Is NP is closed under the kleene-star operation?
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
In today’s lecture let’s focus on a few that you guys had trouble on in the midterms (and the most recent stuff which 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
Is NP closed under the Kleene-star operation?

Yes
\[ \text{NP}^* = \{ x_1, x_2, x_3 \} \]

\[ \text{NP}^* = \{ x_1, x_2, x_3 \} \]

\[
\text{NTM} - \text{Breaks the clean star string into parts}
\]

\[
\text{verify each part with original NTM solver for problem P_x}
\]
Given an asymptotically tight bound for:

\[ \sum_{i=1}^{n} i^3 \leq O(n^4) \]

Work in the function \( A(n) \) for \( \frac{n^3}{n} \) it times you call the function \( n \).

\[ O(A(n) \cdot B(n)) \]
Practice: Regular expressions

Find the regular expression for the language:

\( \{ w \in \{0, 1\}^* | \text{w does not contain 00 as a substring} \} \)

\[
\begin{align*}
\text{Regex} : & \quad 1^* + 1^* (011^*)^* 1^* + 1^* (101^*)^* 1^* + 0 \\
\text{regularity can be expressed by DFA, NFA, Regex} \\
\text{any language that can be constructed} \\
\text{by the base languages using a finite #} \\
\end{align*}
\]
Is the following language regular?

$L = \{ w | w \text{ does not contain the substring } 00 \text{ nor } 11 \}$
Is the following language regular?

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

Any DFA that represents this language:

$\delta(1, i-1) = 1 \text{ for } w \in L$

Contradiction
Let $M$ be the following NFA:

Which of the following statements about $M$ are true?

1. $M$ accepts the empty string
2. $(s, 010) = \{s, a, c\}$
3. $\text{reach}(a) = \{s, a, c\}$
4. $M$ rejects the string $11100111000$
5. $L(M) = (00)^\ast + (111)^\ast$
Let M be the following NFA:

Which of the following statements about M are true?

1. M accepts the empty string $\varepsilon$

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 $\varepsilon$ -
2. $\delta(s, 010) = \{s, a, c\}$ -

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 $\varepsilon$ -
2. $\delta(s, 010) = \{s, a, c\}$ -
3. $\varepsilon - \text{reach}(a) = \{s, a, c\}$ -
Let M be the following NFA:

Which of the following statements about M are true?

1. M accepts the empty string $\varepsilon$ -
2. $\delta(s, 010) = \{s, a, c\}$ -
3. $\varepsilon - \text{reach}(a) = \{s, a, c\}$ -
4. M rejects the string $11100111000$ -
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 $\varepsilon$ -
2. $\delta(s, 010) = \{s, a, c\}$ -
3. $\varepsilon - \text{reach}(a) = \{s, a, c\}$ -
4. $M$ rejects the string $11100111000$ -
5. $L(M) = (00)^* + (111)^*$ -
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 -
Given $\Sigma = 0, 1$, the language $L = \{0^n1^n | n \geq 0\}$ is represented by which grammar?

(a) 

$S \rightarrow 0T1 | 1$

$T \rightarrow T0 | \epsilon$

(b) 

$S \rightarrow 0S1$

(c) 

$S \rightarrow 0S1 | 0S | S1 | \epsilon$

(d) 

$S \rightarrow AB1$

$A \rightarrow 0$

$B \rightarrow S | \epsilon$

(e) None of the above
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.
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
Recall the linear time selection algorithm 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:

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

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 $\varphi$ with $n$ variables, and $k$ clauses

*Question:* Is there a truth assignment to the variables such that $\varphi$ 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:

\[ x = \text{SAT} \]
\[ y = \text{Clique} \]

\[ I_x = \phi \]
\[ I_y = G, k \]

If \( \phi \) is sat, the graph \( G \) has a clique of size \( \leq k \).
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

\[(x_1 \vee x_2 \vee \bar{x}_3) \land (x_4 \vee x_5 \vee x_6)\]

\(k\) clauses \(k\) literals that satisfy the \(Y\)
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 ($x_i$ and $\overline{x_i}$)
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 $\overline{x_1}$)

\[
\varphi = (x_1 \lor x_2) \land (\overline{x_1} \lor \overline{x_2}) \land (\overline{x_1} \lor x_2)
\]

\[
\land \left( x_1 \lor \overline{x_2} \right)
\]
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 $\overline{x}_1$)

$$\varphi = (x_1 \lor x_2) \land (\overline{x}_1 \lor \overline{x}_2) \land (\overline{x}_1 \lor x_2)$$
Hence the reduction creates an 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 $\overline{x}_1$)

$$\varphi = (x_1 \lor x_2 \lor \overline{x}_3) \land (\overline{x}_1 \lor x_2 \lor \overline{x}_3) \land (\overline{x}_1 \lor x_2 \lor x_3)$$
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 $\overline{x}_1$)

Let $\varphi = (x_1 \lor x_2 \lor \overline{x}_3) \land (\overline{x}_1 \lor x_2 \lor \overline{x}_3) \land (\overline{x}_1 \lor x_2 \lor x_3)$
Very similar to 3SAT to independent set reduction:

Figure 1: Graph for $\varphi = (\neg x_1 \lor x_2 \lor x_3) \land (x_1 \lor \neg x_2 \lor x_3) \land (\neg x_1 \lor x_2 \lor x_4)$
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 \hspace{1cm} NP \hspace{1cm} NP-hard \hspace{1cm} NP-complete
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:

\[ \text{LONGESTPATH} \leq_P \text{SP1} \]

Reduction: Make all edges negative
Longest Path \( Z \): Find the longest path Heaviest in \( G \)

Input: \( G \)

Output: weight of the longest path

\[ P \neq \text{NP} \quad \text{NP-hard} \quad \text{NP-complete} \]
Multi-section questions
Does there exist some language $L \subseteq \{0, 1\}^*$ where:

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

$L$ is decidable but $L^*$ is undecidable
Does there exist some language \( L \subseteq \{0,1\}^* \) where:

- \( L \) is neither regular nor NP-hard
Does there exist some language $L \subseteq \{0, 1\}^*$ where:

$L$ is in P, but $L$ has a infinite fooling set
Savitch’s Theorem
One last thought before you go....(my favorite theorem)

Proved by Walter Savitch in

**Lemma**

Savitch’s Theorem: \( \text{NSPACE}(f(n)) \subseteq \text{DSPACE}(f(n)^2) \)
One last thought before you go....(my favorite theorem)

Proved by Walter Savitch in

**Lemma**
Savitch’s Theorem: $\text{NSPACE}(f(n)) \subseteq \text{DSPACE}(f(n)^2)$

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 $G^M_x$
  - We know the tape can decide $x$ in $f(n)$ space. Therefore there are $2^{O(f(n))}$ configurations
  - Therefore $G^M_x$ has $2^{O(f(n))}$ vertices
- A deterministic Turing machine can run STCON on that graph resulting in $O\left((\log (2^{O(f(n)}))^2\right) \equiv O(f(n)^2)$ space
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
“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