CIS1921

Lecture 2: Solvers & Encoding

Reminders

- Homework 0 due Monday, January 27, 11:59PM
- OH Schedule: *Thomas:* Sun 3-4pm, *Cindy*: Tues 8-9pm, *Ishaan:* Wed 9:30-10:30pm
- OH held virtually on OHQ – be sure to add the class!
- Homework 1 will be posted this weekend. 2 weeks to complete!
- Make sure you have access to Ed
- Gradescope: **8KPRG5**

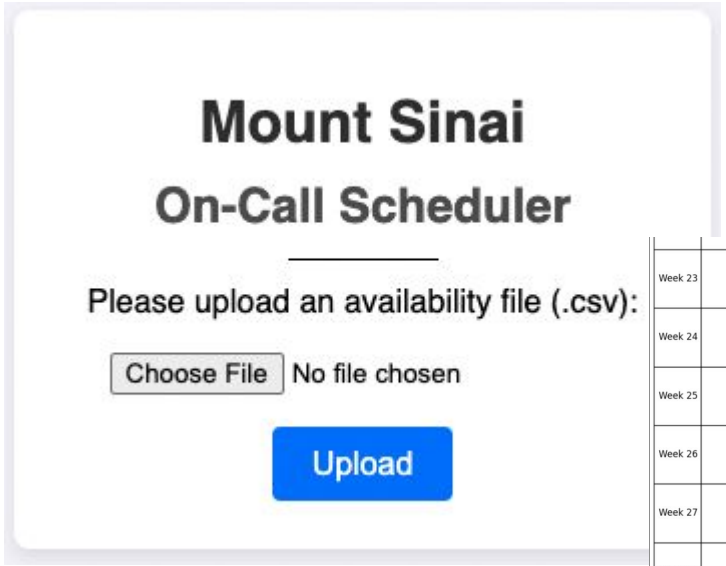
Accommodations

- If you have any special accommodations (i.e. things that would prevent you from submitting a HW at a certain time), please let me know ASAP.

A couple past final projects...

