Pre-lecture brain teaser

Is NP is closed under the kleene-star operation?

ECE-374-B: Lecture 25 - Final Review

Instructor: Nickvash Kani

University of Illinois at Urbana-Champaign

Final Topics

Topics for the final exam include:

- Everything on Midterm 1:
  - Regular expressions
  - DFAs, NFAs,
  - Fooling Sets and Closure properties
  - CFGs and PDAs
  - CSGs and LBAs
- Turing Machines
- MST Algorithms
- Everything on Midterm 2
  - Asymptotic Bounds
  - Recursion, Backtracking
  - Dynamic Programming
  - DFS/BFS
  - DAGs and TopSort
  - Shortest path algorithms
- Everything on Midterm 3
  - Reductions
  - P, NP, NP-hardness
  - Decidability

Final Topics

In today's lecture let's focus on a few that you guys had trouble on in the midterms (and the most recent stuff whih you'll be tested on).

- Everything on Midterm 1:
    - **Regular expressions**
    - **DFAs, NFAs,**
    - **Fooling Sets and Closure properties**
    - **CFGs** and **PDAs**
    - CSGs and LBAs
- Turing Machines
- MST Algorithms
- Everything on Midterm 2
    - **Asymptotic Bounds**
    - Recursion, Backtracking
    - Dynamic Programming
    - DFS/BFS
    - DAGs and TopSort
    - Shortest path algorithms
- Everything on Midterm 3
    - Reductions
    - **P, NP, NP-hardness**
    - Decidability

Pre-lecture brain teaser

Is NP is closed under the kleene-star operation?

Practice: Asymptotic bounds

Given an asymptotically tight bound for:

$$\sum_{i=1}^{n} i^3 \tag{1}$$

Practice: Regular expressions

Find the regular expression for the language:

$$\{w \in \{0,1\}^* | w \text{does not contain 00 as a substring}\} \tag{2}$$

Practice: Fooling Sets

Is the following language regular?

$$L = \{w \mid w \text{ does not contain the substring } 00 \text{ nor } 11\}$$

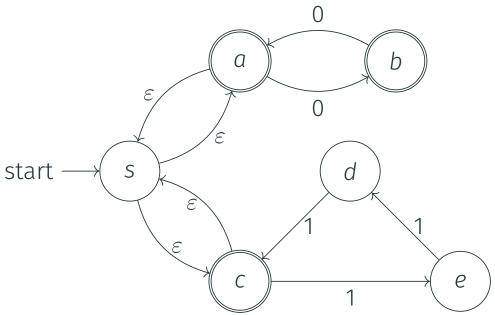
Practice: Fooling Sets

Is the following language regular?

$$L = \{w | w \text{ has an equal number of } 0\text{'s and } 1\text{'s } \}$$

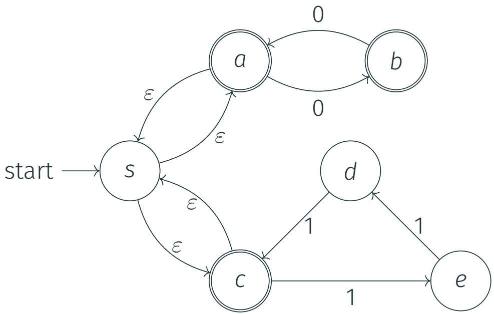
Practice: NFAs and DFAs

Let M be the following NFA:



Which of the following statements about M are true?

Let M be the following NFA:

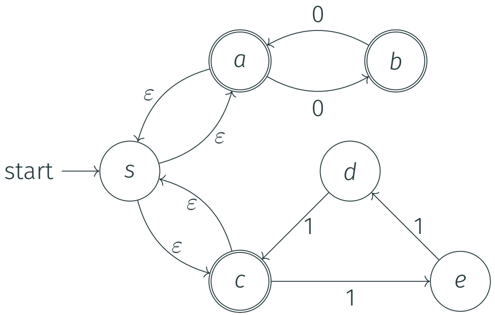


Which of the following statements about M are true?

1. M accepts the empty string ε -

Practice: NFAs and DFAs

Let M be the following NFA:

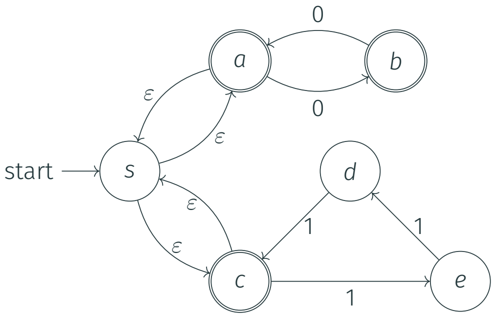


Which of the following statements about M are true?

