

Craig Stewart Holman, Ph.D.

13050 Dahlia Circle, Apt. 311
Eden Prairie, Minnesota 55344

www.patterncraft.com

(952) 388-8727
craig@patterncraft.com

Services Offered

- Research in computer science, especially in algorithms and graph theory
- Algorithm engineering: design, implementation, and tuning
- Software architecture, design, and development in C#, C++, C, PL/SQL, rule systems; software engineering
- Object-oriented design of frameworks, class libraries, components, and applications; design patterns
- Review of architectures, designs and code; review and tuning of development methodologies
- Instruction in C#, C++, C, .NET, PL/SQL, rule systems, object-oriented analysis and design, optimization, testability, GUI design, and other aspects of software development
- Creativity, innovation, and performance

Available immediately.

Education

Ph.D. Northwestern University, Computer Science, 1990

M.S. Northwestern University, Computer Science, 1988

B.S. Northeastern Illinois University, Information Science, Mathematics minor, 1986

B.A. University of Illinois at Urbana-Champaign, English Literature, Mathematics minor, 1981

Dissertation : Elements of an Expert System for Determining the Satisfiability of General Boolean Expressions

Thesis : A Method for Determining the Satisfiability of General Boolean Expressions

Qualifiers: Artificial Intelligence, Formal Theory, Programming Languages and Compiler Theory

Research

I have been conducting private research for 22 years without publishing in journals. My research has been focused on developing algorithms for challenging problems, including several NP-complete problems, particularly in graph theory. In 2008, I began a research blog at www.patterncraft.com. Entries on most of the following topics may be found in this blog. I am writing At Play, a book on some of my research in graph theory, especially on cliques, that aims at enticing readers to engage in mathematics and algorithms.

Boolean Satisfiability – My research for my Master's thesis resulted in a new algorithm for determining whether a general boolean expression (not necessarily in CNF or DNF) is satisfiable. The algorithm assumes that an expression evaluates to true, reverse propagates that constraint, constructs a constraint tree, and from that constructs a directed acyclic graph of constraints having a single source and sink. The expression is satisfiable iff there exists a consistent set of constraints along a path from the source to the sink. Assuming that the expression evaluates to false can be used for falsifiability testing. The constraint calculus that I defined was proven to be both sound and complete. The algorithm has some ordering parameters and I did some preliminary studies of the effects of settings for these parameters on algorithm efficiency. I also sought and found some metrics that were predictive of satisfiability, falsifiability, and the amount of work the algorithm would need to determine the logical class of an expression.

Boolean Simplification – My research for my doctoral dissertation resulted in a new algorithm for simplifying a general boolean expression to a canonical logically equivalent expression. The reduction process operates upon a constraint tree. I found a set of eight reductive graph transformations that form a complete set of reductions, meaning that if the eight reductions are applied in any order to any application sites in the tree until no further application is possible, the same canonical constraint tree will result. This property allowed me to be concerned with efficiency rather than minimality. Proving that this set of reductions was complete and providing an algorithm for the reduction process were the central results of my dissertation. This algorithm has ordering parameters that were not explored at that time. I sought and found some metrics that were predictive of the amount of work required for reduction and the size of the resulting reduced expressions. It was during this research that I first considered the problem of how to choose the best tools for a task.

k-Clique Exists – Determining whether a graph contains a clique of size k remains my favorite problem - it has fascinated me for the past eighteen years. I have worked on six new algorithms for its solution. A clique is a complete subgraph of a graph - there is an edge between every pair of nodes in a clique. The approach that I've been focusing on for the past several years began as an investigation of a classic searching technique. It changed into a studies on the reduction of the search space and has become an exercise in squeezing blood from a stone. The reduction of the search space now quite often solves the problem, rendering a search unnecessary.

Craig Stewart Holman, Ph.D.

Maximum Cliques and Maximum Clique Size – These are the problems of finding the maximum cliques of a graph (the cliques having the largest size) and the size of these maximum cliques. The approach that I've been using is an adaptation of my splitting algorithm for finding the maximal cliques of a graph.

Maximal Cliques – A maximal clique is a clique that is not a proper subgraph of any other clique of the graph. The Maximal Cliques problem is the construction of the set of the maximal cliques of a graph. I found a way to use a splitting technique to build up the set of maximal cliques. A few years ago, when Vodafone, a European telecommunications company, was looking for a better maximal cliques problem, I ran two of their sample graphs through my algorithm. It was able to construct complete sets of maximal cliques for the graphs within two minutes, while their algorithm was having trouble producing incomplete sets over the course of a weekend. They confirmed the correctness of my solutions down to the threshold that their algorithm had been permitted to use.

