Dynamic-Data-Driven Real-Time Computational Mechanics Environment

John Michopoulos[1], Charbel Farhat[2], and Elias Houstis[3,4]

[1] Special Projects Group, Code 6303, U.S. Naval Research Laboratory, U.S.A.
john.michopoulos@nrl.navy.mil
[2] Dept. of Aerospace Engineering Sciences, University of Colorado at Boulder, U.S.A
charbel.farhat@colorado.edu
[3] Computer Sciences Department, Purdue University, U.S.A.
[4] Dept. of Comp. Eng. and Telecommunications, University of Thessaly, Greece
enh@cs.purdue.edu

Abstract. The proliferation of sensor networks in various areas of technology has enabled real-time behavioral monitoring of various physical systems in various length and time scales. The opportunity to use these data dynamically for improving speed, accuracy, and general performance of predictive behavior modeling simulation is of paramount importance. The present paper identifies enabling modeling methods and computational strategies that are critical for achieving real-time simulation response of very large and complex systems. It also discusses our choices of these technologies in the context of sample multidisciplinary computational mechanics applications.

1 Introduction

The main objective of the effort described in this paper is to establish and use a strategy for selecting, generating, improving, and applying methodologies capable of enabling data-driven real-time simulation of large, complex, multi-field, multi-domain physical systems. This paper reports on the initial success in achieving this goal, under the additional fundamental requirement that sensor networks will be providing live data originating from the actual physical systems that are simulated by the exercise of their corresponding computational models.

This effort is a part of a larger effort of developing a Data Driven Environment for Multiphysics Applications (DDEMA) [1,2,3,4], as a computationally implementable framework in the form of a Multidisciplinary Problem Solving Environment (MPSE). Although DDEMA's scope also entails utilizing dynamic data for adaptive modeling or model selection, this paper only discusses data usage for adaptive simulation, steering, and short future behavioral prediction assuming that the model is known and well established. This model approximates the behavior of a system that belongs in the continuum mechanics engineering domain.

The rest of the paper is organized as follows. In Section 2, we describe the context and required enabling technologies achieving this objective. In Section

M. Bubak et al. (Eds.): ICCS 2004, LNCS 3038, pp. 693–700, 2004.

3, we specify the technologies we selected for DDEMA: precomputed solution synthesis and model reduction, along with multidimensional computational parallelism. Finally, Section 4 presents two preliminary sample applications and concludes this paper.

2 Modeling Methods and Computational Strategies for Improving Simulation Performance

Discussion in this section is limited only to the systemic modeling methodologies that capture system-behavior representation, and the computational strategies of their usage. We do not intend to discuss computational, networking, infrastructure and process mobility strategies and methodologies. However, it should be recognized that these can also have a considerable effect on the performance of a simulation environment. This task has been discussed in [1,2,3,4].

To achieve the main objective as described in the introduction, one has first to define the context of the problem at hand. Figure 1 provides a generic description of this context. It shows a schematic representation of the dataflow relationship among the actual physical system and the corresponding sensor network, the computational model, and the associated simulation environment along with the simulated system. In addition, the data-flow paths have been labeled to allow the distinction of both the real-time solving (RTS) and the precomputed solving and real-time solution synthesis (PS-RTSS) modes of simulation usage. All components in this figure are generic and do not express the architecture of a particular system or technology.

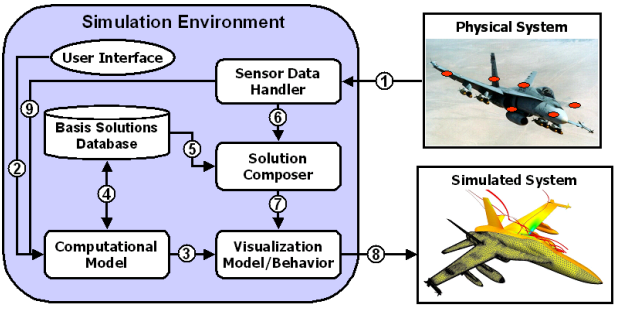

Fig. 1. Relationship among physical system, sensor data, and computational infrastructure for behavioral simulation.

