# ONE-WAY FUNCTIONS AND UP COMPLEXITY CLASS

***ΑΛΓΟΡΙΘΜΟΙ ΚΑΙ ΠΟΛΥΠΛΟΚΟΤΗΤΑ 2***

*ΕΠΙΜΕΛΕΙΑ :ΣΤΟΥΚΑ ΑΙΚΑΤΕΡΙΝΗ-ΠΑΝΑΓΙΩΤΑ*
***ΜΕΤΑΠΤΥΧΙΑΚΟ:****ΜΠΛΑ*

# ΣΥΜΜΕΤΡΙΚΗ ΚΡΥΠΤΟΓΡΑΦΗΣΗ

- Η Alice θέλει να στείλει ένα μήνυμα  m(plaintext) στον Bob μέσα από ένα μη έμπιστο κανάλι και να μην μπορεί να το διαβάσει άλλος εκτός από τον Bob.

# ΣΥΜΜΕΤΡΙΚΗ ΚΡΥΠΤΟΓΡΑΦΗΣΗ

- Η Alice θέλει να στείλει ένα μήνυμα m(plaintext) στον Bob μέσα από ένα μη έμπιστο κανάλι και να μην μπορεί να το διαβάσει άλλος εκτός από τον Bob.
- Έχουν ανταλλάξει από πριν ένα κλειδί μυστικό k.

# ΣΥΜΜΕΤΡΙΚΗ ΚΡΥΠΤΟΓΡΑΦΗΣΗ

- Η Alice θέλει να στείλει ένα μήνυμα m(plaintext) στον Bob μέσα από ένα μη έμπιστο κανάλι και να μην μπορεί να το διαβάσει άλλος εκτός από τον Bob.
- Έχουν ανταλλάξει από πριν ένα κλειδί μυστικό k.
- Η Alice θα χρησιμοποιήσει μία συνάρτηση κρυπτογράφησης συνήθως πολυωνυμικού χρόνου και θα στείλει κρυπτογραφημένο το m σαν ciphertext c= E(m,k).

# ΣΥΜΜΕΤΡΙΚΗ ΚΡΥΠΤΟΓΡΑΦΗΣΗ

- Η Alice θέλει να στείλει ένα μήνυμα m(plaintext) στον Bob μέσα από ένα μη έμπιστο κανάλι και να μην μπορεί να το διαβάσει άλλος εκτός από τον Bob.
- Έχουν ανταλλάξει από πριν ένα κλειδί μυστικό k.
- Η Alice θα χρησιμοποιήσει μία συνάρτηση κρυπτογράφησης συνήθως πολυωνυμικού χρόνου και θα στείλει κρυπτογραφημένο το m σαν ciphertext c= E(m,k).
- Ο Bob θα λάβει c και θα βρει m με συνάρτηση αποκρυπτογράφησης m=D(c,k).

# ΣΥΜΜΕΤΡΙΚΗ ΚΡΥΠΤΟΓΡΑΦΗΣΗ

- Η Alice θέλει να στείλει ένα μήνυμα m(plaintext) στον Bob μέσα από ένα μη έμπιστο κανάλι και να μην μπορεί να το διαβάσει άλλος εκτός από τον Bob.
- Έχουν ανταλλάξει από πριν ένα κλειδί μυστικό k.
- Η Alice θα χρησιμοποιήσει μία συνάρτηση κρυπτογράφησης συνήθως πολυωνυμικού χρόνου και θα στείλει κρυπτογραφημένο το m σαν ciphertext c= E(m,k).
- Ο Bob θα λάβει c και θα βρει m με συνάρτηση αποκρυπτογράφησης m=D(c,k).
- Οι συναρτήσεις E και D θεωρούνται γνωστές σε κάποιον που προσπαθεί να βρει m.

