NUEVO ALGORITMO PARA EL CÁLCULO DE LA MATRIZ INVERSA.

Pablo L. Freyre y Nelson Díaz.
Universidad de la Habana. Cuba.

RESUMEN
En este trabajo se presenta un nuevo algoritmo que permite dada una matriz inversible, definida en el campo Z_2 (matriz booleana), obtener su inversa. El algoritmo transforma la matriz escrita en su forma clásica en polinomios y elementos del campo Z_2, a partir de los cuales se calcula la inversa de la matriz. La ventaja de este algoritmo, con respecto a otros, es que permite resolver el sistema de ecuaciones $X = YA^{-1}$ con Y y A conocidos, donde $A \in GL_n(Z_2)^*$ y X, $Y \in (Z_2)^n$, sin necesidad de calcular explícitamente la matriz inversa, sino utilizando los polinomios asociados a la matriz A. El algoritmo está implementado en lenguaje Mathematica (Versión 4.0).

ABSTRACT
In this work we present a new algorithm that allows, given an invertible matrix defined over Z_2 (boolean matrix), obtain their inverse. The algorithm transforms the matrix written in its classical form into polynomials and elements the field Z_2, starting from which the inverse of the matrix is calculated. The advantage of this algorithm is that it allows to solve equations system in the form $X = YA^{-1}$, with known Y and A, where $A \in GL_n(Z_2)$ and Y, $X \in (Z_2)^n$, with no need to calculate explicitely the inverted matrix, but using instead the inverted polynomials associated with to matrix A. The algorithm is implemented on Mathematica language (Version 4.0).

Key Words: linear ecuaquations systems, modular Polinomials,

MSC 65F18.

1 INTRODUCCIÓN.

En muchas aplicaciones prácticas se presenta la necesidad de resolver un sistema de ecuaciones lineales cuya matriz es definida sobre un campo finito. En particular en la estadística matemática aparecen tales sistemas con matrices definidas en el campo Z_2 conocidas como matrices booleanas. Reviste por tanto interés práctico el desarrollo de algoritmos eficientes para tales fines.

El objetivo de este trabajo es exponer la implementación, en lenguaje Mathematica (Versión 4.0), de un nuevo algoritmo que permite dado una matriz invertible, definida en el campo Z_2, obtener su inversa. El algoritmo resuelve el sistema de ecuaciones Y = XA cuando se conoce Y y A, donde $A \in GL_n(Z_2)$ y X , $Y \in (Z_2)^n$, sin necesidad de calcular explícitamente la matriz inversa. Las operaciones que se realizan en el algoritmo son multiplicación de polinomios módulos polinomios primitivos más operaciones definidas en el campo Z_2.

En el trabajo de Freyre y Díaz (2006) se presenta este algoritmo en pseudo código con su fundamentación para el caso general en que las matrices se encuentran definidas en un campo finito arbitrario. Las operaciones a realizar en el mismo son el cálculo de la inversa de polinomios y la multiplicación de polinomios módulos polinomios primitivos para polinomios definidos en el campo.

* Cuando $A \in GL_n(Z_2)$ decimos que A es una matriz booleana no singular.

En Menezes, Van Oorschot y Varston (1996) se reportan algoritmos específicos para la multiplicación y cálculo de la inversa de polinomios módulo polinomios primitivos definidos en el campo Z_2. La utilización de estos algoritmos combinado con el uso de polinomios primitivos con un número mínimo de coeficientes hace que la velocidad para el cálculo de la matriz inversa y solución del sistema de ecuaciones antes mencionado aumente.

En el presente trabajo se muestran tres ejemplos en los cuales se expone primero el algoritmo para el calculo de la matriz inversa y posteriormente la derivación de este para resolver el sistema de ecuaciones $X = YA^{-1}$.

Desarrollo.
Los algoritmos que a continuación se citan, se encuentran implementados en Lenguaje Mathematica (Versión 4.0) y en ambos casos tenemos que:
n – Es el grado de la matriz.

lpp – Es la lista de polinomios primitivos a utilizarse en el algoritmo, que van en grado descendente desde n hasta 1. Los n polinomios primitivos seleccionados se calculan a priori y pueden ser cualesquiera.

En Peterson W.W. y Weldon J. E. (1972), Golomb W. S. (1982) y Lidl R. y Niederreiter H. (1994) se encuentran tablas con polinomios primitivos definidos en el campo Z_2, de las cuales se pueden seleccionar los polinomios primitivos para ambos algoritmos. En las referencias anteriores además de las tablas se encuentra información sobre los polinomios primitivos.

