

I. INTRODUCCIÓN.

¡Hola!, bienvenido al uso del programa Ps1. Este programa es una herramienta desarrollada para ayudarte en el estudio de 2 materias : “Lenguajes y Autómatas” y “Programación de Sistemas I”.

Inicialmente, este manual te indicará el uso de Ps1 como auxiliar didáctico tanto para la enseñanza como para el aprendizaje de los siguientes temas del curso de “Lenguajes y Autómatas” :

- Expresiones regulares.
- Reglas de Thompson.
- Algoritmo de construcción de subgrupos.
- Algoritmo de particiones.
- Gramáticas.

El uso de Ps1 como auxiliar didáctico para el curso de “Programación de Sistemas I”, se trata en otro documento.

Ps1 es un programa que te facilita la edición de expresiones regulares, y su compilación según las reglas descritas en el libro del dragón : “COMPILADORES, Principios, técnicas y herramientas”, con autores Aho, Sethi y Ullman.

Una vez que has editado y compilado una expresión regular, puedes visualizar la descomposición sintáctica que efectuó Ps1 (en forma de parejas token-lexema) para reconocer cada expresión regular que compone tu definición regular. Realmente esta característica, sólo es útil si tú eres profesor o bien, si eres alumno te servirá como ejemplo de estudio para el tema de analizadores sintácticos que se incluye en la materia de “Programación de Sistemas I” y que seguro te llamará la atención.

Recuerda que a un conjunto de expresiones regulares se le denomina una definición regular, y ésta denota a un lenguaje regular. Los tokens son secuencias de caracteres con un cierto significado semántico. Por ejemplo el token id puede tomar los lexemas siguientes en un segmento de código : iCont, iNum, sNoCon, fPromedioGrupo. Un token situado en el contexto de un lenguaje de programación, es un lenguaje cuyas cadenas cumplen con un cierto patrón. Este patrón generalmente es definido utilizando una definición regular, es decir, utilizando un conjunto de expresiones regulares.

Las definiciones regulares en su construcción siguen una serie de reglas bien específicas. Ps1 usa esta serie de reglas para compilar tu definición regular. Además Ps1 añade otras reglas que deberás respetar para lograr que tus definiciones regulares puedan ser compiladas de manera exitosa. Todo lo anterior se te explica en el capítulo II.

En el capítulo III verás cómo es generado un autómata finito no determinístico, para cada una de tus expresiones regulares que conforman la definición regular

que editaste y compilaste. Este autómata finito no determinístico AFND es construido por Ps1, usando las reglas de THOMPSON. El formato empleado por Ps1 para dibujar estos autómatas es .BMP, por lo que fácilmente podrás agregarlos a un documento Word ya sea directamente o bien, utilizando un editor de imágenes (Paint por ejemplo) donde podrás personalizar el autómata producido por Ps1.

Puedes también modificar los colores de los componentes del autómata además del tamaño del bitmap donde es dibujado el autómata.

El AFND construido empleando las reglas de THOMPSON es visualizado a colores y en blanco y negro según hayas seleccionado en la opción correspondiente. Además, Ps1 dibuja un AFND por cada expresión regular que hayas editado y compilado de manera exitosa.

Ps1 te proporciona la opción de obtener el autómata finito determinístico AFD, a partir del autómata finito no determinístico AFND –reglas de THOMPSON-. El capítulo IV describe los pasos que deberás seguir para obtener lo siguiente :

- Dibujo en un formato .BMP del autómata finito determinístico.
- Descripción paso a paso del algoritmo de construcción de subgrupos utilizado para construir dicho autómata AFD. Ps1 te da la facilidad de poder incluir texto para documentar cada etapa de este algoritmo.
- Visualización de los componentes del AFD. También puedes incluir texto para documentar el reporte de componentes.
- Tabla de transición –función move- del AFD. Se incluye el grupo de estados del AFND correspondiente a cada estado del nuevo AFD.
- Facilidad de cambiar el color del autómata.

Ya sea el texto o el dibujo logrado en cualquiera de las funciones anteriores, puedes manipularlo para insertarlo en un documento de Word, un documento HTML, etc.

El capítulo V es dedicado a explicarte la manera en que puedes obtener el autómata finito determinístico óptimo o reducido. Este AFD reducido o mínimo en cuanto al número de estados que lo componen, Ps1 lo construye usando el algoritmo de particiones sucesivas. La visualización del dibujo del AFD reducido, de los componentes del AFD reducido, de su tabla de transición, son tareas que puedes consultar en este capítulo V. Además puedes grabar en disco, el dibujo del AFD reducido para posteriormente utilizarlo en la construcción de analizadores léxicos.

Por último, puedes utilizar Ps1 para editar y compilar gramáticas. Ps1 te visualiza los componentes de la gramática que editaste y su descomposición sintáctica. También te permite derivar a la izquierda o a la derecha una sentencia que tú proporcionas. Esta característica es útil para que te familiarices en la visualización del proceso de reconocimiento de una sentencia. El capítulo VI trata lo anterior.

II. EXPRESIONES REGULARES.

En este capítulo te mostraré las reglas que debes observar para la edición y compilación de definiciones regulares. Iniciaré por indicarte cómo se traduce una definición regular hecha en clase –en el pintarrón ó en el cuaderno-, para que pueda ser reconocida como válida por Ps1. Seguiremos con las tareas de edición, almacenamiento-salvar- y compilación de una expresión regular. Después te mostraré con ejemplos los errores más comunes encontrados cuando Ps1 compila una definición regular.

2.1 REGLAS PARA CONSTRUCCIÓN DE EXPRESIONES REGULARES.

Debes de observar las siguientes reglas cuando construyas un conjunto de expresiones regulares :

1. ϵ es una expresión regular que denota al lenguaje $\{ \epsilon \}$, o sea, el conjunto que contiene solamente a la cadena vacía.
2. Si a es un símbolo perteneciente al alfabeto Σ , entonces a es una expresión regular que denota al lenguaje $\{ a \}$, lenguaje que sólo contiene a la cadena a .
3. Operaciones para expresiones regulares. Sean r y s dos expresiones regulares que denotan al lenguaje $L(r)$ y $L(s)$, respectivamente. Entonces:
 - (a) $(r) | (s)$ es una expresión regular que denota al lenguaje $L(r) \cup L(s)$. A esta operación se le conoce como *alternancia*.
 - (b) $(r) (s)$ es una expresión regular que denota al lenguaje $L(r) L(s)$. Operación *concatenación*.
 - (c) $(r)^*$ es una expresión regular que denota al lenguaje $(L(r))^*$. Operación *cerradura*.
 - (d) (r) es una expresión regular que denota al lenguaje $L(r)$.

En la siguiente tabla 2.1 se añaden otra reglas muy importantes, que debes considerar cuando edites tu conjunto de expresiones regulares.

Expresión	Descripción
c	cualquier caracter no operador
$\backslash c$	el caracter c literalmente
$"s"$	la cadena s literalmente
$.$	cualquier caracter excepto el nueva línea
$[s]$	cualquier caracter en s
$[c-c]$	Indica un rango ejo. $[A-Z]$ cualquier letra mayúscula.
$[^s]$	cualquier caracter no perteneciente a s
r^*	zero o mas r 's
r^+	uno o mas r 's
$r?$	zero o una r
$r_1 r_2$	concatenación r_1 con r_2
$r_1 r_2$	alternancia r_1 con r_2
(r)	r

Tabla 2.1 Reglas en la edición de expresiones regulares.

Para denotar una expresión regular utiliza el siguiente patrón :

- Empieza con una letra, seguida de 0 o mas letras y/o digitos y/o caracter de subrayado.
- El identificador que denota a la expresión regular lo debes de encerrar – delimitar- entre llaves.

Ejo. 2.1 Supongamos que queremos denotar el lenguaje de todas las letras mayúsculas :

{Letras} -> A | B | C | D | ... | X | Y | Z

Observa que el identificador *Letras* lo hemos encerrados entre llaves. Además, de aquí podemos concluir que la sintáxis que debes seguir para construir una expresión regular es la siguiente :

{IdExprReg} -> Expresión

Donde :

➔ es el operador de definición de expresiones regulares,

Expresión es cualquier combinación de las reglas mencionadas al inicio de esta sección y

{*IdExprReg*} es el identificador de la expresión regular.

En la práctica las alternancias como la anterior A | B | C | D | ... | X | Y | Z, no son aceptadas por Ps1, es decir Ps1 no acepta el operador ... en una expresión. Lo anterior te lleva a inferir que debes de incluir a todas las letras o símbolos – mayúsculas para el ejemplo– que formen parte de una alternancia. Claro que no es recomendable hacerlo así, Ps1 soporta la notación subrango. El ejemplo lo podrás realizar como se ilustra a continuación :

{Letras} -> [A-Z]

Ejo. 2.2 Obtener la definición regular para denotar el lenguaje representado por todos los identificadores que inicien con al menos una letra, seguidos de cualquier número de letras y/o dígitos y/o caracter de subrayado.

Letras -> A | B | ... | Z | a | b | ... | z

Digitos -> 0 | 1 | 2 | ... | 8 | 9

Sub -> _

Id -> Letras (Letras | Digitos | Sub) *

La anterior es la definición regular que denota el lenguaje del token Id. Debes utilizar las reglas para identificadores de expresiones regulares y para rangos o subrangos. La definición que deberás introducir a la computadora será :

{Letras} -> [A-Za-z]

{Digitos} -> [0-9]

{Sub} -> _
{Id} -> {Letras} ({Letras} | {Digitos} | {Sub}) *

En la práctica los identificadores son reconocidos hasta que se encuentre un caracter diferente a letra, a dígito y al subrayado. Agrega una expresión concatenada al final de la expresión dada para el token Id.

{Id} -> {Letras} ({Letras} | {Digitos} | {Sub}) * [^A-Za-z0-9_]



Operador [^] de exclusión

Ejo. 2.3 Escribe una definición regular para los operadores relacionales en C.

La expresión regular puedes presentarla como :

OpRel -> < | <= | > | >= | != | ==

En la aplicación Ps1 se introducirá :

{OpRel} -> [<>!=] = | [<>] [^=]

¿Por qué no es válida la siguiente expresión?

{OpRel} -> [<>!=] =?

Ejo 2.4 Veamos la definición regular que aparece en el libro del dragón para el token *num* :

Digito -> 0 | 1 | ... | 9

Digitos -> Digito Digito *

FraccionOpcional -> . Digitos | ∈

ExponenteOpcional -> (E (+ | - | ∈) Digitos) | ∈

Num -> Digitos FraccionOpcional Exponente Opcional

Aquí debemos tener cuidado con las expresiones que contienen el punto . , el + y el -, ya que los tres son operadores que indican operaciones sobre expresiones regulares. En este caso representan el símbolo por si mismos, únicamente. Debes añadir el símbolo \ antes del caracter según lo indica la 2da. regla de la tabla 2.1. Además debes eliminar el uso del caracter ∈ utilizando la propiedad ?, ya que Ps1 no acepta dicho caracter. La definición regular debes escribirla mas o menos así :

{Digito} -> [0-9]

{Digitos} -> {Digito} {Digito} *

{FraccionOpcional} -> (\. {Digitos}) ?
{ExponenteOpcional} -> ([Ee] [\+|-] ? {Digitos}) ?
{Num} -> {Digitos} {FraccionOpcional} {ExponenteOpcional}

2.2 EDITANDO UNA DEFINICIÓN REGULAR.

Para entrar al proceso de edición de un conjunto de expresiones regulares, debes de considerar si es una definición regular nueva o bien ésta ya existe.

