Frank Hoffmann · David J. Hand
Niall Adams · Douglas Fisher
Gabriela Guimaraes (Eds.)

Advances in Intelligent Data Analysis

LNCS 2189

4th International Conference, IDA 2001
Cascais, Portugal, September 2001
Proceedings

Springer

Frank Hoffmann David J. Hand Niall Adams
Douglas Fisher Gabriela Guimaraes (Eds.)

Advances in Intelligent Data Analysis

4th International Conference, IDA 2001
Cascais, Portugal, September 13-15, 2001
Proceedings

Springer

Volume Editors

Frank Hoffmann
Royal Institute of Technology, Centre for Autonomous Systems
10044 Stockholm, Sweden
E-mail: hoffmann@nada.kth.se

David J. Hand
Niall Adams
Imperial College, Huxley Building
180 Queen's Gate, London SW7 2BZ, UK
E-mail: {d.j.hand,n.adams}@ic.ac.uk

Douglas Fisher
Vanderbilt University, Department of Computer Science
Box 1679, Station B, Nashville, TN 37235, USA
E-mail: dfisher@vuse.vanderbilt.edu

Gabriela Guimaraes
New University of Lisbon, Department of Computer Science
2825-114 Caparica, Portugal
E-mail: gg@di.fct.unl.pt

Cataloging-in-Publication Data applied for

Die Deutsche Bibliothek - CIP-Einheitsaufnahme

Advances in intelligent data analysis : 4th international conference ;
proceedings / IDA 2001, Cascais, Portugal, September 13 - 15, 2001. Frank
Hoffmann ... (ed.). - Berlin ; Heidelberg ; New York ; Barcelona ; Hong Kong ;
London ; Milan ; Paris ; Tokyo : Springer, 2001
(Lecture notes in computer science ; Vol. 2189)
ISBN 3-540-42581-0

CR Subject Classification (1998): H.3, I.2, G.3, I.5.1, I.4.5, J.2, J.1, J.3

ISSN 0302-9743
ISBN 3-540-42581-0 Springer-Verlag Berlin Heidelberg New York

Springer-Verlag Berlin Heidelberg New York
a member of BertelsmannSpringer Science+Business Media GmbH

http://www.springer.de

Typesetting: Camera-ready by author, data conversion by Boller Mediendesign
Printed on acid-free paper SPIN: 10840583 06/3142 5 4 3 2 1 0

Preface

These are the proceedings of the fourth biennial conference in the *Intelligent Data Analysis* series. The conference took place in Cascais, Portugal, 13–15 September 2001. The theme of this conference series is the use of computers in intelligent ways in data analysis, including the exploration of intelligent programs for data analysis. Data analytic tools continue to develop, driven by the computer revolution. Methods which would have required unimaginable amounts of computing power, and which would have taken years to reach a conclusion, can now be applied with ease and virtually instantly. Such methods are being developed by a variety of intellectual communities, including statistics, artificial intelligence, neural networks, machine learning, data mining, and interactive dynamic data visualization. This conference series seeks to bring together researchers studying the use of intelligent data analysis in these various disciplines, to stimulate interaction so that each discipline may learn from the others. So as to encourage such interaction, we deliberately kept the conference to a single track meeting. This meant that, of the almost 150 submissions we received, we were able to select only 23 for oral presentation and 16 for poster presentation. In addition to these contributed papers, there was a keynote address from Daryl Pregibon, invited presentations from Katharina Morik, Rolf Backhofen, and Sunil Rao, and a special 'data challenge' session, where researchers described their attempts to analyse a challenging data set provided by Paul Cohen. This acceptance rate enabled us to ensure a high quality conference, while also permitting us to provide good coverage of the various topics subsumed within the general heading of intelligent data analysis.

We would like to express our thanks and appreciation to everyone involved in the organization of the meeting and the selection of the papers. It is the behind-the-scenes efforts which ensure the smooth running and success of any conference. We would also like to express our gratitude to the sponsors: Fundação para a Ciência e a Tecnologia, Ministério da Ciência e da Tecnologia, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, Fundação Calouste Gulbenkian and IPE Investimentos e Participações Empresariais, S.A.