1. M accepts the empty string ε -
2. $\delta(s, 010) = \{s, a, c\}$ -

Practice: NFAs and DFAs

Let M be the following NFA:

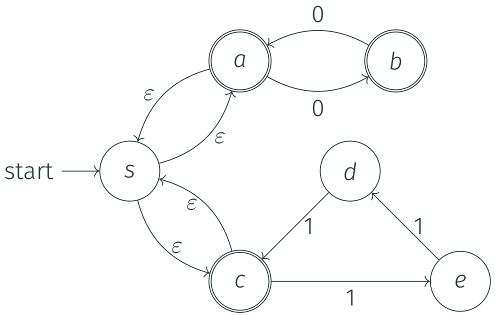


Which of the following statements about M are true?

1. M accepts the empty string ε -
2. $\delta(s, 010) = \{s, a, c\}$ -
3. $\varepsilon - \mathrm{reach}(a) = \{s, a, c\}$ -

Practice: NFAs and DFAs

Let M be the following NFA:

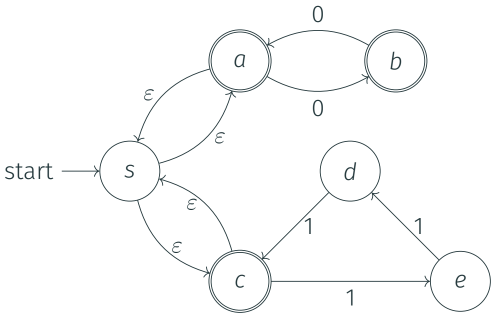


Which of the following statements about M are true?

1. M accepts the empty string ε -
2. $\delta(s, 010) = \{s, a, c\}$ -
3. $\varepsilon - \mathrm{reach}(a) = \{s, a, c\}$ -
4. M rejects the string 11100111000 -

Practice: NFAs and DFAs

Let M be the following NFA:



1. M accepts the empty string ε -
2. $\delta(s, 010) = \{s, a, c\}$ -
3. $\varepsilon - \mathrm{reach}(a) = \{s, a, c\}$ -
4. M rejects the string 11100111000 -
5. $L(M) = (00)^* + (111)^*$ -

Which of the following statements about M are true?

Practice: Closure

Which of the following is true for **every** language $L \subseteq \{0,1\}^*$

1. L^* is non-empty -
2. L^* is regular -
3. If L is NP-Hard, then L is not regular -
4. If L is not regular, then L is undecidable -

Context-Free Languages

Given $\Sigma = 0, 1$, the language $L = \{0^n 1^n | n \geq 0\}$ is represented by which grammar?

(a)

$$S \rightarrow 0T1|1$$
$$T \rightarrow T0|\varepsilon$$

(b)

$$S \rightarrow 0S1$$

(c)

$$S \rightarrow 0S1|0S|S1|\varepsilon$$

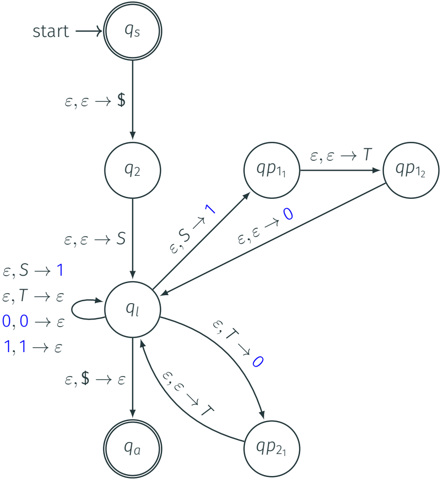
(d)