1. Si es una definición regular nueva selecciona los items del menú *Archivo / Nuevo*. Ps1 te responde con una ventana de edición como es mostrado en la figura 2.1 :

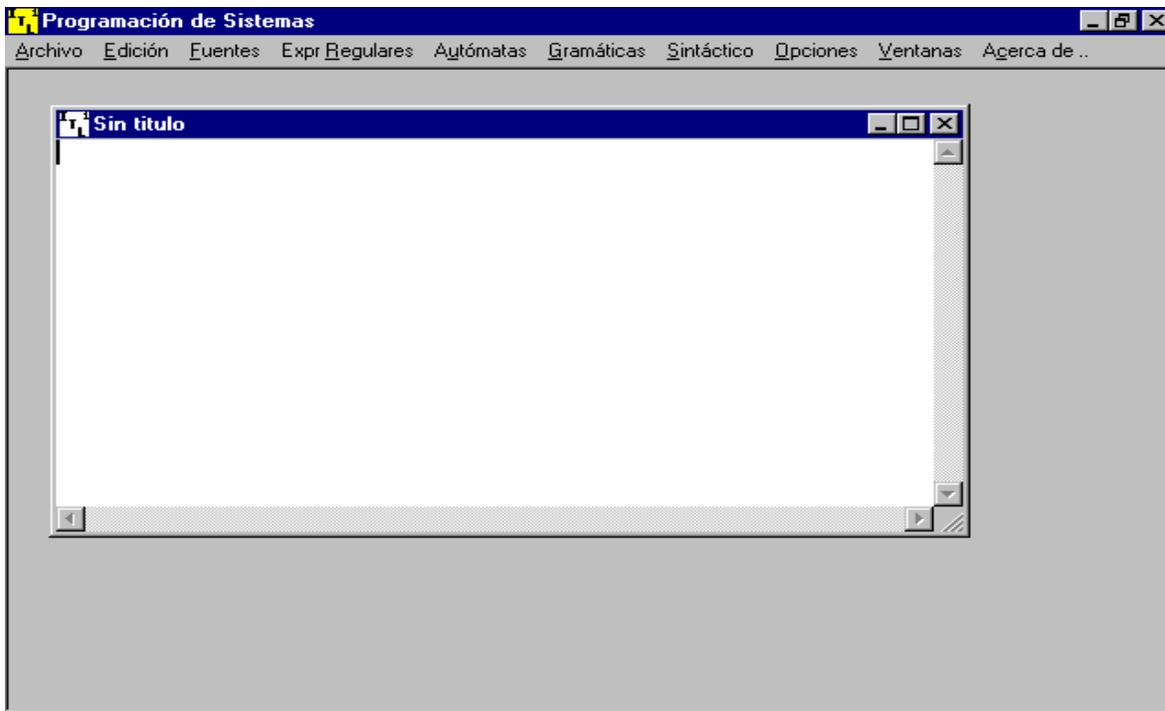


Fig. 2.1 Nueva edición.

2. Si el archivo donde se encuentra la definición regular ya existe, selecciona los items del menú *Archivo / Abrir* . Ps1 te muestra la caja de diálogo para abrir archivos según se te muestra en la Fig 2.2 . Ya que hayas seleccionado el archivo con extensión .exr Ps1 te responde con la ventana de edición con la definición regular en ella, Fig. 2.3.

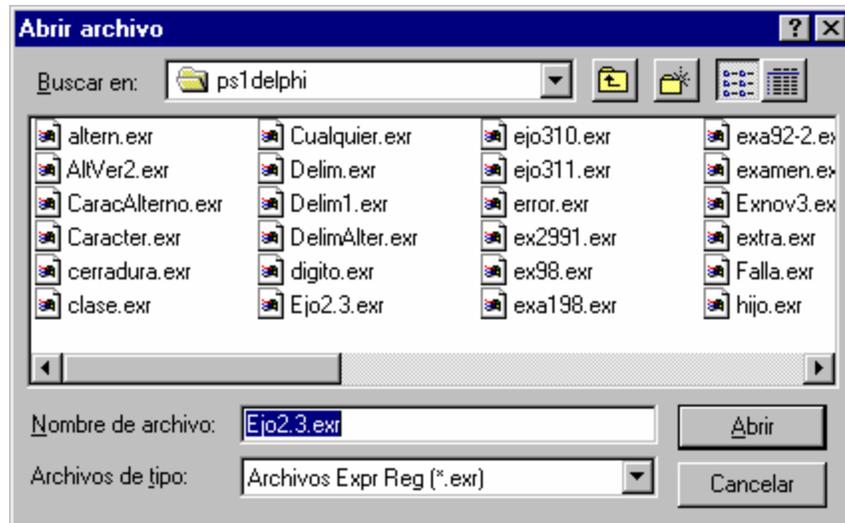


Fig. 2.2 Cuadro de diálogo para abrir expresiones regulares.

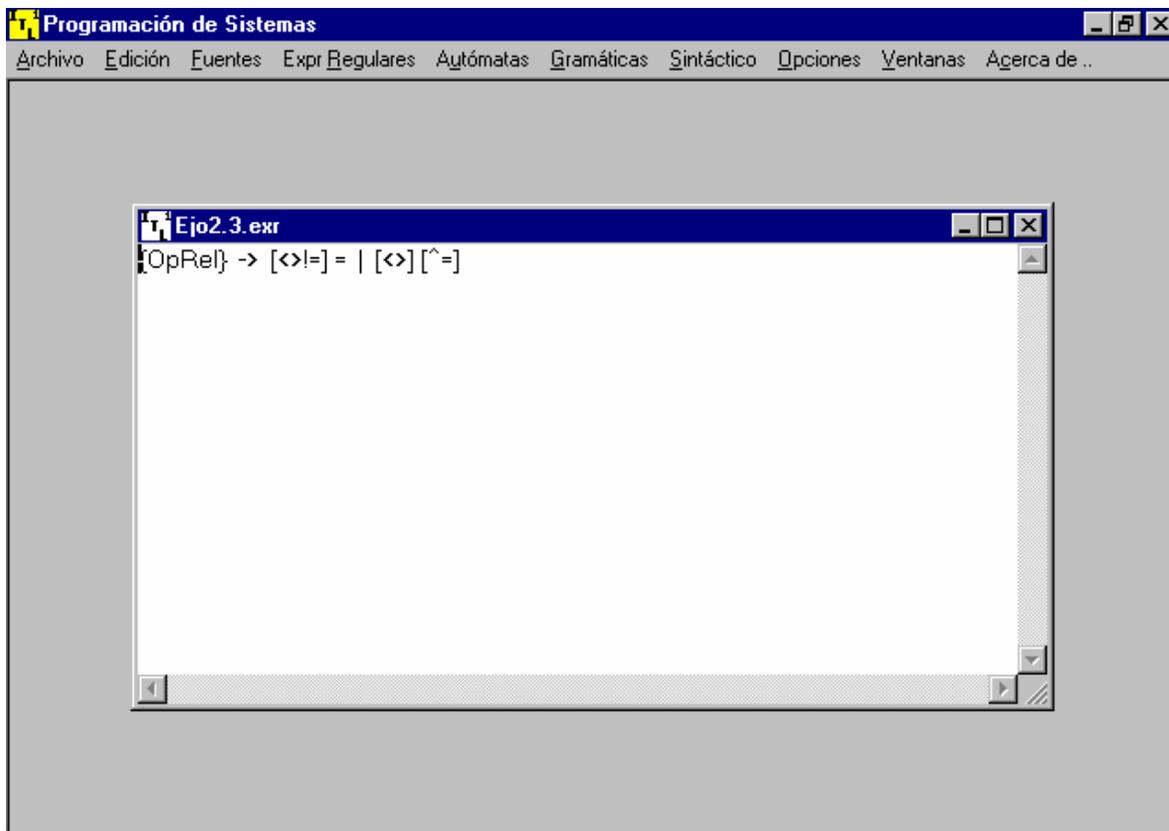


Fig. 2.3 Ventana para edición de expresiones regulares.

2.3 SALVANDO UNA DEFINICIÓN REGULAR.

Una vez que hayas editado un conjunto de expresiones regulares, estarás interesado en guardar tu información para posteriores referencias. Puedes hacerlo fácilmente usando la selección de los items *Archivo / Salvar como ó Archivo / Salvar*.

Nota : Debes guardar tus archivos que contienen las definiciones regulares en el mismo directorio en que reside el programa ejecutable Ps1. Esta restricción vive con la primera versión de Ps1.

Ps1 te mostrará el CUADRO DE DIÁLOGO Salvar como según se ilustra en la Fig. 2.4. La extensión que debes considerar en el nombre del archivo es .exr. Si no la incluyes en el nombre, Ps1 te la añade por omisión.

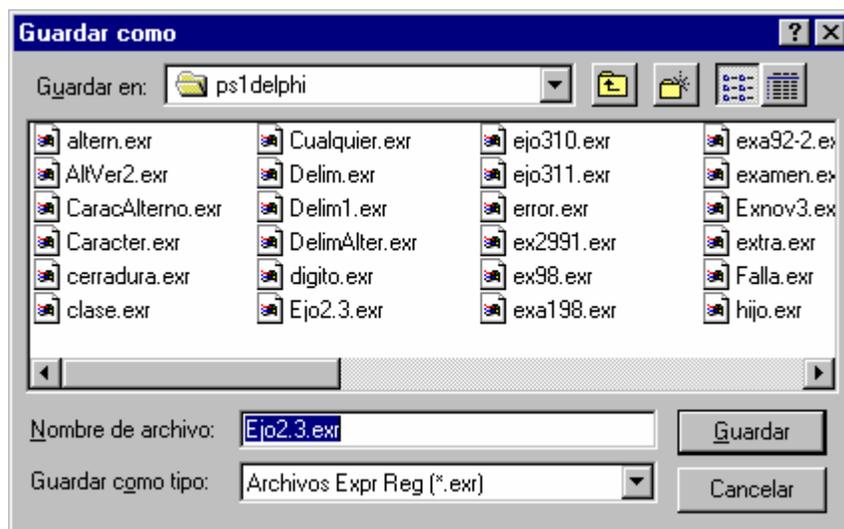


Fig. 2.4 Cuadro de diálogo Salvar como.

2.4 COMPILANDO UNA DEFINICIÓN REGULAR.

La compilación de una definición regular es tarea muy sencilla. Sólo debes asegurarte que antes de compilar hayas salvado tu definición regular si ésta es nueva. Cada vez que salves tu ventana de edición, estas asegurando que la opción de compilación se habilite.

Entonces para compilar tu definición regular sólo selecciona del menú principal los items *Expr Regulares / Compilar*.

Si no has cometido ningún error en la construcción de la definición regular obtendrás un cuadro de mensaje donde se te indica que la compilación ha sido exitosa, fig. 2.5.

Si hubieras incurrido en errores, Ps1 te visualiza el primer error que encuentre, fig. 2.6. Deberás corregirlo y volver a compilar. Este proceso se repetirá hasta que Ps1 no encuentre más errores en tu ventana de edición.



Fig. 2.5 Aviso de Ps1 al efectuar una compilación exitosa.

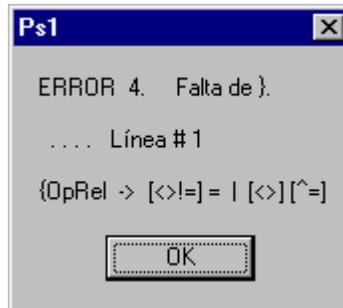


Fig. 2.6 Mensaje de error al compilar una expresión regular.

