European Journal of Operational Research 253 (2016) 543–556

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Invited Review

Origin and early evolution of corner polyhedra

Ralph Gomory*

New York University, Stern School of Business, United States

ARTICLE INFO

Article history:
Received 27 March 2015
Accepted 1 March 2016
Available online 8 March 2016

Keywords:
Integer programming
Cutting
Linear programming
Corner polyhedra

ABSTRACT

Corner Polyhedra are a natural intermediate step between linear programming and integer programming. This paper first describes how the concept of Corner Polyhedra arose unexpectedly from a practical operations research problem, and then describes how it evolved to shed light on fundamental aspects of integer programming and to provide a great variety of cutting planes for integer programming.

1. Introduction and background

1.1. Intent of this article

This article is a personal account of my experiences with Corner Polyhedra and some closely related integer programming problems. It will not be a survey of the related literature, a survey which, because of my intermittent connection with the subject, I really could not write in an informed and balanced way.[1]

The article starts by describing work on the practical problems of paper mills. The necessity of dealing with the very large size of these problems motivated the invention of what is now called column generation. Then the results obtained using these methods turned out to have a completely unexpected periodicity.

Explaining that periodicity led through various stages of understanding to the creation of the polyhedra that I named Corner Polyhedra. We will see how the Corner Polyhedra then took on a life of their own, giving insight into the structure of integer polyhedra and generating new families of cutting planes for general integer programming.

Throughout I will do my best to describe the surroundings and motivation that drove this evolution and to make the various steps as clear as possible and illustrate them by examples. I will also occasionally point to directions which seem to me to have unrealized possibilities.

1.2. Background: applied mathematics and operations research

In the 1950's Operations Research was a new and exciting part of Applied Mathematics. It was appealing to me because it promised to extend the reach of Mathematics beyond the traditional fields of Science and Engineering and closer to ordinary life. And that did happen.

But in addition, many times, Operations Research work motivated by practical needs, has also turned out to be mathematically beautiful.

Many believe that Applied Mathematics, and especially Operations Research, is mainly a routine use of mathematics. Many believe that operations researchers find a problem, apply to it some well understood piece of mathematics, and the answer comes out. That ends it, the problem is solved.

This certainly can happen, but often applied work is much more complicated than that. In applied work finding a way to formulate a problem mathematically can be difficult in itself. Then if you do succeed in finding a mathematical formulation, its sheer size and complexity may overwhelm standard approaches. You may have to split out tractable parts and leave the rest, or you may have to find a way to approximate, or you may have to invent.

Sometimes you may succeed in all this only to find that your hard won solution is met with hostility by those who might be affected by it, or alternatively, you may be fortunate and find that those affected by your work are surprisingly eager to adopt it for reasons quite unconnected with what you have done.

And, every now and then, you may turn up something unexpected, something you stumble across while pursuing something

* Tel.: +1 9142388522.
E-mail address: gomory@sloan.org

[1] This paper is based on my lecture "Forty Years of Corner Polyhedra" given on July 11, 2012 at EURO XXV. I want to thank the many researchers who have sent me their papers over the years, including the many years that I have spent away from integer programming, being engaged in other work. Trying to mention their work and place it in the proper context would be a very large and very worthwhile enterprise, but I do not attempt it in this paper.

http://dx.doi.org/10.1016/j.ejor.2016.03.001
0377-2217/

else. When I think of this sort of thing I always think of Christopher Columbus.

Although Columbus lived in a world very different from ours, he had some problems that resemble those we have today. In modern terms we would say that Columbus had major difficulties getting his project funded, and then, after his plan to reach China and India by sailing west was finally accepted, he failed to get there.

Columbus promised to find a new route to the Indies. But Columbus didn't find a new route to the Indies. Instead he discovered a whole new world. This too can happen.

1.3. Some modern explorers

About 370 years after the Queen of Spain sent Columbus off toward the Indies, a modern ruler, IBM, sent off another smaller group of explorers. Our group of explorers was chartered to see if the mathematical methods of Operations Research could find and conquer rich new territories for computers.

Among our explorers were Benoit Mandelbrot, Paul Gilmore, and T.C. Hu. Later we were joined by Philip Wolfe and Ellis Johnson.