# ΣΥΜΜΕΤΡΙΚΗ ΚΡΥΠΤΟΓΡΑΦΗΣΗ

- Η Alice θέλει να στείλει ένα μήνυμα m(plaintext) στον Bob μέσα από ένα μη έμπιστο κανάλι και να μην μπορεί να το διαβάσει άλλος εκτός από τον Bob.
- Έχουν ανταλλάξει από πριν ένα κλειδί μυστικό k.
- Η Alice θα χρησιμοποιήσει μία συνάρτηση κρυπτογράφησης συνήθως πολυωνυμικού χρόνου και θα στείλει κρυπτογραφημένο το m σαν ciphertext c= E(m,k).
- Ο Bob θα λάβει c και θα βρει m με συνάρτηση αποκρυπτογράφησης m=D(c,k).
- Οι συναρτήσεις E και D θεωρούνται γνωστές σε κάποιον που προσπαθεί να βρει m.
- Η ασφάλεια έγκειται στο μυστικό κλειδί k που είναι γνωστό μόνο στην Alice και τον Bob.

# ONE TIME PAD

- Plaintext $m=(m_0,m_2,..m_{n-1})$ $m_i \epsilon \{o,1\}$
- Key= $k=(ko,k_1,\ldots,k_{n-1})$ $k_i \epsilon\{0,1\}$
- Ciphertext $c=(c_0,\ldots..,c_{n-1})$ $c_i \epsilon \{0,1\}$
- Encryption : $c_i=XOR(m_i,k_i)= m_i + k_i \mod 2$
- Decryption $m_i=XOR(c_i,k_i)$
- Αν ισχύει $Pr(k_i =0)=Pr(k_i=1)=1/2$ τότε έχουμε ότι $Pr(C=c|M=m_i)=Pr(C=c|M=m_j)$ για κάθε $m_i , m_j \epsilon M$ και για κάθε $c \epsilon C$ ( τέλεια μυστικότητα κατά Shannon)

# ONE TIME PAD

- Plaintext $m=(m_0,m_2,..m_{n-1})$ $m_i \epsilon \{o,1\}$
- Key= $k=(ko,k_1,\ldots,k_{n-1})$ $k_i \epsilon\{0,1\}$
- Ciphertext $c=(c_0,\ldots..,c_{n-1})$ $c_i \epsilon \{0,1\}$
- Encryption : $c_i=XOR(m_i,k_i)= m_i + k_i \text{ mod}2$
- Decryption $m_i=XOR(c_i,k_i)$
- Αν ισχύει $Pr(k_i =0)=Pr(k_i=1)=1/2$ τότε έχουμε ότι $Pr(C=c|M=m_i)=Pr(C=c|M=m_j)$ για κάθε $m_i$ ,$m_j \epsilon M$ και για κάθε $c \epsilon C$ ( τέλεια μυστικότητα κατά Shannon)
- Δηλαδή ο επιτιθέμενος δεν παίρνει καμία πληροφορία από $c_i$ για $m_i$ ,αφού το συγκεκριμένο $c_i$ μπορεί να είναι ciphertext οποιουδήποτε plaintext αν έχει χρησιμοποιηθεί το ανάλογο κλειδί.

# ONE TIME PAD

- Plaintext $m=(m_0,m_2,..m_{n-1})$ $m_i \epsilon \{o,1\}$
- Key= $k=(ko,k_1,\ldots,k_{n-1})$ $k_i \epsilon\{0,1\}$
- Ciphertext $c=(c_0,\ldots..,c_{n-1})$ $c_i \epsilon \{0,1\}$
- Encryption : $c_i=XOR(m_i,k_i)= m_i + k_i \text{ mod}2$
- Decryption $m_i=XOR(c_i,k_i)$
- Αν ισχύει $Pr(k_i =0)=Pr(k_i=1)=1/2$ τότε έχουμε ότι $Pr(C=c|M=m_i)=Pr(C=c|M=m_j)$ για κάθε $m_i$ ,$m_j \epsilon M$ και για κάθε $c \epsilon C$ ( τέλεια μυστικότητα κατά Shannon)
- Δηλαδή ο επιτιθέμενος δεν παίρνει καμία πληροφορία από $c_i$ για $m_i$ ,αφού το συγκεκριμένο $c_i$ μπορεί να είναι ciphertext οποιουδήποτε plaintext αν έχει χρησιμοποιηθεί το ανάλογο κλειδί.
- Προσοχή: αυτό ισχύει αν το κλειδί χρησιμοποιηθεί μία φορά.