As shown in Fig. 1, path $[2-4-3-8]$ and path $[1-9-3-8]$ represent the RTS mode, while path $[2-4] \wedge [(5,6)-7-8]$ corresponds to the PS-RTSS mode. The latter path consists of the conjunction of the precomputed solving part represented by the sub-path $[2-4]$, and the real-time solution synthesis part represented by the sub-path $[(5,6) \wedge 7-8]$. The path syntax

used here has been devised to avoid the complications of temporal logic syntax while allowing the ability to capture synchronicity and asynchronicity. Paths in regular parentheses separated by a coma represent synchronous parallel-in-time dataflow, while square brackets are used to denote complete paths, and $\wedge$ is used to denote asynchronous logical conjunction of paths.

The two-way dataflow between the *Basis Solutions Database* and *Computational Model* in Fig.1 underscores the complexity of the relationship that can exist between these two modules. Indeed, computational models generate in general a basis database, but such a database can also be exploited to construct, for example, reduced-order models (ROM). Furthermore, a key property of the *Computational Model* module is that it may include inside it many levels of recursive application of the entire *Simulation Environment* operating in PS-RTSS mode. The difference from the current incarnation of the *Computational Model* module is that for each level below, a ROM participates in the corresponding instance of the *Computational Model* module.

Dynamic-data can improve the simulation via a number of various manners. These include the updating of computational models and the reduction of simulation uncertainty to enhance simulation validity, and the enabling of original ideas for parallel processing in the time-domain and of the construction of reduced-order models to improve computational efficiency.

Thus, it is important to select, create, and utilize methodologies and strategies that can inherently take advantage of real-time data. The most important aspect of what is referred to as real-time simulation is that the time between the time of user action (or the arrival of new data into the system), and the time the system responds back with the corresponding adjustment of behavioral simulation, has to be as small as possible. Therefore, the common requirement for all processes involved in the simulation is that they have to be as fast as possible. Here we examine how the modeling methods and the computational strategies can be combined for achieving real-time simulation response. Thus, a distinction between a computational strategy and a modeling methodology needs to be drawn here. A computational strategy refers to the abstract methodology employed for optimal usage of the computational and networking infrastructure. A modeling methodology refers to the particular behavior model (usually in equational form) and its corresponding computational implementation.

A detailed survey of behavior-modeling methods capable of enabling real- or near-real-time simulation lies well outside the scope of the present paper. However, before selecting, inventing, improving, or implementing a method, it is reasonable to expect that certain common attributes have been identified and a taxonomic classification has been constructed.

We have identified 36 behavior modeling methods to date that lend themselves to accelerated systemic behavior computations. Our list [5] is continuously growing, thus not allowing us to include it in the present paper. We have also created a classification taxonomy based on certain attributes of these methods, and have established four classes of selection criteria based on required attributes.

The first classification level contains the *Dimensional Reduction Methods* (DRM); all those which do not fall in this category are the *Rest*. The common characteristic of all DRMs that classifies them as such is that they all attempt to project some quantifiable feature (state-variable field and space- or time-domain of applicability) of the problem to a discrete lower-dimensional space. They subsequently recover the global systemic behavior via appropriate reconstruction from the reduced projections.

The DRMs are further subdivided into the *Reduced Basis Behavior Decomposition Methods* (RBBDM), and the *Reduced Domain Decomposition Methods* (RDDM). The common characteristic of all RBBDMs is that they often represent behavioral field variables as a composition of sub-behaviors defined in lower-dimensional spaces, mostly through the use of specialized basis functions. Similarly, the common characteristic of all RDDMs is that they decompose the domain space of application of behavior to subdomains, and subsequently reconstruct systemic behavior from the composition of the subdomains.

RBBDMs are model reduction methods subdivided further into those which are physics-based, and those which are physics-blind. Examples of physics-based RBBDMs are methods such as Proper Orthogonal Decomposition (POD), Hankel-norm approximation, balanced truncation, Singular perturbation, Cross grammian methods, Principal Components, Karhunen-Loeve, Multiresolution Green's Function, Hierarchical Multiresolution, Differential Geometry Components, Examples of physics-blind RBBDMs are methods such as Multilayered Perceptrons, Radial Basis Networks, Local Polynomials (Splines etc.), Neural Networks, Polynomial Chaos, Support Vector Machines, etc.

