ISyE/Math/CS 728
Integer Optimization

Course Introduction

Prof. Jim Luedtke
University of Wisconsin-Madison

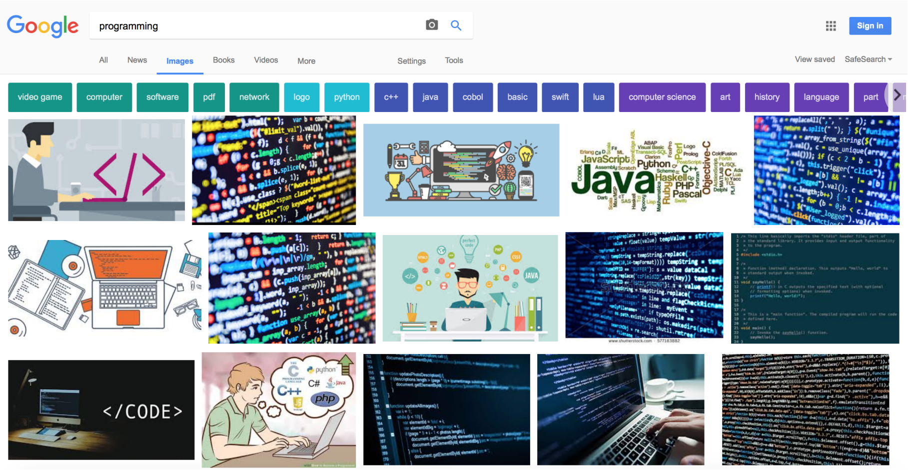
Significant use from the book *Integer Programming* by M. Conforti, G. Cornuéjols, and G. Zambelli

Integer Programming

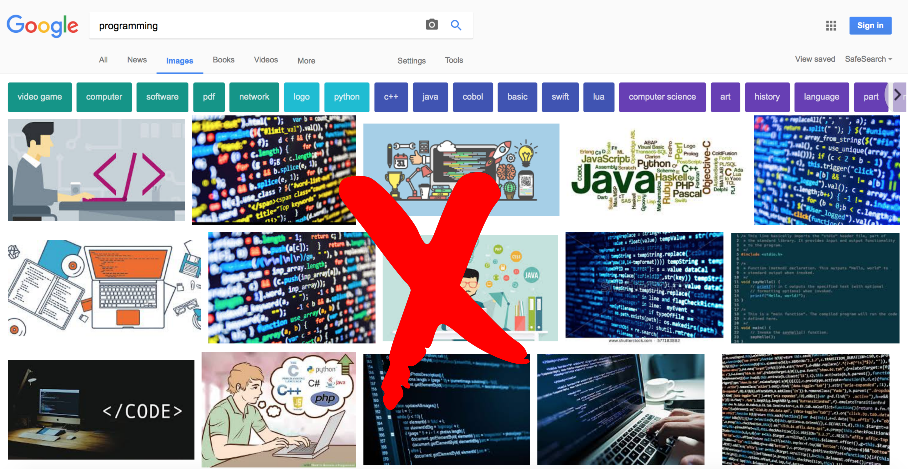
Integer ~~Programming~~ Optimization

~~Programming~~ Optimization

Programming



Programming



It is NOT about programming languages...

Optimization

It is about making decisions.

Optimization

It is about making OPTIMAL decisions.

Optimization

It is about making OPTIMAL decisions.

Lately “Programming” is getting replaced by “Optimization”.

What is Integer Optimization about?

- Integer Optimization is about decision making with linear functions and with integers.
- More precisely:
  - All the functions that we consider (objective and constraints) are linear functions.
  - Some of the decisions can take only certain integer values.

An Integer Optimization problem (a.k.a. integer program)

$$
\begin{array}{rl}
\text{minimize} & 2x_1 - x_2 + 4x_3 \\
\text{subject to} & x_1 + x_2 + x_4 \leq 2 \\
& 3x_2 - x_3 = 5 \\
& x_3 + x_4 \geq 3 \\
& x_1 \geq 0 \\
& x_3 \leq 0 \\
& x_1, x_2 \quad \text{integral (i.e., } x_1, x_2 \in \mathbb{Z}\text{).}
\end{array}
$$

Why is optimization with integers important?

