Closure Properties of P and NP
Closure Properties of P and NP
Closure Properties of P and NP

- Union
Closure Properties of P and NP

- Union
- Intersection
Closure Properties of P and NP

- Union
- Intersection
- Complement
Closure Properties of P and NP

- Union
- Intersection
- Complement
- Concatenation
Closure Properties of P and NP

- Union
- Intersection
- Complement
- Concatenation
- Kleene star
Closure Properties of P
Closure of $P$ Under Union

**Thm** If $L_1 \in P$ and $L_2 \in P$ then $L_1 \cup L_2 \in P$. 
**Thm** If \( L_1 \in P \) and \( L_2 \in P \) then \( L_1 \cup L_2 \in P \). 
\( L_1 \in P \) via TM \( M_1 \) which works in time \( p_1(n) \). 
\( L_2 \in P \) via TM \( M_2 \) which works in time \( p_2(n) \).
Closure of P Under Union

**Thm** If $L_1 \in P$ and $L_2 \in P$ then $L_1 \cup L_2 \in P$.

$L_1 \in P$ via TM $M_1$ which works in time $p_1(n)$.

$L_2 \in P$ via TM $M_2$ which works in time $p_2(n)$.

The following algorithm recognizes $L_1 \cup L_2$ in poly time.

1. Input($x$) (We assume $|x| = n$).
2. Run $M_1(x)$, output is $b_1$ (this takes $p_1(n)$).
3. Run $M_2(x)$, output is $b_2$, (this takes $p_2(n)$).
4. If $b_1 = Y$ OR $b_2 = Y$ then output Y, else output N.

This algorithm takes $\sim p_1(n) + p_2(n)$, which is poly.

Note Key is that the set of polynomials is closed under addition.
Closure of $P$ Under Union

**Thm** If $L_1 \in P$ and $L_2 \in P$ then $L_1 \cup L_2 \in P$.

$L_1 \in P$ via TM $M_1$ which works in time $p_1(n)$.

$L_2 \in P$ via TM $M_2$ which works in time $p_2(n)$.

The following algorithm recognizes $L_1 \cup L_2$ in poly time.

1. Input($x$) (We assume $|x| = n$.)
2. Run $M_1(x)$, output is $b_1$ (this takes $p_1(n)$)
3. Run $M_2(x)$, output is $b_2$, (this takes $p_2(n)$)
4. If $b_1 = Y$ OR $b_2 = Y$ then output $Y$, else output $N$. 

This algorithm takes $\sim p_1(n) + p_2(n)$, which is poly.

Note Key is that the set of polynomials is closed under addition.
Closure of P Under Union

**Thm** If \( L_1 \in P \) and \( L_2 \in P \) then \( L_1 \cup L_2 \in P \).

\( L_1 \in P \) via TM \( M_1 \) which works in time \( p_1(n) \).

\( L_2 \in P \) via TM \( M_2 \) which works in time \( p_2(n) \).

The following algorithm recognizes \( L_1 \cup L_2 \) in poly time.

1. Input(\( x \)) (We assume \(|x| = n\).)
2. Run \( M_1(x) \), output is \( b_1 \) (this takes \( p_1(n) \))
3. Run \( M_2(x) \), output is \( b_2 \), (this takes \( p_2(n) \))
4. If \( b_1 = Y \) OR \( b_2 = Y \) then output \( Y \), else output \( N \).

This algorithm takes \( \sim p_1(n) + p_2(n) \), which is poly.
Closure of $P$ Under Union

**Thm** If $L_1 \in P$ and $L_2 \in P$ then $L_1 \cup L_2 \in P$.

$L_1 \in P$ via TM $M_1$ which works in time $p_1(n)$.

$L_2 \in P$ via TM $M_2$ which works in time $p_2(n)$.

The following algorithm recognizes $L_1 \cup L_2$ in poly time.

1. Input($x$) (We assume $|x| = n$.)
2. Run $M_1(x)$, output is $b_1$ (this takes $p_1(n)$)
3. Run $M_2(x)$, output is $b_2$, (this takes $p_2(n)$)
4. If $b_1 = Y$ OR $b_2 = Y$ then output $Y$, else output $N$.

This algorithm takes $\sim p_1(n) + p_2(n)$, which is poly.

**Note** Key is that the set of polynomials is closed under addition.
Closure of P Under Intersection

**Thm** If $L_1 \in P$ and $L_2 \in P$ then $L_1 \cap L_2 \in P$. 

This algorithm takes $\sim p_1(n) + p_2(n)$, which is poly.

Note Key is that the set of polynomials is closed under addition.
Closure of P Under Intersection

**Thm** If $L_1 \in P$ and $L_2 \in P$ then $L_1 \cap L_2 \in P$. $L_1 \in P$ via TM $M_1$ which works in time $p_1(n)$. $L_2 \in P$ via TM $M_2$ which works in time $p_2(n)$.
Closure of P Under Intersection

**Thm** If $L_1 \in P$ and $L_2 \in P$ then $L_1 \cap L_2 \in P$.

$L_1 \in P$ via TM $M_1$ which works in time $p_1(n)$.

$L_2 \in P$ via TM $M_2$ which works in time $p_2(n)$.