Graph Isomorphism – I'm close to completing a new algorithm for determining whether two labeled, undirected, and unweighted graphs are identical except for their labels. For each graph, it constructs a canonical representation that incrementally partitions the nodes of a graph into an equivalence class based upon (what will be) a universal invariant. The basic algorithm is beautiful and works efficiently ($O(N^4)$), but I found that, for some graphs, a supplemental signature is needed. Most of my time has been spent on this supplemental signature. I think I've nearly got it pinned down - I'm almost done with the proofs and have to rework the implementation of the supplement. The algorithm should be easily extended to handle directed and/or weight graphs.

Factoring Composite Integers Into Two Factors – Inspired by the no-longer-active RSA Labs Factoring Challenge, I've worked on a new algorithm framework for factoring a composite integer into two factors. This is a beautiful problem.

The Adaptive Market Pattern – This is the pattern for a multi-level free-market framework for selecting and applying the best tools for tasks that learns from experience and incorporates the elements of prediction, estimation, bidding, reputation, and adaptation. I first became interested in this problem when thinking about how to choose the best tool (algorithm + parameters) for determining the satisfiability of general boolean expressions. I have frequently encountered this problem and decided to generalize it into a pattern.

Problem-Solving Services – For several years, I've been interested in how a commercial problem-solving service could be organized. What I envision is software as a service, accessible through the internet, that would publish a catalog of problems it could solve (including many NP-complete and other challenging problems), accept instances of these problems to solve, and use the best algorithms and its experience with solving similar problems to attempt to solve these instances as efficiently as it can. An overview is at www.Patterncraft.com/Blog/Blog-090527.html

Active Blackboards – A blackboard is a software construct that can hold and serve objects of many varieties. At its simplest, it is a container of named objects. I'm interested in blackboards that can do far more. I've had fun thinking about templates for blackboards (e.g. crime investigation), blackboards that can perform reasoning, and blackboards that can perform tasks as appropriate. Blackboards should have the ability to source and history of all facts and deductions as well as a measure of confidence that can be justified. Blackboards should also be able to support What-If analysis. I could really use some of these enhanced blackboards for implementing some algorithms, the Adaptive Market pattern, and problem-solving systems.

Emergent Network Dataflow Processing – When I took my first course in computer architecture, my professor introduced the topic of dataflow processing by describing it as a pool of fundamental processing elements out of which could be woven as needed a network for solving a specific problem through which data could flow. I thought that was beautiful and was disappointed that the current implementations of dataflow processing were much more prosaic. I decided several years ago that the design and simulation of such an emergent-network dataflow processor would be great fun.

The Business Object Model and Rule Systems Methodology – While a Rule Systems Architect, I designed an engine-agnostic business rule repository; worked towards a comprehensive business object model; designed and prototyped a task-oriented engine-agnostic business object model repository; developed a process for managing business rules, including identifying rule candidates, synthesizing rules; figured out how traditional programming can coexist with a rule systems approach; and adapted a light version of the Rational Unified Process so that it accommodates a rule systems approach.

Other topics – Subgraph Isomorphism, Universal Traversal Sequences, Ramsey Theory (Graph theory aspect), The Riemann Hypothesis, The Netflix Challenge, and Eternity II

Craig Stewart Holman, Ph.D.

