
τ_{ij}	=	viscous stress tensor
δ_{ij}	=	Kronecker delta
κ	=	thermal conductivity
ε	=	rate of dissipation of *k*
$\dot{\omega}$	=	chemical heat release source terms

I. Introduction

The traditional design and analysis practice for advanced propulsion systems, particularly chemical rocket engines, relies heavily on expensive full-scale prototype development and testing. Over the past decade, use of high-fidelity analysis and design tools such as CFD early in the product development cycle has been identified as one way to alleviate testing costs and to develop these devices better, faster and cheaper. In the design of advanced propulsion systems, CFD plays a major role in defining the required performance over the entire flight regime, as well as in testing the sensitivity of the design to the different modes of operation. Computational modeling tools, along with selected ground and flight testing, are expected to be the prime engineering approach of choice in the development of space propulsion systems. Thus, increased emphasis is being placed on developing and applying CFD models to simulate the flow field environments and performance of advanced propulsion systems. This necessitates the development of next generation computational tools which can be used effectively and reliably in a design environment by non-CFD specialists.

A critical sub-system in the performance, life-cycle cost, and robustness of the overall rocket engine is the thrust chamber assembly. This assembly includes the manifolds which route propellant around and through the assembly, the injectors, the thrust chamber, the nozzle, and the ignition system. The goal of the present work is to develop a computational tool capable of real-fluid simulation of the entire rocket engine starting from both fuel and oxidizer ducts, through each respective manifold, including the fuel regeneration loop, and to the injector and combustion chamber, ending at the nozzle exit of the engine. To reflect the true physics, this tool must be multidisciplinary, able to handle reacting-fluid flow and solid-phase heat transfer simultaneously. There is a critical need for a robust simulation capability of steady and unsteady flows with real-fluid properties in combustion devices and other related flows such as ducts and manifolds, utilizing tightly coupled multidisciplinary physics in a unified framework. The work presented here is aimed at addressing this need.

The computational tool presented here is called Loci-STREAM [1]. It is a pressure-based, Reynolds-averaged Navier-Stokes (RANS) code for generalized unstructured grids, which is designed to handle all-speed flows (incompressible to supersonic) and is particularly suitable for solving multi-species flow in fixed-frame combustion devices. Loci-STREAM integrates proven numerical methods for generalized grids and state-of-the-art physical models in a novel rule-based programming framework called *Loci* [2] which allows: (a) seamless integration of multidisciplinary physics in a unified manner, and (b) automatic handling of massively parallel computing. The objective is to be able to routinely simulate problems involving complex geometries requiring large unstructured grids and complex multidisciplinary physics.

II. Methodology

Over the last decade significant new technologies have been developed which allow the design of reliable programs for simulating the complex physics involved in turbulent flows. In the authors' opinion, a state-of-the-art CFD solver for liquid rocket engines must have: (a) a generalized unstructured grid (arbitrary polyhedral cells) capability to allow ultimate flexibility in grid generation for complex geometries and to allow simulation of various sub-components of modern engineering devices in a coupled manner, (b) good scalability on modern parallel computers, especially distributed memory clusters, (c) robust and efficient time-stepping schemes, (d) reliable engineering grade turbulence models for both steady and unsteady flows, (e) reacting flow capability, (f) real-fluid models, and (g) a sound programming framework which reduces the complexity of assembling the various modules of the code while ensuring maximum parallel efficiency in an automatic manner.

A. Rule-based Programming Framework: Loci

The development of this CFD solver technology is facilitated by new developments in the area of programming frameworks. For these tools we use the a framework for application development called *Loci* [2] that is designed to reduce the complexity of assembling large-scale continuum mechanics applications and their integration to provide a multidisciplinary simulation environment. The Loci framework adds a logic-relational programming paradigm to the mix of programming tools available to the application developer. In this framework the unstructured relationships between entities that arise in unstructured representations of meshes and algorithms are represented as relations (a theoretical construct that can be considered practically as a table that relates items of one field to items of another.) Computations in the Loci framework are represented by logical inference rules with accompanying computations that describe relationships between values in the application, and thus relies on the theoretical foundation of logical-relational programming models such as *DataLog*[3]. In some sense, these rules are similar to the rules used by the UNIX make utility to control the compilation of object modules. Inference rules in *Loci* are essentially equations that

arise from the specification of the numerical discretization scheme and associated physics. For example, for the thermally perfect equation of state(EoS) we have the equation $p=\rho RT$, but we also have the implication that any entity that simultaneously had values for ρ, R, and T, also has a value for p as defined by the thermally perfect EoS. The Loci framework provides facilities for specifying these types of relationships between values which *Loci* can use to reason about program derivations. In the context of this programming model, an application is divided into three distinct parts: 1) a collection of facts that describe the problem such as the geometry, mesh, and properties of materials, 2) a knowledge base represented by a collection of transformation rules which can be used to derive answers to specific queries regarding the given facts, and 3) a user query that specifies what specific value is required (e.g. what is the heat flux through this surface?). The Loci framework assembles the computations specified by the rules (2) such that an answer to the query (3) can be obtained from the given facts (1), or alternatively the framework may find that no logically consistent composition of rules will answer the given query. As such, the Loci framework provides a powerful foundation for building highly reconfigurable applications where the compositions of components can be easily changed to meet the given simulation needs. For example, multiphysics simulations can be composed by combining the rules for different applications along with a small set of additional rules describing the intra-application interactions. Currently we have implemented finite-volume, finite-element, and discontinuous-Galerkin models within the framework as well as performed multiphysics simulations by composing the rules as described above.

