1 Initiative in Scientific Symbolic Computation

To support the Vision 2020 for applied Information Technology, we propose an initiative in Scientific Symbolic Computation. The development of symbolic algorithms for manipulating discrete and continuous structures. The recognition of abstract patterns and the technical development of their properties is a keystone of the technological engine of our society. To prepare students for the advancement of these enterprises, they must be trained in paradigms that result in new ideas, structures and patents. Finally, the properties must be made algorithmic providing the raw materials and design specifications for the software that runs the engine of technology.

We propose the organization of a group of faculty across disciplinary boundaries around the common theme of Scientific Symbolic Computation. This IT-intensive area will prepare students for the applications of symbolic computation (see below), can further research into applications of symbolic description of phenomena and develop necessary accompanying algorithms.

Symbolic computation has its roots in the long history of the development of notation and symbols for mathematical structures. With the advent of the stored program computer, it came of age with the development of software capable of performing the manipulation of the formulae that is the toolkit of every practicing mathematician, scientist and engineer.

For example, within the biological world, there is great need for better tools to elucidate the hidden structure and patterns important to DNA and, in particular the human genome. A current major problem of the moment is that of protein folding. Another application of the symbolic approach has been the use of knot and link polynomials to differentiate between different proposed mechanisms for DNA processes. The computation of these invariants requires exact computations using a symbolic, formal property of skein relations and cannot be approached by an numerical approximation. More generally, similar symbolic techniques can be used for the description of 3 (and higher)-dimensional problems of configurations of molecules.

Additional application of the algebraic symbolic method arise in mechanical engineering and robotics. We mention several current problems: A robot arm involves segments and joints. The joints come in two types, revolute, which rotate, and prismatic, which can extend and retract. The problem in design and control is to describe the points of space which can be reached by such a robotic arm. It turns out that the situation is governed by systems of non-linear polynomial equations of the type studied in algebraic geometry. While this is just a beginning example, it points out a current serious interaction between current engineering problems and modern mathematics. Our second example are the Bezier cubics, certain parametric cubic curves, that are used in computer aided geometric design to create complex shapes like automobile hoods or airplane wings. Remarkably, these same curves are also used in the

page description language PostScript. Yet these cubic curves are the daily bread and butter of algebraic geometry. In a similar vein, the configuration spaces of graphs are used to model the motion of multiple robots within a network.

Much of the research in graph theory during the last couple of decades has been motivated by rapid advances in information technology. Graphs are extensively used in a vast array of practical applications,including the design of electronic circuits (Very Large ScaleIntegration), design and analysis of communication networks, and real-time routing information flow. The new developments in graph theory, especially those related to the Graph Minors project of Robertson and Seymour, have provided theoretical background for efficient algorithms. In particular, some important problems, like the k-linkage problem, which previously were suspected intractable, have been shown to be solvable in polynomial time. Moreover, most problems have been shown to have linear-time algorithms when restricted to graphs with special structure: bounded tree-width. However, much remains to be done: It is widely believed that we are just beginning to utilize the theoretical advances. Most of the new algorithms are still awaiting efficient implementations and are likely to be significantly improved in the near future; other efficient algorithms are still expected to emerge. Even though graphs provide an abstraction that is invaluable in making real-life problems suitable for computation, designing and implementing graph algorithms is difficult, as it requires a solid grasp of symbolic methods and significant programming experience. Some fundamental problems, like isomorphism testing, are still intractable and require significant programming experience for selecting most beneficial heuristics. Moreover, manipulating graphs requires processing a combination two different aspects exhibited by graphs: symbolic and graphic. The problem is exacerbated by incompatibility of various implementation projects around the world, due to a lack of standard for a dynamic data structure for representing graphs.

One of the persons who has both the theoretical knowledge of graph theory and experience in designing and implementing graph algorithms is Professor Oporowski of the LSU Mathematics Department. He designed and implemented software for solving some of his research projects, including an important problem in circuit design, which asks to find all graphs that require more than one crossing when drawn on the plane. Parts of his software laid the foundations for SunGraph, an interactive graph manipulation package that was later expanded by a group of researchers and graduate students of Georgia Institute of Technology. Professor Oporowski plans to continue his work in graph algorithms, both in laying the foundations for new algorithms and in implementing them for use by other researchers.

Another application area of current interest within the LSU Department of Mathematics concerns solving systems of polynomial equations. First we note that polynomial systems arise in many (perhaps most, if linear equations are considered polynomial, which they are) applications. Examples are computer vision and robot motion.

An active area is the search for efficient algorithms for solving them. This impinges both on traditional numerical analysis and symbolic methods.

One symbolic method of recent intense study is the theory and application of resultants. This has involved both theoretical studies (e.g. Gelfand, Kapranov, Zelevinski, Sturmfels, D. Cox) and research in computer science and engineering (J. Canny at Berkeley, I. Emiris (INRIA, France), D. Manocha (UNC Chapel Hill), R. Goldman at Rice and T. Saxena (SUNY Albany). The point to emphasize here is that very sophisticated mathematical tools from algebra and algebraic geometry are necessary to understand this (and this expertise exists at LSU) and that engineers and computer scientists have actually begun to learn (and to refine and extend) these methods for implementation in practical algorithms. Currently there are two graduate students (of Prof. Hoffman) in mathematics working on projects related to this: H. Wang who is working on a method devised by Cederberg and Cox for implicitization of algebraic equations and M. Holcomb who is studying the topology of configuration spaces of planar linkages.

Another facet of this initiate could be the development, testing and refinement of algorithms for symbolic computation for a specific field of theoretical inquiry: algebraic number theory, combinatorics, algebraic geometry, link theory, configuration spaces, inter alia. With an appropriately general and novel focus, there is potential for development of commercial software, if support for testing, refinement and distribution is initially provided. Two such promising directions, still in their infancy, are the development of interval arithmetic and of a robust implementation of parallelization techniques for a symbolic computation engine.

In summary, the development of a faculty cluster focused on scientific symbolic computation would build on expertise already on campus and provide the foundation for the enhanced transfer of the fruits of academic research to the wider information technology enterprise.