Software Engineering Experience

Languages and Programming Technologies

C	25 years	Microsoft, Borland, DeSmet, and UNIX V, taught to undergraduates, IBM developers
C++	19 years	Microsoft, Borland, UNIX V; taught to undergraduate and graduate students
Visual C++	14 years	1.52 – 2008; .NET 3.5, STL, Boost, Win32 API, MFC, COM, OLE automation (Word, Excel), graphics, multithreading, messaging, GUI, enhanced controls, DLL, exception handling, serialization, memory-mapped files, private heaps, Unicode, embedded SQL, SNMP, NT services, performance optimization; taught to undergraduates and others
C# and .NET	2 years	Visual Studio 2005 and 2008. .NET 2 - 3.5; Visual Studio Tools for Office (VSTO); add-ins for Office 2003 and 2007 applications; Fluent Ribbon tabs for Office 2007 add-ins; robust multithreading, GUI, SQL, TCP, XML. Experience with namespaces including System.Collections (Generic, Specialized), ComponentModel, Data, Diagnostics, Drawing, Globalization, IO, Messaging, Reflection, Runtime (InteropServices, Remoting (Channels.Tcp), Serialization.Formatters), Text, Threading, Windows.Forms (Xml); Microsoft.Office (Core, DocumentFormat (OpenXml, OpenXml.Packaging), Interop (Excel, PowerPoint, Word); Microsoft.Windows.Forms; Microsoft.Win32. Exploring ASP.NET, LINQ, WPF, and WF.
Open XML	1 year	Experience parsing and manipulating Microsoft's new open format for Office files, including Microsoft Word 2007, PowerPoint 2007, and Excel 2007.
Rule Systems	21 years	Extensive coursework in Artificial Intelligence, Mathematical Logic, Automated Theorem Proving, Database Theory, and Deductive Databases; member of the Deductive Database Research Group at Northwestern University; teaching assistant for Automated Reasoning course; taught graduate course in Automated Reasoning; Otter; 2 years at Blue Cross Blue Shield of MN driving an enterprise business rules initiative, including vision, architecture, methodology, design, and prototyping; Pegasystems
Database	20 years	Theory, design, deductive database systems, dBase, embedded SQL, Oracle PL/SQL; Oracle 7i-9i; MS SQL 2005 stored procedures, SQL Express; taught undergraduate and graduate courses in database design and theory
Significant experience:		XML (DOM and SAX), HTML, Pascal, BASIC, Lua
Moderate experience :		Java; Assembler (6502), LISP, Logo, Modula-2, Prolog, SNOBOL
Limited experience :		Assembler (370, 3090, 8088), COBOL, FORTRAN, PL/I, Python, Visual Basic 5

Operating Environments

Windows Vista, XP, 2000, NT 4.0, NT 3.51, 98, 95, and 3.x; DOS, Unix System V, Linux, OS/2, MVS

Object-Oriented Analysis, Design, and Development

OOA, OOD, OO development (18 years); design patterns (13 years); Booch, Rational Rose; UML, ORM; Agile

Quality Assurance and Reviewing

Design and code reviewer (13 years); computer science teacher (review and evaluation) (8 years)

Standards and Methodology

Rule systems (2.5 years), C++ standards (2 years), development methodology (6 years), IS Standards (1.5 years)

Current Projects

Algorithm Development Graph Isomorphism, Maximal Cliques, k-Clique Exists, Factoring composite integers into two factors, Boolean expression satisfiability and simplification. Emphasis on finishing the implementation, analysis, and proofs of a new algorithm for determining Graph Isomorphism and writing up and posting research results. (Visual C++ 2005 – 2008, Visual C# 2005 – 2008, .NET 3.5, personal)

Algorithm Implementation Porting several of my libraries (e.g. Set and Graph) and applications that implement my Graph Isomorphism and Maximal Cliques algorithms from C++ to C#. (Visual C# 2008, .NET 3.5, personal)

Eternity II Solving the Eternity II puzzle. (Visual C++ 2008, personal)

Craig Stewart Holman, Ph.D.

Employment

SpeechGear

Northfield, Minnesota

Senior Software Developer, December 2007 – December 2008.

SpeechGear develops software for translation between natural languages.

Document Workbench Enhanced the natural language translation workbench that also provides translation services to SpeechGear's add-ins for Microsoft PowerPoint and Word. Enhancements included access to translation memory domains by integration with SQL Express, domain stacking, acronym expansion, robustness, and performance. (Visual C# 2005, .NET 2, MS SQL 2005, 2007–8, SpeechGear)

Document Add-ins for Microsoft Word Redesigned and rewrote natural language translation add-ins for Word 2003 and 2007. Devised and implemented a new capability for document-translation systems: the retention of nearly all formatting elements, including font properties and style. For example, the sentences

Some text is **bold**, some is underlined, and some is *italic*. Red text can **startle** cats.

could be translated and automatically formatted as

Algunos texto está escrito en **negrita**, subrayado algunos y algunos en *cursiva*. Texto rojo pueda **asustas** gatos.