The following algorithm recognizes $L_1 \cup L_2$ in poly time.
Closure of P Under Intersection

**Thm** If \( L_1 \in P \) and \( L_2 \in P \) then \( L_1 \cap L_2 \in P \).

\( L_1 \in P \) via TM \( M_1 \) which works in time \( p_1(n) \).

\( L_2 \in P \) via TM \( M_2 \) which works in time \( p_2(n) \).

The following algorithm recognizes \( L_1 \cup L_2 \) in poly time.

1. Input(\( x \)) (We assume \( |x| = n \).)
2. Run \( M_1(x) \), output is \( b_1 \) (this takes \( p_1(n) \))
3. Run \( M_2(x) \), output is \( b_2 \), (this takes \( p_2(n) \))
4. If \( b_1 = Y \) AND \( b_2 = Y \) then output \( Y \), else output \( N \).
Closure of P Under Intersection

**Thm** If $L_1 \in P$ and $L_2 \in P$ then $L_1 \cap L_2 \in P$.
$L_1 \in P$ via TM $M_1$ which works in time $p_1(n)$.
$L_2 \in P$ via TM $M_2$ which works in time $p_2(n)$.

The following algorithm recognizes $L_1 \cup L_2$ in poly time.

1. Input($x$) (We assume $|x| = n$.)
2. Run $M_1(x)$, output is $b_1$ (this takes $p_1(n)$)
3. Run $M_2(x)$, output is $b_2$, (this takes $p_2(n)$)
4. If $b_1 = Y$ AND $b_2 = Y$ then output $Y$, else output $N$.

This algorithm takes $\sim p_1(n) + p_2(n)$, which is poly.
Closure of P Under Intersection

**Thm**  If \( L_1 \in P \) and \( L_2 \in P \) then \( L_1 \cap L_2 \in P \).

\( L_1 \in P \) via TM \( M_1 \) which works in time \( p_1(n) \).

\( L_2 \in P \) via TM \( M_2 \) which works in time \( p_2(n) \).

The following algorithm recognizes \( L_1 \cup L_2 \) in poly time.

1. Input(\( x \)) (We assume \( |x| = n \).)
2. Run \( M_1(x) \), output is \( b_1 \) (this takes \( p_1(n) \))
3. Run \( M_2(x) \), output is \( b_2 \), (this takes \( p_2(n) \))
4. If \( b_1 = Y \) AND \( b_2 = Y \) then output \( Y \), else output \( N \).

This algorithm takes \( \sim p_1(n) + p_2(n) \), which is poly.

**Note**  Key is that the set of polynomials is closed under addition.
Closure \( P \) Under Concatenation

**Thm** If \( L_1 \in P \) and \( L_2 \in P \) then \( L_1L_2 \in P \).
Closure P Under Concatenation

**Thm** If $L_1 \in P$ and $L_2 \in P$ then $L_1L_2 \in P$.

$L_1 \in P$ via TM $M_1$ which works in time $p_1(n)$.

$L_2 \in P$ via TM $M_2$ which works in time $p_2(n)$. 

The following algorithm recognizes $L_1L_2$ in poly time.

1. Input($x$) (We assume $|x| = n$.) Let $x = x_1 \cdots x_n$.

2. For $0 \leq i \leq n$
   - Run $M_1(x_1 \cdots x_i)$ and $M_2(x_{i+1} \cdots x_n)$. If both say Y then output Y and STOP. (Time: $p_1(i) + p_2(n-i) \leq p_1(n) + p_2(n)$.)

3. Output N

This algorithm takes $\leq (n+1) \times (p_1(n) + p_2(n))$ which is poly.

Note Key is that the set of polynomials is closed under addition and mult by $n$. 
Closure P Under Concatenation

**Thm** If $L_1 \in P$ and $L_2 \in P$ then $L_1 L_2 \in P$.

$L_1 \in P$ via TM $M_1$ which works in time $p_1(n)$.

$L_2 \in P$ via TM $M_2$ which works in time $p_2(n)$.

The following algorithm recognizes $L_1 L_2$ in poly time.
Closure P Under Concatenation

**Thm** If \( L_1 \in P \) and \( L_2 \in P \) then \( L_1L_2 \in P \).

\( L_1 \in P \) via TM \( M_1 \) which works in time \( p_1(n) \).

\( L_2 \in P \) via TM \( M_2 \) which works in time \( p_2(n) \).

The following algorithm recognizes \( L_1L_2 \) in poly time.

1. **Input** (We assume \( |x| = n \).) Let \( x = x_1 \cdots x_n \)

2. For \( 0 \leq i \leq n \)
   
   2.1 Run \( M_1(x_1 \cdots x_i) \) and \( M_2(x_{i+1} \cdots x_n) \). If both say \( Y \) then output \( Y \) and STOP. (Time:
   
   \( p_1(i) + p_2(n - i) \leq p_1(n) + p_2(n) \).

3. Output \( N \)
Closure $P$ Under Concatenation

**Thm** If $L_1 \in P$ and $L_2 \in P$ then $L_1L_2 \in P$.

$L_1 \in P$ via TM $M_1$ which works in time $p_1(n)$.