One benefit of the Loci framework is that the synthesized applications are transparently parallelized without specific directives or user intervention. The Loci scheduler automatically schedules communication between processors as needed to satisfy the schedule of rule computations. In addition to automatic parallelization, the resulting application is verified to be logically consistent against a simple model of correct programs (e.g. that all values are uniquely defined and that provided assertions of the numerical model are met.) Rules can be packaged in run-time loadable shared object modules. These modules can then be combined to produce multidisciplinary simulations by instructing applications to load specified modules. The resulting system provides a very modular method for assembling large scale simulation software.

The Loci framework consists of a C++ library that defines basic utilities types, container types used to create distributed fact databases, a class structure for creating Loci rules, and a planner that assembles rules and facts into a parallel application schedule. In addition to the Loci library, there is a preprocessor called 'lpp' that allows one to develop multi-paradigm programs that by extending standard object oriented C++ code with rule specifications. For example, the thermally perfect EoS described earlier is represented by the following Loci code:

```
$rule pointwise(p<-rho,Rtilde,T) {
  $p = $rho*$Rtilde*$T ;
}
```

Here the '$' sign is used to activate the Loci program elements. The '$rule' identifier signals that a rule specification will follow. There are several types of rule semantics: pointwise, singleton, unit, and apply, that are used to specify how to interpret the given equation. In the case of the EoS, it is expected that this equation applies point-by-point to any entity that satisfies the inputs, and thus is a pointwise rule. After the type of the rule, the signature identifies the inputs and outputs of the rules. All variables before the left arrow are the values that will be computed by the rule, while the values on the right side represent the values needed to perform the computation. The details of all of the rule types and their specification are too detailed to describe here. For an introduction to these concepts along with a description of how the scheduling process is achieved refer to our paper cited earlier[2].

Currently modules for finite-element thermal-stress analysis, an Eulerian dispersed phase finite-volume solver, and a density-based finite-volume flow solver have been implemented within the framework and coupled to produce multidisciplinary simulations. Currently work is underway to develop an arbitrary order discontinuous-Galerkin Navier-Stokes solver and a Control-Angle Discrete Ordinates Method (CA-DOM) non-gray radiation module. Once these modules are completed, they will add to the library of components that can be composed to form advanced simulations. This paper describes the implementation of a pressure-based algorithm for realistic simulations of rocket components under high pressures that is fully implemented within the Loci framework.

B. Pressure-Based Algorithm

The flow solver is based on the SIMPLE (Semi-Implicit Method for Pressure-Linked Equations) algorithm [4]. It uses a control volume approach with a collocated arrangement for the velocity components and the scalar variables like pressure. Pressure-velocity decoupling is prevented by employing the momentum interpolation approach; this involves adding a fourth-order pressure dissipation term while estimating the mass flux at the control volume interfaces. The

velocity components are computed from the respective momentum equations. The velocity and the pressure fields are corrected using a pressure correction (p') equation. The correction procedure leads to a continuity-satisfying velocity field. The whole process is repeated until the desired convergence is reached [5,6].

C. Governing Equations

The governing equations that are used are the equations of mass continuity, momentum, energy and species transport:

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho u_j\right) = 0 \tag{1}$$

$$\frac{\partial}{\partial t}\left(\rho u_i\right) + \frac{\partial}{\partial x_j}\left(\rho u_j u_i\right) = -\frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ji}}{\partial x_j} \tag{2}$$

$$\frac{\partial}{\partial t}\left(\rho H\right) + \frac{\partial}{\partial x_j}\left(\rho u_j H\right) = -\frac{\partial q_j}{\partial x_j} + \frac{\partial}{\partial x_j}\left(u_i \tau_{ij}\right) + \frac{\partial p}{\partial t} \tag{3}$$

$$\frac{\partial}{\partial t}\left(\rho Y_i\right) + \frac{\partial}{\partial x_j}\left(\rho u_j Y_i\right) = \frac{\partial}{\partial x_j}\left(\mu \frac{\partial Y_i}{\partial x_j}\right) + \dot{\omega}, \quad i = 1, NS \tag{4}$$

where ρ is density, u_i is velocity vector, p is pressure, Y_i is the mass fraction of species i (out of a total of NS species), τ_{ij} is viscous stress tensor, q_j is heat flux vector; H is the total (or stagnation) enthalpy given by

$$H = h + \frac{1}{2} u_i u_i \tag{5}$$

with h being the specific enthalpy which is defined later.

The constitutive relation between stress and strain rate for a Newtonian fluid is used to relate the components of the stress tensor to velocity gradients:

$$\tau_{ij} = \left(\mu + \mu_t\right)\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3}\frac{\partial u_l}{\partial x_l}\cdot \delta_{ij}\right) - \frac{2}{3}\rho k \delta_{ij} \tag{6}$$

where μ is the molecular viscosity and μ_t is the turbulent (eddy) viscosity to be defined later. The heat flux vector is obtained from Fourier's law:

$$q_j = -\left(\kappa + \frac{\mu_t}{\mathrm{Pr}_t}\right)\frac{\partial T}{\partial x_j} \tag{7}$$

where δ_{ij} is the Kronecker delta and κ is the thermal conductivity; Pr_L is the laminar Prandtl number defined as:

$$\mathrm{Pr}_L = \frac{C_p \mu}{\kappa} \tag{8}$$

For turbulence closure, the model employed is the k-ε model. The eddy viscosity is estimated from the turbulent kinetic energy (k) and the rate of dissipation of turbulent kinetic energy (ε) by the following relationship:

$$\mu_t = \frac{C_\mu f_\mu \rho k^2}{\varepsilon} \tag{9}$$

The k and ε are estimated by their own transport equations which can be written, in Cartesian coordinates, as the following:

$$\frac{\partial}{\partial t}\left(\rho k\right) + \frac{\partial}{\partial x_i}\left(\rho u_i k\right) = \frac{\partial}{\partial x_i}\left[\left(\mu + \frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_i}\right] + P_k - \rho\varepsilon \tag{10}$$

$$\frac{\partial}{\partial t}\left(\rho \varepsilon\right) + \frac{\partial}{\partial x_i}\left(\rho u_i \varepsilon\right) = \frac{\partial}{\partial x_i}\left[\left(\mu + \frac{\mu_t}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_i}\right] + C_1 P_k \frac{\varepsilon}{k} - C_2 \rho \frac{\varepsilon^2}{k} \tag{11}$$

where P_k is the production of k from the mean flow shear stresses and is given by

$$P_k = \tau_{ij}\frac{\partial u_i}{\partial x_j} = \mu_t\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right)\cdot\frac{\partial u_i}{\partial x_j} \tag{12}$$

The term $\dot{\omega}$ represents the chemical heat release source terms which are obtained from the laws of mass action. A set of chemical reactions can be expressed as follows for the *i*th species of the *j*th reaction, in terms of the stoichiometric coefficients (ν'_{ij} and ν''_{ij} for the reactants and products, respectively):

$$\sum_{i=1}^{NS}\nu'_{ij}M_i \underset{K_{b_j}}{\overset{K_{f_j}}{\rightleftharpoons}} \sum_{i=1}^{NS}\nu''_{ij}M_i \ , \qquad j=1,...,NR \tag{13}$$

The net rate of change of the molar concentration, X_{ij}, of species i due to reactions j is given by

$$X_{ij} = \left(\nu''_{ij}-\nu'_{ij}\right)\left[K_{f_j}\prod_i\left(\frac{\rho Y_i}{M_{Wi}}\right)^{\nu'_{ij}} - K_{b_j}\prod_i\left(\frac{\rho Y_i}{M_{Wi}}\right)^{\nu''_{ij}}\right] \tag{14}$$

and the net species production rate, $\dot{\omega}_i$, is obtained by summing over all reactions:

$$\dot{\omega}_i = M_{Wi}\sum_{j=1}^{NR}X_{ij} \tag{15}$$

The forward rate of reaction is given by the modified Arrhenius law:

$$K_{f_j} = A_j T^{B_j}\exp\left(-\frac{E_j}{RT}\right) \tag{16}$$

and the corresponding backward reaction can be obtained from

$$K_{b_j} = \frac{K_{f_j}}{K_{eq_j}} \tag{17}$$

where K_{eq_j} is the equilibrium coefficient given by

$$K_{eq_j} = \left(\frac{RT}{p_0}\right)^{\Delta\nu_j}.\exp\left(-\frac{\sum_{i=1}^{NS}\left(\nu''_j-\nu'_j\right)G_i}{RT}\right) \tag{18}$$

where G_i is the Gibb's free energy and

$$\Delta\nu_j = \sum_{i=1}^{NS}\nu'_{ij} - \sum_{i=1}^{NS}\nu''_{ij} \tag{19}$$

In the above equations, A_j, B_j and E_j are constants.

D. Thermodynamic Properties for Ideal Gases

The static enthalpy of the fluid at any point is given by

$$h = \sum Y_i h_i \tag{20}$$

where the static enthalpy for each ideal gas specie is related to the temperature (*T*) as

$$h_i = h_{0i} + \int_{T_0}^{T} C_{pi}(T).dT \tag{21}$$

Here h_{0i} is the standard heat of formation of species *i* which is defined as the heat evolved when one mole of the species is formed from its elements in their respective standard states at 298.15 K and 1 atmosphere. C_{pi} is the specific-heat (for constant pressure processes) for species *i* which is computed, in the temperature range of 300-5000 K, using a fourth-order polynomial function of temperature, as follows:

$$C_{pi} = A_i + B_i T + C_i T^2 + D_i T^3 + E_i T^4 \tag{22}$$

where A_i - E_i are curve-fit constants. C_{pi} is linearly extrapolated when the fluid temperature is below 300 K or above 5000 K. An equation of state is required to relate the density to the thermodynamic variables; for a mixture of ideal gases, we use the following:

$$\rho = \frac{p}{RT/M_w} \tag{23}$$

where the molecular weight of the mixture is computed from

$$M_w = \left(\sum \frac{Y_i}{M_{wi}} \right)^{-1} \tag{24}$$

and R is the universal gas constant.