Examples of RDDMs are methods like the partial realization, Padé methods, multi-resolution geometry patches, general domain decomposition and the recently-developed PITA [6].

Examples in the category of all of the *Rest* methods are Recursive Kalman Filtering Estimation, Capacitance Matrix, Reaction and Lagrangian Constraint Optimization, and Equationless Lifting Operator.

All these methods have a great potential for enabling real-time computations, but are often reliable only for time-invariant systems. Dynamic data enables their adaptation, when the system simulated is time-varying, using interpolation and/or sensitivity approaches. Certain common desired attributes of all of the above methods can be used to construct selection criteria. We marked [5] all methods according to the following performance criteria: capability for real-time simulations in terms of both RTS and PS-RTSS, capability for modeling uncertainty, capability for adaptive model formation, and finally capability for parallel implementation.

3 Data-Driven Acceleration Choices of DDEMA

According to the behavioral model classification of the previous section, we have limited ourselves to those methods which satisfy only the first and last criteria of our selection strategy. The dynamic-data-driven aspect of the system to be

simulated is relevant to the applied methods and strategies both trivially and non-trivially, depending on our particular choices. For example, the RTS strategy can utilize the dynamic data to alter initial and/or boundary conditions, to construct or update a ROM, and to provide seed values to a PARAREAL (parallel real-time) solution method [7] or PITA (parallel implicit time-integrator algorithm) [6] that parallelizes in the time-domain the dynamic solution of a ROM problem. It should be noted that for a ROM, a time-parallel solution approach is more efficient than a conventional degree-of-freedom-parallel approach (also known as space-parallel approach) because by definition a ROM contains few degrees of freedom and therefore can benefit only to a certain extent from parallel processing at the degree-of-freedom level.

The PS-RTSS strategy utilizes the data only at the synthesis stage in their capacity to select proportions of precomputed solutions. It is important to remember that PS-RTSS is a computational strategy and not a modeling method. It still requires utilizing a behavioral model of the system. It uses this model to compute and store in a database behavioral solutions corresponding to the bases of low-dimensional space parameterizations of the generalized input as it is exercised by the actions of the user- or the sensor-data. At the time of real-time usage of the simulator, the *Solution Composer* module takes the sensor data or/and the user's actions and synthesizes a solution that corresponds to these choices, based on the stored database content. The solution synthesis is done as a postprocessing of elementary solutions, and therefore involves evaluative computation — and not solving computation which is only performed in the precomputation stage. This strategy uses models in an off-line mode prior to the real-time simulation period, and therefore is more insensitive to the efficiency of the computational method that implements the behavioral model of the system to be simulated. Clearly, all possible computational methods for implementing the systemic behavior are acceptable in this strategy. However, it is reasonable to expect that the more efficient a method is, the more it contributes to the creation of an efficient system that is inexpensive to use from an overall perspective, regardless of the distinction between the precomputation solving and the real-time computation modes.

4 Sample Applications

4.1 PS-RTSS of an Underwater Composite Vessel

Preliminary results of this strategy have been obtained for the case of a cylindrical I-beam ring-stiffened composite pressure vessel with semi-spherical steel end-caps (see Fig. 2). The model of this vessel represents one of the components of a submarine-attached "Dry-Deck-Shelter" (DDS) utilized in underwater naval missions. Embedded sensors provide datastreams that encode strain-field measurements used to assess generalized loading conditions on the vessel in terms of three basis loading cases that span the loading space [8]. The first is the external pressure that is proportional to the depth of the submerged cylinder. The other two basis cases are bending moments about axis-x L_x and axis-y L_y (see

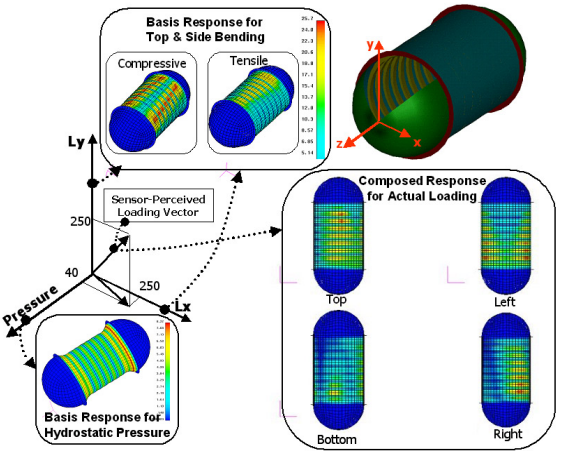

Fig. 2. Material softening distributions from precomputed basis solutions and sensor-controlled reconstruction for an underwater composite pressure vessel, under conditions of hydrostatic pressure and explosion-induced horizontal and vertical bending.