First answer

Many decisions involve deciding a quantity that is indivisible:

- Number of airplanes to produce.
- Number of floors in a building.
- What about number of cents to invest in a stock?

Sometimes a continuous approximation is good enough.

Why is optimization with integers important?

Slam dunk answer

We can use $0, 1$ (binary) variables for a variety of purposes:

- Modeling yes/no decisions.
- Enforcing logical conditions.
- Modeling fixed costs.
- Modeling piecewise linear functions.

Usually a continuous approximation is not good enough.

The usefulness of binary variables

Suppose for a set of elements to choose from, $i \in S$, we have the binary variables:

$$x_i = \begin{cases} 1 & \text{if element } i \text{ chosen} \\ 0 & \text{otherwise.} \end{cases}$$

How can we model these restrictions with linear inequalities?

1. If we do choose i, we must choose j.

The usefulness of binary variables

Suppose for a set of elements to choose from, $i \in S$, we have the binary variables:

$$x_i = \begin{cases} 1 & \text{if element } i \text{ chosen} \\ 0 & \text{otherwise.} \end{cases}$$

How can we model these restrictions with linear inequalities?

1. If we do choose i, we must choose j.
2. We must either choose both i and j, or neither.

The usefulness of binary variables

Suppose for a set of elements to choose from, $i \in S$, we have the binary variables:

$$x_i = \begin{cases} 1 & \text{if element } i \text{ chosen} \\ 0 & \text{otherwise.} \end{cases}$$

How can we model these restrictions with linear inequalities?

1. If we do choose i, we must choose j.
2. We must either choose both i and j, or neither.
3. We can choose at most t items from a set S of items.

The usefulness of binary variables

Suppose for a set of elements to choose from, $i \in S$, we have the binary variables:

$$x_i = \begin{cases} 1 & \text{if element } i \text{ chosen} \\ 0 & \text{otherwise.} \end{cases}$$

How can we model these restrictions with linear inequalities?

1. If we do choose i, we must choose j.
2. We must either choose both i and j, or neither.
3. We can choose at most t items from a set S of items.
4. If we choose any one item in a set S, we must choose k ($k \notin S$).

The usefulness of binary variables

Suppose for a set of elements to choose from, $i \in S$, we have the binary variables:

$$x_i = \begin{cases} 1 & \text{if element } i \text{ chosen} \\ 0 & \text{otherwise.} \end{cases}$$

How can we model these restrictions with linear inequalities?

1. If we do choose i, we must choose j.
2. We must either choose both i and j, or neither.
3. We can choose at most t items from a set S of items.
4. If we choose any one item in a set S, we must choose k ($k \notin S$).
5. If we choose all items in a set S, we must choose k ($k \notin S$).

Example: The assignment problem

- There are n jobs to be performed by n workers.
- We know the cost c_{ij} of assigning job i to worker j for each pair i, j.
- What is the cheapest way of assigning one job to each of the n workers?
- This problem can be easily written as an Integer Optimization problem.

Assignment problem: Formulation

Decision variables:

- $x_{ij} = 1$ if job i is assigned to worker j, $= 0$ otherwise

Assignment problem: Formulation

Decision variables:

- $x_{ij} = 1$ if job i is assigned to worker j, $= 0$ otherwise

Objective: Minimize total cost of assignments

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

Assignment problem: Formulation

Decision variables:

- $x_{ij} = 1$ if job i is assigned to worker j, $= 0$ otherwise

Objective: Minimize total cost of assignments

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

Constraints: Each job assigned to one worker

$$\sum_{j=1}^{n} x_{ij} = 1, \quad \forall i = 1, \ldots, n$$

Assignment problem: Formulation

Decision variables:

- $x_{ij} = 1$ if job i is assigned to worker j, $= 0$ otherwise

Objective: Minimize total cost of assignments

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

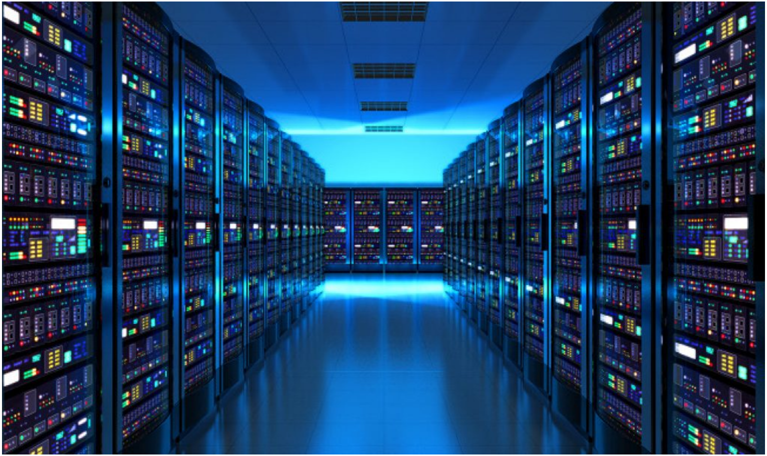
Constraints: Each job assigned to one worker

$$\sum_{j=1}^{n} x_{ij} = 1, \quad \forall i = 1, \dots, n$$

Constraints: Each worker is assigned one job

$$\sum_{i=1}^{n} x_{ij} = 1, \quad \forall j = 1, \dots, n$$

Can growing computing power help?



Can growing computing power help?

- Consider the assignment problem.
- Simple: check each possible assignment and select the best.

Can growing computing power help?

- Consider the assignment problem.
- Simple: check each possible assignment and select the best.
- But how long will it take? Let's use an IBM BlueGene/L:

Can growing computing power help?

- Consider the assignment problem.
- Simple: check each possible assignment and select the best.
- But how long will it take? Let's use an IBM BlueGene/L:

#*people*	Solutions to check	Time
3	6	0
10	3.6×10^6	0
15	1.3×10^{12}	1 sec
30		
70		

Can growing computing power help?

- Consider the assignment problem.
- Simple: check each possible assignment and select the best.
- But how long will it take? Let's use an IBM BlueGene/L:

#*people*	Solutions to check	Time
3	6	0
10	3.6×10^{6}	0
15	1.3×10^{12}	1 sec
30	2.7×10^{32}	
70		

Can growing computing power help?

- Consider the assignment problem.
- Simple: check each possible assignment and select the best.
- But how long will it take? Let's use an IBM BlueGene/L:

#*people*	Solutions to check	Time
3	6	0
10	3.6×10^6	0
15	1.3×10^{12}	1 sec
30	2.7×10^{32}	14 billion years[a]
70		

[a] That's four times the age of the universe!

Can growing computing power help?

- Consider the assignment problem.
- Simple: check each possible assignment and select the best.
- But how long will it take? Let's use an IBM BlueGene/L:

#*people*	Solutions to check	Time
3	6	0
10	3.6×10^6	0
15	1.3×10^{12}	1 sec
30	2.7×10^{32}	14 billion years[a]
70	1.2×10^{100} [b]	

[a] That's four times the age of the universe!
[b] That's more than the number of particles in the observable universe!

So how hard is Integer Optimization?

- We will see that this particular integer program can be solved very efficiently (by solving a linear program).

So how hard is Integer Optimization?

- We will see that this particular integer program can be solved very efficiently (by solving a linear program).
- Could we be more clever and solve all integer programs efficiently? (After all, we can solve linear programs...)

So how hard is Integer Optimization?

- We will see that this particular integer program can be solved very efficiently (by solving a linear program).
- Could we be more clever and solve all integer programs efficiently? (After all, we can solve linear programs...)
- Solving general integer programs can be much more difficult than solving linear programs.
  - There is a whole theory (complexity) that supports this claim.
  - You will learn a tiny bit of about this.

Integer optimization is everywhere!

The knapsack problem

- You are choosing what to bring in your backpack.
- You can carry at most b pounds.
- You have n possible items.
- Item i would give benefit of value c_i and weight a_i.
- What items to pack?