E. Operator Splitting Algorithm for Finite Rate Chemistry

An operator splitting method, similar to the one employed in Refs. [7,8] is implemented to handle the stiffness resulting from the species production rates in the species transport equations. The governing equations for chemically reacting flows form a system of stiff non-linear partial differential equations. Thus, a fully implicit treatment of chemistry source terms in the context of the pressure-based method employed in Loci-STREAM, wherein the governing equations are solved sequentially, is not feasible. Instead, an operator-splitting is employed to effectively handle the disparate time scales associated with the chemical reactions and the transport process. The predictor-corrector steps for solving the species transport equations associated with the combustion process are appended at the end of the fluid dynamics equations; this is justified based on the assumption that the chemical reaction time scales are too small for convective and diffusive processes to be significant. The time step is assumed to be small enough that the increased local temperature due to reactions is not felt by the neighboring cells during a characteristic time of the fluid. Thus, the chemistry time scale is bound by the fluid flow characteristic time, which is comparable to the integration time step of the flow solver. Thus, the chemical kinetics can be resolved by employing a stiff ODE solver which typically employs several steps for integration and still be coupled with the fluid dynamics equations in an effective manner.

In the predictor step, the following system of stiff ODEs is solved:

$$\rho \frac{dY_i^*}{dt} = \dot{\omega}_i \left(Y_1^*, Y_2^*, \cdots, Y_{NS}^* \right), \qquad i = 1, NS \tag{25}$$

The stiff ODE solver employed is CVODE [9]. The source code for CVODE is distributed by Lawrence Livermore Laboratory and is freely available via the internet. Once the ODE solution is obtained, the effective production rates to be used in the species transport equation are obtained as follows:

$$\left(\dot{\omega}_{eff} \right)_i = \rho \frac{\left(Y_i^* - Y_i^n \right)}{\Delta t_f} \tag{26}$$

Finally, the following species transport equation is solved in an implicit manner:

$$\frac{\partial}{\partial t}\left(\rho Y_i \right) + \frac{\partial}{\partial x_j}\left(\rho u_j Y_i \right) = \frac{\partial}{\partial x_j}\left(\mu \frac{\partial Y_i}{\partial x_j} \right) + \left(\dot{\omega}_{eff} \right)_i, \qquad i = 1, NS \tag{27}$$

F. Real-Fluid Model

There is a critical need for methodology to determine the effect of real-fluid transport and thermodynamic properties in CFD flow solvers for accurately simulating rocket propulsion system performance and operational characteristics. It is recognized that the majority of the advanced liquid propulsion problems involve fluids that cannot be accurately modeled using the ideal-gas law for each species. The working fluids in conventional rocket engines are in the real-fluid regime, and advanced engines now under development will operate under even higher pressures which will make these real-fluid effects even more important.

In the present algorithm, thermal and caloric equations of state (EOS) developed by Hirschfelder, Buehler, McGee and Sutton [10,11] (hereafter referred to as the HBMS EOS) in the late 1950s are employed. These equations provide a

description of the complete PVT surface, including the gas-phase, dense-gas phase and liquid phase regions. In the following, we give a brief review of the HBMS EOS and discuss its role in the Loci-STREAM algorithm. Full details of the derivation of the EOS can be found in the papers of Hirschfelder et al [10,11]. Figure 1 shows a projection of the PVT surface for a generic pure material onto the axes of reduced pressure and reduced density. This space is conveniently, although arbitrarily, divided into three regions defined as follows, each containing its own equation which specifies the relation of the thermodynamic variables pressure, temperature and density:

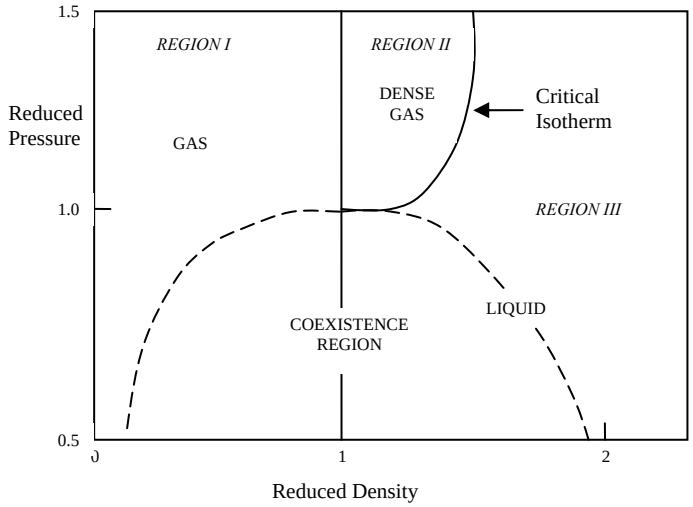

Figure 1. Projection of the P-V-T surface for a pure material on reduced density and pressure axes.

- Region I: Gas. All temperatures; density less than the critical, $\rho_c \le 1$
- Region II: High Density Gas. Temperature above the critical, $T_c \ge 1$; density greater than the critical, $\rho_c \ge 1$
- Region III: Liquid. Temperature below the critical, $T_c \le 1$; density greater than the critical, $\rho_c \ge 1$

It is important to note that certain continuity constraints on the derivatives of the thermodynamic variables between the zones have been enforced in the derivation of the equations to ensure that all derived thermodynamic variables are continuous at the boundaries between the regions.

