THE ACADEMY CORNER

No. 25

Bruce Shawyer

All communications about this column should be sent to Bruce Shawyer, Department of Mathematics and Statistics, Memorial University of Newfoundland, St. John's, Newfoundland, Canada. A1C 5S7.

Abstracts • Résumés

Canadian Undergraduate Mathematics Conference 1998 — Part 3

The Brachistochrone Problem

Nils Johnson
The University of British Columbia

The brachistochrone problem is to find the curve between two points down which a bead will slide in the shortest amount of time, neglecting friction and assuming conservation of energy. To solve the problem, an integral is derived that computes the amount of time it would take a bead to slide down a given curve $y(x)$. This integral is minimized over all possible curves and yields the differential equation $y(1 + (y')^2) = k^2$ as a constraint for the minimizing function $y(x)$. Solving this differential equation shows that a cycloid (the path traced out by a point on the rim of a rolling wheel) is the solution to the brachistochrone problem. First proposed in 1696 by Johann Bernoulli, this problem is credited with having led to the development of the calculus of variations. The solution presented assumes knowledge of one-dimensional calculus and elementary differential equations.

The Theory of Error-Correcting Codes

Dennis Hill
University of Ottawa

Coding theory is concerned with the transfer of data. There are two issues of fundamental importance. First, the data must be transferred accurately. But equally important is that the transfer be done in an efficient manner. It is the interplay of these two issues which is the core of the theory of error-correcting codes.

Typically, the data is represented as a string of zeros and ones. Then a *code* consists of a set of such strings, each of the same length. The most fruitful approach to the subject is to consider the set $\{0, 1\}$ as a two-element field. We will then only

consider codes which are vector spaces over this field. Such codes will be called *linear codes*.

Since there is no way to always ensure that the message has been transferred accurately, it is important to add redundant structure so that, except for extreme cases, the receiver will realize that there has been an error. This is called *error-detection*. It would be even better if the receiver were able to not only detect that an error has occurred, but to determine the location of the incorrect digit or digits, and thus determine the original message. This is the notion of *error correction*.

This theory uses in a fundamental way ideas from linear algebra over finite fields, such abstract algebra as principal ideal domains, and metric space theory. One of the reasons this is such an exciting subject is that it combines principles of abstract algebra with very concrete and important applications. Among the many applications are the minimization of noise from compact disc recordings, transmission of data from satellites, and transmission of financial data.

I will introduce the basic concepts of coding theory, focusing on linear codes. I will discuss several more specialized topics, such as the most important methods of error correction. I will also discuss a number of specific examples. I will conclude with a brief discussion of *convolutional codes*, one of the more recent advances in the field.

Fixed Points and Diagonalization: An Abstraction of Gödel's Theorem

Todd A. Kemp
University of Calgary

Gödel's Incompleteness Theorem is one of the most provocative results in Mathematical Logic. It may be summarized "In any *'sufficiently powerful'* consistent theory, there must always be theorems (true sentences) which are *unprovable*."

In this paper, I address the question of what this heuristic phrase *'sufficiently powerful'* means. I will appeal to abstract conditions put forward by Smullyan, and show that any system meeting them also satisfies Gödel's Theorem. Further, after outlining some of the basic terminology and major results in the area, I will demonstrate that Robinson's Arithmetic—a well-known Gödel system—indeed meets Smullyan's conditions, hence lending some credibility to the claim that these conditions are not only sufficient but necessary.

Smullyan also goes on to show that his abstract form of Gödel's Theorem is actually a special case of a general Fixed Point Theorem—one which is also the primitive framework behind a major result in Recursive Function Theory, and Smullyan's favourite puzzle—the Mockingburd Puzzle. I will discuss this Fixed Point Theorem, and how it relates to Gödel's theorem.

Misuse of Statistics

Theodoro Koulis
McGill University

Statistics play a big role in the sciences. Statistics provide scientists with tools that help them verify the significance of their data. However, since these tools are widely distributed in popular software packages such as Microsoft Excel, they are often misused. Using some test statistics, I will show how some basic procedures can be misdirected and how conclusions can be misinterpreted. These will be:

1. The sign test and hypothesis testing:
   Foundations of hypothesis testing (the likelihood ratio principle)
2. Goodness of fit test statistics:
   Good and bad measures: Pearson's χ^2 and Kolmogorov-Smirnov tests
3. Correlation and rank correlation (if time permits)

An Introduction to Random Walks from Pólya to Self-Avoidance

Michael Kozdron
University of British Columbia

This paper provides an introduction to random walks. We begin with some basic definitions and culminate with the classical theorem of Pólya that a simple random walk in $\mathbb{Z}^d$, $d \geq 3$ is transient and recurrent otherwise. We then discuss the more contemporary topic of self-avoiding random walks and survey some currently open problems. This paper assumes only a minimal background in probability including the notion of a random variable and an expectation.

Of Graphs and Coffi Grounds: Decompiling Java

Patrick Lam
McGill University

Java programmers write applications and applets in plain English-like text, and then apply a java compiler to the text to obtain *class files*. Class files, which are typically transmitted across the Web, are a low-level representation of the original text; they are not human-readable. Consider a compiler as a function from text to class files. My goal is to compute the inverse function: given the compiled class file, I wish to find the best approximation to the original text possible. This is called decompilation.

Given a class file, one can build an unstructured graph representation of the program. The main goal of my work is to develop graph algorithms to convert these unstructured graphs into structured graphs corresponding to high-level program constructs, and I will discuss these in detail. I shall also mention some results concerning possibility and impossibility which show that decompilation is always possible if the program may be lengthened.