September 2001

Frank Hoffmann
David J. Hand
Niall Adams
Gabriela Guimaraes
Doug Fisher

Organization

IDA 2001 was organized by the department of Computer Science, New University of Lisbon.

Conference Committee

General Chair:	Douglas Fisher (Vanderbilt University, USA)
Program Chairs:	David J. Hand (Imperial College, UK)
	Niall Adams (Imperial College, UK)
Conference Chair:	Gabriela Guimaraes (New University of Lisbon, Portugal)
Publicity Chair:	Frank Höppner (Univ. of Appl. Sciences Emden, Germany)
Publication Chair:	Frank Hoffmann (Royal Institute of Technology, Sweden)
Local Chair:	Fernando Moura-Pires (University of Evora, Portugal)
Area Chairs:	Roberta Siciliano (University of Naples, Italy)
	Arno Siebes (CWI, The Netherlands)
	Pavel Brazdil (University of Porto, Portugal)

Program Committee

Niall Adams (Imperial College, UK)
Pieter Adriaans (Syllogic, The Netherlands)
Russell Almond (Educational Testing Service, USA)
Thomas Bäck (Informatik Centrum Dortmund, Germany)
Riccardo Bellazzi (University of Pavia, Italy)
Michael Berthold (Tripos, USA)
Liu Bing (National University of Singapore)
Paul Cohen (University of Massachusetts, USA)
Paul Darius (Leuven University, Belgium)
Fazel Famili (National Research Council, Canada)
Douglas Fisher (Vanderbilt University, USA)
Karl Froeschl (University of Vienna, Austria)
Alex Gammerman (Royal Holloway, UK)
Adolf Grauel (University of Paderborn, Germany)
Gabriela Guimaraes (New University of Lisbon, Portugal)
Lawrence O. Hall (University of South Florida, USA)
Frank Hoffmann (Royal Institute of Technology, Sweden)
Adele Howe (Colorado State University, USA)
Klaus-Peter Huber (SAS Institute, Germany)
David Jensen (University of Massachusetts, USA)
Joost Kok (Leiden University, The Netherlands)
Rudolf Kruse (University of Magdeburg, Germany)
Frank Klawonn (University of Applied Sciences Emden, Germany)

Hans Lenz (Free University of Berlin, Germany)
David Madigan (Soliloquy, USA)
Rainer Malaka (European Media Laboratory, Germany)
Heikki Mannila (Nokia, Finland)
Fernando Moura Pires (University of Evora, Portugal)
Susana Nascimento (University of Lisbon, Portugal)
Wayne Oldford (University of Waterloo, Canada)
Albert Prat (Technical University of Catalunya, Spain)
Peter Protzel (Technical University Chemnitz, Germany)
Giacomo della Riccia (University of Udine, Italy)
Rosanna Schiavo (University of Venice, Italy)
Kaisa Sere (Abo Akademi University, Finland)
Roberta Siciliano (University of Naples, Italy)
Rosaria Silipo (Nuance, USA)
Floor Verdenius (ATO-DLO, The Netherlands)
Stefan Wrobel (University of Magdeburg, Germany)
Hui XiaoLiu (Brunel University, UK)
Nevin Zhang (Hong Kong University of Science and Technology, Hong Kong)

Sponsoring Institutions

Fundação para a Ciência e a Tecnologia, Ministério da Ciência e da Tecnologia
Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa
Fundação Calouste Gulbenkian
IPE Investimentos e Participações Empresariais, S.A.

Table of Contents

The Fourth International Symposium on Intelligent Data Analysis

Feature Characterization in Scientific Datasets

Elizabeth Bradley,[1] Nancy Collins,[1*] and W. Philip Kegelmeyer[2]

[1] University of Colorado, Department of Computer Science, Boulder, CO 80309-0430
lizb,collinn@cs.colorado.edu,

[2] Sandia National Laboratories, P.O. Box 969, MS 9951, Livermore, CA, 94551-0969
wpk@ca.sandia.gov

Abstract. We describe a preliminary implementation of a data analysis tool that can characterize features in large scientific datasets. There are two primary challenges in making such a tool both general and practical: first, the definition of an interesting feature changes from domain to domain; second, scientific data varies greatly in format and structure. Our solution uses a hierarchical feature ontology that contains a base layer of objects that violate basic continuity and smoothness assumptions, and layers of higher-order objects that violate the physical laws of specific domains. Our implementation exploits the metadata facilities of the SAF data access libraries in order to combine basic mathematics subroutines smoothly and handle data format translation problems automatically. We demonstrate the results on real-world data from deployed simulators.