$L_2 \in P$ via TM $M_2$ which works in time $p_2(n)$.

The following algorithm recognizes $L_1L_2$ in poly time.

1. Input($x$) (We assume $|x| = n$.) Let $x = x_1 \cdots x_n$

2. For $0 \leq i \leq n$
   
   2.1 Run $M_1(x_1 \cdots x_i)$ and $M_2(x_{i+1} \cdots x_n)$. If both say Y then output Y and STOP. (Time:
   $p_1(i) + p_2(n - i) \leq p_1(n) + p_2(n)$.)

3. Output N

This algorithm takes $\leq (n + 1) \times (p_1(n) + p_2(n))$ which is poly.
Closure P Under Concatenation

**Thm**  If $L_1 \in P$ and $L_2 \in P$ then $L_1L_2 \in P$.

$L_1 \in P$ via TM $M_1$ which works in time $p_1(n)$.

$L_2 \in P$ via TM $M_2$ which works in time $p_2(n)$.

The following algorithm recognizes $L_1L_2$ in poly time.

1. Input($x$) (We assume $|x| = n$.) Let $x = x_1 \cdots x_n$

2. For $0 \leq i \leq n$
   2.1 Run $M_1(x_1 \cdots x_i)$ and $M_2(x_{i+1} \cdots x_n)$. If both say Y then output Y and STOP. (Time: $p_1(i) + p_2(n - i) \leq p_1(n) + p_2(n)$.)

3. Output N

This algorithm takes $\leq (n + 1) \times (p_1(n) + p_2(n))$ which is poly.

**Note** Key is that the set of polynomials is closed under addition and mult by $n$. 


Closure of P Under Complementation

**Thm** If \( L \in P \) then \( \overline{L} \in P \).
Closure of P Under Complementation

**Thm** If \( L \in P \) then \( \overline{L} \in P \).

\( L \in P \) via TM \( M \) which works in time \( p(n) \).
Closure of \( P \) Under Complementation

**Thm** If \( L \in P \) then \( \overline{L} \in P \).

\( L \in P \) via TM \( M \) which works in time \( p(n) \).

The following algorithm recognizes \( \overline{L} \) in poly time.

1. Input(\( x \)) (We assume \( |x| = n \).)
2. Run \( M(x) \). Answer is \( b \).
3. If \( b = Y \) then output N, if \( b = N \) then output Y.

Run time is \( \sim p(n) \), a poly.
**Thm**  If $L \in P$ then $\overline{L} \in P$.

$L \in P$ via TM $M$ which works in time $p(n)$.

The following algorithm recognizes $\overline{L}$ in poly time.

1. Input($x$) (We assume $|x| = n$.)
2. Run $M(x)$. Answer is $b$.
3. If $b = Y$ then output N, if $b = N$ then output Y.

Run time is $\sim p(n)$, a poly.

**Note**  No note needed.
Closure of $P$ Under $*$

**Thm** If $L \in P$ then $L^* \in P$.

**Proof**
First lets talk about what you **should not** do.
Closure of $P$ Under $*$

**Thm** If $L \in P$ then $L^* \in P$.

**Proof**
First let's talk about what you **should not** do.

**A contrast**

- $x \in L^*$? Look at ??? ways to have $x = z_1 \cdots z_m$. 
**Thm** If \( L \in P \) then \( L^* \in P \).

**Proof**

First let's talk about what you **should not** do.

**A contrast**

- \( x \in L^* \)? Look at \( ??? \) ways to have \( x = z_1 \cdots z_m \).
  - Break string into 1 piece: \( \binom{n}{0} \) ways to do this.
  - Break string into 2 pieces: \( \binom{n}{1} \) ways to do this.
  - Break string into 3 pieces: \( \binom{n}{2} \) ways to do this.
  
  \[ \vdots \]
  - Break string into \( n \) piece: \( \binom{n}{n} \) ways to do this.
Thm  If $L \in P$ then $L^* \in P$.

Proof
First lets talk about what you **should not** do.

A contrast

- $x \in L^*$? Look at $\ldots$ ways to have $x = z_1 \cdots z_m$.
  - Break string into 1 piece: $\binom{n}{0}$ ways to do this.
  - Break string into 2 pieces: $\binom{n}{1}$ ways to do this.
  - Break string into 3 piece: $\binom{n}{2}$ ways to do this.
  
  $\vdots$

Break string into $n$ piece: $\binom{n}{n}$ ways to do this.
So total number of ways to break up the string is

$$\binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{n}.$$ 

What is another name for this?
That Weird Sum: A Story

B is Bill, D is Darling.

B: D, how many subsets are there of \( \{1, \ldots, n\} \)?
B is Bill, D is Darling.

B: D, how many subsets are there of \( \{1, \ldots, n\} \)?

D: You can either choose 0 elements or choose 1 element, so

\[
\sum_{k=0}^{n} \binom{n}{k} = 2^n.
\]

Now, You got sum, I got \( 2^n \). What does that mean?

D: That one of us is wrong.

B: No. It means our answers are equal: \( 2^n = \sum_{k=0}^{n} \binom{n}{k} \).

D: Really!