$$S \rightarrow AB1$$
$$A \rightarrow 0$$
$$B \rightarrow S|\varepsilon$$

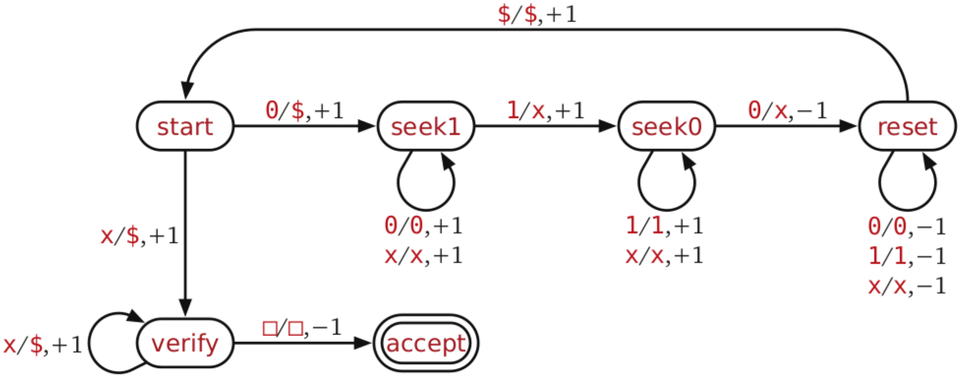
(e) None of the above

Push-down Auto-mata

What is the context-free grammar of the following push-down automata:



You have the following Turing machine diagram that accepts a particular language whose alphabet $\Sigma = \{0, 1\}$. Please describe the language.



Linear Time Selection

Recall the linear time selection logarithm that uses the medians of medians. I use the same algorithm, but instead of lists of size 5, I break the array into lists of size 7 and do the median-of-medians as normal. The running time for my new algorithm is:

(a) $O(\log(n))$
(b) $O(n)$
(c) $O(n\log(n))$
(d) $O(n^2)$
(e) None of the above

Linear Time Selection

Recall the linear time selection logarithm that uses the medians of medians. I use the same algorithm, but instead of lists of size 5, I break the array into lists of size 7 and do the median-of-medians as normal. The running time for my new algorithm is:

(a) $O(\log(n))$
(b) $O(n)$
(c) $O(n\log(n))$
(d) $O(n^2)$
(e) None of the above

Why did we choose lists of size 5? Will lists of size 3 work?

(Hint) Write a recurrence to analyze the algorithm's running time if we choose a list of size k.

We looked at the BasicSearch algorithm:

```
Explore(G,u):
    Visited[1..n] ← FALSE
    // ToExplore, S: Lists
    Add u to ToExplore and to S
    Visited[u] ← TRUE
    while (ToExplore is non-empty) do
        Remove node x from ToExplore
        for each edge xy in Adj(x) do
            if (Visited[y] = FALSE)
                Visited[y] ← TRUE
                Add y to ToExplore
                Add y to S
    Output S
```

We said that if ToExplore was a:

- Stack, the algorithm is equivalent to
- Queue, the algorithm is equivalent to

What if the algorithm was written recursively (instead of the while loop, you recursively call explore). What would the algorithm be equivalent to?

Let G = (V,E) be a connected, undirected graph with edge weights w, such that the weights are distinct, i.e., no two edges have the same weight. Which of the following is necessarily true about a minimum spanning tree of G?

(a) If T_1 and T_2 are MSTs of G then $T_1 = T_2$, i.e., the MST is unique.

(b) There are MSTs T_1 and T_2 such that $T_1 \neq T_2$ i.e, MST is not unique.

(c) There is an edge e that is **unsafe** that belongs to a MST.

(d) There is a **safe** edge that does not belong to a MST of G.

Consider the two problems:

Problem: 3SAT

Instance: Given a CNF formula φ with n variables, and k clauses
Question: Is there a truth assignment to the variables such that φ evaluates to true

Problem: Clique

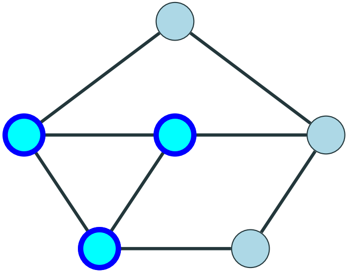
Instance: A graph G and an integer k.
Question: Does G has a clique of size $\geq k$?

Reduce **3SAT** to **CLIQUE**

Reduction: 3SAT to Clique

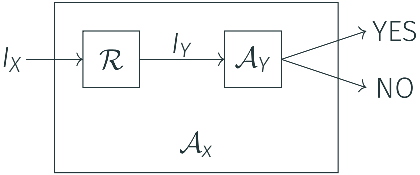
Given a graph G, a set of vertices V' is:

clique: every pair of vertices in V' is connected by an edge of G.