The thermal equation of state is explicit in density and temperature and is given by:

$$\frac{p}{p_c} = \sum_{j=1}^{4} T_r^{\,j-2} \sum_{i=1}^{6} B_{ij} \rho_r^{\,i-2} \;;\; T_r = \frac{T}{T_c} \;;\; \rho_r = \frac{\rho}{\rho_c} \tag{28}$$

In addition to the equations relating pressure, density and temperature, so-called "departure functions" for the enthalpy are provided for each of the three regions, which allow one to compute the enthalpy once the values of pressure, density and temperature are known. For completeness, these equations are listed as follows (see references above for complete specification of additional symbols):

Region 1 (Gas):

$$\frac{(H-H_0)}{RT} = z_c \left[\rho \left(2k_0 t^{-1} + 3k_1 t^{-2} \right) + k_2 \rho^2 \left(1 - 2t^{-2} \right) \right] + \frac{\left(b\rho - b'\rho^2 \right)}{\left(1 - b\rho + b'\rho^2 \right)} \tag{29}$$

Region II (Dense Gas):

$$\frac{(H-H_0)}{RT} = -z_c \sum_{j=0}^{3} t^{j-2} \left[\left(Q_j(\rho)/\rho \right) - Q_j(1) - (j-2)\left(W_j(\rho) - W_j(1) \right) \right] + \frac{(H_1 - H_0)}{RT} \tag{30}$$

Region III (Liquid):

$$\begin{aligned} \frac{(H-H_0)}{RT} = z_c \sum_{j=0}^{3} t^{j-2} \left\{ \frac{Q_j(\rho)}{\rho} - \frac{Q_j(\rho_l)}{\rho_l} - (j-2)\left[W_j(\rho) - W_j(\rho_l) \right] - (j-1) Q_j(\rho_l) \left[\rho^{-1} - \rho_l^{-1} \right] \right\} \\ - \left(\rho^{-1} - \rho_l^{-1} \right) t \left(\frac{d\rho_l}{dt} \right) z_c \sum_{j=0}^{3} t^{j-2} R_j(\rho_l) - z_c \left(\rho_v^{-1} - \rho^{-1} \right) \left(\frac{dp_v}{dt} \right) + \frac{(H_v - H_0)}{RT} \end{aligned} \tag{31}$$

In the above equations, p is the pressure, T is the temperature, ρ is the density. The above equations are based on the theorem of corresponding states for real fluids, which means that the p-v-T relations for all species are similar if these variables are normalized with their values at the critical condition (these normalized values are called reduced values and are indicated by the subscript "r"). H and H_0 are the real-fluid and ideal-fluid enthalpies, respectively, of a given species. Z_c is the compressibility for a given species at the critical point. B_{ij} and R are the coefficients of the thermal property polynomial and the gas constant, respectively, for a given species. For homogeneous spray simulations, the mixture properties are obtained by using the additive-volume method when multi-component fluid/vapor mixtures are present in the flowfield, i.e., multiphase mixtures are treated as ideal solutions. For mixtures under conditions when the species become ideal gases, the thermodynamic data obtained from the CEC code (Gordon & McBride [12]) are used. The sound speed of a multi-component mixture is calculated from the above mentioned real-fluid property sub-model to properly account for compressibility effects.

When solving the compressible Navier-Stokes equations for a particular geometry with the medium consisting of a pure compressible substance, Loci-STREAM computes the specific enthalpy from the energy equation and the pressure via the pressure-correction equation. In order to obtain the thermodynamic quantities temperature and density, one must "query" the equation of state. For an ideal gas medium in which the enthalpy is strictly a function of temperature, one must invert the enthalpy/temperature equation (in which enthalpy is a non-linear function of temperature) to get temperature given the enthalpy. One can then, in a separate step, compute the density from the standard ideal-gas EOS since the pressure and temperature are now available. On the other hand, for a medium which is being modeled by the HBMS EOS, enthalpy is no longer a function of temperature alone, but also density. Thus, given enthalpy and pressure, one must simultaneously invert both the EOS and the enthalpy relation to obtain the density and temperature. For a pure substance, this amounts to a Newton-iteration procedure for two non-linear equations of density and temperature, although other more sophisticated numerical techniques can be used.

G. Generalized Grid Capability

The finite-volume framework that is provided as part of the Loci infrastructure allows for the solution on generalized grids, that is grids composed of arbitrary polyhedra. We take advantage of this capability in the development of mesh refinement tools that employ general elements. The primary advantage of using general elements for mesh refinement is that it eliminates problems associated with terminating the refinement. Depending on the specific strategy, refinement in one element may produce a hanging node or a face mismatch on an adjacent element that is not refined. The approach mimics Cartesian mesh refinement strategies. In fact, Cartesian mesh refinement produces what we would term general elements because of the apparent face mismatch between elements. If hanging nodes are not allowed, neighbor cells must be refined according to some strategy until all hanging nodes are eliminated. The refinement strategy is facilitated through a face splitting algorithm whereby each of the faces of a cell are split using the face centroid. The cell is refined through insertion of a point at the cell centroid. The resulting algorithm can be used for isotropic refinement of arbitrary cell types as illustrated in figure 2.