On-Call Scheduler for Hospitals

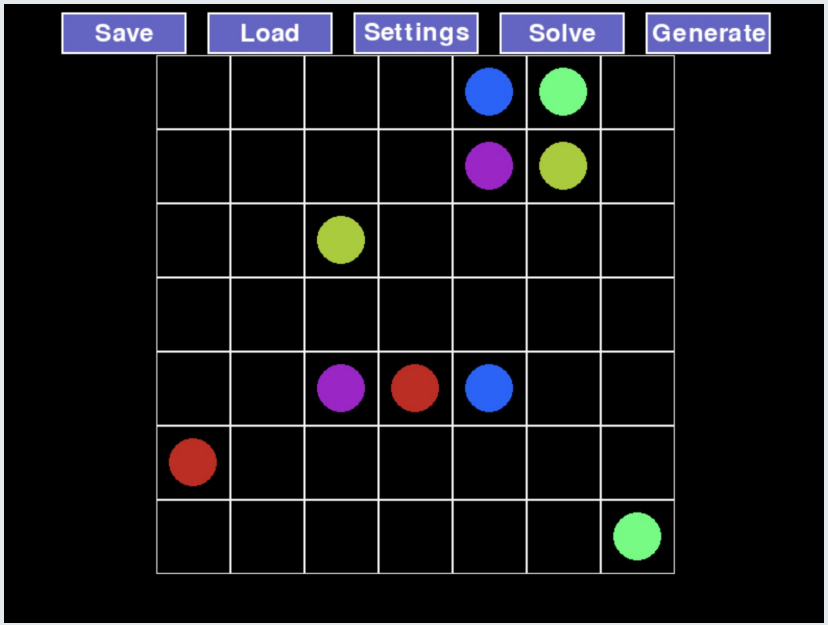
Christopher Wun & Alex Jiang

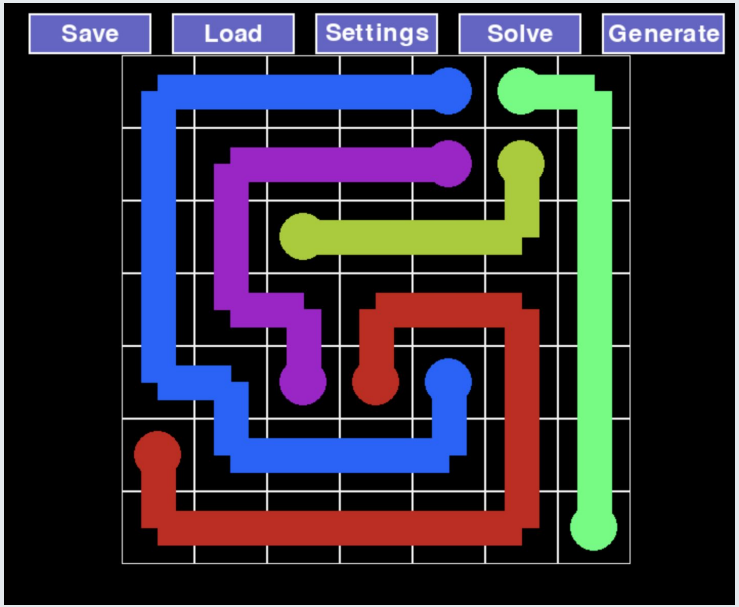


	Uemura						
Week 23	Andrade Bharani Javier Sheehan Wey	Hansen Uemura	Wagner Mark	Mulholland Arabelo	Yamada	Hossain	Kamal
Week 24	Andrade Bharani Biewald Chai Sheehan	Sanon Uemura	Wagner Mark	Mulholland Arabelo	Hansen	Hossain	Kamal
Week 25	Bharani Choi Sheehan Uemura Wagner	Javier Sanon	Pelleg Mark	Andrade Mehta	Masutani	Amir	Kamal
Week 26	Choi Gelfman Sheehan Uemura Wagner	Afezolli Sanon	Pelleg Mark	Andrade Mehta	Masutani	Hossain	Kamal
Week 27	Bharani Chen Fung Mehta Wagner	Apoeso Javier	Choi Mark	Belland Arabelo	Yamada	Hossain	TBA
Week 28	Bharani Chen Mehta Pelleg Wagner	Apoeso Javier	Choi Mark	Rousseau Arabelo	Yamada	Amir	TBA
Week 29	Apoeso Crooms Mehta Pelleg Popp	Hansen Sanon	Sheehan Mark	Rousseau Arabelo	Yamada	Hossain	Kamal
	Apoeso						