# ONE TIME PAD

- Plaintext $m=(m_0,m_2,..m_{n-1})$ $m_i \epsilon \{o,1\}$
- Key= $k=(ko,k_1,\ldots,k_{n-1})$ $k_i \epsilon\{0,1\}$
- Ciphertext $c=(c_0,\ldots..,c_{n-1})$ $c_i \epsilon \{0,1\}$
- Encryption : $c_i=XOR(m_i,k_i)= m_i +k_i \text{ mod}2$
- Decryption $m_i=XOR(c_i,k_i)$
- Αν ισχύει $Pr(k_i =0)=Pr(k_i=1)=1/2$ τότε έχουμε ότι $Pr(C=c|M=m_i)=Pr(C=c|M=m_j)$ για κάθε $m_i$ ,$m_j \epsilon M$ και για κάθε $c \epsilon C$ ( τέλεια μυστικότητα κατά Shannon)
- Δηλαδή ο επιτιθέμενος δεν παίρνει καμία πληροφορία από $c_i$ για $m_i$ ,αφού το συγκεκριμένο $c_i$ μπορεί να είναι ciphertext οποιουδήποτε plaintext αν έχει χρησιμοποιηθεί το ανάλογο κλειδί.
- Προσοχή: αυτό ισχύει αν το κλειδί χρησιμοποιηθεί μία φορά.
- Αν και το σύστημα έχει τέλεια ασφάλεια το αρνητικό είναι ότι απαιτείται κλειδί όσο με το μήκος του plaintext ,οπότε αφού μπορούν να ανταλλάξουν Alice και Bob τόσο μεγάλο κλειδί γιατί να μην ανταλλάξουν και plaintext ?

# PUBLIC KEY CRYPTOGRAPHY

- Στην περίπτωση που η Alice και ο Bob δεν έχουν κάποιο μυστικό κοινό κλειδί από πριν μπορεί να ακολουθηθεί η παρακάτω διαδικασία :1)Ο Bob παράγει ένα ζευγάρι $(p_b,s_b)$ ,δημοσιεύει $p_b$ ,το οποίο είναι το public key του και κρατάει μυστικό το $s_b$ ,το οποίο είναι το private key του και το ξέρει μόνο εκείνος.
  2)Η Alice του στέλνει το $c=E(p_b,m)$,που είναι το κρυπτογράφημα του m
  3) Ο Bob παίρνει το $m =D(s_b,c)$
  4) Η ασφάλεια έγκειται στο γεγονός ότι να υπολογιστεί το $s_b$ από $p_b$ είναι υπολογιστικά δύσκολο ,όπως και ο υπολογισμός του m από το c, χωρίς το $s_b$.

# PUBLIC KEY CRYPTOGRAPHY

- Στην περίπτωση που η Alice και ο Bob δεν έχουν κάποιο μυστικό κοινό κλειδί από πριν μπορεί να ακολουθηθεί η παρακάτω διαδικασία :1)Ο Bob παράγει ένα ζευγάρι $(p_b,s_b)$ ,δημοσιεύει $p_b$ ,το οποίο είναι το public key του και κρατάει μυστικό το $s_b$ ,το οποίο είναι το private key του και το ξέρει μόνο εκείνος.
  2)Η Alice του στέλνει το $c=E(p_b,m)$,που είναι το κρυπτογράφημα του m
  3) Ο Bob παίρνει το $m =D(s_b,c)$
  4) Η ασφάλεια έγκειται στο γεγονός ότι να υπολογιστεί το $s_b$ από $p_b$ είναι υπολογιστικά δύσκολο ,όπως και ο υπολογισμός του m από το c, χωρίς το $s_b$.
- Το public key cryptosystem ανήκει στο FNP(Αν μου δώσουν (m,c), μπορώ σε πολυωνυμικό ντετερμινιστικό χρόνο να ελέγξω αν $c=E(p_b,m)$.

