REVIEW

David Kauchak
CS 140 – Fall 2022

1

Admin

All assignments graded and returned

Assignment 11 due Wednesday!

Dr. Dave: normal mentor hours through 12/12

Last mentor session: Millie, Wednesday 10am-12
(I'm still trying to add some over the weekend)

2

Admin

Final
- posted on Gradescope on Monday morning
- due Tuesday at 11:59pm
- time-limited (3 hours)
- You may use:
  - the book
  - your notes
  - the class notes
  - ONLY these things
- Do NOT discuss it with anyone until after Tuesday at 11:59pm

3

Test taking advice

- Read the questions carefully!
- Don't spend too much time on any problem
  - if you get stuck, move on and come back
- When you finish answering a question, reread the question and make sure that you answered everything the question asked
- Think about how you might be able to reuse an existing algorithm/approach
- Show your work (I can't give you partial credit if I can't figure out what went wrong)
- Don't rely on the book/notes for conceptual things
  - Do rely on the notes for a run-time you may not remember, etc.

4

High-level approaches

Algorithm tools
- Divide and conquer
  - assume that we have a solver, but that can only solve sub-problems
  - define the current problem with respect to smaller problems
  - Key: sub-problems should be non-overlapping
- Dynamic programming
  - Same as above
  - Key difference: sub-problems are overlapping
  - Once you have this recursive relationship:
    - figure out the data structure to store sub-problem solutions
    - work from bottom up (or memoize)

5

High-level approaches

Algorithm tools cont.
- Greedy
  - Same idea: most greedy problems can be solve using dynamic programming (but generally slower)
  - Key difference: Can decide between overlapping sub-problems without having to calculate them (i.e. we can make a local decision)
- Flow
  - Matching problems
  - Numerical maximization/minimization problems

6

Data structures

A data structure
- Stores data
- Supports access to/questions about data efficiently
  - the different bias towards different actions
- No single best data structure

Fast access/lookup?
- If keys are sequential: array
- If keys are non-sequential or non-numerical: hashtable
- Guaranteed run-time/ordered: balanced binary search tree

7

Data structures

Min/max?
- heap

Fast insert/delete at positions?
- linked list

Others
- stacks/queues
- extensible data structures
- disjoint sets

8

Graphs

Graph types
- directed/undirected
- weighted/unweighted
- trees, DAGs
- cyclic
- connected

Algorithms
- connectedness
- contains a cycle
- traversal
  - dfs
  - bfs

9

Graphs

Graph algorithms cont.
- minimum spanning trees
- shortest paths
  - single source
  - all pairs
- topological sort
- flow

10

Other topics…

Analysis tools
- recurrences
- big-O

NP-completeness
- proving NP-completeness
- reductions

11

NP Terminology

P: set of problems that can be solved in polynomial time

NP: set of problems that can be verified in polynomial time (i.e., given a problem instance and a solution, verify that it is a solution)

NP-Hard: A problem is NP-Hard if any other NP-Hard problem can be reduced to the problem in polynomial time

NP-Complete: A problem is NP-Complete if it is *both* in NP and NP-Hard.

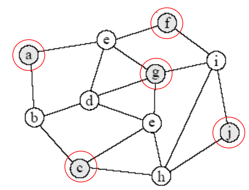
12

Reduction direction

13

Independent-Set revisited

Given a graph G = (V, E) is there a subset V'⊆ V of vertices of size |V '| = k that are independent, i.e. for any pair of vertices u, v ∈ V' there exists no edge between any of these vertices

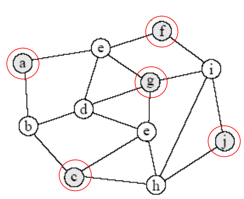


Is Independent-Set NP-Complete?

19

Independent-Set revisited

Given a graph G = (V, E) is there a subset V'⊆ V of vertices of size |V '| = k that are independent, i.e. for any pair of vertices u, v ∈ V' there exists no edge between any of these vertices



Reduce 3-SAT to Independent-Set

20

3-SAT $\leq_p$ Independent-Set

Given a 3-CNF formula, convert it into a graph

$$(a \vee \neg a \vee \neg b) \wedge (c \vee b \vee d) \wedge (\neg a \vee \neg c \vee \neg d)$$

For the boolean formula in 3-SAT to be satisfied, at least one of the literals in each clause must be true

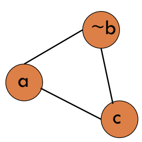
In addition, we must make sure that we enforce a literal and its complement must not both be true.

21

3-SAT $\leq_p$ Independent-Set

Given a 3-CNF formula, convert into a graph

For each clause, e.g. (a OR ~b OR c) create a clique containing vertices representing these literals



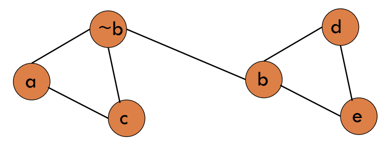
- for the Independent-Set problem to be satisfied it can only select one variable
- to make sure that all clauses are satisfied, we set k = number of clauses

22

3-SAT $\leq_p$ Independent-Set

Given a 3-CNF formula, convert into a graph

To enforce that only one variable and its complement can be set we connect each vertex representing x to each vertex representing its complement ~x



23

Proof

"yes" for 3-SAT -> "yes" for INDEPENDENT-SET

Given a 3-SAT problem with k clauses and a valid truth assignment, show that f(3-SAT) has an independent set of size k. (Assume you know the solution to the 3-SAT problem and show how to get the solution to the independent set problem)

Since each clause is an OR of variables, at least one of the three must be true for the entire formula to be true. Therefore each 3-clique in the graph will have at least on node that can be selected.