2.5 ERRORES COMUNES AL CONSTRUIR EXPRESIONES REGULARES.

- ERROR 3. Falta de la llave inicial que delimita al identificador de la expresión regular.

Veamos la definición regular para el token Id :

```
{Dig} -> [0-9]
Letra} -> [A-Za-z]
{Sub} -> _
{Id} -> {Letra} ( {Letra} | {Dig} | {Sub} ) * [^_A-Za-z0-9]
```

En la línea 2 el identificador de la expresión regular no inicia con el símbolo {. Ps1 responde al compilar con el mensaje de error de la fig. 2.7.



Fig. 2.7 ERROR 3: token no conocido.

- ERROR 4. Falta de la llave que termina al identificador de una expresión regular.

Ahora quitemos la segunda llave y dejemos la primera en la expresión regular del token Id.

```
{Dig} -> [0-9]
{Letra -> [A-Za-z]
{Sub} -> _
{Id} -> {Letra} ( {Letra} | {Dig} | {Sub} ) * [^_A-Za-z0-9]
```

Ps1 responde con el mensaje de error de la fig. 2.8.



Fig. 2.8 ERROR 4. Falta de }.

El mismo mensaje es exhibido por Ps1 cuando insertas antes de la llave final o entre el propio identificador, uno o mas caracteres de espacio. Haz la prueba añadiendo un espacio antes de la llave en la línea dos :

```
{Dig} -> [0-9]
{Letra } -> [A-Za-z] // o bien {L etra} -> [A-Za-z]
{Sub} -> _
{Id} -> {Letra} ( {Letra} | {Dig} | {Sub} ) * [^_A-Za-z0-9]
```

↓

↑

Inserción de un espacio antes de la llave final

- ERROR 9. Falta de las dos llaves que delimitan al identificador de una expresión regular.

Ahora quitemos las dos llaves en la línea dos de la definición regular para el token Id. Ps1 visualiza el cuadro de diálogo de error de la fig. 2.9.

```
{Dig} -> [0-9]
Letra -> [A-Za-z]
{Sub} -> _
```

{Id} -> {Letra} ({Letra} | {Dig} | {Sub}) * [^_A-Za-z0-9]

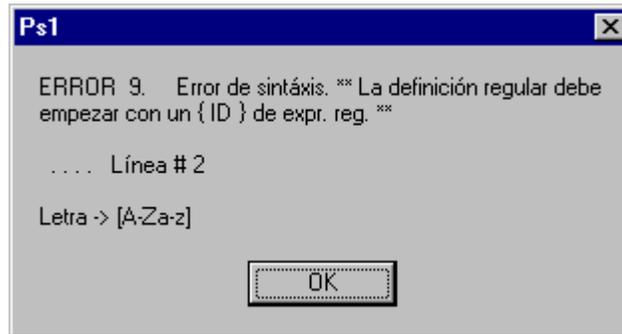


Fig. 2.9 ERROR 9. Falta de las dos llaves en el identificador de una expresión regular.

- ERROR 2. Identificador mal construido.

Este error –fig. 2.10- se presenta cuando después de la primera llave del identificador de una expresión regular, añades uno o mas espacios. Veamos lo siguiente :

{Dig} -> [0-9]

{ Letra} -> [A-Za-z]



Inserción de uno o mas blancos después de la primera llave

{Sub} -> _

{Id} -> {Letra} ({Letra} | {Dig} | {Sub}) * [^_A-Za-z0-9]

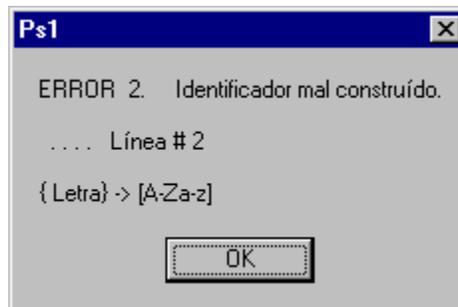


Fig. 2.10 ERROR 2. Identificador mal construido.

- ERROR 10. Falta del operador ->.

Este error ocurre cuando después del identificador de la expresión regular olvidas añadir el operador ->, o bien cuando lo construyes mal, es decir si insertas un espacio entre el – y el >, o bien te falta alguno de ellos, fig. 2.11.

{Dig} -> [0-9]
{Letra} [A-Za-z]



Falta del operador ->

{Sub} -> _
{Id} -> {Letra} ({Letra} | {Dig} | {Sub}) * [^_A-Za-z0-9]

{Dig} -> [0-9]
{Letra} -> [A-Za-z]



Espacio entre los dos símbolos del operador ->

{Sub} -> _
{Id} -> {Letra} ({Letra} | {Dig} | {Sub}) * [^_A-Za-z0-9]

{Dig} -> [0-9]
{Letra} > [A-Za-z]



Falta de uno de los dos símbolos del operador ->

{Sub} -> _
{Id} -> {Letra} ({Letra} | {Dig} | {Sub}) * [^_A-Za-z0-9]

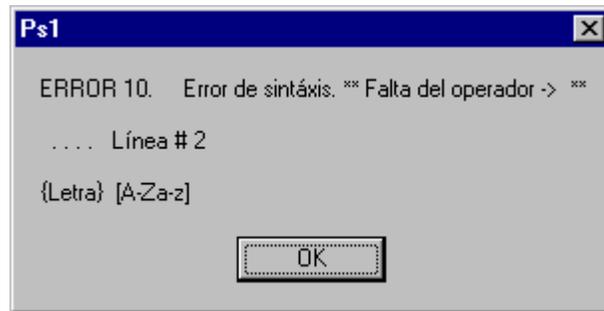
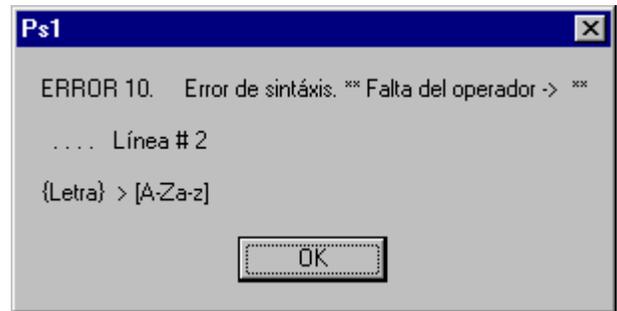
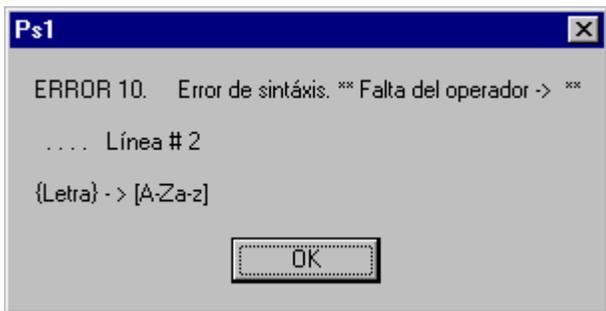


Fig. 2.11. ERROR 10. Falta de operador ->

- ERROR 5. Falla en la construcción de una cadena.

Este error sucede cuando olvidas iniciar una cadena con el símbolo de comillas “.

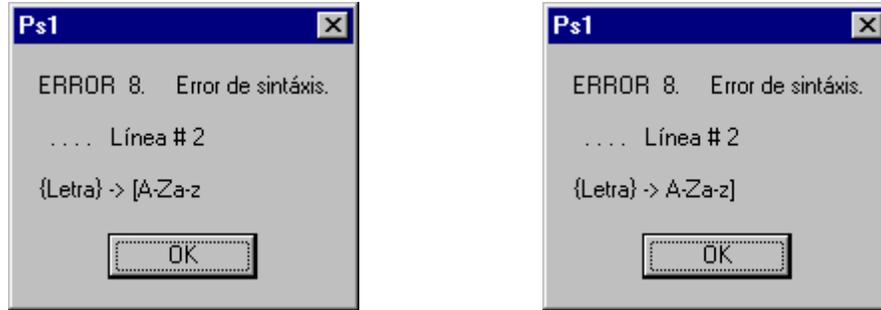


Fig. 2.14 ERROR 8. Error de sintáxis falta de corchete.

- ERROR 7. Falta de caracter en \car.

Cuando usas la regla de caracter en forma literal \car y que olvidas teclear el caracter después del símbolo \, Ps1 te responde con el mensaje de error 7, fig. 2.15. Observa las siguientes líneas :

```
{Dig} -> [0-9]
{Letra} -> A-Za-z \ ← Falta el caracter después del símbolo \
{Sub} -> _
{Id} -> {Letra} ( {Letra} | {Dig} | {Sub} ) * [^_A-Za-z0-9]
```

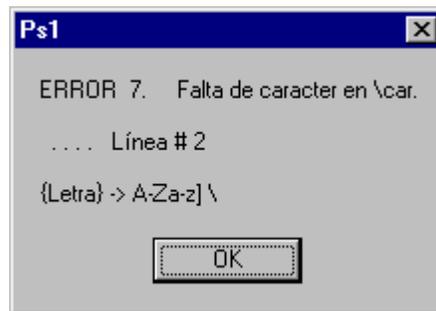


Fig. 2.15. ERROR 7. Falta de caracter en \car.

- ERROR 11. Expresión regular no definida.

Este error es muy común. Se presenta cuando usas un identificador de una expresión regular en el miembro derecho de una definición, siendo que aún no la habías definido en renglones previos. Por ejemplo, observa la línea 4 de la definición regular para el token Id :

```
{Dig} -> [0-9]
{Letra} -> A-Za-z
{Sub} -> _
{Id} -> {Letra} ( {Letr} | {Dig} | {Sub} ) * [^_A-Za-z0-9] ← Expresión regular no definida, fig. 2.16
```

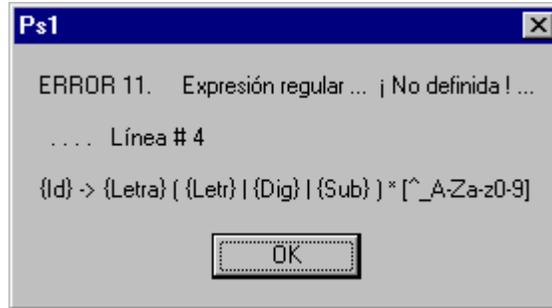


Fig. 2.16. ERROR 11. Expresión regular no definida.

III. REGLAS DE THOMPSON –AFND–.

Existen un conjunto de reglas que permiten obtener el autómata finito no determinístico a partir de una definición regular. Se les denominan reglas de Thompson. Estas reglas son fáciles de aplicar y como son rigurosas –formales–, son susceptibles de ser tratadas por un programa de computadora. Ps1 te proporciona la característica de lograr el AFND dado que has editado y compilado una definición regular, según lo visto en el anterior capítulo.

Las reglas de Thompson son explicadas en el capítulo III págs 106 a 128, correspondientes al volumen 2 del manual técnico.

En las siguientes secciones te ilustraré acerca de visualización en pantalla del AFND construido en base a las reglas de Thompson, de la conmutación entre pantallas de edición de la definición regular y de visualización del autómata finito no determinístico, además de cómo resolver el problema de un mal dibujo del autómata cuando las dimensiones del bitmap no soportan al total del AFND. También te mostraré cómo seleccionar los colores del AFND según tu agrado.

