Closure Properties of P and NP

Exposition by William Gasarch—U of MD
Closure of P

Exposition by William Gasarch—U of MD
**Thm** If $L_1 \in P$ and $L_2 \in P$ then $L_1 \cup L_2 \in P$. 

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

This algorithm takes $\sim p_1(n) + p_2(n)$, which is poly.
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 \( N \).
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 $L_1 \cap L_2 \in P$. 
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 $p_2(n)$.
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 \( N \).
**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 $\mathcal{P}$ under Intersection

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

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

$L_2 \in \mathcal{P}$ 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.
Thm If \( L_1 \in P \) and \( L_2 \in P \) then \( L_1 L_2 \in P \).
Closure of 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 $L_1 \cup L_2$ 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. (This takes $p_1(i) + p_2(n-i) \leq p_1(n) + p_2(n)$).

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 $n$. 

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 $L_1 \cup L_2$ 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. (This takes $p_1(i) + p_2(n-i) \leq p_1(n) + p_2(n)$).
3. Output N
Closure of Concatenation

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

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

$L_2 \in \mathbb{P}$ 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 $|x| = n$.) 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. (This takes $p_1(i) + p_2(n - i) \leq p_1(n) + p_2(n)$).

3. Output N

This algorithm takes $\leq (n + 1) \times (p_1(n) + p_2(n))$ which is poly.
Closure of 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 $L_1 \cup L_2$ 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. (This takes $p_1(i) + p_2(n - i) \leq p_1(n) + p_2(n)$.)
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 Complementation

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

---

The following algorithm recognizes $L$ 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 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 $L$ 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.
**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.
**Thm** If \( L \in P \) then \( L^* \in P \).

**Proof**

First let's talk about what you **should not** do.
Closure of $P$ Under $\ast$

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

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

**A contrast**

- $x \in L_1L_2$? Look at $n + 1$ ways to have $x = z_1z_2$. 

Closure of P Under *

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

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

**A contrast**

- $x \in L_1L_2$? Look at $n + 1$ ways to have $x = z_1z_2$.
- $x \in L^*$? Look at ??? ways to have $x = z_1 \cdots z_m$.  


Closure of $P$ Under $\ast$

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

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

**A contrast**

- $x \in L_1L_2$? Look at $n + 1$ ways to have $x = z_1z_2$.
- $x \in L^*$? 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 piece: $\binom{n}{2}$ ways to do this.

$\vdots$

Break string into $n$ piece: $\binom{n}{n}$ ways to do this.
Closure of $P$ Under $\ast$

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

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

**A contrast**

- $x \in L_1L_2$? Look at $n + 1$ ways to have $x = z_1z_2$.
- $x \in L^\ast$? 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.

... 
Break string into $n$ pieces: $\binom{n}{n}$ 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\}?

D: You can either choose 0 elements or choose 1 element, so $(n^0) + (n^1) + \cdots + (n^n)$.

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: $2^n = (n^0) + (n^1) + \cdots + (n^n)$.

D: Really!

B: Yes, really!
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
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** 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:** 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

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

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 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 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 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 $L$. 
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] = 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] = TRUE

output A[n]

$O(n^2)$ calls to $M$ on inputs of length $\leq n$. Runtime $\leq O(n^2 p(n))$. 
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 $A[j] \text{ AND } M(x_{j+1} \cdots x_i) = Y$ 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 of NP

Exposition by William Gasarch—U of MD
We will now show that NP is closed under $\cup$, $\cap$, $\cdot$, and $\ast$. 
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.
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 $\ast$. 
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 $\ast$.

3. Note that we did not include complementation. We’ll get to that later.
Thm If $L_1 \in \text{NP}$ and $L_2 \in \text{NP}$ then $L_1 \cup L_2 \in \text{NP}$. 
Closure of NP under Union

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

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

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

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

\[
\quad |y| = p_1(|x|) + p_2(|x|) + 1. \quad y = y_1 \$ y_2 \text{ where } |y_1| = p_1(|x|) \text{ and } |y_2| = p_2(|X|).
\]

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

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

1. $|y| = p_1(|x|) + p_2(|x|) + 1. \ y = y_1 \$ y_2$ where $|y_1| = p_1(|x|)$ and $|y_2| = p_2(|X|)$.  

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

**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.

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

- $|y| = p_1(|x|) + p_2(|x|) + 1.$ $y = y_1 \& y_2$ where $|y_1| = p_1(|x|)$ and $|y_2| = p_2(|x|)$.
- $(x, y_1) \in B_1 \lor (x, y_2) \in B_2)$

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

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

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

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

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

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

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

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

- $|y| = p_1(|x|) + p_2(|x|) + 1$. $y = y_1 \$ y_2$ where $|y_1| = p_1(|x|)$ and $|y_2| = p_2(|X|)$.

- $(x, y_1) \in B_1 \land (x, y_2) \in B_2$)
Closure of NP under Intersection

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

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

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

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

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

1. $|y| = p_1(|x|) + p_2(|x|) + 1$. $y = y_1 \| y_2$ where $|y_1| = p_1(|x|)$ and $|y_2| = p_2(|X|)$.
2. $(x, y_1) \in B_1 \land (x, y_2) \in B_2)$

**Witness:** $|y| = p_1(|x|) + p_2(|x|) + 1$ is short.
Closure of NP under Intersection

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

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

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

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

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

- $|y| = p_1(|x|) + p_2(|x|) + 1$. $y = y_1y_2$ where $|y_1| = p_1(|x|)$ and $|y_2| = p_2(|x|)$.  
- $(x, y_1) \in B_1 \land (x, y_2) \in B_2)$

Witness: $|y| = p_1(|x|) + p_2(|x|) + 1$ is short.  
Verification: $(x, y_1) \in B_1 \land (x, y_2) \in B_2)$, is quick.
Closure of Concatenation

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

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

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

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

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

- $x = x_1x_2$
- $|y_1| = p_1(|x_1|)$
- $|y_2| = p_2(|x_2|)$
- $(x_1, y_1) \in B_1$
- $(x_2, y_2) \in B_2$
Closure of NP Under $\ast$

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

\[
\{ x : (\exists z_1, \ldots, z_k, y_1, \ldots, y_k) \}
\]

- \( x = z_1 \cdots z_k \)
- \( (\forall i)[|y_i| = p(|z_i|)] \)
- \( (\forall i)[(z_i, y_i) \in B] \)
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.)

3. The question of whether or not NP is closed under complementation is 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 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.)

3. The question of whether or not 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 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.
Most Complexity Theorists think $\text{NP}$ is not closed under complementation.
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.