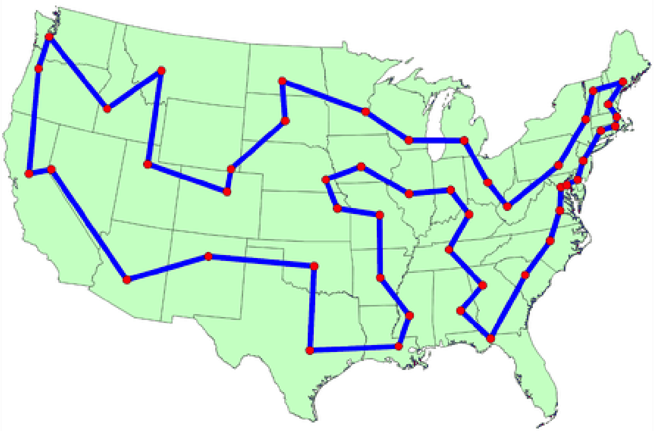
The knapsack problem

$$
\begin{aligned}
\max \quad & \sum_{i=1}^{n} c_i x_i \\
\text{s. t.} \quad & \sum_{i=1}^{n} a_i x_i \leq b \\
& x \in \{0,1\}^n.
\end{aligned}
$$

- You are choosing what to bring in your backpack.
- You can carry at most b pounds.
- You have n possible items.
- Item i would give benefit of value c_i and weight a_i.
- What items to pack?

The traveling salesman problem (TSP)

- Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city and returns to the origin city?



More examples

- Design a supply chain network: decide where to open facilities and which customers they should serve.

More examples

- Design a supply chain network: decide where to open facilities and which customers they should serve.
- Portfolio optimization: limit number of stocks held.

More examples

- Design a supply chain network: decide where to open facilities and which customers they should serve.
- Portfolio optimization: limit number of stocks held.
- Stochastic optimization with risk: ensure percent of outcomes which are "bad" is small.

More examples

- Design a supply chain network: decide where to open facilities and which customers they should serve.
- Portfolio optimization: limit number of stocks held.
- Stochastic optimization with risk: ensure percent of outcomes which are “bad” is small.
- LTL vehicle routing: which loads should be combined on trucks and in what sequence should the loads be delivered?

More examples

- Design a supply chain network: decide where to open facilities and which customers they should serve.
- Portfolio optimization: limit number of stocks held.
- Stochastic optimization with risk: ensure percent of outcomes which are "bad" is small.
- LTL vehicle routing: which loads should be combined on trucks and in what sequence should the loads be delivered?
- Sports scheduling: assign games to days.

More examples

- Design a supply chain network: decide where to open facilities and which customers they should serve.
- Portfolio optimization: limit number of stocks held.
- Stochastic optimization with risk: ensure percent of outcomes which are “bad” is small.
- LTL vehicle routing: which loads should be combined on trucks and in what sequence should the loads be delivered?
- Sports scheduling: assign games to days.
- Chemical processing network design: which process types should be used and how should they be connected?

More examples

- Design a supply chain network: decide where to open facilities and which customers they should serve.
- Portfolio optimization: limit number of stocks held.
- Stochastic optimization with risk: ensure percent of outcomes which are "bad" is small.
- LTL vehicle routing: which loads should be combined on trucks and in what sequence should the loads be delivered?
- Sports scheduling: assign games to days.
- Chemical processing network design: which process types should be used and how should they be connected?
- Machine learning: sparse linear regression, Bayesian network structure learning, sparse PCA

Couse details

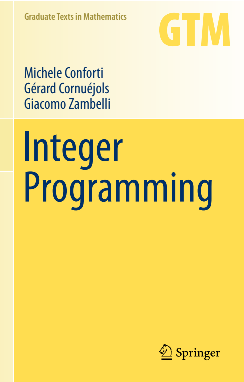
Purpose of this course

- Present the mathematical foundations of Integer Optimization.
- Understand cases that can be solved efficiently.
- Emphasis on the techniques that are most successful in current software implementations: convexification and enumeration.

What this course is NOT about:

- We won't spend much time on how to model problems as Integer Optimization problems. (See ISyE 524)
  - We study what makes a model "good".
  - We will see some advanced modeling techniques.
- We won't cover combinatorial algorithms (see ISyE 425) or specialized algorithms for "easy" integer optimization problems (see CS 577).

Recommended textbook



Integer Programming,
by Michele Conforti, Gérard Cornuéjols, and Giacomo Zambelli (2014).

- Winner of the 2015 INFORMS Lanchester Prize for best contribution to Operations Research in the past 3 years.
- I will try to use consistent notation, and provide references to related sections of this book
- Some parts of course will not closely follow the book

What we will cover (subject to change)

- Algorithmic frameworks: Branch & bound and cutting planes
- Modeling techniques, and comparing models
- Integer programming software
- Perfect formulations
- Decomposition methods
- Polyhedral theory and strong valid inequalities
- General purpose valid inequalities

Warning on course difficulty

This is a Ph.D.-level course $\Rightarrow$ Much of the material in this course is deep.