2.- ALGORITMO PARA EL CÁLCULO DE LA MATRIZ INVERSA.

El algoritmo para el cálculo de la matriz inversa consta de dos pasos:

- Algoritmo que expresa la matriz a través de polinomios y elementos del campo Z_2.
- Algoritmo para el cálculo de la matriz inversa.

2.1 algoritmo que expresa la matriz a través de polinomios y elementos de z_2.

Programación del algoritmo:
m – Es la matriz a la que se le va a calcular su inversa.
vbc – Son los valores de los elementos del campo Z_2 y de los coeficientes de los polinomios definidos en el campo Z_2 asociados a la matriz m.

```
Clear[Creavbc]
Creavbc[n_, i_, v_, vbc_, lpp_] :=
Block[{x = 0, t, z, y = PadLeft[{}, n]},
  z = lpp[[i]][Take[vbc[[i]], {i, n}]];
  t = lpp[[i]][Take[y, {i, n}]];
  If[TrueQ[z[[1]] != t[[1]]], t = lpp[[i]][Take[v, {i, n}]]*(z^-1);
    If[TrueQ[t == 0], t = lpp[[i]][Take[y, {i, n}]]];
    x = lpp[[1]][Take[v, {1, i - 1}]] –
        lpp[[1]][Take[vbc[[i]], {1, i - 1}]]*lpp[[1]][t[[1]][[{1}]]];
  ];
  If[TrueQ[x == 0], x = lpp[[1]][PadLeft[{}, n]]];
  Return[Join[Take[x[[1]], {1, i - 1}], t[[1]]]];
]

Clear[Fpolinomial]
Fpolinomial[n_, m_, lpp_] :=
Block[{vbc = {}, i, j, vec, y = PadLeft[{}, n]},
  For[j = 1, j <= n, j++,
    i = 0; vec = m[[j]];
    While[(i = i + 1) < j,
```

```
      vec = Creavbc[n, i, vec, vbc, lpp];
      If[Take[vec, {i, n}] == Take[y, {i, n}], Print["Verifique si la matriz es
                                                    inversible."]; Return[y]]
    ];
    If[Take[vec, {i, n}] == Take[y, {i, n}], Print["Verifique si la matriz es inversible.
                                                    "]; Return[y]];
    AppendTo[vbc, vec]
  ];
  Return[vbc];
]
```

ALGORITMO PARA EL CÁLCULO DE LA MATRIZ INVERSA.

Programación del algoritmo:

vbc – Son los valores de los elementos del campo Z_2 y de los coeficientes de los polinomios definidos en el campo Z_2 asociados a la matriz anterior.

m – Es la matriz inversa.

```
Clear[ILb]
ILb[n_, i_, v_, vbc_, lpp_] :=
Block[{x, t, z},
  t = lpp[[i]][Take[v, {i, n}]]*(lpp[[i]][Take[vbc[[i]], {i, n}]]^-1);
  If[TrueQ[t == 0], t = lpp[[i]][PadLeft[{}, n + 1 - i]]];
  x = lpp[[1]][Take[v, {1, i - 1}]] –
      lpp[[1]][Take[vbc[[i]], {1, i - 1}]]*lpp[[1]][t[[1]][[{1}]]];
  If[TrueQ[x == 0], x = lpp[[1]][PadLeft[{}, n]]];
  Return[Join[Take[x[[1]], {1, i - 1}], t[[1]]]];
]

Clear[Invmatriz]
Invmatriz[n_, vbc_, lpp_] :=
Block[{m = {}, v = IdentityMatrix[n], i, j, vec},
  For[j = 1, j <= n, j++,
    i = 0; vec = v[[j]];
    While[(i = i + 1) < n + 1,vec = ILb[n, i, vec, vbc, lpp]];
    AppendTo[m, vec]
  ];
  Return[m];
]
```

3 ALGORITMO PARA RESOLVER LA ECUACION X = YA^{-1} CONOCIENDO LOS VALORES DE LA MATRIZ A Y EL VECTOR Y.

Programación del algoritmo:

y – Valor del vector Y.

m – Es la matriz A.