B: Yes, really!
That Weird Sum: A Story

B is Bill, D is Darling.

B: D, how many subsets are there of \( \{1, \ldots, n\} \)?

D: You can either choose 0 elements or choose 1 element, so

\[
\binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{n}.
\]
That Weird Sum: A Story

B is Bill, D is Darling.

B: D, how many subsets are there of \( \{1, \ldots, n\} \)?

D: You can either choose 0 elements or choose 1 element, so

\[
\binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{n}.
\]

B: Another Way: 1 is IN or OUT, 2 is IN or OUT, etc, so \(2^n\). Now,
You got sum, I got \(2^n\). What does that mean?
That Weird Sum: A Story

B is Bill, D is Darling.
B: D, how many subsets are there of \( \{1, \ldots, n\} \)?
D: You can either choose 0 elements or choose 1 element, so

\[
\binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{n}.
\]

B: Another Way: 1 is IN or OUT, 2 is IN or OUT, etc, so \(2^n\). Now,
You got sum, I got \(2^n\). What does that mean?
D: That one of us is wrong.
That Weird Sum: A Story

B is Bill, D is Darling.

B: D, how many subsets are there of \( \{1, \ldots, n\} \)?

D: You can either choose 0 elements or choose 1 element, so

\[
\binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{n}.
\]

B: Another Way: 1 is IN or OUT, 2 is IN or OUT, etc, so \(2^n\). Now,
You got sum, I got \(2^n\). What does that mean?

D: That one of us is wrong.

B: No. It means our answers are equal:

\[
2^n = \binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{n}.
\]

D: Really!

B: Yes, really!
Back to Our Story

Back to our problem:
The technique of looking at all ways to break up $x$ into pieces takes roughly $2^n$ steps, so we need to do something clever.
Back to Our Story

Back to our problem:
The technique of looking at all ways to break up $x$ into pieces takes roughly $2^n$ steps, so we need to do something clever.

**Dynamic Programming** We solve a harder problem but get lots of information we don’t need in the process.
Back to Our Story

Back to our problem:
The technique of looking at all ways to break up $x$ into pieces takes roughly $2^n$ steps, so we need to do something clever.

**Dynamic Programming** We solve a harder problem but get lots of information we don’t need in the process.

**Original Problem** Given $x = x_1 \cdots x_n$ want to know if $x \in L^*$
Back to Our Story

Back to our problem:
The technique of looking at all ways to break up $x$ into pieces takes roughly $2^n$ steps, so we need to do something clever.

**Dynamic Programming** We solve a harder problem but get lots of information we don’t need in the process.

**Original Problem** Given $x = x_1 \cdots x_n$ want to know if $x \in L^*$

**New Problem** Given $x = x_1 \cdots x_n$ want to know:
Back to Our Story

Back to our problem:
The technique of looking at all ways to break up $x$ into pieces takes roughly $2^n$ steps, so we need to do something clever.

**Dynamic Programming** We solve a harder problem but get lots of information we don’t need in the process.

**Original Problem** Given $x = x_1 \cdots x_n$ want to know if $x \in L^*$

**New Problem** Given $x = x_1 \cdots x_n$ want to know:

- $e \in L^*$
- $x_1 \in L^*$
- $x_1x_2 \in L^*$
- $\vdots$
- $x_1x_2 \cdots x_n \in L^*$.  

Intuition

$x_1 \cdots x_i \in L^*$ IFF it can be broken into TWO pieces, the first one in $L^*$, and the second in $L^*$.  

Back to our problem:
The technique of looking at all ways to break up \( x \) into pieces takes roughly \( 2^n \) steps, so we need to do something clever.

**Dynamic Programming**  We solve a harder problem but get lots of information we don’t need in the process.

**Original Problem**  Given \( x = x_1 \cdots x_n \) want to know if \( x \in L^* \)

**New Problem**  Given \( x = x_1 \cdots x_n \) want to know:

\[ e \in L^* \]
\[ x_1 \in L^* \]
\[ x_1 x_2 \in L^* \]
\[ \vdots \]
\[ x_1 x_2 \cdots x_n \in L^*. \]

**Intuition**  \( x_1 \cdots x_i \in L^* \) IFF it can be broken into TWO pieces, the first one in \( L^* \), and the second in \( L \).
Final Algorithm

$A[i]$ stores if $x_1 \cdots x_i$ is in $L^*$. $M$ is poly-time Alg for $L$, poly $p$. 
Final Algorithm

$A[i]$ stores if $x_1 \cdots x_i$ is in $L^*$. $M$ is poly-time Alg for $L$, poly $p$.

Input $x = x_1 \cdots x_n$
$A[0] = \text{TRUE}$
for $i = 1$ to $n$ do
  for $j = 0$ to $i - 1$ do
    if $A[j]$ AND $M(x_{j+1} \cdots x_i) = Y$ then $A[i] = \text{TRUE}$
  output $A[n]$
Final Algorithm

$A[i]$ stores if $x_1 \cdots x_i$ is in $L^*$. $M$ is poly-time Alg for $L$, poly $p$.