Fig.2). They can correspond to maneuvering inertial loads, or to the quasi-static bending effect of a pressure wave induced by a remote underwater explosion.

In precomputation mode, the simulator computes and stores into the database the material softening and other state-variable field distributions over the entire geometry of the structure loaded by each one of the three loading cases. This was done by finite element analysis (FEA) of the associated model for the three distinct loading conditions. The results are shown in Fig. 2. For the bending cases, both the tensile and compressive sides of the vessels are shown. The difference between the tension and compression sides of the vessel is due to the nonlinear constitutive response of the composite material that was predetermined by a mechatronically driven automated approach [9,10].

In the real-time mode, our simulator received strain measurement input from at least three sensors [8]. A hypothetical data report from these three sensors determined that the applied loading condition on the structure is defined by the loading vector $(40\boldsymbol{P}, 250\boldsymbol{L_x}, 250\boldsymbol{L_y})$ as shown in Fig. 2. The intrinsic power of this strategy is that the simulator does not have to run an additional FEA to determine the corresponding state variable field distributions and pay the solution cost. The *Solution Composer* module described in Fig. 1 multiplies the basis solutions stored in the database by the coefficients of the components of the loading vector bases and sums the results in just one pass. The visualization results through the *Visualization* module are instantaneous and satisfy the technical needs of the user. In this case, the user may be the submarine commander, in which case the simulator is on-board the structure it simulates and can be used as data-driven computational decision support system [2,3,4].

4.2 Combined RTS/PS-RTSS of the Aeroelastic Behavior of the AGARD Wing 445.6

Preliminary results of this hybrid strategy have been obtained for a flutter analysis of the AGARD Wing 445.6. This wing is an AGARD standard aeroelastic configuration with a 45 degrees quarter-chord sweep angle, a panel aspect ratio of 1.65, a taper ratio of 0.66, and a NACA 65A004 airfoil section. The model selected here is the so-called 2.5-ft weakened model 3 whose measured modal frequencies and wind-tunnel flutter test results are reported in [11], and for which a full-order aeroelastic computational model with 178,758 degrees of freedom is described in [12]. Embedded sensors provide datastreams that encode free-stream pressure, density, and Mach number.

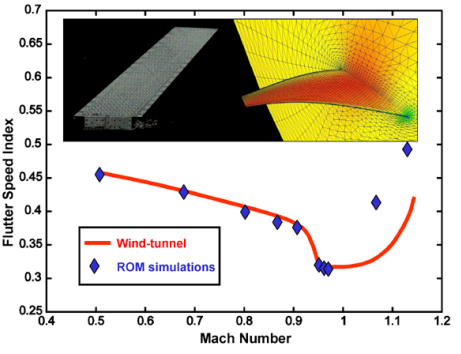

Fig. 3. AGARD Wing 445.6: flutter speed indices simulated by a hybrid RTS/PS-RTSS strategy.

In a first precomputation mode, the simulator used the POD approach described in [13] to generate and store in the *Basis Solutions Database* four aeroelastic ROMs with 200 degrees of freedom each, at the following four free-stream Mach numbers: $M_\infty = 0.499$, $M_\infty = 0.901$, $M_\infty = 0.957$, and $M_\infty = 1.141$. Then, it considered a scenario of flight acceleration from the subsonic regime ($M_\infty = 0.499$) to the supersonic regime ($M_\infty = 1.141$). During this acceleration, it received eight requests from the Mach number sensor for predicting the flutter speed in order to clear the target flight envelope. Four of these requests were at the Mach numbers mentioned above, and four others at $M_\infty = 0.678$, $M_\infty = 0.954$, $M_\infty = 0.960$, and $M_\infty = 1.072$. For each of the latter free-stream Mach numbers, the *Solution Composer* generated in near-real-time a corresponding new ROM by interpolating the angles between the subspaces of the stored ROMs [13], and predicted in real-time the corresponding flutter speed index. The visualization results through the *Visualization* module are reported in Fig. 3 and compared with the flutter speed index curve obtained from wind-tunnel test data [11]. The aeroelastic results generated by this RTS/PS-RTSS hybrid strategy are shown to be in good agreements with the test data.

