Lógica y Programación

Diagramas de Decisión Binarios

J.-A. Alonso, F.-J. Martín-Mateos, J.-L. Ruiz-Reina

Dpto. Ciencias de la Computación e Inteligencia Artificial

Universidad de Sevilla

Forma normal condicional

- Definición:

$$x \twoheadrightarrow y_0, y_1 = (x \wedge y_0) \vee (\neg x \wedge y_1)$$

- Todas las conectivas lógicas se pueden expresar en función de $\twoheadrightarrow$ y los valores de verdad 0 y 1, de forma que las *condiciones* se evalúan únicamente sobre variables:
  - $x \equiv x \twoheadrightarrow 1, 0$
  - $\neg x \equiv x \twoheadrightarrow 0, 1$
  - $x \wedge y \equiv x \twoheadrightarrow (y \twoheadrightarrow 1, 0), 0$
  - $x \vee y \equiv x \twoheadrightarrow 1, (y \twoheadrightarrow 1, 0)$
  - $x \rightarrow y \equiv x \twoheadrightarrow (y \twoheadrightarrow 1, 0), 1$
  - $x \leftrightarrow y \equiv x \twoheadrightarrow (y \twoheadrightarrow 1, 0), (y \twoheadrightarrow 0, 1)$
- Una Forma Normal Condicional es una fórmula construida enteramente con la conectiva $\twoheadrightarrow$ y los valores de verdad 0 y 1, de forma que las *condiciones* se evalúan únicamente sobre variables

Expansión de Shannon

- $F[x/v]$ representa la fórmula obtenida a partir de F sustituyendo x por v, donde $v \in \{0, 1\}$
- Expansión de Shannon de F con respecto a x:

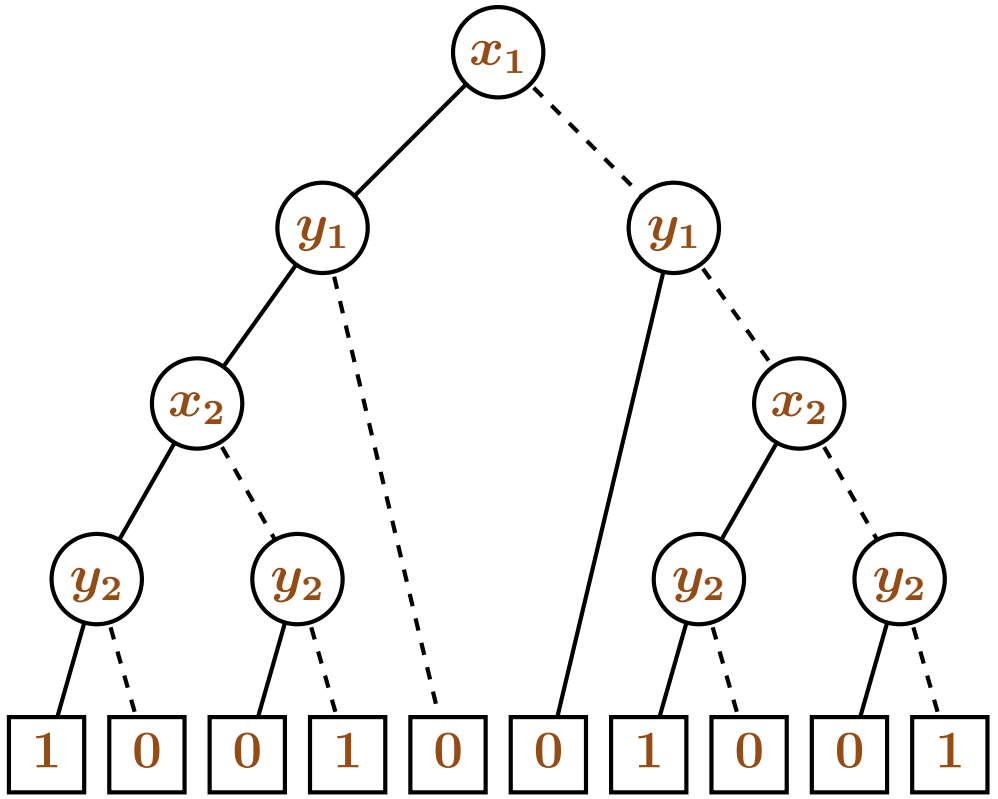
$$F \equiv x \longrightarrow F[x/1], F[x/0]$$

- Transformación a forma normal condicional usando la expansión de Shannon
- Implementación: **`(expansion-shannon F)`**

Árboles de decisión

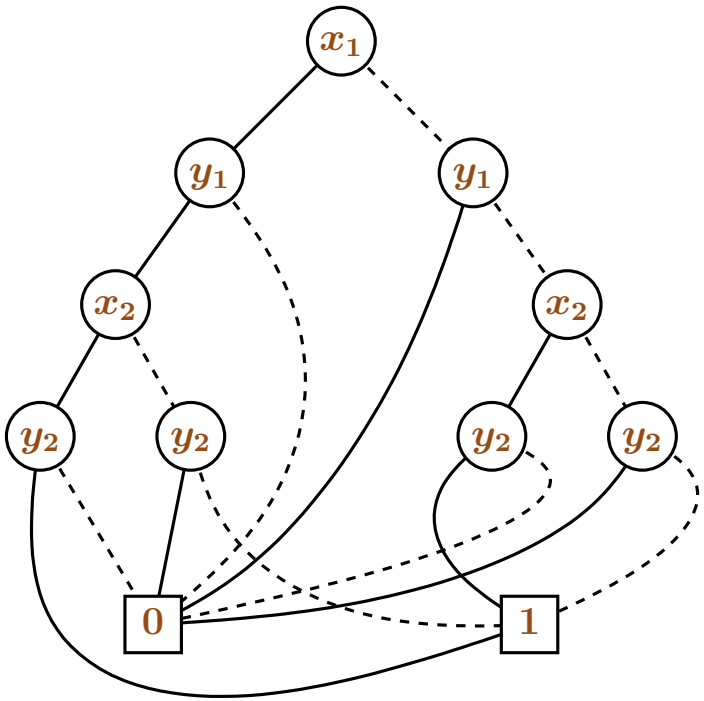
- Ejemplo: $F = (x_1 \leftrightarrow y_1) \wedge (x_2 \leftrightarrow y_2)$