Input $x = x_1 \cdots x_n$


$A[0] = \text{TRUE}$

for $i = 1$ to $n$ do
  for $j = 0$ to $i - 1$ do
    if $A[j]$ AND $M(x_{j+1} \cdots x_i) = Y$ then $A[i] = \text{TRUE}$

output $A[n]$

$O(n^2)$ calls to $M$ on inputs of length $\leq n$. Runtime $\leq O(n^2p(n))$. 
**Final Algorithm**

$A[i]$ stores if $x_1 \cdots x_i$ is in $L^*$. $M$ is poly-time Alg for $L$, poly $p$.

Input $x = x_1 \cdots x_n$


$A[0] = \text{TRUE}$

for $i = 1$ to $n$ do

    for $j = 0$ to $i - 1$ do

        if $A[j]$ AND $M(x_{j+1} \cdots x_i) = Y$ then $A[i] = \text{TRUE}$

output $A[n]$

$O(n^2)$ calls to $M$ on inputs of length $\leq n$. Runtime $\leq O(n^2 p(n))$.

**Note** Key is that the set of polynomials is closed under mult by $n^2$. 
Closure Properties of NP
We will now show that NP is closed under $\cup$, $\cap$, $\cdot$, and $\ast$. 
We will now show that NP is closed under $\cup$, $\cap$, $\cdot$, and $\ast$.

1. Our proofs will use that poly’s are closed under stuff, as did the proofs of closure under P. But we will not state this.
Closure of NP Under . . .

We will now show that NP is closed under $\cup$, $\cap$, $\cdot$, and $\ast$.

1. Our proofs will use that poly’s are closed under stuff, as did the proofs of closure under P. But we will not state this.

2. None of the proofs is anywhere near as hard as the proof that P is closed under $\ast$. 
We will now show that NP is closed under $\cup$, $\cap$, $\cdot$, and $\ast$.

1. Our proofs will use that poly’s are closed under stuff, as did the proofs of closure under P. But we will not state this.

2. None of the proofs is anywhere near as hard as the proof that P is closed under $\ast$.

3. Note that we did not include complementation. We’ll get to that later.
Closure of NP Under Union

**Thm** If $L_1 \in \text{NP}$ and $L_2 \in \text{NP}$ then $L_1 \cup L_2 \in \text{NP}$. 

The following defines $L_1 \cup L_2$ in an NP-way.

$L_1 \cup L_2 = \{ x : (\exists y) \left[ |y| = p_1(|x|) + p_2(|x|) + 1 \land y = y_1 \lor y_2 \right] \}$

Witness $|y| = p_1(|x|) + p_2(|x|) + 1$ is short.

Verification $(x, y_1) \in B_1 \lor (x, y_2) \in B_2$ is quick.
Closure of NP Under Union

**Thm** If $L_1 \in \text{NP}$ and $L_2 \in \text{NP}$ then $L_1 \cup L_2 \in \text{NP}$.

$L_1 = \{ x : (\exists y_1)[|y_1| = p_1(|x|) \land (x, y_1) \in B_1]\}$

$L_2 = \{ x : (\exists y_2)[|y_2| = p_2(|x|) \land (x, y_2) \in B_2]\}$
Closure of NP Under Union

**Thm** If $L_1 \in \text{NP}$ and $L_2 \in \text{NP}$ then $L_1 \cup L_2 \in \text{NP}$.

$L_1 = \{x : (\exists y_1)[|y_1| = p_1(|x|) \land (x, y_1) \in B_1]\}$

$L_2 = \{x : (\exists y_2)[|y_2| = p_2(|x|) \land (x, y_2) \in B_2]\}$

The following defines $L_1 \cup L_2$ in an NP-way.

$L_1 \cup L_2 = \{x : (\exists y) [\]$
Closure of NP Under Union

Thm  If $L_1 \in \text{NP}$ and $L_2 \in \text{NP}$ then $L_1 \cup L_2 \in \text{NP}$.
$L_1 = \{x : (\exists y_1)[|y_1| = p_1(|x|) \land (x, y_1) \in B_1]\}$
$L_2 = \{x : (\exists y_2)[|y_2| = p_2(|x|) \land (x, y_2) \in B_2]\}$

The following defines $L_1 \cup L_2$ in an NP-way.
$L_1 \cup L_2 = \{x : (\exists y)$
$[|y| = p_1(|x|) + p_2(|x|) + 1 \land$
$(x, y_1) \in B_1 \lor (x, y_2) \in B_2)\}$
Closure of NP Under Union

**Thm** If $L_1 \in \text{NP}$ and $L_2 \in \text{NP}$ then $L_1 \cup L_2 \in \text{NP}$.

$L_1 = \{ x : (\exists y_1)[|y_1| = p_1(|x|) \land (x, y_1) \in B_1] \}$

$L_2 = \{ x : (\exists y_2)[|y_2| = p_2(|x|) \land (x, y_2) \in B_2] \}$

The following defines $L_1 \cup L_2$ in an NP-way.