3.1 OBTENCIÓN DEL AFND.

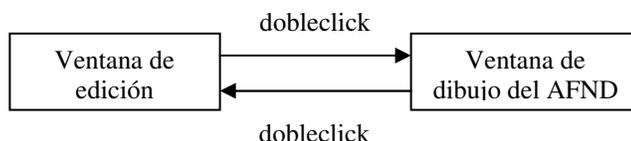
Una vez que Ps1 ha detectado que una definición regular ha sido compilada sin errores, habilita las opciones del menú *Autómatas / Thompson(AFND)*. Selecciona este par de items del menú para obtener el dibujo del AFND construido aplicando las reglas de Thompson, fig. 3.1.

En la figura 3.1 sólo es visto el autómata final, es decir el autómata que reconoce la definición regular $\{Id\} \rightarrow \{Letra\} (\{Letra\} | \{Digito\} | \{Sub\}) * [^A-Za-z0-9_]$. Ps1 añade el autómata según las reglas de Thompson para cada línea que compone a la definición regular. Para que veas lo anterior mencionado, sólo navega con las barras de desplazamiento sobre el dibujo del autómata.

El dibujo del autómata se despliega sobre una superficie de 1000 x 1000 pixeles. Hay ocasiones en que esta superficie no es suficiente para desplegar el dibujo de un autómata. Cuando ésto suceda, el autómata no será dibujado de manera correcta y será necesario que modifiques las dimensiones de la imagen en la que es dibujado el autómata. Esta modificación es tratada en las siguientes secciones.

3.2 CONMUTACIÓN ENTRE VENTANA DE EDICIÓN Y VENTANA DE DIBUJO DEL AFND.

Frecuentemente es necesario observar los cambios que produce una edición de cierta definición regular, en el AFND de Thompson. Luego de ver el AFND, volver a la edición para corregir o salvar los cambios. Esta conmutación entre las ventanas de edición y de dibujo del autómata, es lograda de una manera muy sencilla. Sólo debes hacer un doble click sobre la superficie de la ventana donde te encuentres y cambiarás a la otra ventana.



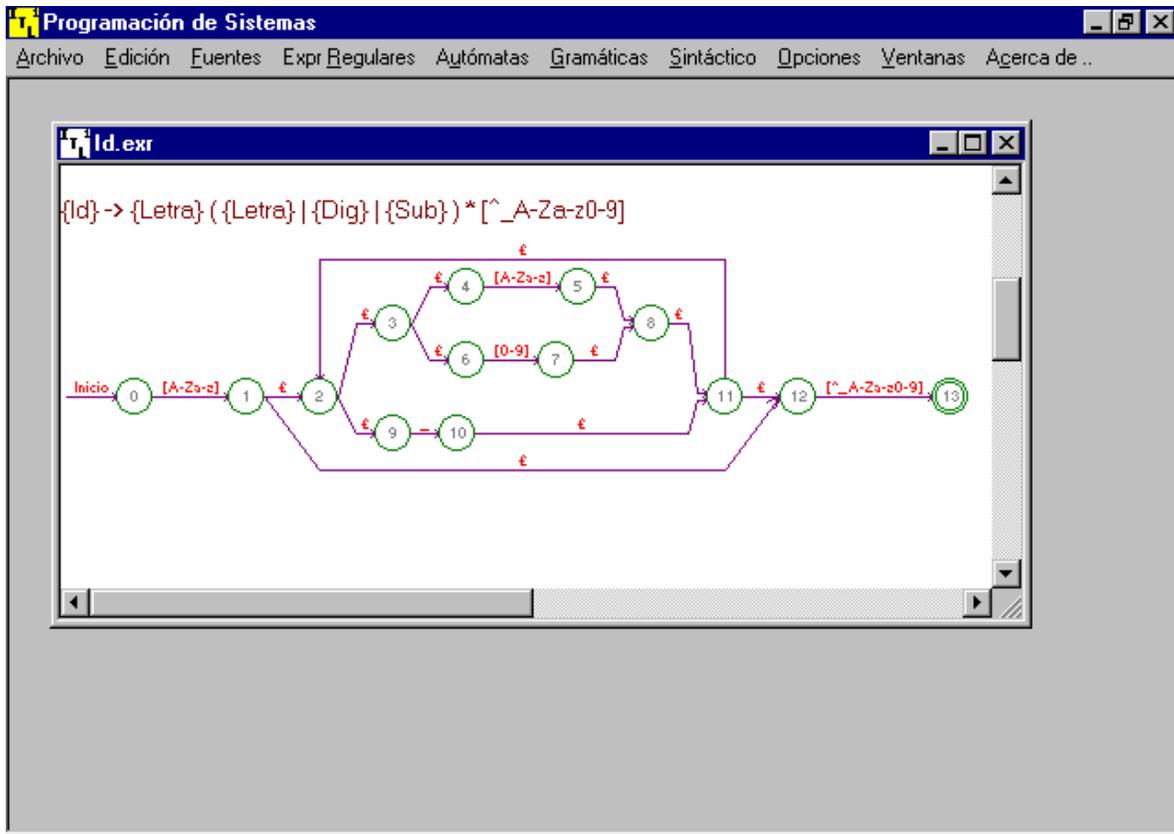


Fig. 3.1 AFND para la definición regular Id.

3.3 MODIFICACIÓN DE LAS DIMENSIONES DEL BITMAP DEL AFND.

Ahora veremos lo que sucede en ocasiones en que la definición regular produce un AFND cuyas dimensiones son mayores a las tomadas por Ps1 por omisión (10000 x 10000 pixeles), y el dibujo del AFND es mutilado por Ps1.

Edita la siguiente definición regular, la cual es la resolución del problema propuesto No. 10 del capítulo I página 32 del manual técnico volumen II.

```
{Vocal} -> [AaEeliOoUu]
{Par} -> [02468]
{IniAux} -> {Vocal}| {Par}
{Inicio} -> {IniAux} 6
{Fin} -> \" | @
{Non} -> [13579]
{Dig} -> [0-9]
{SecAux} -> {Non} {Dig}* {Non} | {Non}
{SecInter} -> {SecAux} ?
{Prob10} -> {Inicio} {SecInter} {Fin}
```

Este problema viene en el disco que incluye ejemplos de Ps1, en el archivo Prob10.exr.

Selecciona los ítems **Autómatas** | **Thompson(AFND)** y navega con la barra de desplazamiento vertical hasta el último autómata. ¡A verdad!!?? ... seguramente ya te habrás percatado que el dibujo del autómata está mutilado o no se alcanza a visualizar en su totalidad, fig.3.2.

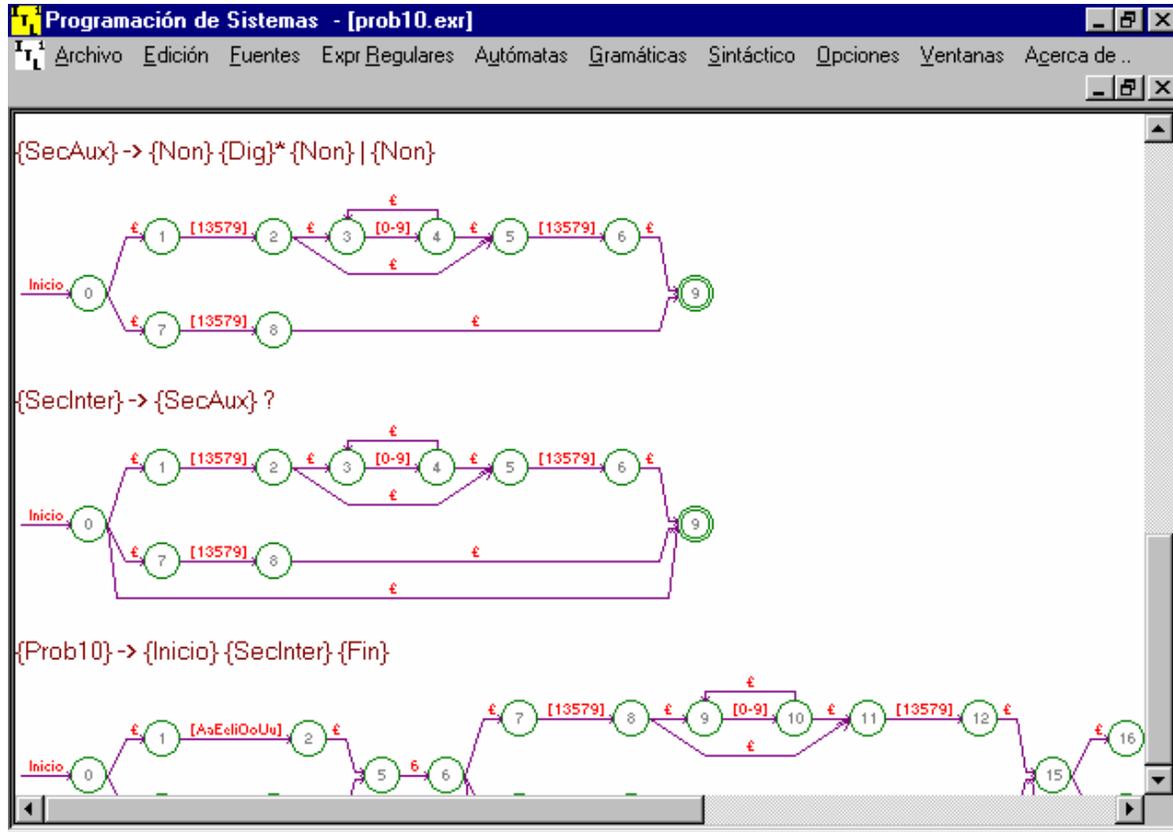


Fig. 3.2 Dimensiones del bitmap no suficientes.

Ps1 te permite manipular AFND con dimensiones hasta de 2000 x 2000 pixeles. Para aumentar la dimensión del bitmap selecciona los ítems del menú *Opciones / Mapa de bits AFND*. Ps1 te contesta con la caja de diálogo de la fig 3.3.

Teclea en la dimensión de altura el valor de 2000.

Ahora Ps1 te visualiza el AFND en un bitmap con dimensiones 1000 x 2000. El dibujo del AFND ahora no tendrá errores y puedes visualizarlo navegando con las barras de desplazamiento vertical y horizontales, fig. 3.4.

3.4 SELECCIÓN DE COLORES DEL AFND.

Es posible que efectúes cambio en el color de los diferentes componentes del dibujo de un AFND. Los ítems del menú que te permiten realizar lo anterior son las mismas que para cambiar la dimensión del bitmap, *Opciones / Mapa de bits AFND*. Observa la figura 3.3 en su parte izquierda. Puedes seleccionar el color de los arcos, etiquetas de los arcos, círculos y números de los estados, y de la descripción de la expresión regular.