# PUBLIC KEY CRYPTOGRAPHY

- Στην περίπτωση που η Alice και ο Bob δεν έχουν κάποιο μυστικό κοινό κλειδί από πριν μπορεί να ακολουθηθεί η παρακάτω διαδικασία :1)Ο Bob παράγει ένα ζευγάρι $(p_b,s_b)$ ,δημοσιεύει $p_b$ ,το οποίο είναι το public key του και κρατάει μυστικό το $s_b$ ,το οποίο είναι το private key του και το ξέρει μόνο εκείνος.
  2)Η Alice του στέλνει το $c=E(p_b,m)$,που είναι το κρυπτογράφημα του m
  3) Ο Bob παίρνει το $m =D(s_b,c)$
  4) Η ασφάλεια έγκειται στο γεγονός ότι να υπολογιστεί το $s_b$ από $p_b$ είναι υπολογιστικά δύσκολο ,όπως και ο υπολογισμός του m από το c, χωρίς το $s_b$.
- Το public key cryptosystem ανήκει στο FNP(Αν μου δώσουν (m,c), μπορώ σε πολυωνυμικό ντετερμινιστικό χρόνο να ελέγξω αν $c=E(p_b,m)$.
- Επιθυμούμε να μην ανήκει στο FP.

# ONE WAY FUNCTIONS

- Ορισμός : one way συνάρτησης:1) Η f είναι μία συνάρτηση από strings σε strings.
- 2)Είναι 1-1 και για όλα τα $x \in \Sigma^*$ $|x|^{1/k} \leq |f(x)| \leq |x|^k$ για κάποιο $k>0$
- 3)$f \in FP$.
- 4)$f^{-1}$ δεν ανήκει στο FP αλλά στο FNP.
- Δεν γνωρίζουμε αν υπάρχει one way function.

# ΠΑΡΑΔΕΙΓΜΑΤΑ

- ***Fmult***(p,C(p),q,C(q))=p·q, n=p·q
- Εώς τώρα δεν έχει βρεθεί αποδοτικός αλγόριθμος εύρεσης παραγόντων του n αν p,q μεγάλοι πρώτοι.
- Ένας αλγόριθμος που επιλύει αυτό το πρόβλημα είναι η μέθοδος P του Pollard,που στηρίζεται στην ιδέα ότι αν $x_i$ και $x_j$ στοιχεία του Zn διαφορετικά μεταξύ τους και $gcd(x_i-x_j,n)>1$ τότε $gcd(x_i-x_j,n)=p$ ή $gcd(x_i-x_j,n)=q$.

# ΠΑΡΑΔΕΙΓΜΑΤΑ

- ***Fmult***(p,C(p),q,C(q))=p·q, n=p·q
- Εώς τώρα δεν έχει βρεθεί αποδοτικός αλγόριθμος εύρεσης παραγόντων του n αν p,q μεγάλοι πρώτοι.
- Ένας αλγόριθμος που επιλύει αυτό το πρόβλημα είναι η μέθοδος P του Pollard,που στηρίζεται στην ιδέα ότι αν $x_i$ και $x_j$ στοιχεία του Zn διαφορετικά μεταξύ τους και $\gcd(x_i-x_j,n)>1$ τότε $\gcd(x_i-x_j,n)=p$ ή $\gcd(x_i-x_j,n)=q$.
- ***Fexp***(p ,C(p),r,x)=(p, C(p),$r^x \bmod p$)
- Το πρόβλημα του διακριτού λογαρίθμου θεωρείται δύσκολο σε υποομάδα του Zp* τάξης μεγάλου πρώτου αριθμού .π.χ αν p=2·q+1 μια τέτοια υποομάδα είναι $<g^{(p-1)/q}>$ τάξης q.

# RSA ENCRYPTION

- Frsa (x,e,p,C(p),q,C(q))=($x^e$ modp·q ,p·q,e)
- 1)Εύρεση πρώτων p,q μεγάλου μήκους(έλεγχος αν είναι πρώτος με μέθοδο Miller –Rabin)
- 2)Υπολογίζουμε n=p·q και φ(n)=(p-1)·(q-1)
- 3)Βρίσκουμε e ϵ U(Zφ(n)): gcd(e,φ(n))=1
- 4)Υπολογίζουμε d τω e·d=1 mod φ(n) με επεκταμένο ευκλείδειο αλγόριθμο που είναι αποδοτικός.
- Public key :e,n
- Private key :d
- Encryption : enc(m)=$m^e$ mod n,όπου m ανήκει στο Zn
- Decryption:dec(c)=$c^d$ mod n
- Παρατηρούμε ότι για κάθε m στο Zn έχουμε ορθότητα.