New features also included the translation of most Word document components (e.g. footnotes, comments), translation caching, retranslation prevention, automatic protection from translation of text in scripts that should not be translated, user-specified protection of text from translation, interaction with Word via interop assemblies, and a tab on the fluent ribbon for the 2007 add-in. (Visual C# 2005, .NET 2, VSTO, 2007–8, SpeechGear)

Document Add-ins for Microsoft PowerPoint Redesigned and enhanced natural language translation add-ins for PowerPoint 2003 and 2007. New features included translation caching, retranslation prevention, automatic protection from translation of text in scripts that should not be translated, interaction with PowerPoint via interop assemblies, and a tab on the ribbon for the 2007 add-in. (Visual C# 2005, .NET 2, VSTO, 2007–8)

Translation Client/Server Prototype Developed prototypes of a translation server that can translate between several pairs of natural languages simultaneously and and multiple clients that use the translation server for translation. (Visual C# 2005, .NET 3.5, Message Queuing, 2008)

HTML Translator Developed an application for translating HTML files between natural languages. This application is a client of the translation server. (Visual C# 2005, .NET 3.5, 2008)

Open XML Parser Developed a hierarchy of classes for parsing, representing, and manipulating Microsoft Office PowerPoint and Word documents that are in the new Open XML file format. The implementation was not complete but was sufficient to provide a proof of concept. (Visual C# 2005, .NET 3.5, 2008)

National Cinemedia

Eden Prairie, Minnesota

Software Engineer, contractor, November 2006 – March 2007.

National Cinemedia prepares and distributes the half-hour programs that are shown before movies in many theaters.

Content Manager Advertising content manager for movie theaters. (Visual C++ 2005, 2006–7)

Code generator for XML-ready classes Generates source for classes from brief XML-encoded class descriptions. The generated classes are fully interoperable with the rich DOM and SAX wrapper classes described below (e.g. object tree from SAX, object tree from and to DOM). (Visual C++ 2005, 2006–7)

Rich wrapper classes for DOM and SAX for XML manipulation (Visual C++ 2005, 2006)

Blue Cross Blue Shield of Minnesota

Eagan, Minnesota

Rule Systems Architect, November 2003 – September 2006.

Business Object Model Repository Vision, requirements, and architecture for a complete BOM repository, with special emphasis on business rules; user-interface prototype for task-centric repository (Oracle 9i, PL/SQL, Apache, Web application, 2005–6)

Business Rules Initiative Vision, architecture, methodology, and design (2003–6)

Near Real-time Pharmacy Claims Processing Analysis and design (Pegasystems, 2006)

Connectivity Coordinator Filled new role, tasked with eliminating backlog and improving processes. Coordinated efforts of several groups responsible for establishing connections with clients for the transmission and processing of electronic files. Task was successfully completed. (2005)

Craig Stewart Holman, Ph.D.

Employment (continued)

Return

Wayzata, Minnesota

Senior Software Engineer, March 2002 – August 2003 (contractor for first five months).

Return facilitates reverse logistics, handling and accounting for product that is returned from retailers.

Web-based Reporting Architecture, design, and implementation (Oracle 7i, PL/SQL, Apache, Web Application, 2003–4)

Data Auditing Architecture, design, implementation (Oracle 7i, PL/SQL, Apache, Web Application, 2002–3)

Jasc Software

Eden Prairie, Minnesota

Senior Software Engineer, July 1998 – November 2001 (contractor for first five months).

Paint Shop Pro Versions 6, 7 and 8, design and implementation (Visual C++ 6, STL, COM, MFC, 2001)

Digital Video Editor Design and implementation of composition serialization and aspects of the user interface for a new product that was promising but later abandoned. (Visual C++ 6, STL, COM, MFC, 2000)

Animation Shop Versions 2 and 3, feature design and implementation, Windows 2000 certification research and preparation (Visual C++ 6, MFC, 2000)

Best Buy

Eden Prairie, Minnesota

Senior Technical Analyst, July 1996 – April 1998.

Enterprise-wide event forwarding Designed and implemented several programs for logging events, summarizing event logs, and forwarding events from event logs to HP OpenView via SNMP (Visual C++ 5, GUI, MFC, COM, NT service, multithreading, ODBC, embedded SQL, string patterns, DLLs, 1997–98, Best Buy)

ReplicationLogMonitor (Visual C++ 5, MFC, GUI, multithreading, scripting, 1997)

SQL Gateway Enhancements (Visual C++ 5, NT service, MFC, multithreading, sockets, registry, 1997)

Clear With Computers (now Firepond)

Edina, Minnesota

Senior Programmer / Analyst, June 1995 – July 1996.

Clear With Computers developed custom salesforce-automation applications for many corporations, especially in the automotive industry.

Proposal module of a salesforce-automation tool OLE automation of MS Word to generate proposals for truck/tractor purchases based upon information derived from other modules of a custom salesforce-automation tool for Freightliner, (Visual C++ 1.52, MFC, OLE automation, GUI, 1995–96)

Xavier University of Louisiana

New Orleans, Louisiana

Assistant Professor, Computer Science Department, 1993 – 1995.