1 Introduction

Currently, the rate at which simulation data can be generated far outstrips the rate at which scientists can inspect and analyze it. 3D visualization techniques provide a partial solution to this problem, allowing an expert to scan large data sets, identifying and classifying important features and zeroing in on areas that require a closer look. Proficiency in this type of analysis, however, requires significant training in a variety of disciplines. An analyst must be familiar with domain science, numerical simulation, visualization methods, data formats, and the details of how to move data across heterogeneous computation and memory networks, among other things. At the same time, the sheer volume of these data sets makes this analysis task not only arduous, but also highly repetitive. A logical next step is to automate the feature recognition and characterization process so scientists can spend their time analyzing the science behind promising or unusual regions in their data, rather than wading through the mechanistic details of the data analysis. This paper is a preliminary report on a tool that does so.

General definitions of features are remarkably hard to phrase; most of those in the literature fall back upon ill-defined words like "unusual" or "interesting" or

* Supported by the DOE ASCI program through a Level 3 grant from Sandia National Laboratories, and a Packard Fellowship in Science and Engineering.

F. Hoffmann et al. (Eds.): IDA 2001, LNCS 2189, pp. 1–12, 2001.

"coherent." Features are often far easier to *recognize* than to *describe*, and they are also highly domain-dependent. The structures on which an expert analyst chooses to focus — as well as the manner in which he or she reasons about them — necessarily depend upon the physics that is involved, as well as upon the nature of the investigation. Meteorologists and oceanographers are interested in storms and gyres, while astrophysicists search for galaxies and pulsars, and molecular biologists classify parts of molecules as alpha-helices and beta-sheets. Data types vary — pressure, temperature, velocity, vorticity, etc. — and a critical part of the analyst's expert knowledge is knowing which features appear in what data fields.

In this paper, we describe a general-purpose feature characterization tool and validate it with several specific instances of problems in one particular field: finite element analysis data from computer simulations of solid mechanics problems. One of our goals is to produce a practical, useful tool, so we work with data from deployed simulators, in a real-world format: ASCI's SAF, a *lingua franca* used by several of the US national labs to read and write data files for large simulation projects. This choice raised some interoperability issues that are interesting from an IDA standpoint, as discussed in section 2 below. The SAF interface provides access to a geometric description of a computational mesh, including the spatial positions of the mesh points (generally xy or xyz) and the type of connectivity, such as triangles or quads, plus information about the physics variables, such as temperature or velocity. Given such a snapshot, our goal is to characterize the features therein and generate a meaningful report. We began by working closely with domain scientists to identify a simple ontology[1] of distinctive coherent structures that help them understand and evaluate the dynamics of the problem at hand. In finite-element applications, as in many others, there are two kinds of features that are of particular interest to us:

- those that violate the continuity and smoothness assumptions that are inherent in both the laws of physics and of numerical simulation: spikes, cracks, tears, wrinkles, etc. — either in the mesh geometry or in the physics variables.
- those that violate higher-level physical laws, such as the requirement for normal forces to be equal and opposite when two surfaces meet (such violations are referred to as "contact problems").

Note that we are assuming that expert users *can* describe these features mathematically; many of the alternate approaches to automated feature detection that are described in section 5 do not make this assumption. The knowledge engineering process is described in section 3.1 and the algorithms that we use to encapsulate the resulting characterizations, which rely on fairly basic mathematics, are described in section 3.2. We have tested these algorithms on roughly a half-dozen data sets; the results are summarized in section 4.