# RSA DECRYPTION

- 1) m ϵ U(Zn),δηλαδή gcd(m,n)=1 .Τότε $m^{e\cdot d}$ modn=$m^{1+k\varphi(n)}$mod n=m mod n=m,αφού ισχύει $m^{\varphi(n)}$ modn=1modn από θεώρημα Euler
- 2)p διαιρεί m και q δεν διαιρεί m. Τότε ισχύει m≡ 0 modp ,άρα m $^{e\cdot d}$ mod p=m mod p και m $^{e\cdot d}$ mod q= $(m^{(q-1)\cdot k})^{p-1}$ ·m mod q =m mod q ,από θεώρημα Fermat,αφού gcd (q,m)=1.Άρα $m^{e\cdot d}$ mod n=m mod n=m.
- 3) p, q διαιρεί m,άρα και n διαιρεί m .Τότε m $^{e\cdot d}$ modn=m mod n =m,αφού m ≡0modn.

# RSA DECRYPTION ΚΑΙ ΣΧΕΣΗ ΜΕ ΑΛΛΑ ΔΥΣΚΟΛΑ ΠΡΟΒΛΗΜΑΤΑ

- RSA – decrypt (c,e,n)≤FindSecrExp(e,n)≤φ(n)-computation ≡ factoring (n)
- RSA – decrypt (c,e,n)≤FindSecrExp(e,n) . Αυτό ισχύει γιατί αν γνωρίζουμε d μπορούμε να κάνουμε αποκρυπτογράφηση με τη συνάρτηση decr.
- FindSecrExp(e,n) ≤ φ(n)- computation . Αυτό ισχύει γιατί αν ξέρω φ(n) βρίσκω d ,που είναι ο αντίστροφος του e στην U(Zφ(n)) με επεκταμένο αλγόριθμο του Ευκλείδη.
- φ(n)- computation ≤ factoring (n) ,γιατί φ(n)=(p-1)·(q-1)
- Factoring(n) ≤ φ(n)- computation ,γιατί αν ξέρω φ(n) το σύστημα n=p · q και φ(n)=(p-1) ·(q-1) έχει δύο αγνώστους p και q.
- Αν ξέρω d τότε μπορώ με πολυωνυμικό πιθανοτικό αλγόριθμο να υπολογίσω παραγοντοποίηση . Ο αλγόριθμος στηρίζεται στην ιδέα του Miller Rabin και στο γεγονός ότι αν βρω u διάφορο του ±1 τω $k^2$=1mod n τότε gcd(k-1,n)=p ή q.

# UP ΚΑΙ Η ΣΧΕΣΗ ΤΗΣ ΜΕ ΤΗΝ ΚΡΥΠΤΟΓΡΑΦΙΑ

- Ορισμός :unambiguous ονομάζουμε την μη ντετερμινιστική μηχανή Turing με τον περιορισμό ότι για κάθε input υπάρχει το πολύ ένα μονοπάτι που οδηγεί σε YES.
- UP είναι η κλάση των γλωσσών που είναι decidable από πολυωνυμικές unambiguous μηχανές Turing.
- UP=(∃!/∀) coUP=(∀/∃!)
- Ισχύει P ≤ UP ≤ NP
- Αν ορίσουμε το RSAdecryption σαν πρόβλημα απόφασης δηλαδή να έχει input <n,e,c,kϵZn> και output YES αν dec(n,e,c)= $c^d$ mod n=m≤k και NO αλλιώς, τότε τι σχέση έχει με τη UP και την coUP?

# UP=P ΤΟΤΕ ΔΕΝ ΥΠΑΡΧΟΥΝ ONE WAY FUNCTIONS.

- Θα αποδείξουμε το αντιθετοαντίστροφο,δηλαδή θα υποθέσουμε ότι υπάρχει μια one way function f και θα βρούμε γλώσσα $L_f$ που ανήκει στη UP και δεν ανήκει στην P.