x – Valor del vector X.

```
Clear[Resolver]
Resolver[n_,m_,y_,lpp_]:=
Block[{vbc, i, x},
   vbc=Fpolinomial[n,m,lpp];
   If[TrueQ[vbc    PadLeft[{},n]],Print["Verifique si la matriz
                                          inversible."];Return[vbc]];
   i=0;x=y;While[(i=i+1)<n+1,x=ILb[n,i,x,vbc,lpp]];
   Return[x];
]
```

En los algoritmos anteriores se han utilizado las funciones propias del lenguaje Matemática para el cálculo de la inversa de polinomios y la multiplicación de polinomios módulos polinomios primitivos, pero en una implementación específica se puede aumentar la velocidad de procesamiento utilizando los algoritmos que se reportan en Menezes, Van Oorschot y Varston (1996) para la multiplicación y cálculo de la inversa de polinomios módulo polinomios primitivos definidos en el campo Z_2.

Ejemplos:

En los tres ejemplos que se exponen a continuación para matrices invertibles de grado 16, 8 y 4 primero se calcula la matriz inversa y posteriormente dada la matriz m y el vector y se resuelve el sistema de ecuaciones $x = y\ m^{-1}$ sin necesidad de calcular explícitamente la matriz inversa. Los polinomios primitivos utilizados en los ejemplos fueron tomados de la tabla C.2 del Peterson W.W. y Weldon J. E. (1972).

1.- Cálculo de la matriz inversa de la matriz m que se expresa a continuación:

$$\begin{pmatrix}
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\
1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\
1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0
\end{pmatrix}$$

Los polinomios primitivos son: $1 + x + x^3 + x^{12} + x^{16}$; $1 + x + x^{15}$; $1 + x + x^6 + x^{10} + x^{14}$;
$1 + x + x^3 + x^4 + x^{13}$; $1 + x + x^4 + x^6 + x^{12}$; $1 + x^2 + x^{11}$; $1 + x^3 + x^{10}$;
$1 + x^4 + x^9$; $1 + x^2 + x^3 + x^4 + x^8$; $1 + x^3 + x^7$; $1 + x + x^6$; $1 + x^2 + x^5$;
$1 + x + x^4$; $1 + x + x^3$; $1 + x + x^2$; $1 + x$.

```
<< "Algebra`FiniteFields`"
lpp ={GF[2,{1,1,0,1,0,0,0,0,0,0,0,0,1,0,0,0,1}],
    GF[2,{1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1}],
    GF[2,{1,1,0,0,0,0,0,1,0,0,0,1,0,0,0,1}], GF[2,{1,1,0,1,
        1,0,0,0,0,0,0,0,0,1}], GF[2,{1,1,0,0,1,0,1,0,0,
        0,0,0,1}], GF[2,{1,0,1,0,0,0,0,0,0,0,0,1}],
    GF[2,{1,0,0,1,0,0,0,0,0,0,1}], GF[2,{1,0,0,0,1,0,0,0,
        0,1}], GF[2,{1,0,1,1,1,0,0,0,1}],
```

```
    GF[2,{1,0,0,1,0,0,0,1}], GF[2,{1,1,0,0,0,0,1}],
    GF[2,{1,0,1,0,0,1}], GF[2,{1,1,0,0,1}],
    GF[2,{1,1,0,1}], GF[2,{1,1,1}], GF[2,{1,1}]}
m = {{0,0,0,0,1,1,1,1,1,1,1,0,0,1,0,1},
   {1,0,0,0,1,1,1,0,0,1,1,1,0,0,1,0},
   {0,0,1,0,0,1,0,0,1,0,1,0,1,1,0,1},
   {1,1,0,0,0,1,0,0,1,0,0,0,1,1,0,0},
   {0,1,1,0,1,0,1,0,0,0,1,1,0,1,0,0},
   {1,1,1,1,0,0,1,0,1,0,0,1,1,1,1,1},
   {0,0,0,1,0,0,0,1,0,0,0,0,1,1,0,1},
   {0,0,0,1,0,1,0,0,1,0,0,0,0,1,0,1},
   {1,0,0,1,1,1,1,0,0,1,1,1,0,1,1,1},
   {1,0,1,1,1,0,0,0,0,1,1,0,0,1,1,1},
   {1,1,0,0,0,1,1,0,1,1,1,1,0,1,0,0},
   {1,1,1,0,0,1,1,0,1,0,1,0,1,1,1,1},
   {0,0,1,1,0,0,0,1,0,0,1,1,0,1,1,0},
   {0,0,0,1,1,0,0,1,1,0,1,0,1,0,0,0},
   {0,0,0,0,0,1,1,0,0,1,0,1,0,0,1,1},
   {1,1,1,1,1,1,1,1,1,0,0,0,1,0,1,1,0}}

vbc = Fpolinomial[16, m, lpp]
Invmatriz[16, vbc, lpp]
MatrixForm[%]
```

$$\begin{pmatrix}
1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\
1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\
1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\
1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0
\end{pmatrix}$$