Columbus' expedition embarked in three ships, the Nina, the Pinta, and the Santa Maria. Our smaller expedition also relied on three ships. Ours were named Linear, Dynamic, and Integer Programming. As it turned out, we needed all three for our voyage.

2. Origins of column generation

2.1. The stock cutting problem

Our little group within IBM's Research Division, was aware of the general stock cutting problem. This is the problem of starting with a stock of large pieces of some material and then cutting those large pieces into needed quantities of smaller sizes while creating as little waste as possible. We understood that a great variety of stock cutting problems could be formulated as linear or integer programming problems; so we wondered if there was something in this area that we could put into actual use.

As a first step we tried a problem we had heard about that involved cutting up big steel girders for bridges; but although we could formulate the problem mathematically, the data from real bridges gave us integer programming problems that were way beyond the scale that anyone at that time could handle.

2.2. The paper trim problem – linear programming

Next Paul Gilmore and I took a look at a different stock cutting problem, the paper trim problem, the problem of cutting the very wide rolls of paper that paper mills produce into the smaller width rolls that people actually use. We had heard that there were special aspects of the paper trim problem that might allow us to use ordinary linear programming instead of integer programming; if true, that would make the problem more tractable.

In the paper trim problem, as in the bridge problem, the actual numbers matter. So here is a description, based on our later experience, of a typical paper trim problem.[2]

Paper comes streaming out of an enormous paper machine in a paper mill and is rolled up on metal spindles. The paper the machine makes has a fixed large width W, which depends on the paper machine. A typical width W is 200 inches or more. The mill's customers want rolls of paper in a variety of much smaller sizes, thirty different customer widths w_j of 20 to 80 inches would be typical. So the mills are obliged to cut up the wide rolls they manufacture into the quantities b_j of these smaller rolls that the customers want.

To produce the right quantities of the smaller rolls, the mills have to cut up their wide rolls in many different ways. Each way to cut up one wide roll is called a cutting pattern. The ith cutting pattern is a list A_i that gives the number $a_{i,j}$ of rolls of a width w_j that the pattern produces.

It was well known that the paper trim problem could be formulated as an integer programming problem. In this standard formulation x_i is the (integer) number of times the ith cutting pattern is used, n is the number of different cutting patterns A_i that are available (this is often a very large number), m is the number of different customer widths w_j that are demanded, and the goal is to choose integers x_i ($i = 1,\ldots,n$) that fill the m customer orders b_j while minimizing the cost.

Cost is taken to be the number of wide rolls required to fill all the orders. So here is a standard formulation:

$$\text{Minimize } V = \sum_{i=1}^{i=n} x_i \quad \text{subject to} \tag{1}$$

$$\sum_{i=1}^{i=n} a_{i,j} x_i \geq b_j \quad \text{for} \quad j = 1, \ldots, m \tag{1A}$$

If we choose, we can add non-negative slack variables to (1A) to produce a formulation free of inequalities.

Now (1) and (1A) together are a very straightforward integer linear programming problem. The only question is: is it too large to handle?

What we had heard about the paper industry was that it was acceptable for the customer requirements, the b_j not to be exact requirements. It was acceptable to produce more than the amounts the customer asked for by a few percent, the customers were willing to take the extra rolls.

What that meant to us was this: if the *linear* programming solution to (1) and (1A) gave us some non-integer x_j, cutting patterns that were used a non-integer number of times, perhaps we could just round up those x_j and the customer would accept the extra rolls that were generated.

That sounded promising.

2.3. Size still a problem

However, even though it was now possible that we were dealing with an ordinary linear programming problem, our mathematical formulation still presented a difficulty: the enormous number of possible cutting patterns. With realistic paper industry problems, with for example 30 different customer widths to choose from, the number of cutting patterns, and therefore the number of columns in the linear programming problem, could easily run into many millions.

Now solving a linear program with millions of columns was well beyond what even the largest computers of that time could do, while the paper mills at that time were yet to purchase even a small computer.

So we had to think hard about how to get the problem down to something that could run on a small computer, and then hope that the cost savings our calculation might produce would cover its cost.

We thought hard about what the simplex method actually does in (1) and (1A) and eventually found an approach.

2.4. A starting solution

Suppose we start the simplex calculation of (1) and (1A) using an arbitrarily chosen set of cutting patterns A_i that we take as an

[2] Gilmore and Gomory (1963) has a detailed description of an actual problem as an appendix.