Reduction: 3SAT to Clique

Bust out the reduction diagram:



Reduction: 3SAT to Clique

Some thoughts:

- Clique is a fully connected graph and very similar to the independent set problem
- We want to have a clique with all the satisfying literals
    - Can’t have literal and its negation in same clique
    - Only need one satisfying literal per clique

Reduction: 3SAT to Clique

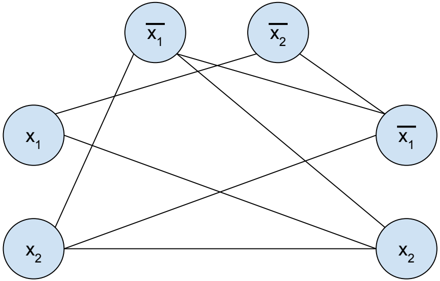
Hence the reduction creates a undirected graph G:

- Nodes in G are organized in k groups of nodes. Each triple corresponds to one clause.
- The edges of G connect all but:
  - nodes in the same triple
  - nodes with contradictory labels (x_1 and $\overline{x_1}$)

Hence the reduction creates a undirected graph G:

- Nodes in G are organized in k groups of nodes. Each triple corresponds to one clause.
- The edges of G connect all but:
  - nodes in the same triple
  - nodes with contradictory labels (x_1 and $\overline{x_1}$)

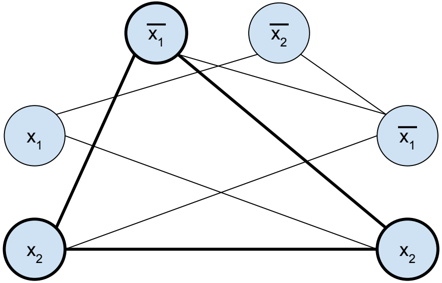
$\varphi = (x_1 \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2)$



Hence the reduction creates a undirected graph G:

- Nodes in G are organized in k groups of nodes. Each triple corresponds to one clause.
- The edges of G connect all but:
  - nodes in the same triple
  - nodes with contradictory labels (x_1 and $\overline{x_1}$)

$\varphi = (x_1 \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2)$

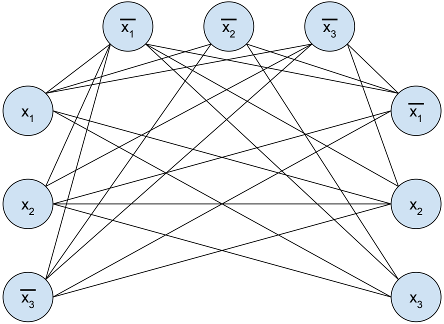


Reduction: 3SAT to Clique

Hence the reduction creates a undirected graph G:

- Nodes in G are organized in k groups of nodes. Each triple corresponds to one clause.
- The edges of G connect all but:
  - nodes in the same triple
  - nodes with contradictory labels (x_1 and $\overline{x_1}$)

$\varphi = (x_1 \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3}) \wedge (\overline{x_1} \vee x_2 \vee x_3)$

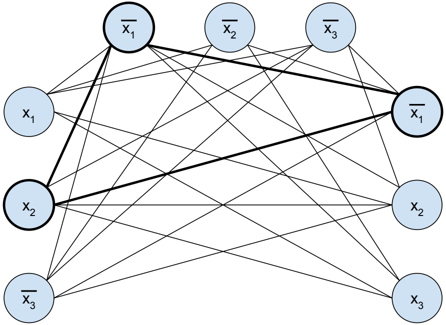


Reduction: 3SAT to Clique

Hence the reduction creates a undirected graph G:

- Nodes in G are organized in k groups of nodes. Each triple corresponds to one clause.
- The edges of G connect all but:
  - nodes in the same triple
  - nodes with contradictory labels (x_1 and $\overline{x_1}$)

$\varphi = (x_1 \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3}) \wedge (\overline{x_1} \vee x_2 \vee x_3)$



Very similar to 3SAT to independent set reduction:

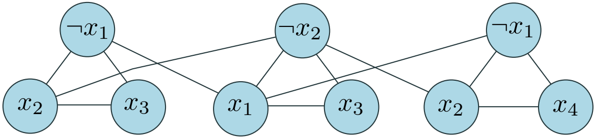


Figure 1: Graph for $\varphi = (\neg x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_4)$

Sample Reduction

Problem (SP1): Determine the shortest *simple* path in a graph. The graph is acyclic but has negative edge weights.

Does this graph belong to: P NP NP-hard NP-complete

Sample Reduction

Problem (SP1): Determine the shortest *simple* path in a graph. The graph is acyclic but has negative edge weights.