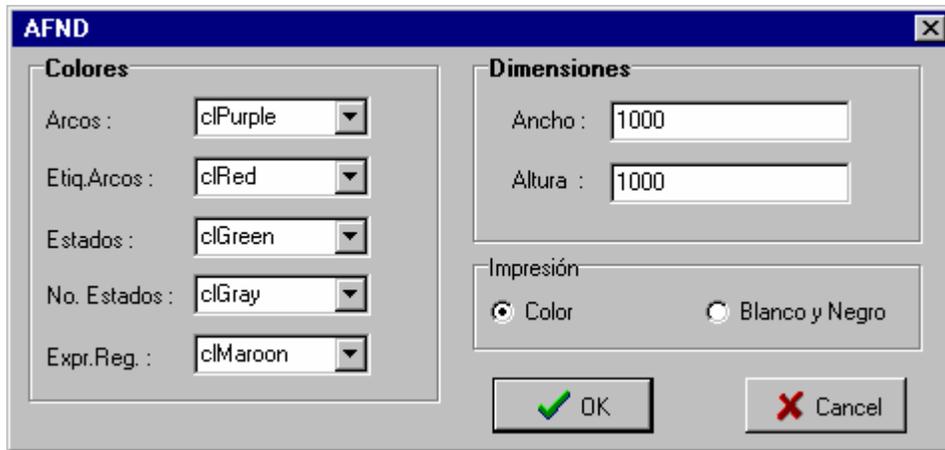


Fig. 3.3 Caja de diálogo para cambio de dimensiones en el bitmap.

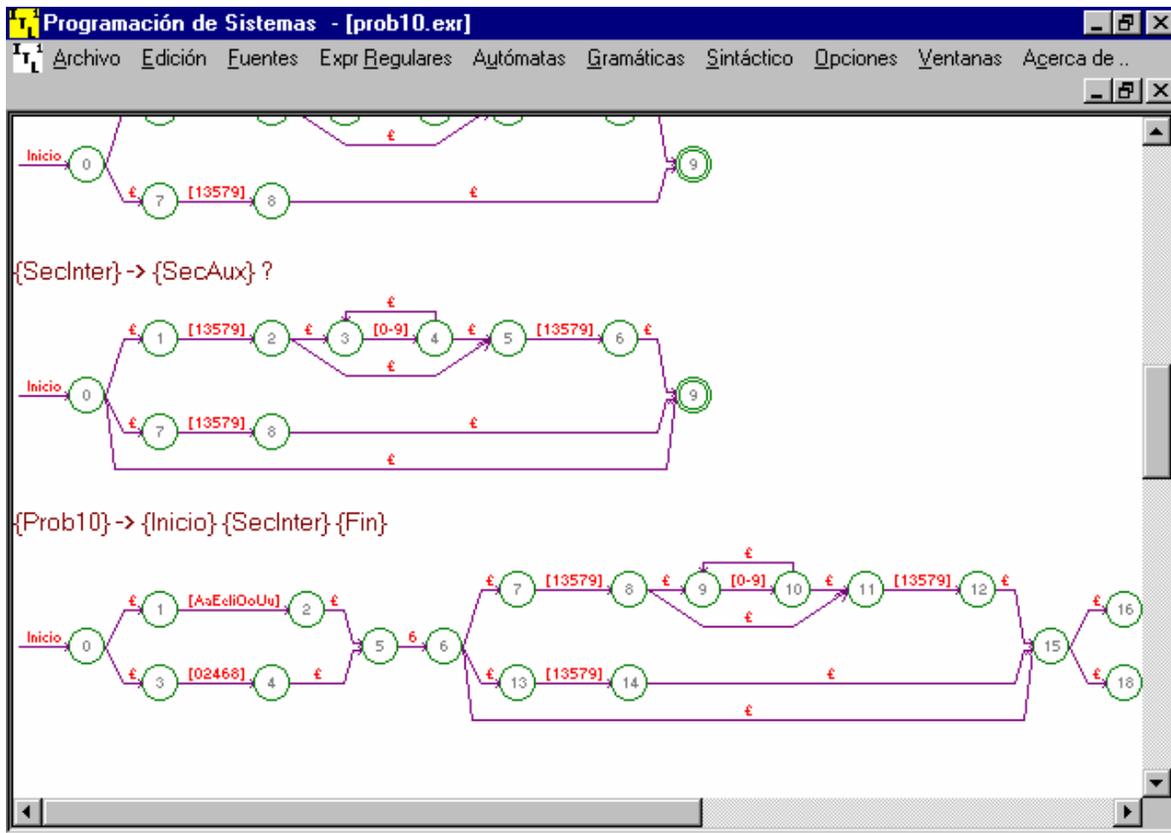


Fig. 3.4 AFND con corrección en dimensiones.

IV. ALGORITMO DE CONSTRUCCIÓN DE SUBGRUPOS AFD.

El algoritmo de construcción de subgrupos toma como entrada un autómata finito no determinístico AFND para construir un autómata finito determinístico AFD.

Ambos reconocen el mismo lenguaje. El algoritmo es descrito ampliamente en el capítulo III del manual técnico volumen II páginas 129 a 157.

En las secciones de este capítulo te diré como obtener el dibujo del AFD, el texto del algoritmo de construcción de subgrupos paso a paso, la tabla de transición del nuevo AFD y la correspondencia de cada estado del AFD con los estados del AFND, así como los componentes del nuevo AFD.

4.1 OBTENCIÓN DEL NUEVO AFD, TABLA DE TRANSICIÓN, ALGORITMO DE CONSTRUCCIÓN DE SUBGRUPOS Y COMPONENTES.

Son muy simples los pasos que debes seguir para tener en pantalla el dibujo del autómata finito determinístico que corresponde a un AFND que es obtenido previamente de una edición de una definición regular.

Ya que Ps1 ha detectado la orden de dibujar el AFND según Thompson, habilita la opción de *Subgrupos (AFD)* del menu *autómatas*. Selecciona estos dos items del menú y Ps1 te visualizará una nueva ventana, fig. 4.1.

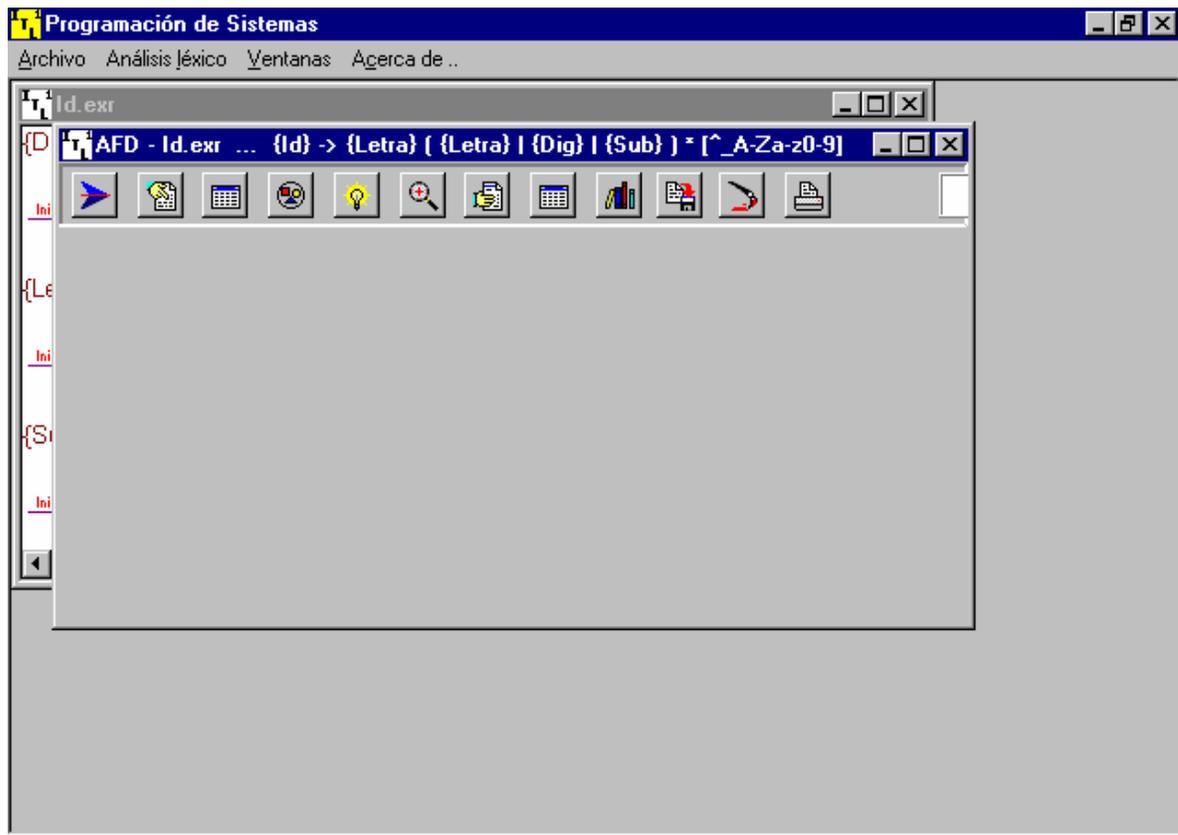


Fig. 4.1 Ventana de funciones AFD de Ps1.

La nueva ventana tiene en su encabezado la expresión regular sobre la cual se construye el autómata finito no determinístico, y por consiguiente éste será el AFND al que se le aplicará el algoritmo de construcción de subgrupos.
Puedes visualizar las dos ventanas a la vez tal y como se te muestra en la figura 4.2, seleccionando los items del menu *Ventanas / Mosaico*.

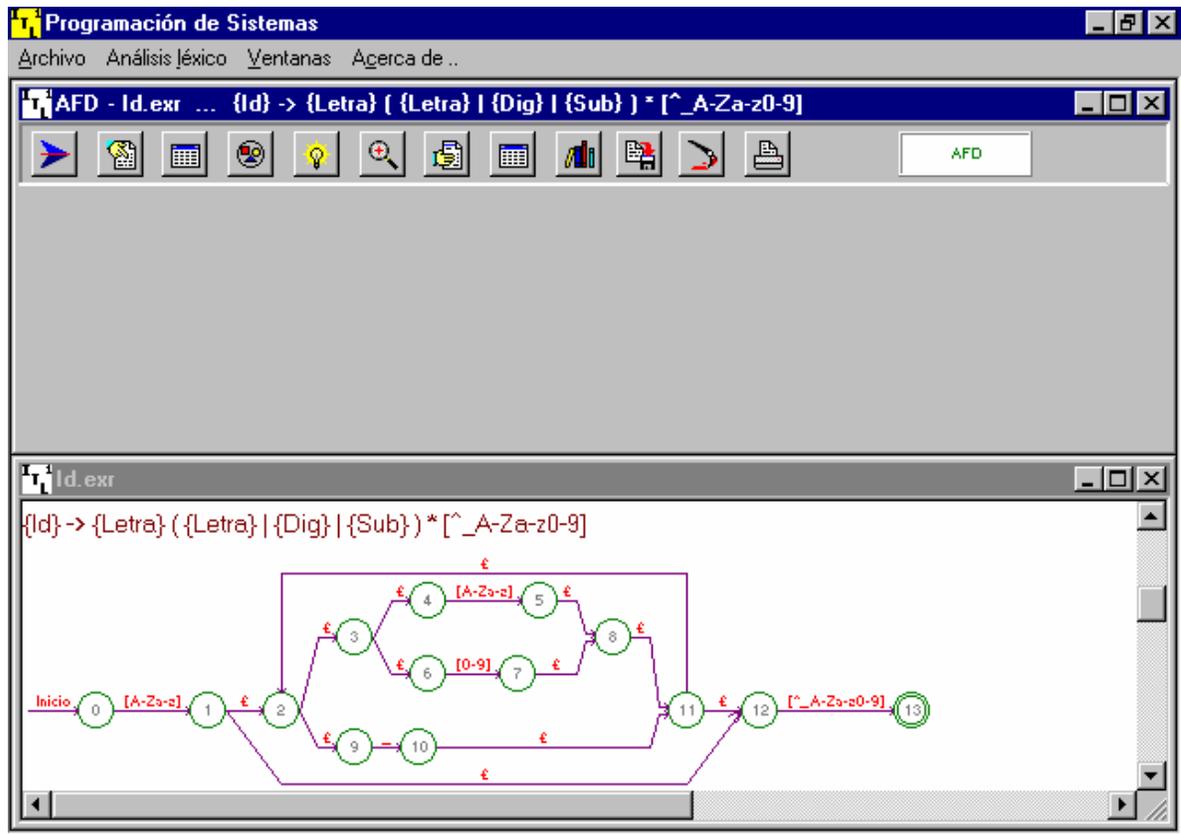


Fig. 4.2 Mosaico de ventanas AFND y AFD.