24

Proof

"yes" for INDEPENDENT-SET -> "yes" 3-SAT

Given a graph with an independent set S of k vertices, show there exists a truth assignment satisfying the boolean formula

- For any variable x_i, S cannot contain both x_i and $\neg x_i$ since they are connected by an edge
- For each vertex in S, we assign it a true value and all others false. Since S has only k vertices, it must have one vertex per clause

25

Master Method

Provides solutions to the recurrences of the form:

$$T(n) = aT(n/b) + f(n)$$

$$\text{if } f(n) = O(n^{\log_b a-\varepsilon}) \text{ for } \varepsilon > 0, \text{ then } T(n) = \Theta(n^{\log_b a})$$
$$\text{if } f(n) = \Theta(n^{\log_b a}), \text{ then } T(n) = \Theta(n^{\log_b a}\log n)$$
$$\text{if } f(n) = \Omega(n^{\log_b a+\varepsilon}) \text{ for } \varepsilon > 0 \text{ and } af(n/b) \le cf(n) \text{ for } c < 1$$
$$\text{then } T(n) = \Theta(f(n))$$

26

Recurrences

$$T(n) = 2T(n/3) + d \qquad T(n) = 7T(n/7) + n$$

$$\text{if } f(n) = O(n^{\log_b a-\varepsilon}) \text{ for } \varepsilon > 0, \text{ then } T(n) = \Theta(n^{\log_b a})$$
$$\text{if } f(n) = \Theta(n^{\log_b a}), \text{ then } T(n) = \Theta(n^{\log_b a}\log n)$$
$$\text{if } f(n) = \Omega(n^{\log_b a+\varepsilon}) \text{ for } \varepsilon > 0 \text{ and } af(n/b) \le cf(n) \text{ for } c < 1$$
$$\text{then } T(n) = \Theta(f(n))$$

$$T(n) = T(n-1) + \log n \qquad T(n) = 8T(n/2) + n^3$$

27

Big O: Upper bound

$O(g(n))$ is the set of functions:

$$O(g(n)) = \left\{ f(n) : \begin{array}{l}\text{there exists positive constants } c \text{ and } n_0 \text{ such that} \\ 0 \le f(n) \le cg(n) \text{ for all } n \ge n_0\end{array} \right\}$$

28

Proving bounds: find constants that satisfy inequalities

Show that $5n^2 - 15n + 100$ is $O(n^2)$

Find constants c and n_0 such that
$5n^2 - 15n + 100 \le cn^2$ for all $n > n_0$

$$cn^2 \ \ge \ 5n^2 - 15n + 100$$
$$c \ \ge \ 5 - 15/n + 100/n^2$$

Let $n_0 = 1$ and $c = 5 + 100 = 105$.
$100/n^2$ only get smaller as n increases and we ignore $-15/n$ since it only varies between -15 and 0

29

Induction on trees

A tree has |V|-1 edges

30

Subset-Sum: dynamic programming

$S = S_1 \ldots S_n$

SS(S,t): true/false, does S contain a subset that sums to t

31

Subset-Sum: dynamic programming

Recursive case:

$SS(S_{1\ldots n}, t) =$

32

Subset-Sum: dynamic programming

Recursive case:

$SS(S_{1\ldots n}, t) = SS(S_{1\ldots n-1}, t) \mid\mid SS(S_{1\ldots n-1}, t-S_n)$

What's changing? What does the structure look like?

33

Subset-Sum: dynamic programming

Recursive case:

$SS(S_{1\ldots n}, t) = SS(S_{1\ldots n-1}, t) \mid\mid SS(S_{1\ldots n-1}, t-S_n)$

34

Subset-Sum: dynamic programming

DP setup:

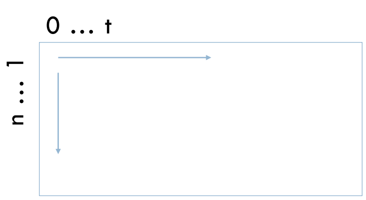
$SS[n, t] = SS[n-1, t] \mid\mid SS[n-1, t-S_n]$

35

Subset-Sum: dynamic programming

DP setup:

$SS[n, t] = SS[n-1, t] \mid\mid SS[n-1, t-S_n]$

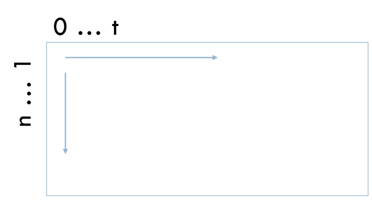


36

Subset-Sum: dynamic programming

DP setup:

$SS[n, t] = SS[n-1, t] \mid\mid SS[n-1, t-S_n]$



runtime?

37

Subset-Sum NP-Complete?!

38