# UP=P ΤΟΤΕ ΔΕΝ ΥΠΑΡΧΟΥΝ ONE WAY FUNCTIONS.

- Θα αποδείξουμε το αντιθετοαντίστροφο,δηλαδή θα υποθέσουμε ότι υπάρχει μια one way function f και θα βρούμε γλώσσα $L_f$ που ανήκει στη UP και δεν ανήκει στην P.
- Αρχικά ορίζουμε λεξικογραφική διάταξη για να μπορούμε να συγκρίνουμε strings :0<1<00<01<10<11<000<..

# UP=P ΤΟΤΕ ΔΕΝ ΥΠΑΡΧΟΥΝ ONE WAY FUNCTIONS.

- Θα αποδείξουμε το αντιθετοαντίστροφο,δηλαδή θα υποθέσουμε ότι υπάρχει μια one way function f και θα βρούμε γλώσσα $L_f$ που ανήκει στη UP και δεν ανήκει στην P.
- Αρχικά ορίζουμε λεξικογραφική διάταξη για να μπορούμε να συγκρίνουμε strings :0<1<00<01<10<11<000<..
- $L_f$={(x,y):υπάρχει z τω f(z)=y και z≤x} Ξέρουμε ότι $z \leq |y|^k$,αφού f είναι one way function.

# UP=P ΤΟΤΕ ΔΕΝ ΥΠΑΡΧΟΥΝ ONE WAY FUNCTIONS.

- Θα αποδείξουμε το αντιθετοαντίστροφο,δηλαδή θα υποθέσουμε ότι υπάρχει μια one way function f και θα βρούμε γλώσσα $L_f$ που ανήκει στη UP και δεν ανήκει στην P.
- Αρχικά ορίζουμε λεξικογραφική διάταξη για να μπορούμε να συγκρίνουμε strings :0<1<00<01<10<11<000<..
- $L_f$={(x,y):υπάρχει z τω f(z)=y και z≤x} Ξέρουμε ότι z≤ | y $|^k$,,αφού f είναι one way function.
- $L_f$ ϵ UP .Φτιάχνουμε μία μη ντετερμινιστική μηχανή Turing που μαντεύει τυχαίο z μήκους ≤ | y $|^k$, και ελέγχουμε αν y=f(z).Αν δεν ισχύει εκτυπώνει NO ,αν ισχύει ελέγχει αν z≤x και αναλόγως εκτυπώνει YES ή NO .Αυτή η μηχανή είναι unambiguous γιατί f είναι 1-1.

# UP=P ΤΟΤΕ ΔΕΝ ΥΠΑΡΧΟΥΝ ONE WAY FUNCTIONS.

- Θα αποδείξουμε το αντιθετοαντίστροφο,δηλαδή θα υποθέσουμε ότι υπάρχει μια one way function f και θα βρούμε γλώσσα $L_f$ που ανήκει στη UP και δεν ανήκει στην P.
- Αρχικά ορίζουμε λεξικογραφική διάταξη για να μπορούμε να συγκρίνουμε strings :0<1<00<01<10<11<000<..
- $L_f$={(x,y):υπάρχει z τω f(z)=y και z≤x} Ξέρουμε ότι z≤ | y |$^k$,,αφού f είναι one way function.
- $L_f$ ϵ UP .Φτιάχνουμε μία μη ντετερμινιστική μηχανή Turing που μαντεύει τυχαίο z μήκους ≤ | y |$^k$, και ελέγχουμε αν y=f(z).Αν δεν ισχύει εκτυπώνει NO ,αν ισχύει ελέγχει αν z≤x και αναλόγως εκτυπώνει YES ή NO .Αυτή η μηχανή είναι unambiguous γιατί f είναι 1-1.
- $L_f$ δεν ανήκει στο P.Για να το αποδείξουμε θα υποθέσουμε ότι ανήκει στο P και θα εφαρμόσουμε με δυαδική αναζήτηση αλγόριθμο για να αντιστρέψουμε f.

# ΑΝ ΔΕΝ ΥΠΑΡΧΟΥΝ ONE WAY FUNCTIONS ΤΟΤΕ UP=P

- Θα αποδείξουμε πάλι το αντιθετοαντίστροφο .