Ahora sólo haz un click sobre el primer botón al extremo izquierdo de la ventana para AFD's, Fig 4.3. El algoritmo de construcción de subgrupos se muestra en la misma ventana. Sólo haz un click sobre el segundo botón de izquierda a derecha de la nueva ventana del AFD, fig. 4.3.

La tabla de transición la obtienes haciendo un click sobre el tercer botón de izquierda a derecha de la ventana del AFD, fig. 4.4.

Por último, haciendo un click sobre el cuarto botón de velocidad situado de izquierda a derecha en la ventana del nuevo AFD, podrás visualizar los componentes del nuevo autómata finito determinístico, fig. 4.4.

Programación de Sistemas - [AFD - Id.exr ... {Id} -> {Letra} ({Letra} | {Dig} | {Sub}) * [^_A-Za-...

Archivo Análisis léxico Ventanas Acerca de ..

AFD

«« Estados AFD »»	«« Estados AFD »»	A-Za-z	0-9	-	!-@[
0	A	B			
1-2-12-3-9-4-6	B	C	D	E	F
5-8-11-2-12-3-9-4-6	C	C	D	E	F
7-8-11-2-12-3-9-4-6	D	C	D	E	F
10-11-2-12-3-9-4-6	E	C	D	E	F
13	F				

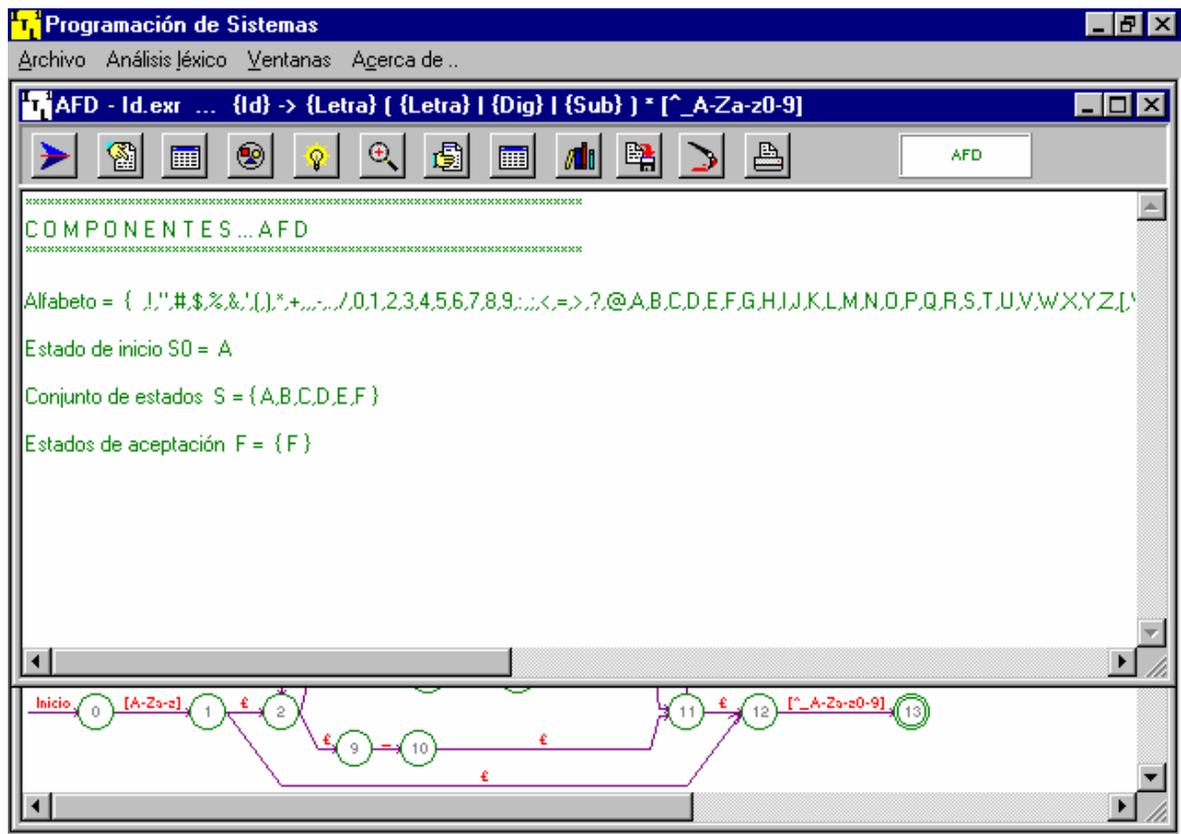


Fig. 4.4 Tabla de transición y componentes del AFD.

V. ALGORITMO DE PARTICIONES – AFD REDUCIDO-

El AFD que se deriva de un AFND a partir de una definición regular, puede ser eficientizado en cuanto al número de estados que lo conforman. El algoritmo de particiones es útil para reducir el número de estados de un autómata finito determinístico. En las páginas 157 a 171 del manual técnico se describen varios ejemplos de la utilización de este algoritmo para reducir autómatas.

Ps1 te proporciona la facilidad con sólo hacer un click, de obtener el dibujo del AFD reducido, sus componentes, su nueva tabla de transición y el texto del algoritmo paso a paso.

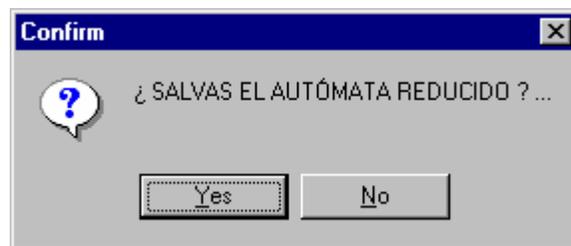
Asimismo puedes grabar la tabla de transición del AFD reducido y el dibujo de dicho autómata, para después utilizarlo en el ensamble de analizadores léxicos, facilidad que el mismo paquete didáctico Ps1 te brinda para el curso de Programación de Sistemas I.

Ahora sólo haz un click sobre el sexto botón de izquierda a derecha de la ventana para AFD's, Fig 5.1. El algoritmo de particiones se muestra en la misma ventana. Sólo haz un click sobre el séptimo botón de izquierda a derecha de la nueva ventana del AFD, fig. 5.1.

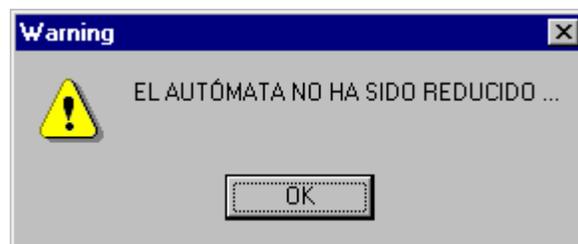
La tabla de transición la obtienes haciendo un click sobre el octavo botón de izquierda a derecha de la ventana del AFD, fig. 5.2.

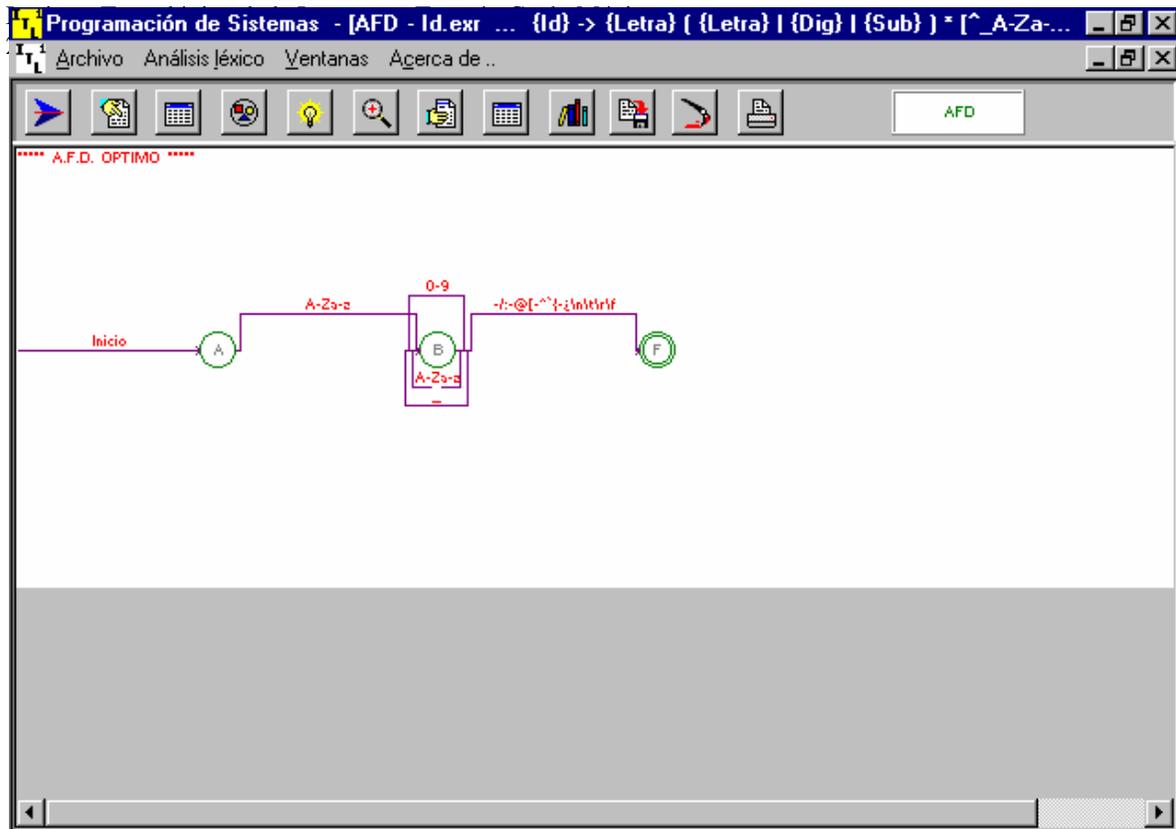
Por último, haciendo un click sobre el noveno botón de velocidad situado de izquierda a derecha en la ventana del nuevo AFD, podrás visualizar los componentes del nuevo autómata finito determinístico reducido, fig. 5.2.

Para grabar el AFD reducido debes de hacer un click sobre el botón que tiene un diskette como imagen –décimo botón-. Ps1 te responderá con el mensaje siguiente :



Si no has derivado el AFD reducido y quieres grabarlo, Ps1 te muestra el siguiente mensaje de error :





```
ALGORITMO DE PARTICIONES ... AFD ÓPTIMO

*** Partición inicial :
G1 = (ABCDE), G2 = (F).

-----
1a. iteración.
-----

*** Partición nueva : ¡ SI hay partición !!! ***
G1 = (A), G2 = (BCDE), G3 = (F).

-----
2a. iteración.
-----

*** PARTICIÓN FINAL : ¡ NO hubo partición !!! ***
G1 = (A), G2 = (BCDE), G3 = (F).
```

Fig. 5.1 AFD reducido y algoritmo de particiones.