$L_1 \cup L_2 = \{ x : (\exists y) [\]

$|y| = p_1(|x|) + p_2(|x|) + 1 \land$

$y = y_1$y_2$ where $|y_1| = p_1(|x|)$ and $|y_2| = p_2(|x|)$\}$
Closure of NP Under Union

**Thm** If $L_1 \in \text{NP}$ and $L_2 \in \text{NP}$ then $L_1 \cup L_2 \in \text{NP}$.

$L_1 = \{x : (\exists y_1)[|y_1| = p_1(|x|) \land (x, y_1) \in B_1]\}$

$L_2 = \{x : (\exists y_2)[|y_2| = p_2(|x|) \land (x, y_2) \in B_2]\}$

The following defines $L_1 \cup L_2$ in an NP-way.

$L_1 \cup L_2 = \{x : (\exists y)[$

$|y| = p_1(|x|) + p_2(|x|) + 1 \land$

$y = y_1 \$ y_2$ where$ |y_1| = p_1(|x|)$ and$ |y_2| = p_2(|x|) \land$

$(x, y_1) \in B_1 \lor (x, y_2) \in B_2)$

$] \}$
Closure of NP Under Union

**Thm** If \( L_1 \in \text{NP} \) and \( L_2 \in \text{NP} \) then \( L_1 \cup L_2 \in \text{NP} \).

\[
L_1 = \{ x : (\exists y_1)[|y_1| = p_1(|x|) \land (x, y_1) \in B_1] \}
\]

\[
L_2 = \{ x : (\exists y_2)[|y_2| = p_2(|x|) \land (x, y_2) \in B_2] \}
\]

The following defines \( L_1 \cup L_2 \) in an NP-way.

\[
L_1 \cup L_2 = \{ x : (\exists y) \n\]

\[
|y| = p_1(|x|) + p_2(|x|) + 1 \land
\]

\[
y = y_1 \$ y_2 \text{ where } |y_1| = p_1(|x|) \text{ and } |y_2| = p_2(|x|) \land
\]

\[
(x, y_1) \in B_1 \lor (x, y_2) \in B_2
\]

**Witness** \(|y| = p_1(|x|) + p_2(|x|) + 1\) is short.
Closure of NP Under Union

**Thm**  If \( L_1 \in \text{NP} \) and \( L_2 \in \text{NP} \) then \( L_1 \cup L_2 \in \text{NP} \).

\[
L_1 = \{ x : (\exists y_1)[|y_1| = p_1(|x|) \land (x, y_1) \in B_1] \}
\]

\[
L_2 = \{ x : (\exists y_2)[|y_2| = p_2(|x|) \land (x, y_2) \in B_2] \}
\]

The following defines \( L_1 \cup L_2 \) in an NP-way.

\[
L_1 \cup L_2 = \{ x : (\exists y) \]
\[
[ |y| = p_1(|x|) + p_2(|x|) + 1 \land \\
 y = y_1 \$ y_2 \text{ where } |y_1| = p_1(|x|) \text{ and } |y_2| = p_2(|x|) \land \\
 (x, y_1) \in B_1 \lor (x, y_2) \in B_2 \}
\]

**Witness**  \(|y| = p_1(|x|) + p_2(|x|) + 1\) is short.

**Verification**  \((x, y_1) \in B_1 \lor (x, y_2) \in B_2\), is quick.
Closure of NP Under Intersection

**Thm** If $L_1 \in \text{NP}$ and $L_2 \in \text{NP}$ then $L_1 \cap L_2 \in \text{NP}$. 
Closure of NP Under Intersection

**Thm** If $L_1 \in \text{NP}$ and $L_2 \in \text{NP}$ then $L_1 \cap L_2 \in \text{NP}$. Similar to UNION.
Thm  If $L_1 \in \text{NP}$ and $L_2 \in \text{NP}$ then $L_1 L_2 \in \text{NP}$.
Closure NP Under Concatenation

**Thm** If $L_1 \in \text{NP}$ and $L_2 \in \text{NP}$ then $L_1 L_2 \in \text{NP}$.

$L_1 = \{x : (\exists y_1)[|y_1| = p_1(|x|) \land (x, y_1) \in B_1]\}$

$L_2 = \{x : (\exists y_2)[|y_2| = p_2(|x|) \land (x, y_2) \in B_2]\}$
Closure NP Under Concatenation

**Thm** If $L_1 \in \text{NP}$ and $L_2 \in \text{NP}$ then $L_1 L_2 \in \text{NP}$.

$L_1 = \{ x : (\exists y_1)[|y_1| = p_1(|x|) \wedge (x, y_1) \in B_1] \}$

$L_2 = \{ x : (\exists y_2)[|y_2| = p_2(|x|) \wedge (x, y_2) \in B_2] \}$

The following defines $L_1 L_2$ in an NP-way.
Closure NP Under Concatenation

**Thm** If \( L_1 \in \text{NP} \) and \( L_2 \in \text{NP} \) then \( L_1L_2 \in \text{NP} \).

\[
L_1 = \{ x : (\exists y_1)[|y_1| = p_1(|x|) \land (x, y_1) \in B_1] \}
\]

\[
L_2 = \{ x : (\exists y_2)[|y_2| = p_2(|x|) \land (x, y_2) \in B_2] \}
\]

The following defines \( L_1L_2 \) in an NP-way.