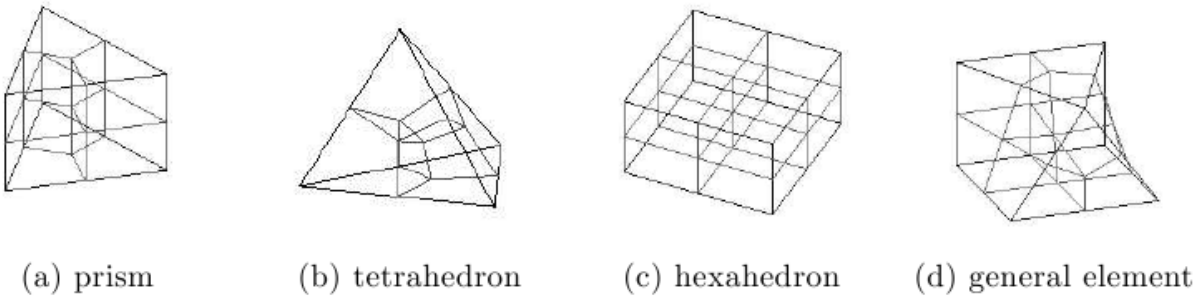

Figure 2: Isotropic refinement plans for typical 3-D elements

Error estimators are used to guide the refinement. Although any error estimator can be formulated, for the present time errors are estimated from the current second order face extrapolation jumps for various dependent and independent variables. For example, in hydrogen-oxygen rocket combustion problems temperature and OH mass fraction reconstruction errors are typically used. A node is marked for refinement if its estimated error is more than a standard deviation over the mean error. (A similar rule can be used for mesh coarsening.) As an example, an adapted grid near the injector post is shown in figure 3. The grid shown consists of quadrilateral and triangular faces along with

"soccer ball-shaped" cell clusters which result from the described isotropic grid adaptation strategy. In this case the error estimators refined the grid near the thin flame front region.

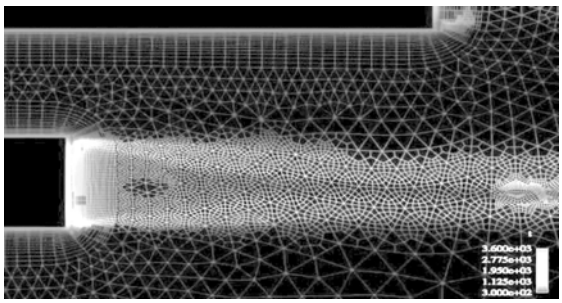

Figure 3. Generalized adapted grid used for a rocket engine injector.

H. Parallel Computing Capability

The scalability of *Loci*-STREAM for parallel architectures is demonstrated by running the code on two different parallel architectures:

(a) An SGI Altix computer which has 16 Itanium 1.3 GHz processors with 16 GB shared memory. A laminar flow past a backward-facing step using a grid consisting of approximately 75,000 nodes is used. A perfectly linear scalability is achieved as shown in Figure 4.

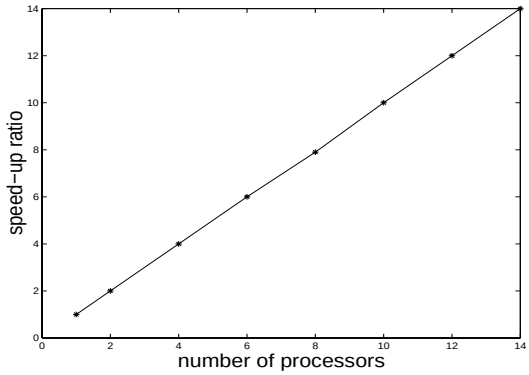

Figure 4. Scalability of Loci-STREAM on a shared memory parallel architecture.

(b) The second parallel architecture used is a computer cluster at NASA Marshall Space Flight center. This cluster consists of Athlon 1800 Hz CPUs with 512 MB RAM per CPU, along with a network switch. It is observed that parallel benefit ceases when the number of cells per CPU drops below 15,000 cells for the 100 Mbps network and 2,500 cells for the 1,000 Mbps network as shown in Figure 5.

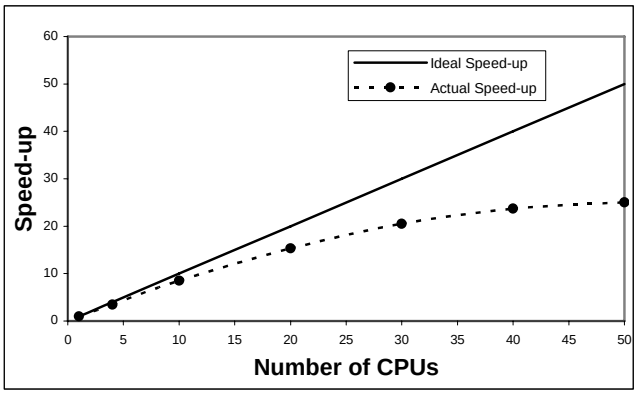

Figure 5. Scalability of Loci-STREAM on a distributed memory networked cluster.

III. Results of Computations

In this section, we present the results of a simulation of gaseous oxygen and hydrogen for an injector geometry supplied by NASA/MSFC which is shown in Figure 6. Due to symmetry, only the top half of the axisymmetric geometry is used for the computation. The grid is supplied by NASA/MSFC. The computation is conducted in the unsteady mode using Euler time-stepping with $\Delta t = 1.0 \times 10^{-5}$ s, along with the SIMPLEC algorithm. A relaxation parameter (0.5) is used for the solution of all the governing equations. Since only the steady state solution is desired, the solution is not forced to converge at each time step. In fact, the most optimum convergence path is achieved by performing just one iteration per time step. The combustion model involves 7-species and 9-reactions, which is shown in Table I. A second-order upwind scheme is used for the convection terms in the momentum, energy and species equations.