[1] Formally, an ontology seeks to distill the most basic concepts of a system down into a set of well defined nouns and verbs (objects and operators) that support effective reasoning about the system.

2 Data Formats and Issues

DMF[15] is a joint interoperability project involving several US national labs. Its goal is to coordinate the many heterogeneous data handling libraries and analysis tools that are used by these organizations, and to produce standards and libraries that will allow others to exploit the results. This project is motivated by the need to perform simulations at the system level, which requires formerly independent programs from various disciplines to exchange data smoothly. The attendant interoperability problems are exacerbated by the growing sophistication and complexity of these tools, which make it more difficult to adapt them to new data formats, particularly if the new format is richer than the old. The specific DMF data interface that we use, called SAF[11], exploits *metadata* — that is, data about the data — to solve these problems. Used properly, metadata can make a dataset *self-describing*. SAF, for example, captures not only the data values, but also the geometry and topology of the computational grid, the interpolation method used inside each computational element, and the relationships between various subsets of the data, among other things. Its interface routines can translate between different data formats automatically, which confers tremendous leverage upon tools that use it. They need only handle one type of data and specify it in their metadata; SAF will perform any necessary translation. In our project, this is important in both input and output. Not only must we handle different kinds of data, but we must also structure and format the results in appropriate ways. As discussed at length in the scientific visualization literature, different users need and want different data types and formats, so reporting facilities must be flexible. Moreover, the consumer of the data might not be a person, but rather another computer tool in a longer processing pipeline. For example, output generated by the characterization routines developed in this paper might be turned into a simple ascii report or formatted into an html page for viewing with a browser by a human expert, and simultaneously fed to a visualization tool for automatic dataset selection and viewpoint positioning. For all of these reasons, it is critical that data be stored in a format that supports the generation and use of metadata, and SAF is designed for exactly this purpose.

Metadata is a much broader research area, and SAF was not the first data model to incorporate and use it. Previous efforts included PDBlib, FITS, HDF, netCDF, VisAD, and DX, among others[4,16,5,8,9] — data formats that enabled analysis tools to reason about metadata in order to handle the regular data in an appropriate manner. While metadata facilities are of obvious utility to the IDA process, they are also somewhat of a Pandora's Box; as simulation tools increase in complexity, effective analysis of their results will require a corresponding increase in the structure, amount, and complexity of the metadata. This raises a host of hard and interesting ontology problems, as well as the predictable memory and speed issues, which are beyond the scope of the current paper.

The SAF libraries are currently in alpha-test release[2]. Because of this, few existing simulation, analysis, and visualization tools understand SAF's native interface. Our early development prototypes, for instance, used the SAF library directly for data access, but had to convert to the OpenDX file format for visualization: the very kind of translation that SAF is intended to obviate. Because visualization is so critical to data analysis, there has been some recent progress in adapting existing visualization tools to parse SAF input. In the first stages of our project, however, such tools did not exist, so we used OpenDX for visualization. We recently began converting to a SAF-aware visualization tool called EnSight[1], but this has not been without problems. Data interface libraries are subject to various chicken-and-egg growing pains. The tools need not understand a format until an interesting corpus of data exists in that format; scientists are understandably unwilling to produce data in a format for which no analysis tools exist. Intelligent data analysis tools that take care of low-level interoperability details can remove many barriers from this process.

3 Intelligent Analysis of Simulation Data

3.1 Knowledge Engineering

In order to automate the feature characterization process, we first needed to understand how human experts perform the analysis. We spent several days with various project analysts at Sandia National Laboratories, observing as they used existing tools on different kinds of data. We focused in on what they found important, how they identified and described those features, how they reasoned about which data fields to examine for a given stage of the process, and how the entire process changed if they were trying to prove or disprove a particular hypothesis. Most of the features of interest to these experts, we found, are clued from local geometry of the simulation mesh; inverted elements with non-positive volume, spikes, wrinkles, dimples, and so on. A smaller set of features of interest are extrema in the physics variables: hot spots and the like. We used this information to specify a simple ontology: that is, a set of canonical features (spikes, tears, cracks, etc.), together with mathematical descriptions of each — the statistical, geometric, and topological properties that define them. We also studied how the experts wrote up, reported, and used their results.

