Semester 1 **Exercises and Solutions for Week 6** 2004

Three-time pads

1. Given $\Delta_1 = XY \ominus ZW$ and $\Delta_2 = XY \ominus QR$, the matrix of relative probabilities $P(XY|\Delta_1, \Delta_2)$ can be computed with this function.

```
function RelativeDifferentialProbabilities(D1,D2,PT)
    // Given 2-character strings D1 and D2 representing
    // XY-ZW and XY-QR, return the 26x26 matrix of
    // probabilities, (P(XY)P(ZW)P(QR)), scaled to a
    // probability function on the set {A,..,Z}^2.
    // PT is a sample plaintext for use in determining
    // the 2-character frequency distribution of English.
    assert #D1 eq 2 and #D2 eq 2;
    AZ := {@ CodeToString(64+i) : i in [1..26] @};
    r1 := Index(AZ,D1[1]); s1 := Index(AZ,D1[2]);
    r2 := Index(AZ,D2[1]); s2 := Index(AZ,D2[2]);
    FDD := RealField()!0;
    DD2 := MatrixAlgebra(RealField(),26)!0;
    F2D := DigraphFrequencyDistribution(PT);
    for i1, j1 in [1..26] do
        i2 := ((i1-r1) mod 26) + 1;
        j2 := ((j1-s1) mod 26) + 1;
        i3 := ((i1-r2) mod 26) + 1;
        j3 := ((j1-s2) mod 26) + 1;
        F3 := F2D[i1,j1] * F2D[i2,j2] * F2D[i3,j3];
        DD2[i1,j1] +:= F3;
        FDD +:= F3;
    end for;
    return (1/FDD)*DD2;
end function;
```

Apply this function to find the plaintexts PT_1, PT_2, and PT_3, where

$$\Delta_1 = PT_1 \ominus PT_2 = \texttt{AHXCOYFBAMKUE}$$
$$\Delta_2 = PT_1 \ominus PT_3 = \texttt{XHXRGEUHPRAHN}$$

You may use *blackcat.txt* as the sample plaintext.

Solution Using the above function, the following lines of input:

```
AZ := &*[ CodeToString(64+i) : i in [1..26] ];
PT := StripEncoding(Read("blackcat.txt"));
Diff1 := "AHXCOYFBAMKUE";
Diff2 := "XHXRGEUHPRAHN";
eps := 0.15;
for k in [1..12] do
    D1 := Diff1[[k,k+1]]; D2 := Diff2[[k,k+1]];
    FD := RelativeDifferentialProbabilities(D1,D2,PT);
    r1 := Index(AZ,D1[1]); s1 := Index(AZ,D1[2]);
    r2 := Index(AZ,D2[1]); s2 := Index(AZ,D2[2]);
    print "k:", k;
    for i1, j1 in [1..26] do
        if FD[i1,j1] ge eps then
            i2 := (i1-r1) mod 26 + 1; j2 := (j1-s1) mod 26 + 1;
            i3 := (i1-r2) mod 26 + 1; j3 := (j1-s2) mod 26 + 1;
            printf "[%o,%o,%o] (%o)\n",
                AZ[[i1,j1]], AZ[[i2,j2]], AZ[[i3,j3]], FD[i1,j1];
        end if;
    end for;
end for;
```

generates the output:

```
k: 1
[TO,TH,WH] (0.2716182630)
k: 2
k: 3
[BE,EC,EN] (0.15547486679)
k: 4
k: 5
[OR,AT,IN] (0.5823337200)
k: 6
[LY,NT,HE] (0.4247418881)
k: 7
[NO,IN,TH] (0.6867756526)
k: 8
[OT,NT,HE] (0.24752628407)
k: 9
[TT,TH,EC] (0.2612827757)
k: 10
[ED,ST,ND] (0.2883526922)
[TO,HE,CO] (0.19345495180)
k: 11
[OM,ES,OF] (0.17007055156)
[OU,EA,ON] (0.2627616522)
k: 12
```

By varying the value of `eps` (epsilon), we piece together likely matching strings for the original plaintexts.

PT_1 :	T	O	B	E	O	R					
PT_2 :	T	H	E	C	A	T					
PT_3 :	W	H	E	N	I	N					
PT_1 :						L	Y	I	T	T	O
PT_2 :						N	T	H	T	H	E
PT_3 :						H	E	B	E	C	O
PT_1 :							N	O	T	T	O
PT_2 :							I	N	T	H	E
PT_3 :							T	H	E	C	O

We conjecture that the correct plaintexts are `TOBEORNOTTO**`, `THECATINTHE**`, and `WHENINTHECO**`, leaving the final characters to pure guesswork.