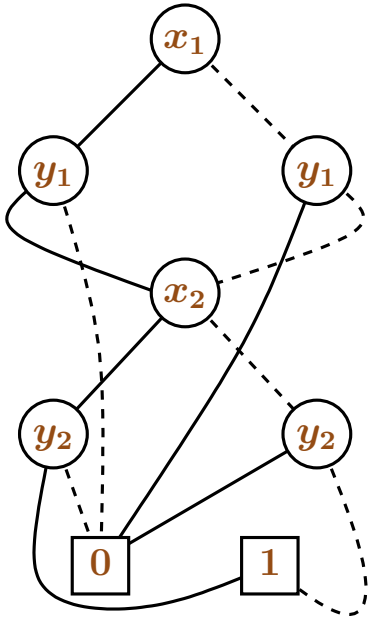
- $F = x_1 \twoheadrightarrow F_1, F_0$
- $F_1 = y_1 \twoheadrightarrow F_{11}, 0$
- $F_0 = y_1 \twoheadrightarrow 0, F_{00}$
- $F_{11} = x_2 \twoheadrightarrow F_{111}, F_{110}$
- $F_{00} = x_2 \twoheadrightarrow F_{001}, F_{000}$
- $F_{001} = y_2 \twoheadrightarrow 1, 0$
- $F_{000} = y_2 \twoheadrightarrow 0, 1$
- $F_{111} = y_2 \twoheadrightarrow 1, 0$
- $F_{110} = y_2 \twoheadrightarrow 0, 1$



Diagramas de decisión binarios

- Un diagrama de decisión binario (DDB) es un grafo acíclico dirigido con
  - Uno o dos nodos terminales etiquetados con 0 o 1
  - Un conjunto de nodos variables con dos hijos. Cada nodo variable está etiquetado con un símbolo proposicional.
- Ejemplo: $F = (x_1 \leftrightarrow y_1) \wedge (x_2 \leftrightarrow y_2)$



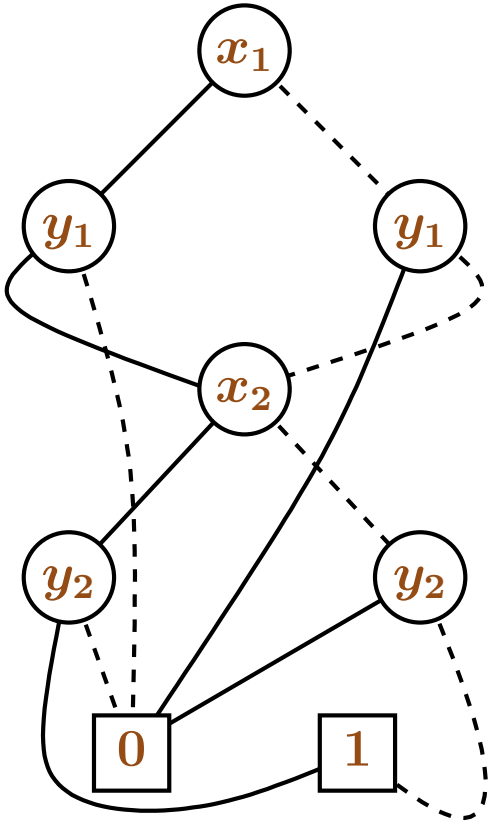


Reducción de Diagramas de Decisión Binarios

- Un DDB está reducido (DDBR) si
  - **unicidad**: No existen dos nodos en el diagrama etiquetados con el mismo símbolo proposicional y con los mismos hijos izquierdo y derecho (nodos duplicados)
  - **no-redundancia**: No existen nodos variables con hijos izquierdo y derecho idénticos (nodos redundantes)