VI. GRAMÁTICAS.

Ps1 te permite editar y compilar gramáticas de contexto libre con recursividad a la izquierda o recursividad a la derecha. Una vez compilada una gramática puedes usar Ps1 para efectuar una derivación a la izquierda o a la derecha de una sentencia previamente definida. Ps1 también te encuentra y visualiza los componentes de la gramática.

6.1 EDICIÓN Y COMPILACIÓN DE UNA GRAMÁTICA.

La edición de una gramática la haces tal y como lo viste en el capítulo de definiciones regulares. Sólo debes cuidar de grabar tu gramática con la extensión .gra, cuando salves tu edición, fig.6.1.

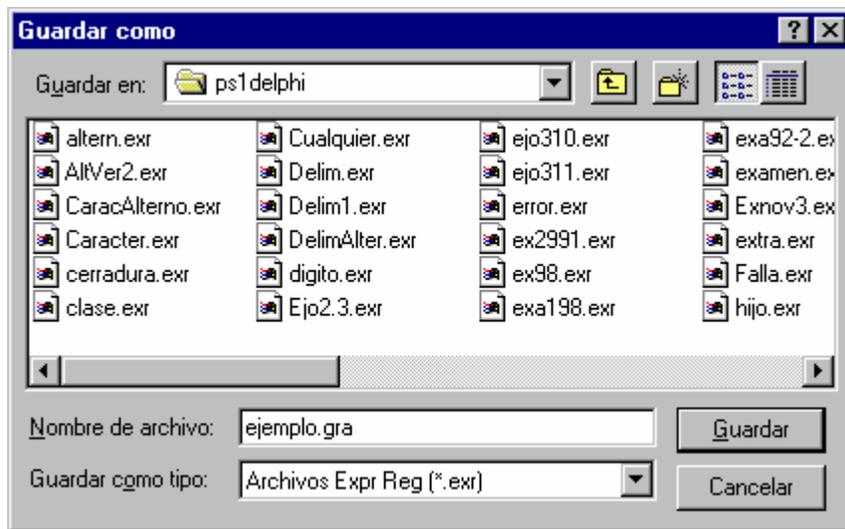


Fig. 6.1 La gramática tiene una extensión .gra.

Una vez que grabas una gramática, puedes compilarla si así lo deseas. Cuando salvas la gramática Ps1 te habilita las opciones del menú *Gramáticas / Compilar*. Si no tienes errores Ps1 te responde con el siguiente mensaje, fig. 6.2 :

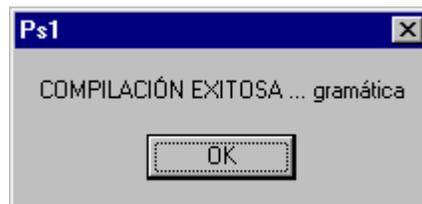


Fig. 6.2 Éxito en la compilación de una gramática.

6.2 COMPONENTES DE UNA GRAMÁTICA.

Selecciona los items del menú Gramáticas | Operaciones y tendrás una nueva ventana, fig. 6.3.

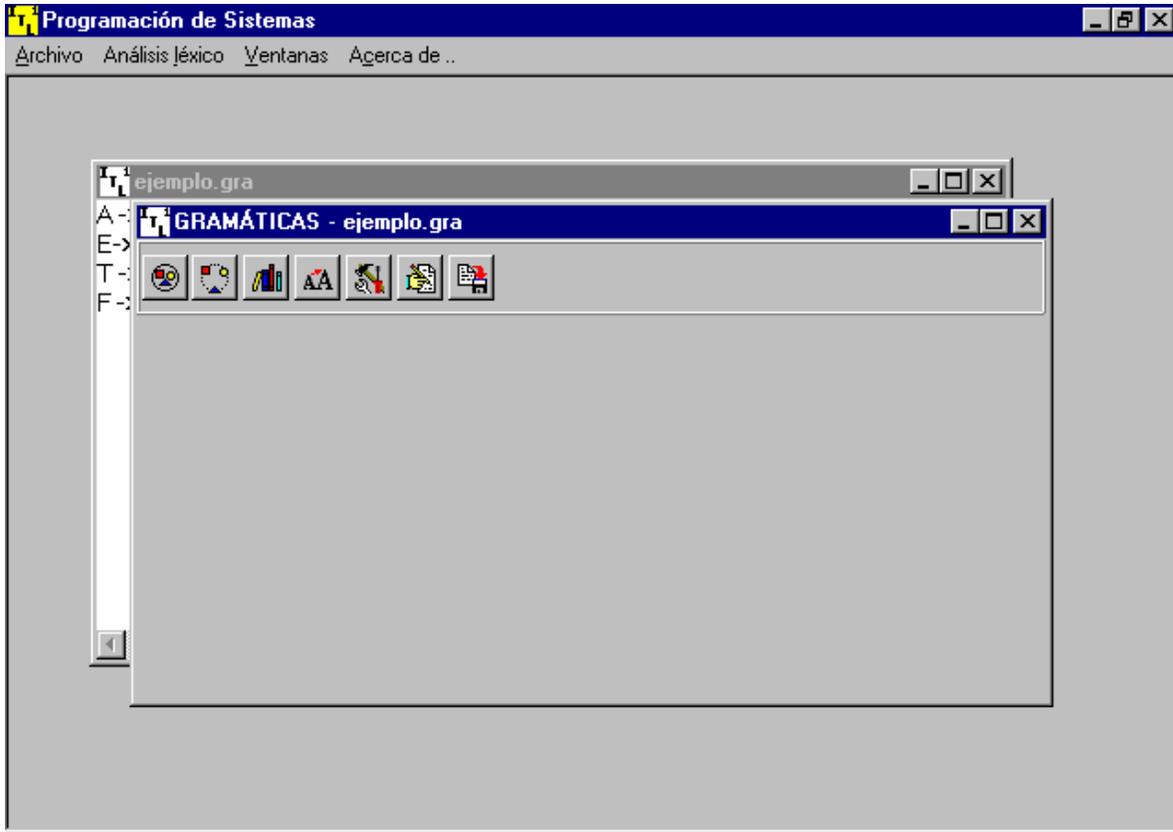


Fig. 6.3 Ventana de operaciones sobre gramáticas.

Puedes visualizar la gramática agrupada haciendo un click sobre el primer botón de la nueva ventana para las gramáticas.

Las producciones una a una –gramática desagrupada-, las puedes ver con un click sobre el segundo botón de la ventana.

Los componentes de la gramática los tienes si haces un click sobre el tercer botón en la ventana de gramáticas.

Lo anterior se muestra en las figuras 6.4 y 6.5.

6.3 DERIVACIÓN A LA IZQUIERDA Y A LA DERECHA DE UNA SENTENCIA.

Para utilizar Ps1 como herramienta para efectuar una derivación de una sentencia, debes primero indicarle el tipo de derivación que vas a emplear. Haciendo un click sobre el quinto botón de la ventana de gramáticas, tendrás la oportunidad de seleccionar el tipo de derivación, fig. 6.6.

La derivación podrás realizarla en la ventana que obtienes si haces un click sobre el cuarto botón en la ventana de las gramáticas, fig. 6.7.

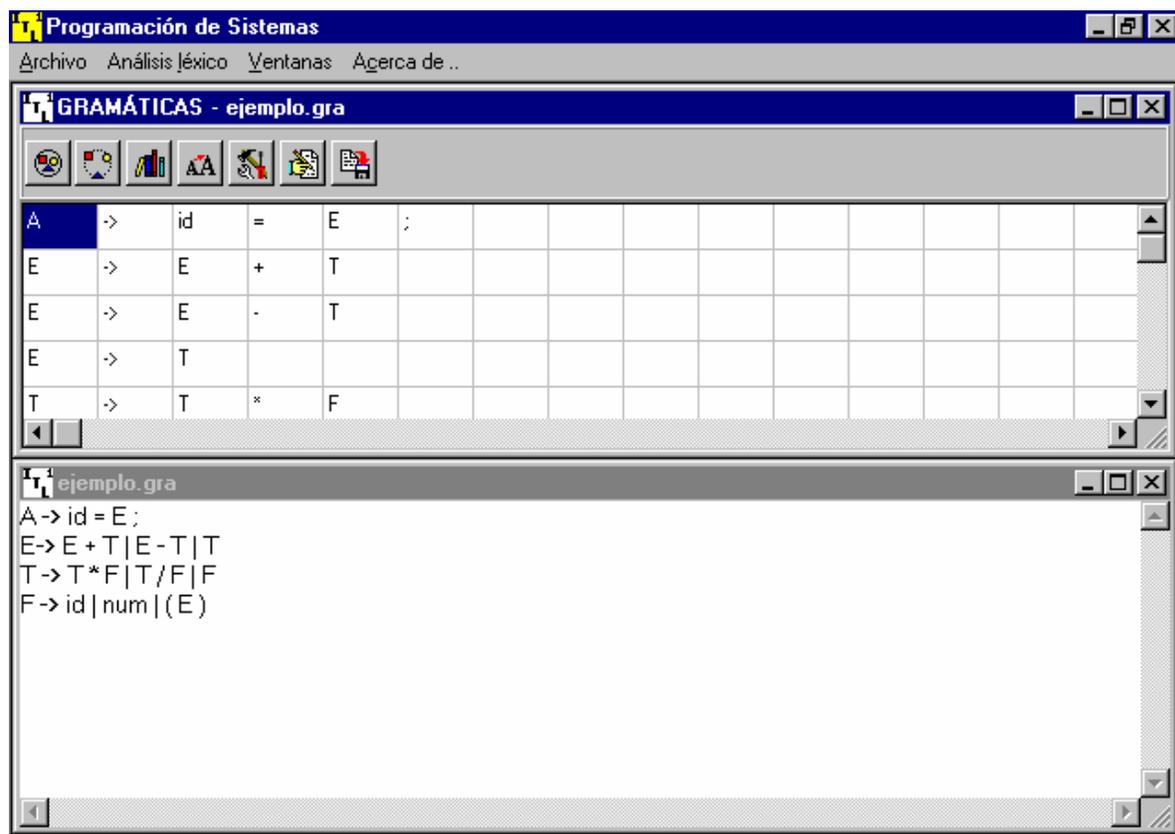
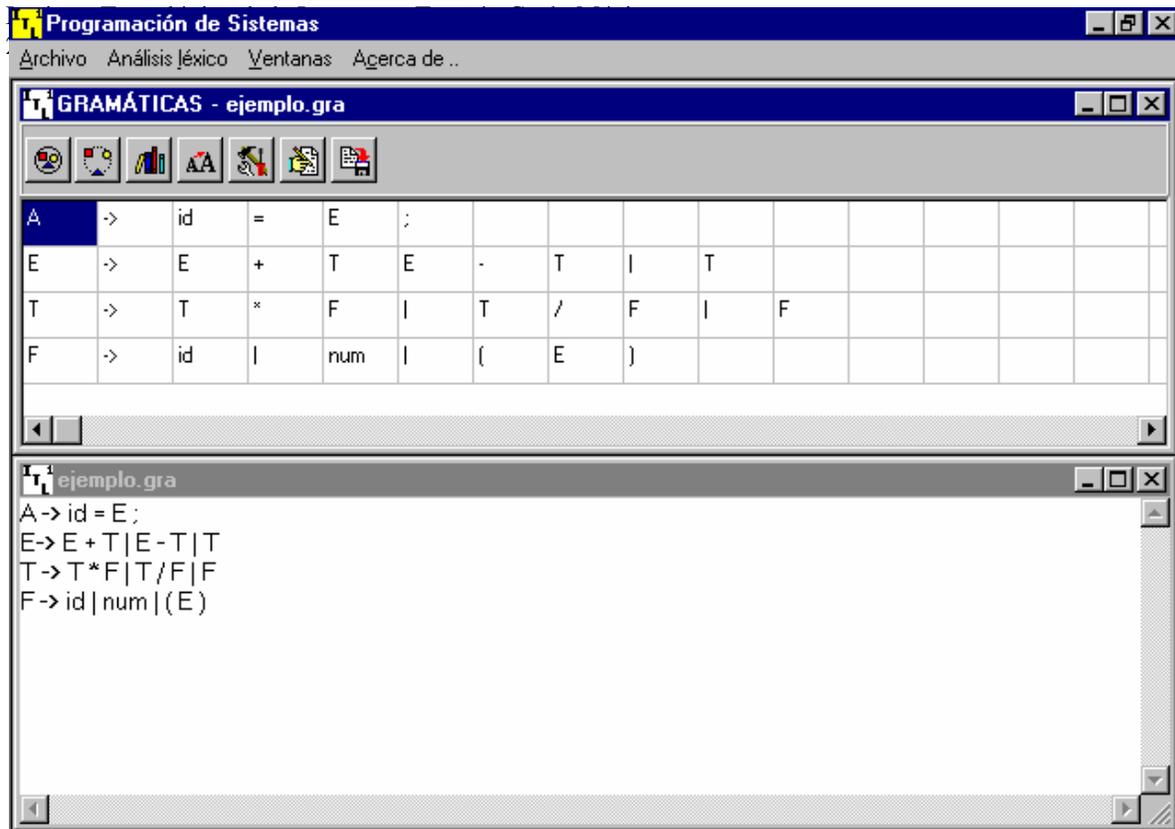