Flow Solver / Generator

Michael Gao & Thomas Ngulube

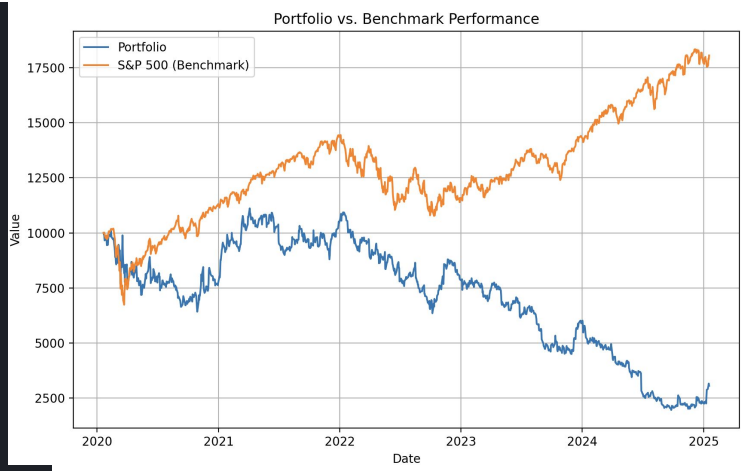




Optimized Portfolio Generator

Madhav Sharma & Ishaan Shah

```
Welcome to the Optimal Portfolio Generator!
Enter your investment preferences.
Investment Amount: $10000
Preferred Portfolio Beta Range (e.g., 0.5 1.0): 0.4 1.1
Maximum number of stocks to invest in: 8

Available Sectors:
1. Industrials
2. Health Care
3. Information Technology
4. Utilities
5. Financials
6. Materials
7. Consumer Discretionary
8. Real Estate
9. Communication Services
10. Consumer Staples
11. Energy
Enter sectors and their percentage limits as comma-separated pairs (e.g., '1,30 2,50'). Leave blank for no limits: 1,30 3,40 8,30
Optimizing portfolio...

Sector Average P/E Ratios:
  Communication Services: 16.84
  Consumer Discretionary: 18.62
  Consumer Staples: 17.95
  Energy: 15.23
  Financials: 17.56
  Health Care: 17.83
  Industrials: 23.65
  Information Technology: 21.52
  Materials: 21.12
  Real Estate: 53.10
  Utilities: 18.10

Optimal Portfolio:
Symbol: WBA, Sector: Consumer Staples, Price: $8.66, Allocation: $10000.00, Beta: 0.67, P/E: 5.48 (Undervalued), Dividend Yield: 11.55%

Total Portfolio Beta: 0.67
Total Portfolio Allocation: $10000.00

P/E Optimization Insights:
  Total Undervalued Stocks Selected: 1
  Total Stocks Selected: 1
  The portfolio prioritizes stocks with P/E ratios below sector averages, indicating a focus on undervaluation.
```



Recap

Last week

- Intro to hard problems
- The SAT problem

This week

- Using SAT solvers in Python
- Factors that affect solver runtimes

Overarching Class Themes

- Accept the fact that the problems we will look at are very hard and "exponential runtime"
  - Take solace in the fact that for many inputs, the problem won't take exponential time
- Every speed-up counts
  - *Take careful consideration of the balance between runtime and complexity*
- There will never be a "right answer"
  - Often, the best thing to do for a problem depends on the problem itself and its data!

Recall: SAT Problem

Given a formula φ of boolean variables, does there exist a truth assignment that makes the entire formula evaluate to True?

- Ex: $(x \vee y) \Rightarrow y$ is satisfiable with $\{x = T; y = T\}$
- Ex: $(x \wedge \overline{x})$ is unsatisfiable

SAT terminology

- Assume only logical symbols are AND, OR, NOT
- **Literal:** a boolean variable (x) or its negation ($\overline{x}$)
  - (x) is called a **positive** literal, and ($\overline{x}$) is a **negative** literal
  - "a variable as it appears in a formula"
- **Clause:** a disjunction/OR of literals
  - e.g. $(\overline{x} \vee y \vee z)$

- Note: we would say that $(\overline{x} \vee y) \wedge (x \vee y)$ has 2 variables and 4 literals

Conjunctive Normal Form

- A boolean formula is in **conjunctive normal form (CNF)** if it is a conjunction/AND of clauses (i.e., an AND of ORs)
  - "a CNF" means "a formula in CNF"
- Ex: which of the following are in CNF?
  - $(\overline{x} \vee y \vee z) \wedge (x \Rightarrow w)$
  - $(\overline{x} \wedge y \wedge z) \vee (\overline{y} \wedge z)$
  - $(\overline{x} \vee y \vee z) \wedge (\overline{y} \vee z)$
  - $\overline{x} \vee y \vee z$
  - $x \wedge \overline{x}$

CNF-SAT: a loss of generality?

- It's convenient for SAT solvers to accept formulas in CNF, but what if we need to solve any other non-CNF boolean formula?
- **Every SAT problem can be converted to a CNF-SAT problem**

DeMorgan's & Distributive Laws

Important Logical Properties

$$\neg(p \vee q) \equiv \neg p \wedge \neg q \qquad \text{(DeMorgan's Law)}$$

$$\neg(p \wedge q) \equiv \neg p \vee \neg q \qquad \text{(DeMorgan's Law)}$$

$$(p \vee q) \vee r \equiv p \vee q \vee r \qquad \text{(Associativity of } \vee\text{)}$$

$$(p \wedge q) \wedge r \equiv p \wedge q \wedge r \qquad \text{(Associativity of } \wedge\text{)}$$

$$(p \wedge q) \vee r \equiv (p \vee r) \wedge (q \vee r) \qquad \text{(Distributive property of } \vee\text{)}$$

$$(p \vee q) \wedge r \equiv (p \wedge r) \vee (q \wedge r) \qquad \text{(Distributive property of } \wedge\text{)}$$

CNF-SAT: a loss of generality?