Acknowledgments. The authors acknowledge the support by the National Science Foundation under grant ITR-0205663.

References

1. Michopoulos, J., Tsompanopoulou, P., Houstis, E., Rice, J., Farhat, C., Lesoinne, M., Lechenault, F., DDEMA: A Data Driven Environment for Multiphysics Applications,in: Proceedings of International Conference of Computational Science - ICCS'03, Sloot, P.M.A., et al. (Eds.) Melbourne Australia, June 2-4, LNCS 2660, Part IV, Springer-Verlag, Haidelberg, (2003) 309-318
2. Michopoulos, J., Tsompanopoulou, P., Houstis, E., Rice, J., Farhat, C., Lesoinne, M., Lechenault, F., Design Architecture of a Data Driven Environment for Multiphysics Applications, in: Proceedings of DETC'03, ASME DETC2003/CIE Chicago IL, Sept. 2-6 2003, Paper No DETC2003/CIE-48268, (2003).
3. Michopoulos, J., Tsompanopoulou, P., Houstis, E., Farhat, C., Lesoinne, M., Rice, J., Joshi, A., On a Data Driven Environment for Multiphysics Applications, Future Generation Computer Systems, in-print (2004).
4. Michopoulos, J., Tsompanopoulou, P., Houstis, E., Farhat, C., Lesoinne, M., Rice, Design of a Data-Driven Environment for Multi-field and Multi-Domain Applications, book chapter in Darema, F. (ed.), Dynamic Data Driven Applications Systems, Kluwer Academic Publishers, Netherlands, in-print (2004).
5. DDEMA-group, Taxonomic Classification of Reduced Order Models, available from http://ddema.colorado.edu/romtaxonomy, (2004).
6. Farhat, C., Chandesris, M., Time-Decomposed Parallel Time-Integrators: Theory and Feasibility Studies for Fluid, Structure, and Fluid-Structure Applications, Internat. J. Numer. Meths., Engrg. 58, (2003) 1397-1434.
7. Lions, J. L., Maday, Y., Turinici, G., Résolution d'EDP par un Schéma en Temps "Pararéel", C. R. Acad. Sci. Paris, Serie I Math. 332, (2001) 661-668.
8. Michopoulos, J.G., Mast, P.W. , Badaliance, R., Wolock, I., Health Monitoring of smart structures by the use of dissipated energy, ASME proc. 93 WAM on Adaptive structures and material systems, G.P. Carman/E. Garcia, eds., ASME, AD-Vol. 35, (1993) 457-462.
9. Michopoulos, J., Computational and Mechatronic Automation of Multiphysics Research for Structural and Material Systems, Invited paper in "Recent advances in Composite Materials" in honor of S.A. Paipetis, by Kluwer Academic publishing (2003) 9-21.
10. Mast, P. , Nash, G., Michopoulos, J., Thomas, R. , Wolock, I., Badaliance, R., Characterization of strain-induced damage in composites based on the dissipated energy density: Part I. Basic scheme and formulation, J of Theor. Appl. Fract. Mech., 22, (1995) 71-96
11. Yates, E., AGARD Standard Aeroelastic Configuration for Dynamic Response, Candidate Configuration I. – Wing 445.6, NASA TM-100492, 1987.
12. Lesoinne, M., Farhat, C., A Higher-Order Subiteration Free Staggered Algorithm for Nonlinear Transient Aeroelastic Problems, AIAA Journal 36, No. 9, (1998) 1754-1756.
13. Lieu, T., Lesoine, M., Parameter Adaptation of Reduced Order Models for Three-Dimensional Flutter Analysis, 42^{nd} AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, (2004).