Fig. 6.4 Gramática agrupada y desagrupada.

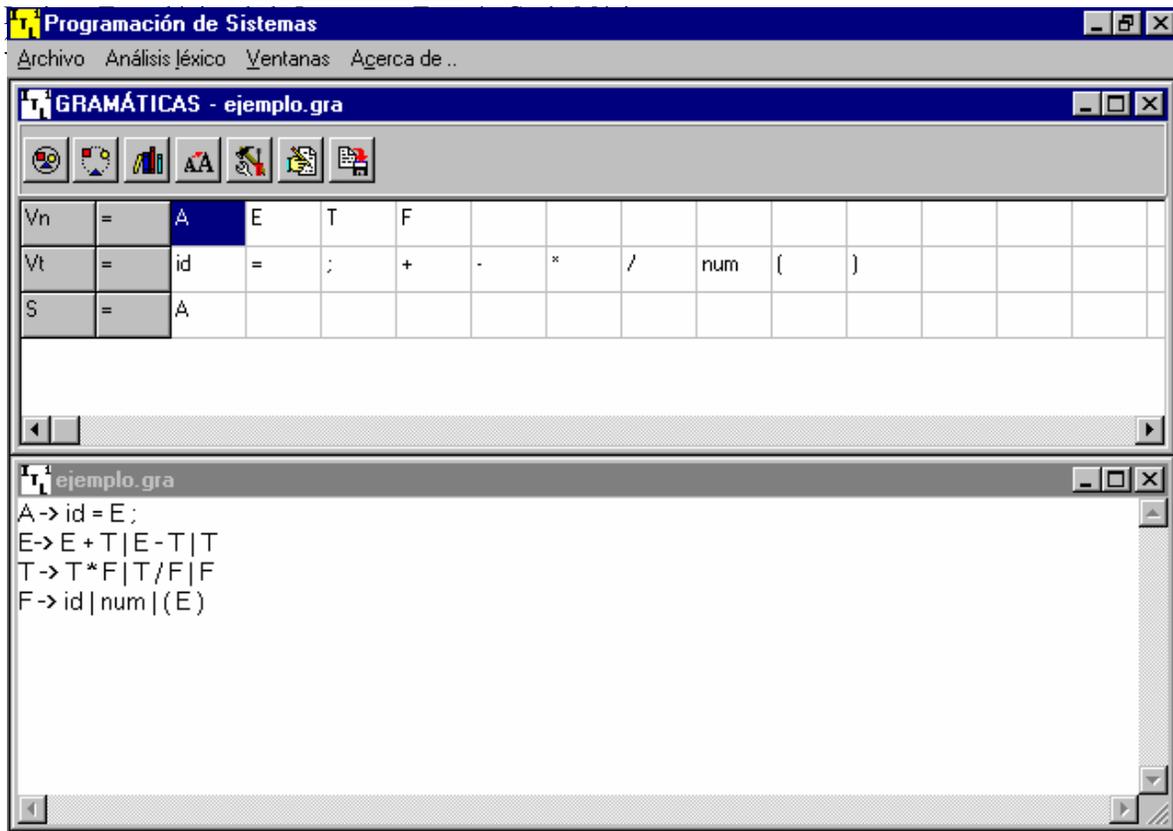


Fig. 6.5 Componentes de la gramática.

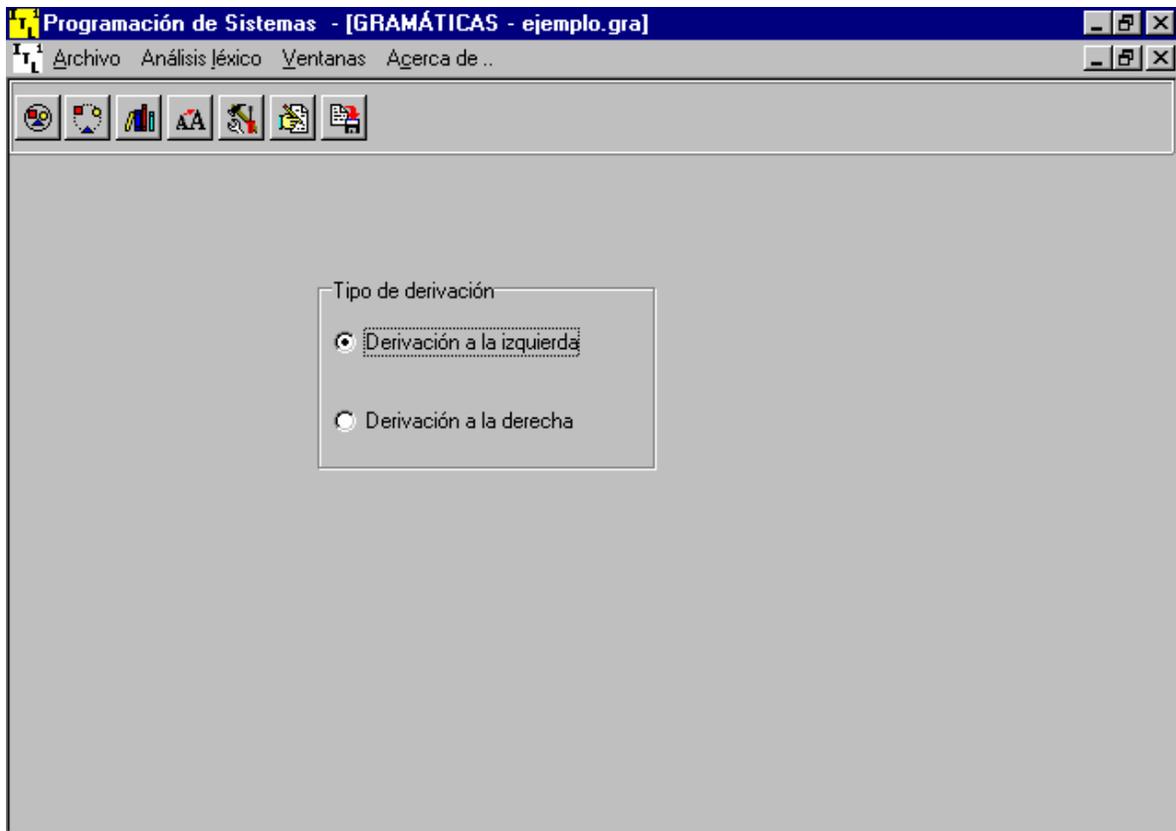


Fig. 6.6 Tipo de derivación

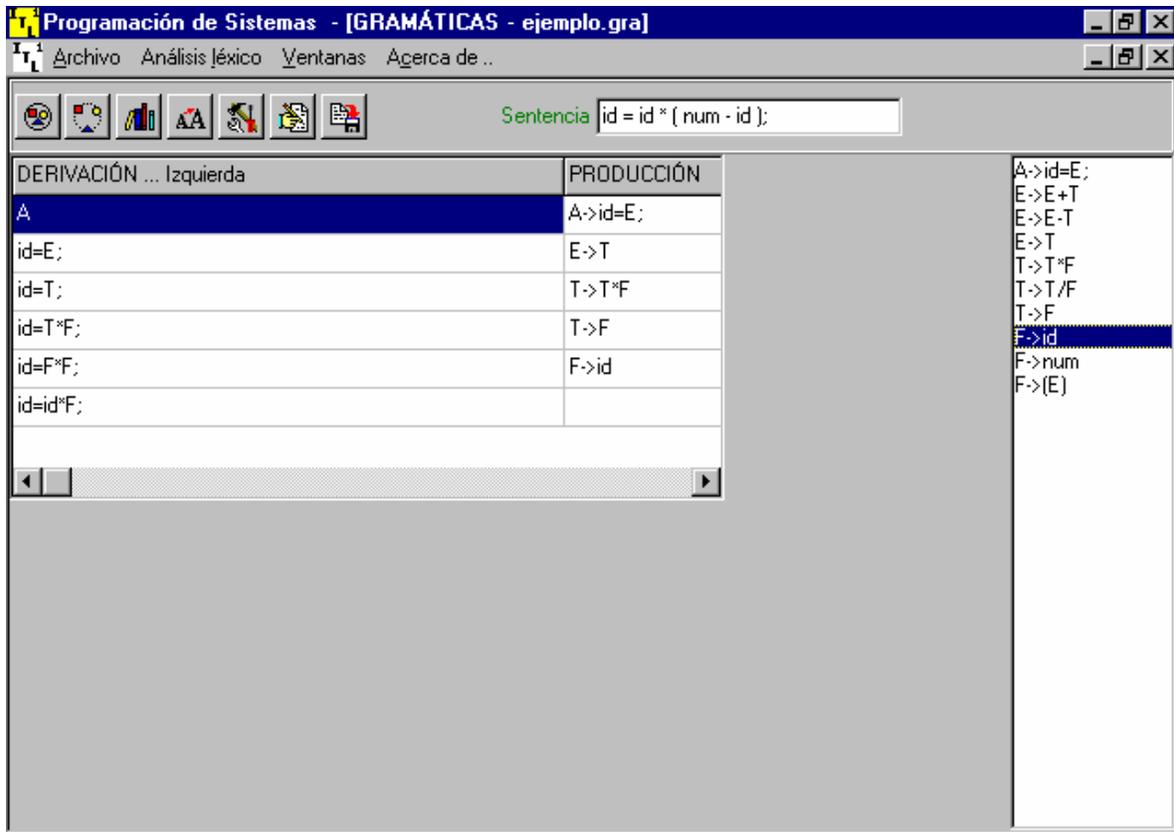


Fig. 6.7 Derivación a la izquierda de una sentencia.