The Sandia analysts view the mechanical modeling process in two stages. The first is model debugging, wherein they ensure that the initial grid is sound, that the coupling is specified correctly between various parts of the model, and that the modeling code itself is operating correctly. The second is the actual simulation, where they examine the data for interesting physical effects: vibrational modes, areas that exceed the accepted stress tolerances, etc. We found that features play important roles in both phases, and that the sets of features actually

[2] We are a designated alpha-test group, and a secondary goal of this project is to provide feedback to the DMF developers, based on our experiences in designing an intelligent data analysis tool around this format.

overlapped. A spike in the results, for instance, can indicate either a numerical failure or a real (and interesting) physical effect. In some cases, reasoning about features let analysts identify model errors that were undetectable by traditional numerical tests like overflow, divide-by-zero, etc. One scientist described a simulation of an automobile engine compartment, including the front bumper. Due to a numerically innocuous error, one of the grid points moved to a location well beyond the back end of the entire car. This obviously non-physical situation — which was immediately visible to the analyst as a feature — flagged the model as faulty, even though no numerical fault occurred.

Note that features can involve the mesh coordinates, the physics variables, and sometimes both. Vertical relief, for instance, is a property of surface geometry, not the value of the physics variables upon that surface. Conversely, calculation of the highest temperature on a surface depends solely on the physics variables. Often, analysts are interested in features that involve both: say, the temperature or wind speed at the highest point on the landscape, or the position of the hottest point. Often, too, their underlying assumptions about geometry and about physics are similar, which can lead to some terminology confusion. A spike in temperature and a spike on the surface are similar in that both violate smoothness assumptions, but the mathematics of their characterization is quite different. This is actually a symptom of a deeper and more interesting property of features: like data analysis itself, they are hierarchical. All surfaces, whether numerical or physical, are generally continuous and smooth, so tears and spikes are likely to be considered to be features in *any* domain. If one knows more about the physics of the problem, other features become interesting as well. In contact problems, for instance — where two surfaces touch one another — the normal forces at the intersection of the two surfaces should be equal and opposite and surfaces should certainly not interpenetrate. Violations of these physical realities are interesting features. To capture these layers of meaning, our feature ontology is hierarchical. It contains a baseline set of features that rest on assumptions that are true of *all* physical systems, together with layers of higher-order features that are specific to individual domains (and sub-domains and so on). Currently, we have finished implementing two such layers: the baseline one mentioned above (spikes *et al.*) and a contact-problem one, which defines deviation from equal-and-opposite as a feature. Both are demonstrated in section 4.

3.2 Algorithms

Given the feature ontology described in the previous section, our next task was to develop algorithms that could find instances of those features in DMF data snapshots and generate meaningful reports about their characteristics. In order to make our work easily extensible, we structured the overall design so as to provide a general-purpose framework into which characterization routines specific to the features of a given domain can be easily installed. In particular, we provide several basic building-block tools that compute important statistical, geometrical, and topological information — about the mesh itself and about the values of the physics variables that are associated with each point in the mesh.

Their results are stored using the SAF library format, complete with metadata that allow them to be combined in different ways to assess a wide variety of features in a range of domains. Often, there is more than one way to find a single feature; a surface spike, for instance, can be characterized using statistics (a point that is several σ away from the mean) or geometry (a point where the slope changes rapidly).

Our current set of basic building blocks is fairly straightforward:

- `normals()`, which takes a DMF dataset and computes the unit-length normal vector to each mesh element.
- `topological-neighbors()`, which takes a DMF dataset and an individual mesh element m and returns a list of mesh elements that share an edge or a vertex with m.
- `geometric-neighbors()`, which takes a DMF dataset, an individual mesh element m and a radius r, and returns a list of mesh elements whose vertices fall entirely within r of the centroid of m.
- `statistics()`, which takes a DMF dataset and a specification of one variable (one of the mesh coordinates or physics variables) and computes the maximum, minimum, mean, and standard deviation of its values.
- `displacements()`, which takes a DMF dataset, finds all neighboring[3] pairs of vertices, measures the xyz distance between them, and reports the maximum, minimum, mean, and standard deviation of those distances

In addition, we provide various vector calculus facilities (e.g., dot products) and distance metric routines.