University of North Dakota

Grand Forks, North Dakota

Assistant Professor, Computer Science Department, 1990 – 1993.

International Business Machines Corporation

Poughkeepsie, New York

Graduate Intern, System Performance / System Workload Analysis Department, Summers 1988, 1989.

Programs for reconstructing / analyzing channel programs from traces Rewrote PL/AS program in C, reducing its runtime from 24 hours of 3090 CPU time to less than 1 minute (C, Assembler, IBM 3090, MVS, 1988)

Northwestern University

Evanston, Illinois

Instructor, Department of Electrical Engineering and Computer Science, Fall quarter, 1989.

Senior Teaching Fellow, Department of Electrical Engineering and Computer Science, 1988 – 1990.

Teaching Assistant, Department of Electrical Engineering and Computer Science, 1987 – 1988.

Programmer, Solid-State Devices Laboratory, Summer 1987.

Honors and Awards

University of North Dakota Outstanding Faculty Advisor Award, 1992, *University of North Dakota*.

Dean of Students Exceptional Performance Award, 1992, *University of North Dakota*.

Senior Teaching Fellowship, 1988 - 1990, *Northwestern University*.

Graduate Teaching Assistantship, 1987 - 1988, *Northwestern University*.

Walter P. Murphy Fellowship, 1986 - 1987, *Northwestern University*.

Bachelor of Science degree awarded with high honors, 1986, *Northeastern Illinois University*.

Craig Stewart Holman, Ph.D.

Supervisory and Mentoring Experience

As an assistant professor in two universities, directed the activities of several teaching assistants, oversaw the research and thesis writing of several graduate students, and directed the independent studies of over twenty students. Invested a great deal of time working with advisees as well as students. Was awarded the University of North Dakota Outstanding Faculty Award, and the Dean of Students Exceptional Performance Award for my work with dyslexic students. Continued mentoring and tutoring several undergraduates after leaving teaching for software development.

In the role of the Rule Systems Architect for Blue Cross Blue Shield of Minnesota, worked with several people in different roles on educating them about what the business-rules approach was, how it could affect the business, and how to implement it effectively. These people included the senior vice president who was responsible for the business rules initiative, the vice president over my area, half a dozen business staff (director and below) who were charged with implementing the initiative on the business side, several enterprise architects, a few application architects, the technical business analysts, the director of the electronic claims processing division, and several developers.

Teaching Experience

Undergraduate : C, C++, Pascal, SQL, Modula-2, Logo; object-oriented analysis, design, and programming; problem analysis, computer architecture, algorithms, operating systems, database theory and design, AI, and automated reasoning.

Graduate : Database theory and deductive database systems, automated reasoning.

Professional : 2 courses on programming in C and presentations on C++ at IBM in Poughkeepsie, NY.

Relevant Coursework

Undergraduate : Algebra, Trigonometry, Calculus and Analytic Geometry I, II, and III, Computer Calculus Lab, Differential Equations, Real Analysis, Complex Variables, Discrete Mathematical Structures, Probability Theory and Applications I, Numerical Analysis, Computer Programming I, Algorithmic Processes, Data Structures, Advanced Programming Techniques, Compiler Theory, Operating Systems Concepts, Operating Systems Theory, UNIX Systems, Networking, IBM 360/370 Assembly Language/Architecture, Advanced Assembler Programming, COBOL Programming, Advanced COBOL Programming, PL/I Programming, General Physics I, II, and III, Intermediate Electricity and Magnetism I, Chemistry I and II, Psychology, Introduction to Linguistics, Descriptive English Grammar

Graduate : Mathematical Logic, Number Theory, Modern Algebra II (Linear Algebra), Theory of Computability, Turing Machines, and Recursive Function Theory, Computer Subsystems, Computer Architecture, Advanced Computer Organization, Robotics, Computer Networks, Database Systems, Relational Database Theory, Artificial Intelligence, Natural Language Processing, Theorem Proving, Automated Theorem Proving, Seminar on Advanced Theorem Proving, Linguistic Theory, Syntactic Analysis (Government-Binding Theory)

Professional training : Business Rules Forum 2004 and 2005 (80 hours), Pegasystems PegaRULES BPM and rule engine development / runtime environment (80 hours), Vitria BusinessWare BPM development environment (80 hours), Microsoft Visual C++ and MFC (80 hours), Object-Oriented Analysis and Design (40 hours), UML (40 hours)

Personal Interests

Baroque music, science, reading, mathematics, Greek history, writing, movies, friends, pets, and good conversation.