Reducción de Diagramas de Decisión Binarios

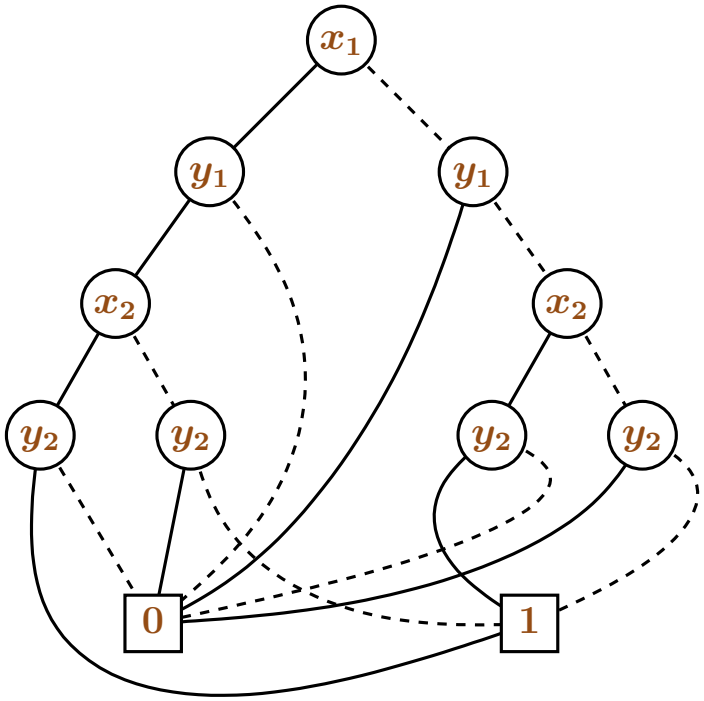
- Ejemplo: $F = (x_1 \leftrightarrow y_1) \wedge (x_2 \leftrightarrow y_2)$

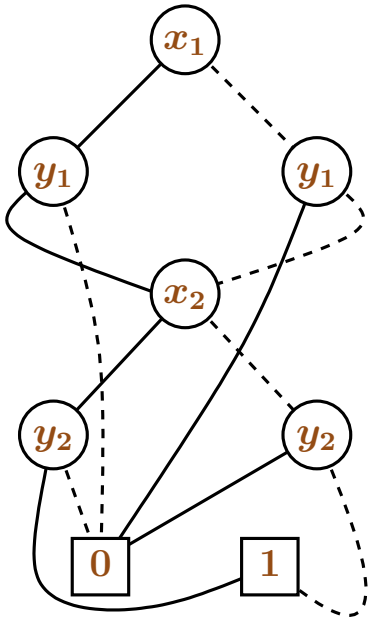


Diagramas de Decisión Binarios Ordenados

- Un DDB está ordenado (DDBO) si existe un orden entre los símbolos proposicionales $x_1 < x_2 < \ldots < x_n$ tal que para todo nodo variable etiquetado con x_i, si uno de sus nodos hijos es un nodo variable etiquetado con x_j entonces $x_i < x_j$
- Ejemplo: $F = (x_1 \leftrightarrow y_1) \wedge (x_2 \leftrightarrow y_2)$

  $x_1 < y_1 < x_2 < y_2$





DDB ordenados y reducidos

- Diagramas de decisión binario reducido y ordenado (DDBRO)
- Propiedades: Una vez fijado el orden entre las variables
  - Toda fórmula es equivalente a un único DDBRO
  - Una fórmula es válida si su DDBRO equivalente está formado por el nodo terminal 1
  - Una fórmula es insatisfacible si su DDBRO equivalente está formado por el nodo terminal 0
  - Los modelos de una fórmula se pueden determinar a partir de los caminos en su DDBRO equivalente que van desde el nodo raíz hasta el nodo terminal 1

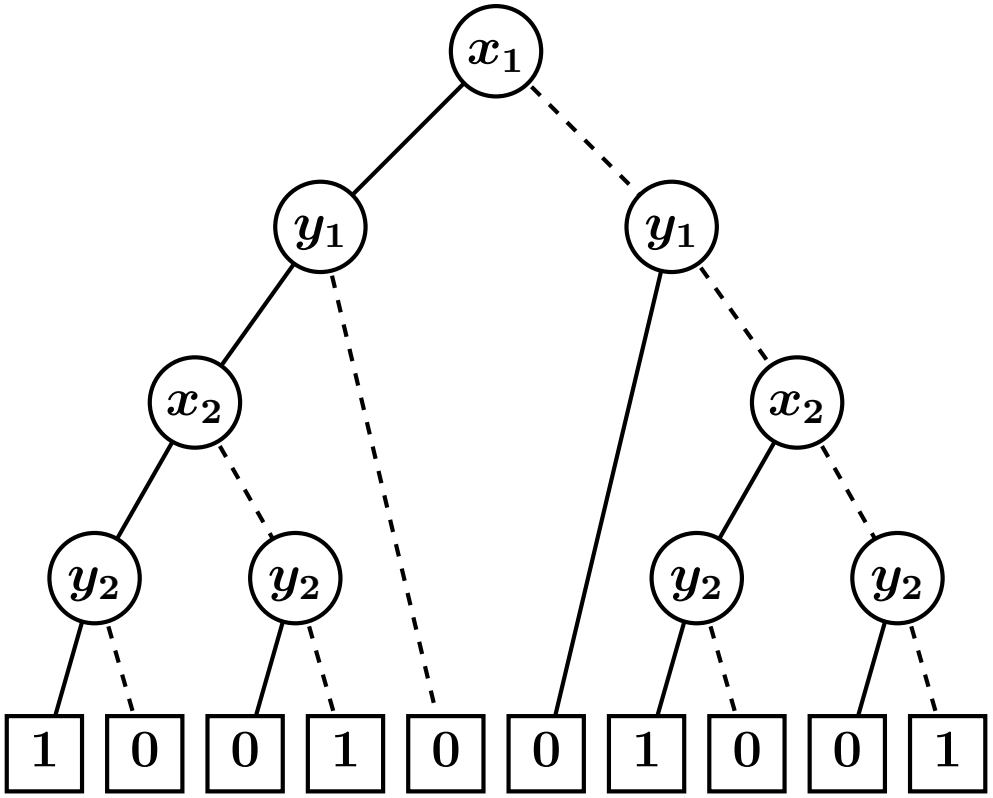
Representación de DDB

- Un DDB se representa con una lista de nodos, donde el primer elemento es el nodo raíz
- Los nodos terminales se representan con las listas `(1 1)` y `(0 0)`
- Un nodo variable se representa con una lista de cuatro elementos `(id,v,h1,h0)`,
  - `id` es un identificador único por nodo (números del 2 en adelante)
  - `v` es el símbolo proposicional que etiqueta el nodo
  - `h1` es el identificador del nodo correspondiente al valor `1` de `v`
  - `h0` es el identificador del nodo correspondiente al valor `0` de `v`