\[
\{ x : (\exists x_1, x_2, y_1, y_2) [\]
\]
**Closure NP Under Concatenation**

**Thm** If \( L_1 \in \text{NP} \) and \( L_2 \in \text{NP} \) then \( L_1L_2 \in \text{NP} \).

\[
L_1 = \{ x : (\exists y_1)[|y_1| = p_1(|x|) \land (x, y_1) \in B_1] \}
\]

\[
L_2 = \{ x : (\exists y_2)[|y_2| = p_2(|x|) \land (x, y_2) \in B_2] \}
\]

The following defines \( L_1L_2 \) in an NP-way.

\[
\{ x : (\exists x_1, x_2, y_1, y_2)[ ▶ x = x_1x_2]
\]

\[
▶ x = x_1x_2
\]
**Thm** If $L_1 \in \text{NP}$ and $L_2 \in \text{NP}$ then $L_1L_2 \in \text{NP}$.

$L_1 = \{x : (\exists y_1)[|y_1| = p_1(|x|) \land (x, y_1) \in B_1]\}$

$L_2 = \{x : (\exists y_2)[|y_2| = p_2(|x|) \land (x, y_2) \in B_2]\}$

The following defines $L_1L_2$ in an NP-way.

\[
\{x : (\exists x_1, x_2, y_1, y_2)[
\begin{align*}
\text{• } x &= x_1x_2 \\
\text{• } |y_1| &= p_1(|x_1|)
\end{align*}
\}
\]
Thm If $L_1 \in \text{NP}$ and $L_2 \in \text{NP}$ then $L_1L_2 \in \text{NP}$.

$L_1 = \{x : (\exists y_1)[|y_1| = p_1(|x|) \land (x, y_1) \in B_1]\}$

$L_2 = \{x : (\exists y_2)[|y_2| = p_2(|x|) \land (x, y_2) \in B_2]\}$

The following defines $L_1L_2$ in an NP-way.

$$\{x : (\exists x_1, x_2, y_1, y_2)[$$

- $x = x_1x_2$
- $|y_1| = p_1(|x_1|)$
- $|y_2| = p_2(|x_2|)$
Closure NP Under Concatenation

**Thm** If $L_1 \in \text{NP}$ and $L_2 \in \text{NP}$ then $L_1L_2 \in \text{NP}$. 

$L_1 = \{x : (\exists y_1)[|y_1| = p_1(|x|) \land (x, y_1) \in B_1]\}$

$L_2 = \{x : (\exists y_2)[|y_2| = p_2(|x|) \land (x, y_2) \in B_2]\}$

The following defines $L_1L_2$ in an NP-way.

\[
\{x : (\exists x_1, x_2, y_1, y_2)[
\begin{align*}
&\quad x = x_1x_2 \\
&\quad |y_1| = p_1(|x_1|) \\
&\quad |y_2| = p_2(|x_2|) \\
&\quad (x_1, y_1) \in B_1
\end{align*}
\}
\]
Thm  If $L_1 \in \text{NP}$ and $L_2 \in \text{NP}$ then $L_1L_2 \in \text{NP}$.

$L_1 = \{ x : (\exists y_1)[|y_1| = p_1(|x|) \land (x, y_1) \in B_1] \}$

$L_2 = \{ x : (\exists y_2)[|y_2| = p_2(|x|) \land (x, y_2) \in B_2] \}$

The following defines $L_1L_2$ in an NP-way.

$$\{ x : (\exists x_1, x_2, y_1, y_2)[$$

- $x = x_1x_2$
- $|y_1| = p_1(|x_1|)$
- $|y_2| = p_2(|x_2|)$
- $(x_1, y_1) \in B_1$
- $(x_2, y_2) \in B_2$

$$] \}$$
**Thm** If $L \in \text{NP}$ then $L^* \in \text{NP}$. 
Closure of NP Under $*$

**Thm** If $L \in \text{NP}$ then $L^* \in \text{NP}$.

$L = \{ x : (\exists y)[|y| = p(|x|) \land (x, y) \in B] \}$
Closure of NP Under *

**Thm**  If $L \in \text{NP}$ then $L^* \in \text{NP}$.

$L = \{ x : (\exists y)[|y| = p(|x|) \land (x, y) \in B] \}$

The following defines $L^*$ in an NP-way

$$\{ x : (\exists z_1, \ldots, z_k, y_1, \ldots, y_k) \}
\begin{align*}
\begin{array}{l}
\quad x = z_1 \cdots z_k \\
\quad (\forall i)[|y_i| = p(|z_i|)] \\
\quad (\forall i)[(z_i, y_i) \in B]
\end{array}
\end{align*}$$
Is NP closed under Complementation

Vote

1. There is a proof that if $L \in \text{NP}$ then $L \in \text{NP}$. (Hence $\text{NP}$ is closed under complementation and we know this.)

2. There is a language $L \in \text{NP}$ with $L \not\in \text{NP}$. (Hence $\text{NP}$ is not closed under complementation and we know this.)