Mass flow rates, species mass fractions and temperatures from experimental data are specified at the two inlets in the domain. The flow is subsonic at the two inlet regions. The boundary conditions at the two inlet regions are listed in Table II. The values of k and ω are specified based on a turbulence length scale (specified as 3% of the inlet diameter), a reference velocity, and a specified turbulence intensity (5%). The flow is subsonic throughout the combustion chamber; however, it becomes supersonic in the diverging portion of the nozzle. Consequently, at the outlet plane, all variables are extrapolated from the interior of the computational domain. All noslip walls are treated as adiabatic. Wall functions are used to bridge the gap between the near-wall region and the fully turbulent core region, for both velocity and temperature profiles. After some trial and error, the near-wall spacing of the final grid is such that the y^+ values lie in the range of 50 to 200. The bottom boundary of the grid is treated as symmetry boundary and all the variables are extrapolated there with the fluxes being set to zero.

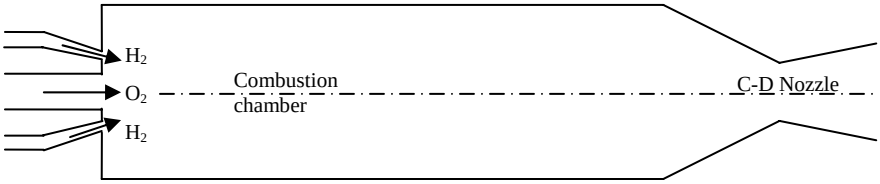

Figure 6. Axisymmetric geometry of the single-element injector.

Table I. Reaction Mechanism involving 7 species and 9 reactions

No.	Reaction	A	B	E
1	$H_2+O_2=OH+OH$	1.700×10^{13}	0	200130.02
2	$OH+H_2=H_2O+H$	2.190×10^{13}	0	21534.555
3	$OH+OH=O+H_2O$	6.023×10^{12}	0	4572.975
4	$O+H_2=H+OH$	1.800×10^{10}	1.0	37248.96
5	$H+O_2=O+OH$	1.220×10^{17}	-0.91	69584.051
6	$O+H+M=OH+M$	1.000×10^{16}	0	0.0
7	$O+O+M=O_2+M$	2.550×10^{18}	-1.0	493798.16
8	$H+H+M=H_2+M$	5.000×10^{15}	0	0.0
9	$H+OH+M=H_2O+M$	8.400×10^{21}	-2.0	0.0

Table II. Boundary conditions specified at the inlets of the injector geometry.

Specified Quantity	Inlet 1	Inlet 2
Species mass fractions	O_2: 0.94616 H_2O: 0.05384	H_2: 0.65056 H_2O: 0.34944
Mass flow rate (kg/s)	0.10685	0.026674
Temperature (K)	644.49	760.73

The initial conditions are specified as follows:

- O_2 channel – species: O_2, Temperature and velocity same as inlet value
- H_2 channel – species: H_2; Temperature and velocity same as inlet value
- Combustion chamber – species: H_2O; Temperature: 700K; Velocity: 10 m/s

No spark (or artificial ignition) was supplied for the computation; ignition starts automatically as the reactants start mixing during the course of the computations. Contours of species concentrations as well as temperature are shown in Figure 7. The prominent features of the flow field in the combustion chamber are a shear layer between the hydrogen and oxygen jets and a large recirculation region downstream of the combustion chamber inlets. Ignition starts immediately downstream of the post region and the flame gets anchored at the post tip. The distribution of the reaction products are shown in Figure 7. It can be observed that the reaction is complete by approximately 40% of the length of the combustion chamber. Downstream of this location, the reactants (hydrogen and oxygen) are mostly consumed and the predominant species is H_2O. The temperature contours in Figure 7 show that the maximum temperature in the combustion chamber is 3565 K. Overall, the computations show that the algorithm is robust.

