



Programación Funcional

Clases teóricas

por Pablo E. “Fidel” Martínez López

7. Inducción y recursión I

To infinity and beyond!



Motivación de inducción y recursión

Motivación de inducción y recursión

- Quedaron planteadas 3 preguntas
 - ¿Cómo definir tipos algebraicos con infinitos elementos?
 - ¿Cómo saber si las ecuaciones son orientadas?
 - ¿Qué falta para poder demostrar en estos casos?

Motivación de inducción y recursión

- ❑ Quedaron planteadas 3 preguntas
 - ❑ ¿Cómo definir tipos algebraicos con infinitos elementos?
 - ❑ ¿Cómo saber si las ecuaciones son orientadas?
 - ❑ ¿Qué falta para poder demostrar en estos casos?
- ❑ Parece ser necesaria ***inducción o recursión***
 - ❑ ¿Realmente entendemos la inducción matemática?
 - ❑ ¿Qué es la inducción? ¿Qué es la recursión?



Inducción estructural

Inducción estructural

- ❏ La ***inducción estructural*** es una *técnica* que responde las 3 preguntas planteadas

Inducción estructural

- ❑ La ***inducción estructural*** es una *técnica* que responde las 3 preguntas planteadas
 - ❑ Permite
 - ❑ Definir tipos algebraicos con infinitos elementos

Inducción estructural

- ❑ La **inducción estructural** es una *técnica* que responde las 3 preguntas planteadas
 - ❑ Permite
 - ❑ Definir tipos algebraicos con infinitos elementos
 - ❑ Definir ecuaciones *orientadas* que usan la función que se está definiendo del lado derecho

Inducción estructural

- ❑ La **inducción estructural** es una *técnica* que responde las 3 preguntas planteadas
 - ❑ Permite
 - ❑ Definir tipos algebraicos con infinitos elementos
 - ❑ Definir ecuaciones orientadas que usan la función que se está definiendo del lado derecho
 - ❑ Probar propiedades sobre elementos de estos conjuntos

Inducción estructural

- ❑ La **inducción estructural** es una *técnica* que responde las 3 preguntas planteadas
 - ❑ Permite
 - ❑ Definir tipos algebraicos con infinitos elementos
 - ❑ Definir ecuaciones orientadas que usan la función que se está definiendo del lado derecho
 - ❑ Probar propiedades sobre elementos de estos conjuntos
- ❑ Operacionalmente la llamamos **recursión estructural**

Inducción estructural

- Ejemplo (dejando Haskell de lado por ahora)
 - Definir el conjunto \mathcal{X} de cadenas de S s terminadas en Z
 - ¿Cómo hacerlo constructivamente?

Inducción estructural

- ❑ Ejemplo (dejando Haskell de lado por ahora)
 - ❑ Definir el conjunto \mathcal{N} de cadenas de S s terminadas en Z
 - ❑ ¿Cómo hacerlo constructivamente?
 - ❑ Podrían ponerse condiciones al conjunto buscado...
 - ❑ **Condición 1:** Z tiene que estar en \mathcal{N} (Tiene 0 S s)
 - ❑ **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}