Does this graph belong to: P NP NP-hard NP-complete

We can show the reduction from LongestPath:

$$\textsc{LongestPath} \leq_P \text{SP1}$$

Reduction: Make all edges negative

Multi-section questions

Practice: Bringing it all together

Does there exist some language $L \subseteq \{0, 1\}^*$ where:

$L^* = (L^*)^*$

Practice: Bringing it all together

Does there exist some language $L \subseteq \{0, 1\}^*$ where:

L is decidable but L^* is undecidable

Practice: Bringing it all together

Does there exist some language $L \subseteq \{0, 1\}^*$ where:

L is neither regular nor NP-hard

Practice: Bringing it all together

Does there exist some language $L \subseteq \{0, 1\}^*$ where:

L is in P, but L has a infinite fooling set

Savitch’s Theorem

Proved by Walter Savitch in

Lemma
Savitch's Theroem: $NSPACE(f(n)) \subseteq DSPACE\left(f(n)^2\right)$

One last thought before you go....(my favorite theorem)

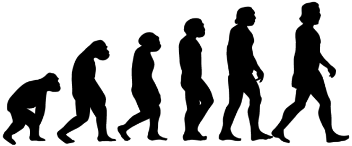
Proved by Walter Savitch in

Lemma
Savitch's Theroem: $NSPACE\,(f(n)) \subseteq DSPACE\left(f(n)^2\right)$

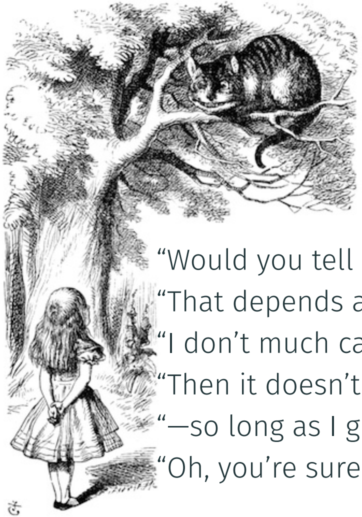
Idea behind the proof:

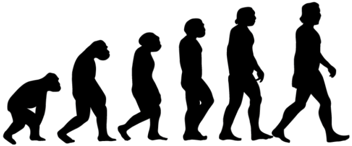
- STCON: finds whether there is a path between two vertices in $O\left((\log(n))^2\right)$ space
- Convert a nondeterministic Turing machine that takes $f(n)$ space into a configuration graph G_x^M
    - We know the tape can decide x in $f(n)$ space. Therefore there are $2^{O(f(n))}$ configurations
    - Therefore G_x^M has $2^{O(f(n))}$ vertices
- A deterministic Turing machine can run STCON on that graph resulting in $O\left(\left(\log\left(2^{O(f(n))}\right)\right)^2\right) \equiv O\left(f(n)^2\right)$ space

Farewell



Farewell

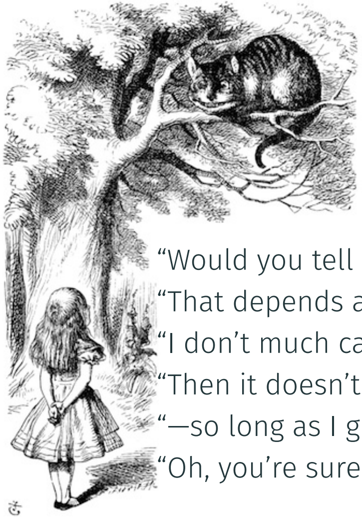




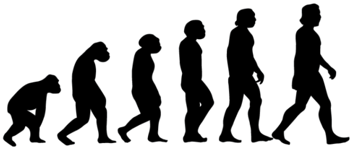
"Would you tell me, please, which way I ought to go from here?"
"That depends a good deal on where you want to get to," said the Cat.
"I don't much care where—" said Alice.
"Then it doesn't matter which way you go," said the Cat.
"—so long as I get somewhere," Alice added as an explanation.
"Oh, you're sure to do that," said the Cat, "if you only walk long enough."

Lewis Carroll, Alice's Adventures in Wonderland

Farewell



“When you’re going through hell, keep going.”
-Winston Churchill



“Would you tell me, please, which way I ought to go from here?”
“That depends a good deal on where you want to get to,” said the Cat.
“I don’t much care where—” said Alice.
“Then it doesn’t matter which way you go,” said the Cat.
“—so long as I get somewhere,” Alice added as an explanation.
“Oh, you’re sure to do that,” said the Cat, “if you only walk long enough.”

Lewis Carroll, Alice’s Adventures in Wonderland