Department of Industrial and Systems Engineering
Spring 2005

Integer Programming

(IE 418)

Meeting: Monday and Wednesday 5:35–6:50PM 451 Mohler Lab

Jeff Linderoth
Office: 325 Mohler
Office hours: Monday 8–9AM, Wednesday 7–8PM, (also by appt.)
Phone: 610-758-4879
E-mail: jtl3@lehigh.edu
Web: http://www.lehigh.edu/~jtl3/teaching/ie418

This course will cover the theory and practice of integer programming. The course is structured into a number of modules that are designed to arm you with the understanding necessary to perform fundamental research in integer programming and also to model and solve integer programs that might arise in your research area.

Required Text

George Nemhauser and Laurence Wolsey. *Integer and Combinatorial Optimization* (John Wiley and Sons, 1988).

Recommended Texts

Alexander Schrijver. *Theory of Linear and Integer Programming* (John Wiley & Sons, 1986).

Laurence Wolsey. *Integer Programming* (Wiley-Interscience, 1998).

Overview

The aim of integer programming is to find optimal decisions in problems where the decisions may only take a certain number of finite values. The area is of enormous practical concern, since "yes-no" decisions are of a discrete nature. The list of application areas that use integer programming is too long to print here.

Integer programming is a mature field, with deep contributions having been made as far back as the early 1960's. Nevertheless, it is still a vibrant field, likely due to the fact that the ability to solve larger and larger integer programs is of enormous practical importance, and thus spurs development in both theory and computation.

This is a first course in integer programming, suitable for students with a graduate knowledge of linear programming and *mathematical sophistication*. Courses on the analysis of algorithms and on graphs and networks will be useful for background, but they are not required.

The course will use the text of Nemhauser and Wolsey to provide the theoretical underpinnings. This text is somewhat dated, so we will supplement the text with recent papers in the field of integer programming to show us the state-of-the-art.

Since advanced software has become such an integral component of integer programming, there will be a *significant* computational component to this course. Familiarity with a programming language such as C or C++ will be extremely helpful, as will knowledge of the Linux operating system, as many of our experiments will be run using the hardware and software resources in the COR@L lab (Room 362 Mohler), wherein the machines run the Linux operating system.

Objectives

- Be able to read, digest, and understand scholarly journal articles on integer programming.
- Become acquainted with "off the shelf" solvers for integer programs, learn what the algorithmic parameters mean, how to tune them for improved performance, and how to customize them for specialized algorithms.
- Understand how integer variables are used for formulating complex mathematical models, and in particular, what makes one valid model "better" than another.
- Know the basic concepts of complexity theory, to be able to understand the difference between an "easy" problem and a "hard" problem.
- Absorb basic polyhedral theory, and use this theory to describe properties of polyhedra arising from integer programs.
- Understand the theory of valid inequalities and the lifting of valid inequalities.
- Learn different classes of valid inequalities used in commercial solvers and also problem specific classes of valid inequalities.
- Understand decomposition-based (problem specific) techniques for large-scale problems.

Requirements and Grading

This course is (hopefully) *not* about getting a good grade. Instead, it should be about challenging yourself and learning about integer programming. It won't be easy! It sounds like a cop out from the teacher, (and maybe it is), but you will learn the most if you study many of these concepts yourself.

Academic Integrity

You are all graduate (or Ph.D.) students. You are all grown-ups. **Do not cheat**. If you have any question or concern about what constitutes cheating or improper collaboration, *please*

contact me. An excellent web site with lots of useful information about the integrity policy and procedures at Lehigh is `http://www.lehigh.edu/~indost/integrity.html`. If I suspect that you are cheating, you will make me sad. Then you will make me mad. Do not do this.

Grading Scheme

The course grade will be based on a weighted average of three components:

- 50% Problem Sets
- 20% Mid Term Exam
- 30% Final Exam

The mid-term exam will be an in-class exam, but the final exam will be a take-home exam. For the problem sets, in general, I will grade some of the problems in detail, and I will scan the solutions to other problems. The problems I grade in detail will be worth roughly twice as much as the problems that I don't grade. I plan on providing detailed solutions to all of the problems, and I plan on giving roughly six or seven problem sets during the course of the semester.

There is a penalty of 10% for each day that an assignment is late, without exception. Once I have graded and returned an assignment, that homework will no longer be accepted. The lowest score on a problem set will only count 50% towards a student's final problem set score. *Please don't whine and ask for extensions on the homeworks.* It's not very becoming.

Very (Tentative) Chronological Syllabus

You will see that there is a two week period in March (lectures on March 14, 16, 21, and 23) in which there is no lecture scheduled. I am traveling to workshops during that time. I have not (yet) arranged for guest lecturers. Stay tuned! I may ask that we extend the lecture time on certain days so that I can cover the requisite material. I am sure that you will all soon love integer programming *so much* that you will all love to stay extra time in order to learn more about it.

In addition to the readings in Nemhauser and Wolsey, I will assign journal articles for you to read that will (hopefully) supplement the text.

Part I: **Integer Programming Basics** *Jan. 19—Feb. 7*

Integer programming modeling concepts. Branch-and-bound for solving integer programs. An introduction to software for solving integer programs.

Readings: I.1, II.4.1, II.4.2, papers

Part II: **Theoretical Underpinnings** *Feb. 9—Feb 28*

Complexity Theory: *Basic complexity classes P, NP, and NP-Complete. The concept of problem reduction to determine complexity.*
Polyhedral Theory: *Concepts of dimension, faces, facets, polyhedral representations, and polarity. The equivalence of separation and optimization*

Readings: **I.5, I.4, I.6.3**

Midterm Exam: March 2

Part III: **Techniques in Branching-Based Algorithms** *Mar. 7—Apr. 11*

Preprocessing and probing in integer programs. Primal heuristics for integer programs. Advanced branching and node selection rules for integer programs. Techniques for obtaining valid inequalities, Gomory's cutting planes, mixed integer rounding, lifting, cover inequalities.

Readings: **II.1.1, II.1.3, II.1.6, II.2.1, II.2.3, II.4.1, II.4.2, II.5.1, II.5.2, II.6.2, II.6.3**

Part IV: **Advanced Topics in Integer Programming** *Apr. 13—Apr. 27*

Lagrangian Relaxation, Bender's Decomposition, Branch-and-Price, Primal and Lattice-based methods for integer programs.

Readings: **II.3.6, II.3.7, II.5.4, I.7, II.6.5, papers**