- **Issue:** How large is the resulting CNF formula?
- **Ex:** $(x_1 \wedge x_2) \vee (x_3 \wedge x_4)$

  $((x_1 \wedge x_2) \vee x_3) \wedge ((x_1 \wedge x_2) \vee x_4)$

  $((x_1 \vee x_3) \wedge (x_2 \vee x_3)) \wedge ((x_1 \vee x_4) \wedge (x_2 \vee x_4))$
  - In general, the CNF of $(x_1 \wedge x_2) \vee (x_3 \wedge x_4) \vee \cdots \vee (x_{2n-1} \wedge x_{2n})$ has 2^n clauses
- This **exponential blowup** will make solving arbitrary non-CNF formulas very difficult... can we do better?

The Tseitin Transformation

- Two boolean formulas are **equisatisfiable** if they are either both satisfiable or both unsatisfiable
  - No small *equivalent* CNF, but we only need to find a small *equisatisfiable* CNF
- For each subformula $\psi = \psi_1 \circ \psi_2$, introduce a new variable x_ψ
  - Here, "subformula" includes the formula φ itself, but excludes all literals
  - The operator $\circ$ represents a boolean connective; i.e., $\wedge$ or $\vee$
- Conjoin (AND) together x_φ with $(x_\psi \Leftrightarrow x_{\psi_1} \circ x_{\psi_2})$ for each ψ
- Convert $(x_\psi \Leftrightarrow x_{\psi_1} \circ x_{\psi_2})$ into an *equivalent* CNF
  - Helpful fact: $(x \Rightarrow y)$ is equivalent to $(\bar{x} \vee y)$

The Tseitin Transformation enjoys linear blowup of the # of clauses

Notation

- Positive (x) and negative ($\overline{x}$) literals
- Recall that we often consider a clause as a **set of literals**, and a CNF as a **set of clauses**
- Therefore might express CNFs in **compact notation**:

$$\{x\overline{z}, yz\overline{x}\} \quad \simeq \quad (x \vee \overline{z}) \wedge (y \vee z \vee \overline{x})$$

MiniSAT

- **MiniSat** is a minimal, open source SAT solver created by Niklas Eén & Niklas Sörensson
  - *An Extensible SAT-solver* (Eén & Sörensson, 2003)
- Silver medal in SAT Comp. 2005…
- …with only ~600 lines of C++ code!

Niklas Eén

Niklas Sörensson

PicoSAT

- **PicoSAT** is an open source SAT solver created by Prof. Armin Biere
  - *(Q)CompSAT and (Q)PicoSAT at the SAT'06 Race (Biere 2006)*
- Armin Biere: *Handbook of Satisfiability*
- Gold medalist in SAT Competition 2007
- Written in C, but bindings to Python, JS, R, etc.
  - We'll use this: can be easily installed with pip!

Armin Biere

Quick demo

DEMO: Solving a CNF with PicoSAT

Let's solve the following formula:

$$\varphi = \left\{5, \overline{4}1, \overline{4}2, 4\overline{12}, \overline{5}43, 5\overline{4}, 5\overline{3}\right\}$$

```
cnf = [[5],[-4,1],[-4,2],[4,-1,-2],[-5,4,3],[5,-4],[5,-3]]
pycosat.solve(cnf) # That was easy!



[1, 2, -3, 4, 5]
```

SAT Encodings

Why do we care about SAT?

SAT Encodings

Why do we care about SAT?

To answer this question, we need to answer a different question first...

How do radio broadcasts work?

SAT Encodings

Why do we care about SAT?

Because we can encode most problems as an instance of CNF-SAT. Let's see how!

Warmup: Boolean Encodings

$x \Rightarrow y$	$\{\overline{x}y\}$
$x \Leftrightarrow y$	$\{\overline{x}y, \overline{y}x\}$
$x \oplus y$	$\{xy, \overline{xy}\}$
if x then y else z	$\{\overline{x}y, xz\}$

Think of SAT more generally…

Let's say I'm in charge of creating a new building for Penn…

Think of SAT more generally…

Let's say I'm in charge of creating a new building for Penn…

And I ask a bunch of people to write down their "wish list" for the building. We want to make sure everyone has at least one wish granted.

Think of SAT more generally...