3. The question of whether or not $\text{NP}$ is closed under complementation is Unknown to Science!

Answer Unknown to Science!
Is NP closed under Complementation

Vote

1. There is a proof that if \( L \in \text{NP} \) then \( \overline{L} \in \text{NP} \). (Hence \( \text{NP} \) is closed under complementation and we know this.)
Is NP closed under Complementation

Vote

1. There is a proof that if \( L \in \text{NP} \) then \( \overline{L} \in \text{NP} \). (Hence NP is closed under complementation and we know this.)

2. There is a language \( L \in \text{NP} \) with \( \overline{L} \notin \text{NP} \). (Hence NP is not closed under complementation and we know this.)
Vote

1. There is a proof that if $L \in \text{NP}$ then $\overline{L} \in \text{NP}$. (Hence \text{NP} is closed under complementation and we know this.)

2. There is a language $L \in \text{NP}$ with $\overline{L} \notin \text{NP}$. (Hence \text{NP} is not closed under complementation and we know this.)

3. The question of whether or not \text{NP} is closed under complementation is \textbf{Unknown to Science}!
Is NP closed under Complementation

Vote

1. There is a proof that if \( L \in \text{NP} \) then \( \overline{L} \in \text{NP} \). (Hence \( \text{NP} \) is closed under complementation and we know this.)

2. There is a language \( L \in \text{NP} \) with \( \overline{L} \notin \text{NP} \). (Hence \( \text{NP} \) is not closed under complementation and we know this.)

3. The question of whether or not \( \text{NP} \) is closed under complementation is **Unknown to Science!**

Answer **Unknown to Science!**
What is the Conventional Wisdom (is there one?)

Vote

1. Most Complexity Theorists think $NP$ is closed under complementation.
2. Most Complexity Theorists think $NP$ is not closed under complementation.
3. There is no real consensus.

Note I have done three polls on what complexity theorists think of $P \text{ vs } NP$ and related issues, so this is not guesswork on my part. Most Complexity Theorists think $NP$ is not closed under complementation.
What is the Conventional Wisdom (is there one?)

Vote

1. Most Complexity Theorists think NP is closed under complementation.
What is the Conventional Wisdom (is there one?)

Vote

1. Most Complexity Theorists think NP is closed under complementation.

2. Most Complexity Theorists think NP is not closed under complementation.
What is the Conventional Wisdom (is there one?)

Vote

1. Most Complexity Theorists think NP is closed under complementation.
2. Most Complexity Theorists think NP is not closed under complementation.
3. There is no real consensus.
What is the Conventional Wisdom (is there one?)

Vote

1. Most Complexity Theorists think $\text{NP}$ is closed under complementation.
2. Most Complexity Theorists think $\text{NP}$ is not closed under complementation.
3. There is no real consensus.

Note I have done three polls on what complexity theorists think of $\text{P vs NP}$ and related issues, so this is not guesswork on my part.
What is the Conventional Wisdom (is there one?)

Vote

1. Most Complexity Theorists think \( \text{NP} \) is closed under complementation.
2. Most Complexity Theorists think \( \text{NP} \) is not closed under complementation.
3. There is no real consensus.

Note  I have done three polls on what complexity theorists think of \( \text{P vs NP} \) and related issues, so this is not guesswork on my part. Most Complexity Theorists think \( \text{NP} \) is not closed under complementation.
Thought Experiment

Most Complexity Theorists think NP is not closed under complementation.

Contrast: Alice is all powerful, Bob is Poly Time.

▶ Alice wants to convince Bob that $\phi \in \text{SAT}$. She can! She gives Bob a satisfying assignment $\vec{b}$ (which is short) and he can check $\phi(\vec{b})$ (which is poly time).

▶ Alice wants to convince Bob that $\phi \not\in \text{SAT}$. What can she do? Give him the entire truth table. Too long!

It is thought that there is no way for Alice to do this.
Thought Experiment

Most Complexity Theorists think NP is not closed under complementation.

**Contrast**  Alice is all powerful, Bob is Poly Time.
Thought Experiment

Most Complexity Theorists think $\text{NP}$ is not closed under complementation.

**Contrast** Alice is all powerful, Bob is Poly Time.

- Alice wants to convince Bob that $\phi \in \text{SAT}$. She can! She gives Bob a satisfying assignment $\vec{b}$ (which is short) and he can check $\phi(\vec{b})$ (which is poly time).
Thought Experiment

Most Complexity Theorists think $\text{NP}$ is not closed under complementation.

**Contrast** Alice is all powerful, Bob is Poly Time.

- Alice wants to convince Bob that $\phi \in \text{SAT}$. She can! She gives Bob a satisfying assignment $\vec{b}$ (which is short) and he can check $\phi(\vec{b})$ (which is poly time).
- Alice wants to convince Bob that $\phi \notin \text{SAT}$. What can she do? Give him the entire truth table. Too long!
Thought Experiment

Most Complexity Theorists think $\text{NP}$ is not closed under complementation.

**Contrast** Alice is all powerful, Bob is Poly Time.

- Alice wants to convince Bob that $\phi \in \text{SAT}$. She can! She gives Bob a satisfying assignment $\vec{b}$ (which is short) and he can check $\phi(\vec{b})$ (which is poly time).

- Alice wants to convince Bob that $\phi \notin \text{SAT}$. What can she do? Give him the entire truth table. Too long!

It is thought that there is no way for Alice to do this.