Resolución del sistema x = y m^{-1} utilizando el algoritmo # 2 y teniendo como datos de entrada: la matriz m expuesta al comienzo del ejemplo y el vector y:

```
y = {0,1,1,0,1,0,0,1,0,0,1,0,0,1,1,1}
x = Resolver[16, m, y, lpp]
x = {0,0,0,0,1,1,0,0,0,0,1,1,1,1,1,1}
```

2.- Cálculo de la matriz inversa de la matriz m que se expresa a continuación:

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

Los polinomios primitivos son: $1 + x^2 + x^3 + x^4 + x^8$; $1 + x^3 + x^7$; $1 + x + x^6$; $1 + x^2 + x^5$; $1 + x + x^4$; $1 + x + x^3$; $1 + x + x^2$; $1 + x$.

```
<< "Algebra`FiniteFields`"
lpp ={GF[2,{1,0,1,1,1,0,0,0,1}], GF[2,{1,0,0,1,0,0,0,1}],
     GF[2,{1,1,0,0,0,0,1}], GF[2,{1,0,1,0,0,1}],
     GF[2,{1,1,0,0,1}], GF[2,{1,1,0,1}], GF[2,{1,1,1}],
     GF[2,{1,1}]}
m = {{1,1,0,0,1,1,0,0}, {0,0,0,0,1,0,1,1},
    {1,1,0,0,1,1,1,0}, {0,1,0,0,1,0,0,0},
    {1,0,1,0,1,1,1,0}, {0,1,1,1,1,1,1,0},
    {0,1,1,0,0,0,1,1}, {0,0,0,0,1,1,1,0}}

vbc = Fpolinomial[8, m, lpp]
Invmatriz[8, vbc, lpp]
MatrixForm[%]
```

$$\begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}$$

Resolución del sistema $x = y\ m^{-1}$ utilizando el algoritmo # 2 y teniendo como datos de entrada: la matriz m expuesta al comienzo del ejemplo y el vector y:

```
y = {1,1,0,0,1,1,0,0}
x = Resolver[16, m, y, lpp]
x = {1,0,0,0,0,0,0,0}
```

3.- Cálculo de la matriz inversa de la matriz m que se expresa a continuación:

$$\begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

Los polinomios primitivos son: $1 + x + x^4$; $1 + x + x^3$; $1 + x + x^2$; $1 + x$.

```
<< "Algebra`FiniteFields`"
lpp ={GF[2,{1,1,0,0,1}], GF[2,{1,1,0,1}], GF[2,{1,1,1}],
    GF[2,{1,1}]}
m = {{0,1,0,1},{1,1,0,1},{1,0,1,1},{1,1,1,1}}

vbc = Fpolinomial[4, m, lpp]
Invmatriz[4, vbc, lpp]
MatrixForm[%]
```

$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix}$$

Resolución del sistema $x = y\, m^{-1}$ utilizando el algoritmo # 2 y teniendo como datos de entrada: la matriz m expuesta al comienzo del ejemplo y el vector y:

```
y = {1,0,1,1}
x = Resolver[16, m, y, lpp]
x = {0,0,1,0}
```

Received April, 2006
Revised February, 2007

REFERENCES

FREYRE P. L. y DÍAZ N. (2006)."Algorithm to decide whether a matrix is singular or not. **Enviado a Journal of Symbolic Computation.**

GOLOMB W. S. (1982). **Shift Register Secuences**. Aegean Park Press. California.

MENEZES A., VAN OORSCHOT P. and VARSTONE S. (1996). **Handbook of Applied Cryptography**. CRC. Press.

LIDL R. and NIEDERREITER H. (1994). **Introduction to Finite Fields and their Applications**. Cambridge University.

PETERSON W.W. and WELDON J. E. (1972**). Error – Correcting Codes**. John Wiley and Sons, Inc. New York. 2ed.