# ΑΝ ΔΕΝ ΥΠΑΡΧΟΥΝ ONE WAY FUNCTIONS ΤΟΤΕ UP=P

- Θα αποδείξουμε πάλι το αντιθετοαντίστροφο .
- Έστω ότι UP διάφορο του P.Τότε υπάρχει L Є UP/P και U η unambiguous μηχανή Turing που την κάνει decide.Τότε ορίζουμε την one way $f_u$ τέτοια ώστε αν x είναι η αναπαράσταση ενός υπολογισμού της U για input y που καταλήγει σε YES τότε $f_u(x)=1y$, αλλιώς σε κάθε άλλη περίπτωση $f_u(x)=0x$.

# ΑΝ ΔΕΝ ΥΠΑΡΧΟΥΝ ONE WAY FUNCTIONS ΤΟΤΕ UP=P

- Θα αποδείξουμε πάλι το αντιθετοαντίστροφο .
- Έστω ότι UP διάφορο του P.Τότε υπάρχει L Є UP/P και U η unambiguous μηχανή Turing που την κάνει decide.Τότε ορίζουμε την one way $f_u$ τέτοια ώστε αν x είναι η αναπαράσταση ενός υπολογισμού της U για input y που καταλήγει σε YES τότε $f_u(x)=1y$, αλλιώς σε κάθε άλλη περίπτωση $f_u(x)=0x$.
- $f_u$ είναι 1-1

# ΑΝ ΔΕΝ ΥΠΑΡΧΟΥΝ ONE WAY FUNCTIONS ΤΟΤΕ UP=P

- Θα αποδείξουμε πάλι το αντιθετοαντίστροφο .
- Έστω ότι UP διάφορο του P.Τότε υπάρχει L Є UP/P και U η unambiguous μηχανή Turing που την κάνει decide.Τότε ορίζουμε την one way $f_u$ τέτοια ώστε αν x είναι η αναπαράσταση ενός υπολογισμού της U για input y που καταλήγει σε YES τότε $f_u(x)=1y$, αλλιώς σε κάθε άλλη περίπτωση $f_u(x)=0x$.
- $f_u$ είναι 1-1
- $f_u$ єFP, αφού το x περιέχει και την αναπαράσταση του y άρα το y μπορεί να ανακτηθεί γρήγορα από x.Επίσης x και y είναι πολυωνυμικά σχετιζόμενα αφού κάθε μονοπάτι U είναι πολυωνυμικό.

# ΑΝ ΔΕΝ ΥΠΑΡΧΟΥΝ ONE WAY FUNCTIONS ΤΟΤΕ UP=P

- Θα αποδείξουμε πάλι το αντιθετοαντίστροφο .
- Έστω ότι UP διάφορο του P.Τότε υπάρχει L Є UP/P και U η unambiguous μηχανή Turing που την κάνει decide.Τότε ορίζουμε την one way $f_u$ τέτοια ώστε αν x είναι η αναπαράσταση ενός υπολογισμού της U για input y που καταλήγει σε YES τότε $f_u(x)=1y$, αλλιώς σε κάθε άλλη περίπτωση $f_u(x)=0x$.
- $f_u$ είναι 1-1
- $f_u$ єFP, αφού το x περιέχει και την αναπαράσταση του y άρα το y μπορεί να ανακτηθεί γρήγορα από x.Επίσης x και y είναι πολυωνυμικά σχετιζόμενα αφού κάθε μονοπάτι U είναι πολυωνυμικό.
- $f^{-1}$ δεν ανήκει στη FP,γιατί αλλιώς για input y θα βρίσκαμε με πολυωνυμικό ντετερμινιστικό τρόπο πρότυπο του 1y ή του 0y μέσω της fu και θα κάναμε decide L με ντεντερμινιστική πολυωνυμική μηχανή Turing ,άτοπο(L δεν ανήκει στο P).

# ΆΛΛΟΣ ΟΡΙΣΜΟΣ ΓΙΑ ONE WAY FUNCTIONS

- Μια συνάρτηση f:{0,1}* ->{0,1}* πολυωνυμική είναι one way function αν για κάθε probabilistic polynomial time αλγόριθμο A υπάρχει μία negligible function e:N->[0,1] ,έτσι ώστε για κάθε n να ισχύει Pr(A(y)=x' ώστε f(x')=y] <e(n),όπου x $\epsilon_R\{0,1\}^n$ και y=f(x).