Representación de DDB

- Ejemplo: $F = (x_1 \leftrightarrow y_1) \wedge (x_2 \leftrightarrow y_2)$

```
((2 x1 3 4)
 (3 y1 5 0)
 (4 y1 0 6)
 (5 x2 7 8)
 (6 x2 9 10)
 (7 y2 1 0)
 (8 y2 0 1)
 (9 y2 1 0)
 (10 y2 0 1)
 (1 1)
 (0 0))
```



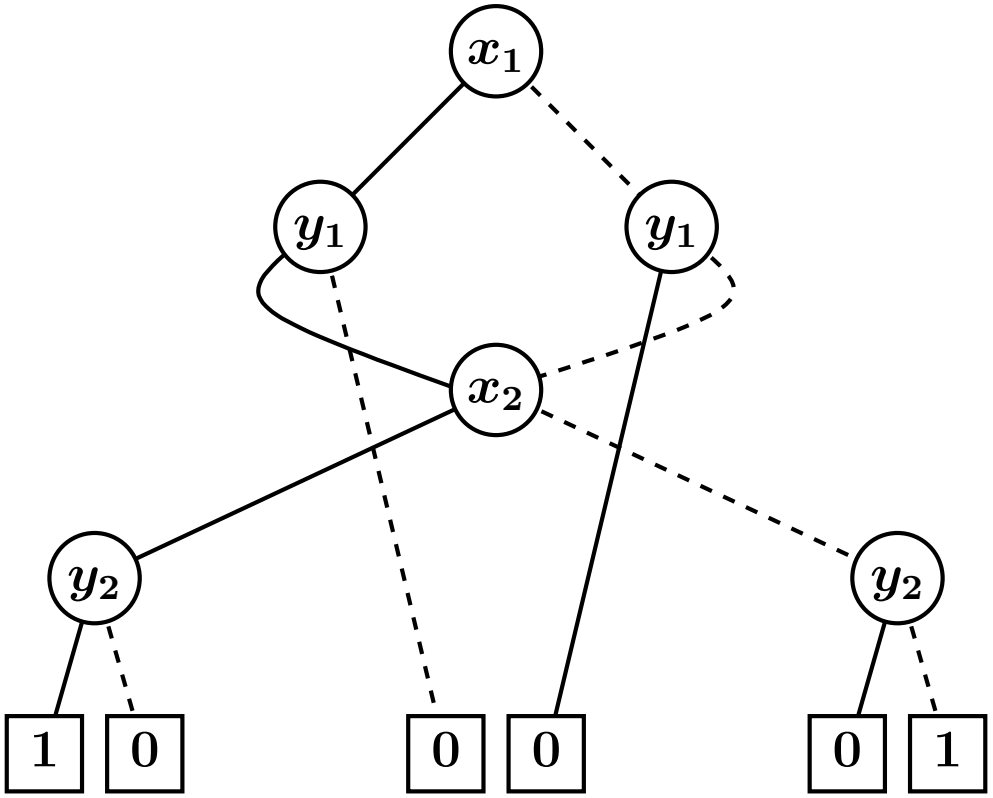
Reducción de DDB

- Ejemplo: $F = (x_1 \leftrightarrow y_1) \wedge (x_2 \leftrightarrow y_2)$

```
((2 x1 3 4)
 (3 y1 5 0)
 (4 y1 0 5)
 (5 x2 7 8)

 (7 y2 1 0)
 (8 y2 0 1)


 (1 1)
 (0 0))
```



Reducción de DDB

- Detección de nodos duplicados
  - Implementación: **`(nodos-duplicados`** D**`)`**
  - Ejemplos:

```
> (nodos-duplicados '((2 x1 0 1) (3 x1 0 1) (1 1) (0 0)))
((2 X1 0 1) (3 X1 0 1))
> (nodos-duplicados '((2 x1 0 1) (3 x2 0 1) (1 1) (1 1)))
((1 1) (1 1))
```

- Eliminación de nodos duplicados
  - Implementación: **`(elimina-duplicados`** D**`)`**
  - Ejemplos

```
> (elimina-duplicados '((2 x1 0 1) (3 x1 0 1) (1 1) (1 1)))
((2 X1 0 1) (1 1))
```

Reducción de DDB

- Detección de nodos redundantes
  - Implementación: **`(nodo-redundante`** D**`)`**
  - Ejemplos:

```
> (nodo-redundante '((2 x1 0 1) (3 x1 0 0) (1 1) (0 0)))
(3 X1 0 0)
> (nodo-redundante '((2 x1 3 3) (3 x2 0 1) (1 1) (1 1)))
(2 X1 3 3)
```

- Eliminación de nodos redundantes
  - Implementación: **`(elimina-redundantes`** D**`)`**
  - Ejemplos

```
> (elimina-redundantes '((2 x1 3 3) (3 x1 1 1) (1 1)))
((1 1))
```