As an example of how these tools work, consider Fig. 1. The vectors computed by `normals()` are shown emanating from the center of each mesh face. In a regular mesh, finding topological neighbors could be trivial. SAF, how-

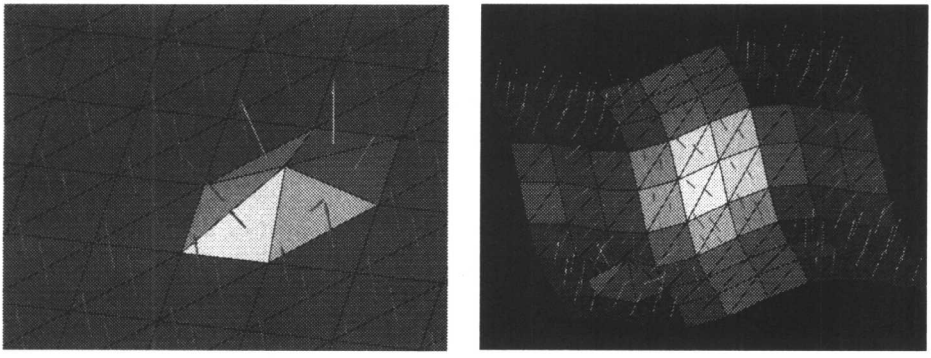

Fig. 1. 3D surface mesh examples, showing the vectors computed by the `normals()` function.

[3] *Topologically* neighboring

ever, is designed to be able to represent irregular and adaptive meshes as well, so the current version of SAF only provides neighbor information implicitly. For this reason, we preprocess the DMF data at the beginning of the characterization run and place it in a data structure that makes the topological information explicit. Our current design maintains a single list of vertices, including xyz position and the values of any associated physics variables. Three other lists point into this vertex list — a face list, an edge list, and a normal list — making it easy to look for shared edges or vertices and deduce neighbor relationships. In the examples in Fig. 1, each triangle has three "face neighbors" and at least three other "vertex neighbors," all of which are returned by `topological-neighbors`. The `geometrical-neighbors` function is a bit more complicated; it calls `topological-neighbors`, measures the Euclidean distances between the vertices of the resulting triangles and the centroid of the original element, discards any element whose vertices do not all fall within the specified distance, and iteratively expands on the others. The `statistics()` and `displacements()` routines use simple traditional methods. The left-hand surface in Fig. 1, for instance, is completely flat, with the exception of the bump in the foreground, and the `statistics()` results reflect the appropriate mean height of the surface and a very small standard deviation. The right-hand surface fluctuates somewhat, so the standard deviation is larger. In both cases, the `displacements()` results would likely be uninformative because the edge lengths of the elements are fairly uniform.

There are a variety of ways, both obvious and subtle, to improve on the toolset described above. We are currently focusing on methods from computational geometry[12] (e.g., Delaunay triangulation) and computational topology, such as the α-shape[7], and we have developed the theoretical framework and some preliminary implementations of these ideas[13,14]. Since features are often easier to *recognize* than to *describe*, we are also exploring the use of machine learning techniques to discover good values for the heuristic parameters that are embedded in these computational geometry and topology algorithms.

4 Results and Evaluation

We have done preliminary evaluations of the algorithms described in the previous section using half a dozen datasets. For space reasons, only two of those datasets are discussed here; please see our website[4] for further results, as well as color versions of all images in this paper. The first dataset, termed `irregular-with-spike`, is shown in Fig. 2. It consists simply of an irregular surface mesh; no physics variables are involved. Such a dataset might, for instance, represent the surface of a mechanical part. As rendered, this surface contains an obvious feature — a vertical spike — to which the eye is immediately drawn. Such a feature may be meaningful for many domain-dependent and -independent reasons: as an indicator of numerical problems or anomalies in the physics models, or perhaps

[4] `http://www.cs.colorado.edu/~lizb/features.html`