We have a **problem** (how to build the building) subject to **constraints** (satisfying the people)

Think of SAT more generally…

We transform a problem into an instance of SAT if we can write our constraints as a SAT formula

Think of SAT more generally...

We transform a problem into an instance of SAT if we can write our constraints as a SAT formula

Think of SAT more generally…

clause!

We transform a problem into an instance of SAT if we can write our constraints as a SAT formula

Think of SAT more generally…

We transform a problem into an instance of SAT if we can write our constraints as a SAT formula

First-Order Logic Encodings

$All(x_1, \ldots, x_n)$	$\{x_1, x_2, \ldots, x_n\}$
$ALO(x_1, \ldots, x_n)$	$\{x_1 x_2 \cdots x_n\}$
$AMO(x_1, \ldots, x_n)$	$\{\overline{x}_i \overline{x}_j \mid 0 \leq i < j \leq n\}$
$ExactlyOne(x_1, \ldots, x_n)$	$ALO(\ldots) \wedge AMO(\ldots)$

Complexity of Encoding

$AMO(x_1, \ldots, x_n)$ $\quad\quad$ $\{\overline{x}_i \overline{x}_j \mid 0 \leq i < j \leq n\}$

- Called the binomial encoding
- How many clauses in this encoding?
- Other encodings that use fewer clauses, but introduce extra variables

Binary AMO Encoding

$AMO(x_1, \ldots, x_n)$

- Retain original variables $x_1, \ldots, x_n$
- Let $m = \lceil \lg n \rceil$, and introduce new variables $b_1, \ldots, b_m$
  - b_j represents j^{th} bit of T, where x_T is the ($\leq$) one True var
- Clauses:

$$\left\{ \left(x_i \Rightarrow \begin{cases} b_j & \text{if } j^{th} \text{ bit of } i \text{ is } 1 \\ \overline{b_j} & \text{if } j^{th} \text{ bit of } i \text{ is } 0 \end{cases} \right) \middle| \, \forall x_i, b_j \right\}$$

- How many extra variables? How many clauses?

Good Encoding Matters

- The performance of a SAT solver can vary **hugely** depending on the encoding.
- Different encodings work better in different cases: **no "universal best encoding."**
- **Start simple** and be more clever if necessary.

Encoding Graph Coloring

Can we color $G = (V, E)$ with $\leq k$ colors?

Encoding Graph Coloring

Can we color $G = (V, E)$ with $\leq k$ colors?

- x_{ic}: if we assign v_i color c
- **Constraint 1:** every vertex has ≥ 1 color

$$\{x_{i1}x_{i2} \dots x_{ik} \mid \forall\, v_i \in V\}$$

Encoding Graph Coloring

Can we color $G = (V, E)$ with $\leq k$ colors?

- x_{ic}: if we assign v_i color c
- **C2:** if $(u, v) \in E$, then u, v have different colors

$$\left\{\overline{x_{ic}x_{jc}} \;\middle|\; \forall\, (v_i, v_j) \in E, c \in [1..k]\right\}$$

DEMO Part 1

From Coloring to Min-Coloring

- What if we wanted to find the $\chi(G)$, the minimum number of colors to color G?
- Binary search
  - Check if G can be colored with 1, 2, 4, 8, 16, … colors
  - Stop at smallest value 2^k such that G is 2^k-colorable
  - Binary search for $\chi(G)$ in range $[2^{k-1},\ 2^k]$
- Requires $O(\lg \chi(G))$ runs of solver

From Coloring to No-Coloring

- UNSAT: Given an integer k, is it that there is NO VALID k-coloring?
  - UNSAT can be **way** harder than SAT.
  - Finding just one satisfying assignment (SAT) vs. showing that none exists (massive search space UNSAT)

But radios?

Next Week

- Start learning how SAT solvers work
- Homework 1 due in two weeks
  - Encoding Sudoku into SAT
  - Extra credit challenge!

Stay Healthy

"For a thinking person, the most serious mental illness is not being sure of who you are."
~Benoit Mandelbrot

References

A. Biere, *Handbook of satisfiability*. Amsterdam: IOS Press, 2009.

J. Burkardt, "CNF Files," *John Burkardt's Home Page*. [Online]. Available: https://people.sc.fsu.edu/~jburkardt/data/cnf/cnf.html.