Reducción de DDB

- La eliminación de nodos redundantes puede generar nodos duplicados
- La eliminación de nodos duplicados puede generar nodos redundantes
- Implementación: **`(reduce-ddb`** D**`)`**
- Ejemplos:

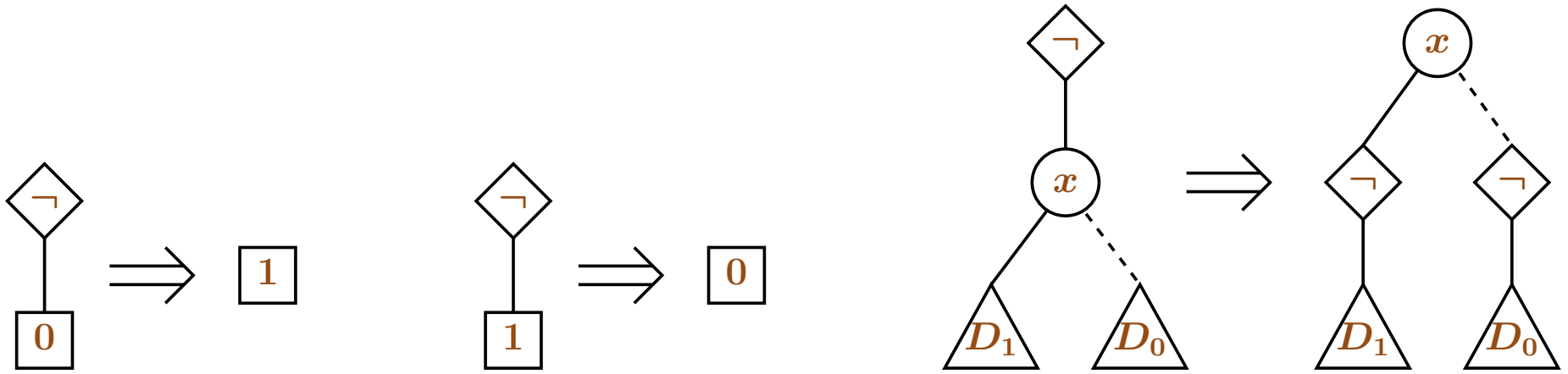
```
> (reduce-ddb '((2 x1 3 4) (3 y1 5 0) (4 y1 0 6) (5 x2 7 8)
                (6 x2 9 10) (7 y2 1 0) (8 y2 0 1) (9 y2 1 0)
                (10 y2 0 1) (1 1) (0 0))
((2 X1 3 4) (3 Y1 5 0) (4 Y1 0 5) (5 X2 7 8)
 (7 Y2 1 0) (8 Y2 0 1) (1 1) (0 0))
```

Operaciones con DDBRO

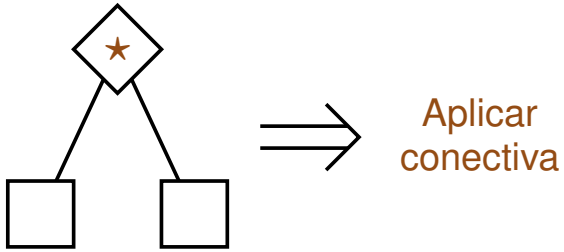
- Conmutatividad de $\twoheadrightarrow$ con respecto al resto de conectivas:
  - $\neg(x \twoheadrightarrow F_1, F_2) \equiv x \twoheadrightarrow \neg F_1, \neg F_2$
  - para $\star \in \{\vee, \wedge, \rightarrow, \leftrightarrow\}$
    - $(x \twoheadrightarrow F_1, F_2) \star G \equiv x \twoheadrightarrow F_1 \star G, F_2 \star G$
    - $(x \twoheadrightarrow F_1, F_2) \star (x \twoheadrightarrow G_1, G_2) \equiv x \twoheadrightarrow F_1 \star G_1, F_2 \star G_2$
- Construcción de un DDBRO para una fórmula F
  - Construir un árbol de decisión a partir de la tabla de verdad de F y reducirlo
  - Construir recursivamente DDBRO para las subfórmulas principales de F y aplicarles la conectiva principal de F

Operaciones con DDBRO

- Negación:

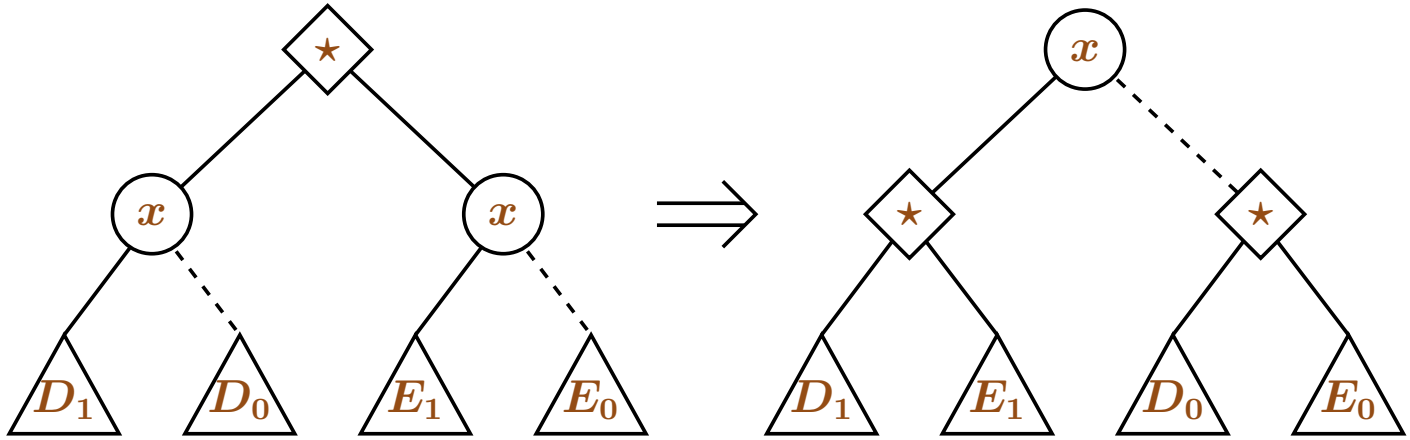


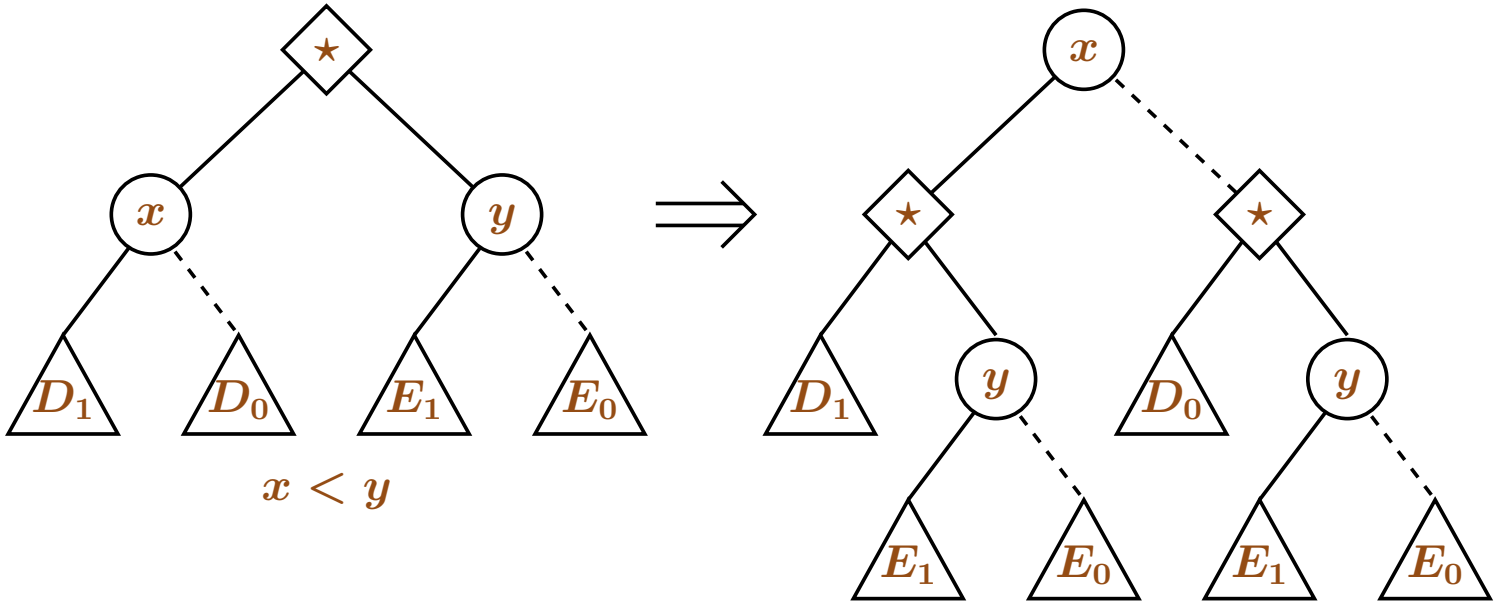
- Operadores binarios:



Operaciones con DDBRO

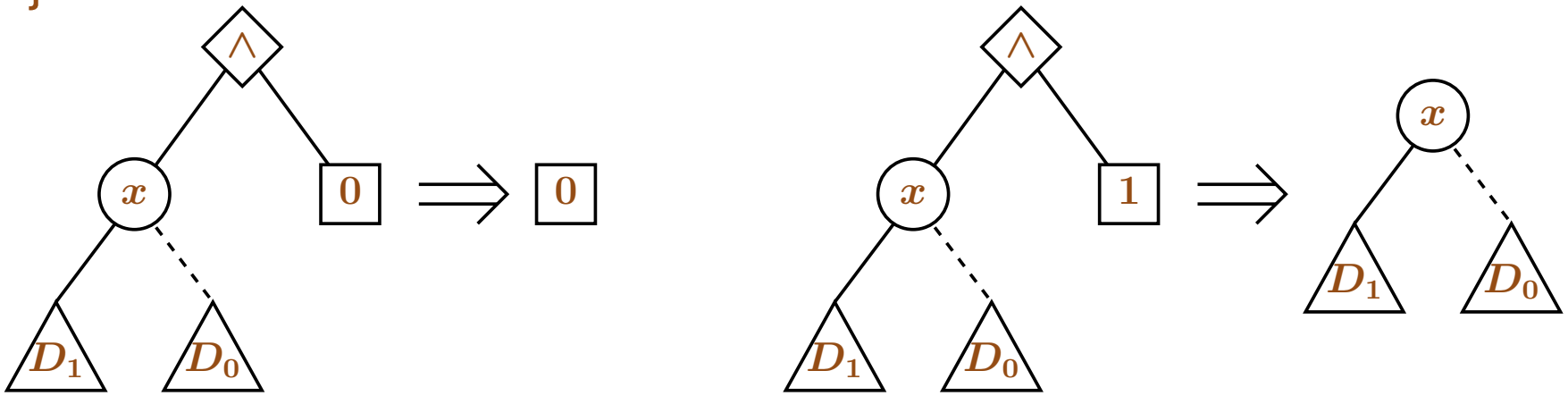
- Operadores binarios:



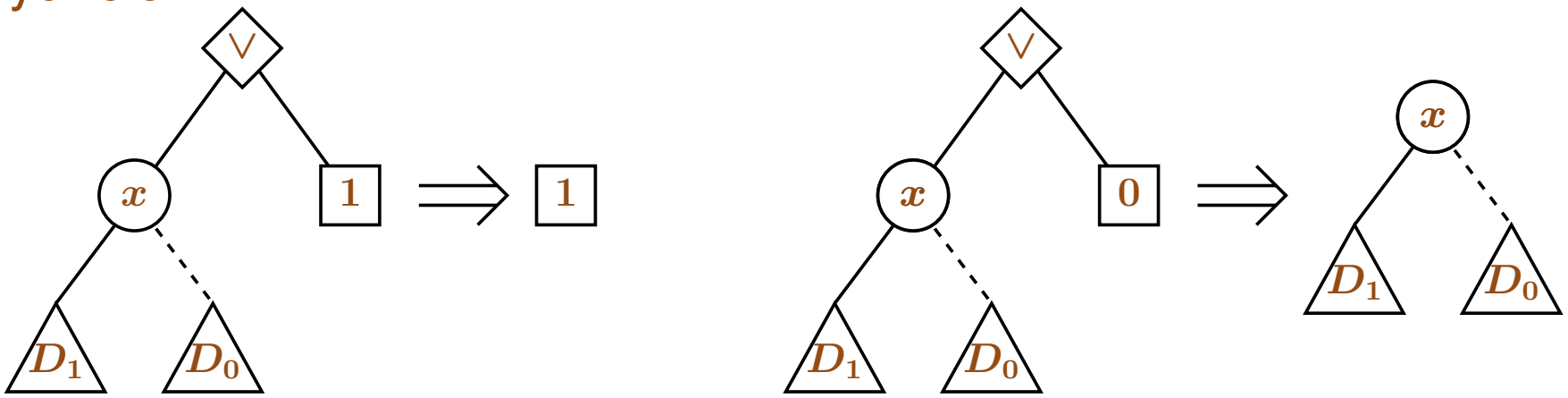


Operaciones con DDBRO

- Conjunción:

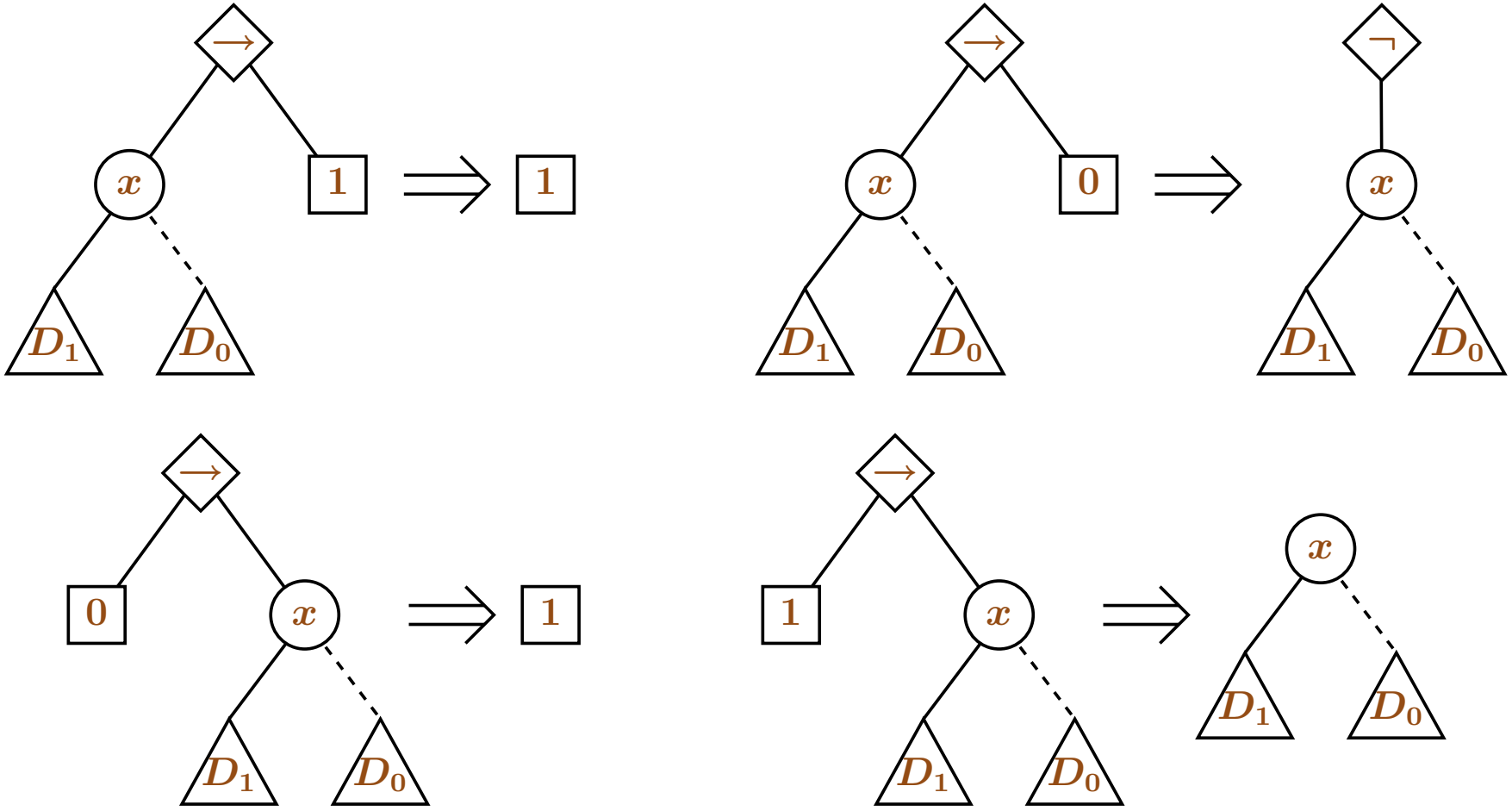


- Disyunción:



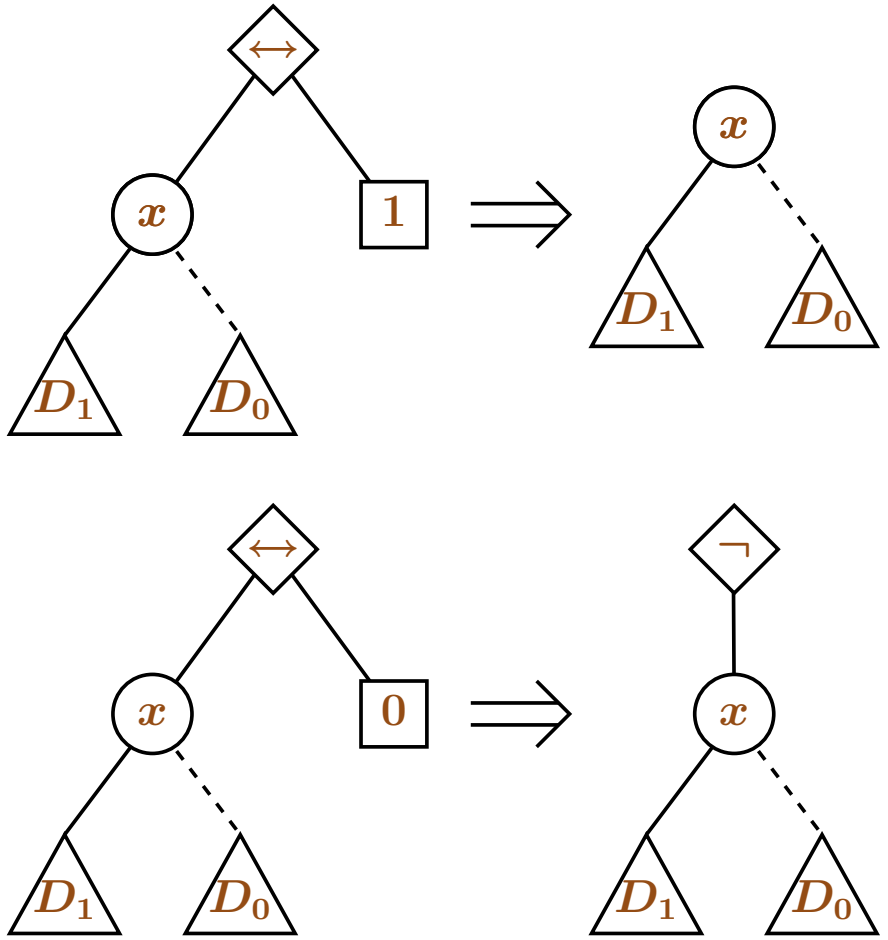
Operaciones con DDBRO

- Implicación:



Operaciones con DDBRO

- Equivalencia:



Operaciones con DDBRO

- Implementación:

  (aplica-negacion D**)**
  (aplica-conjuncion D E**)**
  (aplica-disyuncion D E**)**
  (aplica-implicacion D E**)**
  (aplica-equivalencia D E**)**

Ejercicios

- *Consideremos las fórmulas $F_1 = (x_1 \wedge x_2) \rightarrow \neg x_3$ y $F_2 = \neg x_1 \leftrightarrow (x_2 \vee \neg x_4)$. Se pide:*
  1. *Construir diagramas de decisición binarios reducidos y ordenados para F_1 y F_2 con el siguiente orden entre las variables: $x_1 < x_2 < x_3 < x_4$*
  2. *Aplicar la operación conjunción a los diagramas anteriores*
- *Consideremos las fórmulas $F_1 = (x_1 \rightarrow x_2) \wedge \neg x_3$ y $F_2 = (\neg x_1 \vee x_4) \leftrightarrow \neg x_2$. Se pide:*
  1. *Construir diagramas de decisición binarios reducidos y ordenados para F_1 y F_2 con el siguiente orden entre las variables: $x_1 < x_2 < x_3 < x_4$*
  2. *Aplicar la operación implicación a los diagramas anteriores*

Bibliografía

- Bryant, R.E. Graph-based algorithms for Boolean function manipulation *IEEE Transactions on Computers*, 8(C-35):677–691, 1986.
- Bryant, R.E. Symbolic Boolean manipulation with ordered binary-decision diagrams. *ACM Computing Surveys*, 24(3):293–318, September 1992.