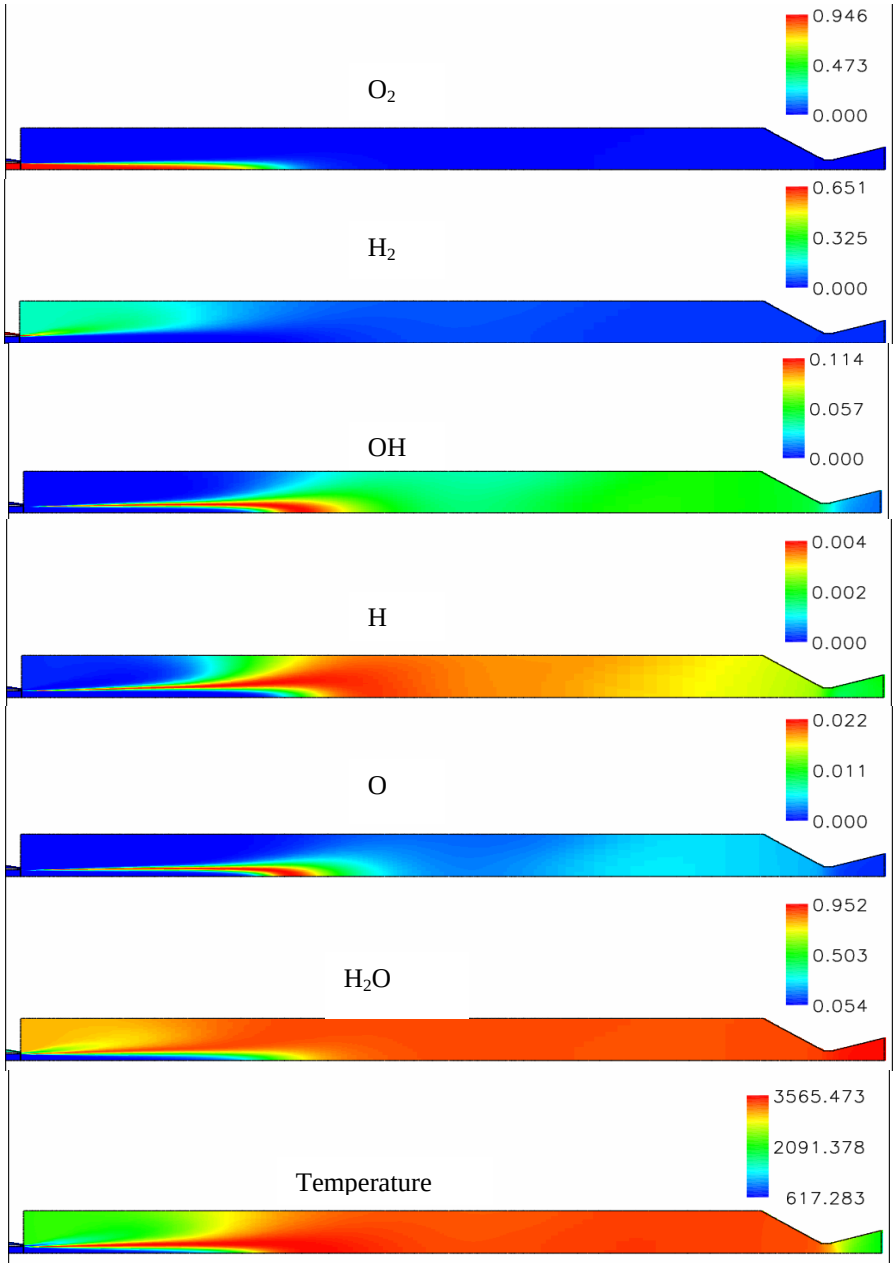

Figure 7. Computational results for reacting flow in the single-element injector.

IV. Summary

We have presented the development of a pressure-based algorithm suitable for simulating high pressure combustion in rocket chambers. We are employing the HBMS equation of state to account for the non-linear thermodynamic properties of liquids, cryogenics, and vapor under the high pressure conditions found in high performance rocket motors. Preliminary results for a gas-gas injector test case are presented to demonstrate the application of the methodology on problems of interest. Simulations of supercritical combustion cases are underway and will be presented in subsequent papers.

One unique feature of this algorithm is that it was developed using the novel multidisciplinary simulation framework called Loci. Not only does the Loci framework provide facilities which will allow this pressure based algorithm to be composed with other Loci modules to create multidisciplinary simulations, it also provides automatic parallelization and consistency checking that can reduce the amount of effort required to develop robust and high performance simulation software.

V. Acknowledgements

This work has been partially supported by the NASA Constellation University Institute Program (CUIP, Ms. Claudia Meyer program monitor). We acknowledge the valuable collaboration with Dr. Jeff West and Mr. Kevin Tucker of the NASA Marshall Space Flight Center.

References

[1] Thakur, S., Wright, J. and Shyy, W., "An Algorithm for Chemically Reacting Flows on Generalized Grids Using a Rule-Based Framework," *43rd AIAA Conference*, Paper No. 2005-0875, Reno, NV (Jan 2005).

[2] Luke, E.A., and T. George, "Loci: A Rule-Based Framework for Parallel Multidisciplinary Simulation Synthesis," *Journal of Functional Programming*, **15**(03): 477-592, (May 2005).

[3] Ullman, J., Principles of Database and Knowledgebase Systems, Computer Science Press, (1988)

[4] Thakur, S, Wright, J. and Shyy, W., "STREAM: A Computational Fluid Dynamics and Heat Transfer Code for Complex Geometries. Part 1: Theory. Part 2: User's Guide," Streamline Numerics, Inc., Gainesville, Florida (1999).

[5] Patankar, S. V., Numerical Heat Transfer and Fluid Flow, Hemisphere, Washington, DC (1980).

[6] Shyy, W., Thakur, S. S., Ouyang, H., Liu, J. and Blosch, E., Computational Techniques for Complex Transport Phenomena, Cambridge Univ. Press, NY (1997).

[7] Shang, H.M., Chen, Y.S., Liaw, P., Chen, C.P. and Wang, T.S., "Investigation of Chemical Kinetics Integration Algorithms for Reacting Flow," 33rd Aerospace Sciences Meeting and Exhibit, *AIAA 95-0806*, Reno, NV (Jan. 1995).

[8] Bussing, T.R.A. and Murman, E.M., "Finite-Volume Method for the Calculation of Compressible Chemically Reacting Flow," *AIAA Journal*, **26**: 1070-1078 (1987).

[9] Cohen, S.D. & Hindmarsh, A.C., "CVODE User Guide," LLNL Report UCRL-MA-118618 (1994).

[10] Hirschfelder, J.O., Buehler, R.J., McGee, H.A., and Sutton, J.R., "Generalized Equations of State for Gases and Liquids," *Industrial and Engineering Chemistry*, Vol. 50, No. 3, pp 375-385 (1958).

[11] Hirschfelder, J.O., Buehler, R.J., McGee, H.A., and Sutton, J.R., "Generalized Thermodynamic Excess Functions for Gases and Liquids," *Industrial and Engineering Chemistry*, Vol. 50, No. 3, pp 386-390 (1958).

[12] Gordon, S. & McBride, B.J., "Computer Program for Calculation of Complex Chemical Equilibrium Compositions, Rocket Performance, Incident and Reflected Shocks, and Chapman-Jouget Detonations," NASA-SP-273 (1971).