- Requires serious math.
- Proofs will be done by me and by you.

Warning on course difficulty

- “Hard but rewarding”

Prerequisites

Essential background that will be assumed:

- Working knowledge of linear algebra.
  See Section 1.5 of the 525 textbook: "Introduction to Linear Optimization" by D. Bertsimas, and J.N. Tsitsiklis. (6 pages)
  - E.g., set theoretic notation, vectors and matrices, matrix inversion, subspaces and bases, affine subspaces, linear independence, and the rank of a matrix.
- Knowledge of Linear Optimization.
  See 525 textbook.
  - E.g., geometry of linear programming, the simplex method, duality theory, network flow problems, complexity of linear programming, and the ellipsoid method.
- Basic proof techniques.
  See Introduction to mathematical arguments in Canvas.
- Mathematical notation.
  See Math symbols cheat sheet in Canvas.

Other Optimization courses

- ISyE/CS 719: Stochastic Programming.
- ISyE/CS 723: Dynamic Programming and Associated Topics.
- ISyE/Math/CS/Stat 726: Nonlinear Optimization I.
- ISyE/CS 727: Convex Analysis.
- ISyE/CS 730: Nonlinear Optimization II.
- ISyE/Math/CS 728: Integer Optimization.
- CS 733: Computational Methods for Large Sparse Systems.
- ISyE 823: Special Topics in Operations Research.

If interested in research in Optimization, Madison is a great place to be :)

Technical details

Class Overview

- Office Hours:
  - Tuesday and Thursday: 10:45–11:45am, right after lecture in same zoom room
- Course home page in Canvas:
  - https://canvas.wisc.edu/courses/230933
  - Syllabus, lecture slides, introductory material, homework assignments and solutions, grades.
  - Use Q&A forum in Piazza module to post questions about the course.
- See syllabus in Canvas for more info

Expectations

I am expected to...

- Teach lectures.
- Be at my office hours.
- Guide your learning process (homework).
- Give you feedback on how you are doing in a timely fashion.

Expectations

I am expected to...

- Teach lectures.
- Be at my office hours.
- Guide your learning process (homework).
- Give you feedback on how you are doing in a timely fashion.

You are expected to...

- Learn.
- Attend lectures and participate. (Ask questions!)
- Do the homework.
- Know and follow academic conduct guidelines.
- Follow online lecture etiquette (e.g., stay muted unless called on, feel free to use chat)

Homework

- There will be approximately one assignment every 2 weeks.
- Strongly encouraged to work in groups of 2 people.
- Submit pdf in Canvas (one submission per group).
- Assignments may not be completely graded.
- Complete solutions will be published in Canvas.
- TA Rui Chen (rchen234@wisc.edu) will deal with assignments and related questions.
- TA office hours:
  - Monday 1-2pm, Wednesday 2-3pm, same zoom link.

$*$ Notify TA this week if you have a conflict with both his office hours.

Grading

Components of grade:

- Homework (25%)
- Midterm (35%) : **March 14, 1pm – March 16, 1pm.**
- Option: Final exam or course project (35%) : **May 2, 1pm – May 4 1pm.**
- Class participation (5%)

Exams will be take-home and may contain a computational implementation component

Optional Project

Students have the option to do a final project in place of final exam

- **Not** allowed to do both!

Projects can/should be done with a partner

- But different partner than homework!

If you wish to do a project, must submit proposed project topic by April 9 at the latest.

- Suggested to submit earlier for approval so you can start earlier

About me...

- B.S., ISyE, UW-Madison, 2001.
- M.S., OR, GA Tech, 2003.
- Ph.D., ISyE, GA Tech, 2007.
- Fall 2007-Summer 2008: IBM Research
- Research Areas: Integer programming (linear and nonlinear), stochastic programming, applications
- Married. Three children, Rowan (13), Cameron (11), Remy (7). One dog, Bowie.
- Interesting: Did cross-country bicycle tour after college
- Hobbies: Mountain biking, board games, kids

About you...

Please complete the Student Survey quiz in Canvas by Jan 31!

Questions about the course?