Inducción estructural

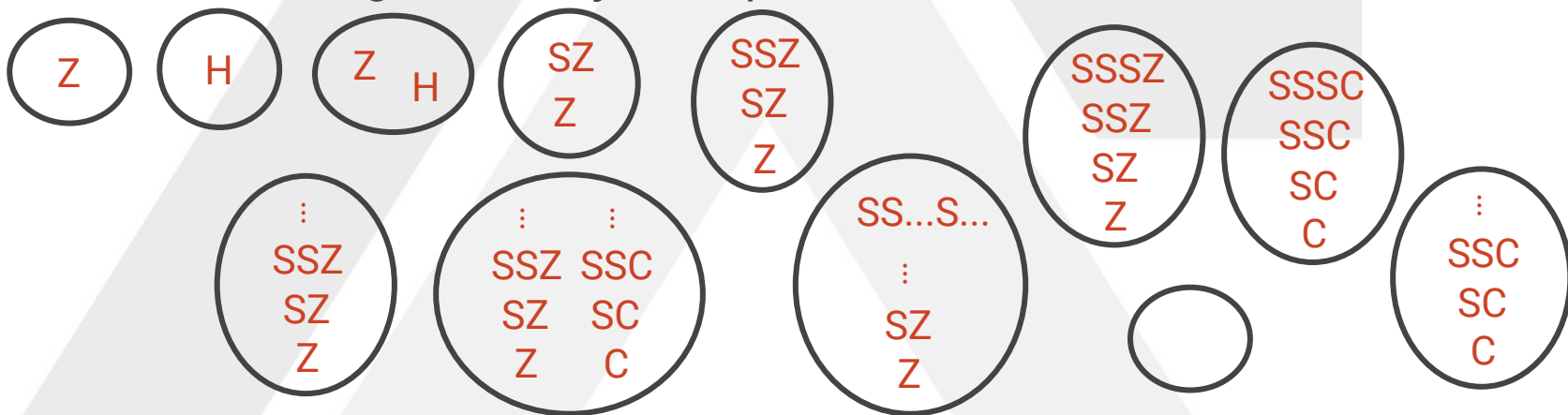
- ❑ Ejemplo (dejando Haskell de lado por ahora)
 - ❑ Definir el conjunto \mathcal{N} de cadenas de S s terminadas en Z
 - ❑ ¿Cómo hacerlo constructivamente?
 - ❑ Podrían ponerse condiciones al conjunto buscado...
 - ❑ **Condición 1:** Z tiene que estar en \mathcal{N} (Tiene 0 S s)
 - ❑ **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
 - ❑ ¿Cómo saber si un conjunto dado es \mathcal{N} ?

Inducción estructural

- ❑ ¿Cómo saber qué conjuntos satisfacen las condiciones?
 - ❑ **Condición 1:** Z tiene que estar en \mathcal{N}
 - ❑ **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
- ❑ Analicemos algunos conjuntos posibles

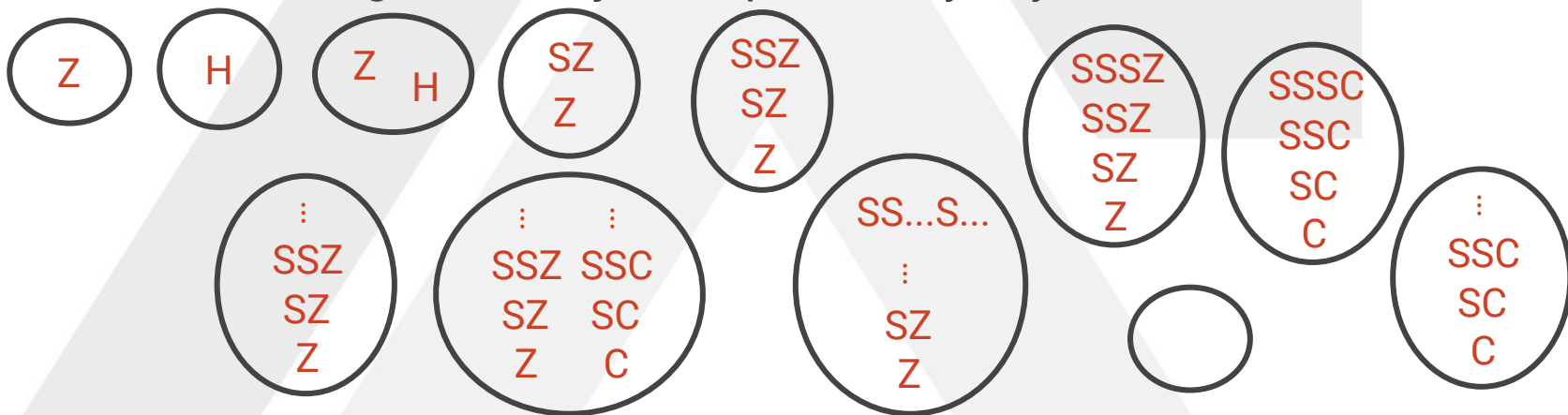
Inducción estructural

- ¿Cómo saber qué conjuntos satisfacen las condiciones?
 - Condición 1:** Z tiene que estar en \mathcal{N}
 - Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
- Analicemos algunos conjuntos posibles



Inducción estructural

- ¿Cómo saber qué conjuntos satisfacen las condiciones?
 - Condición 1:** Z tiene que estar en \mathcal{N}
 - Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
- Analicemos algunos conjuntos posibles y vayamos descartando



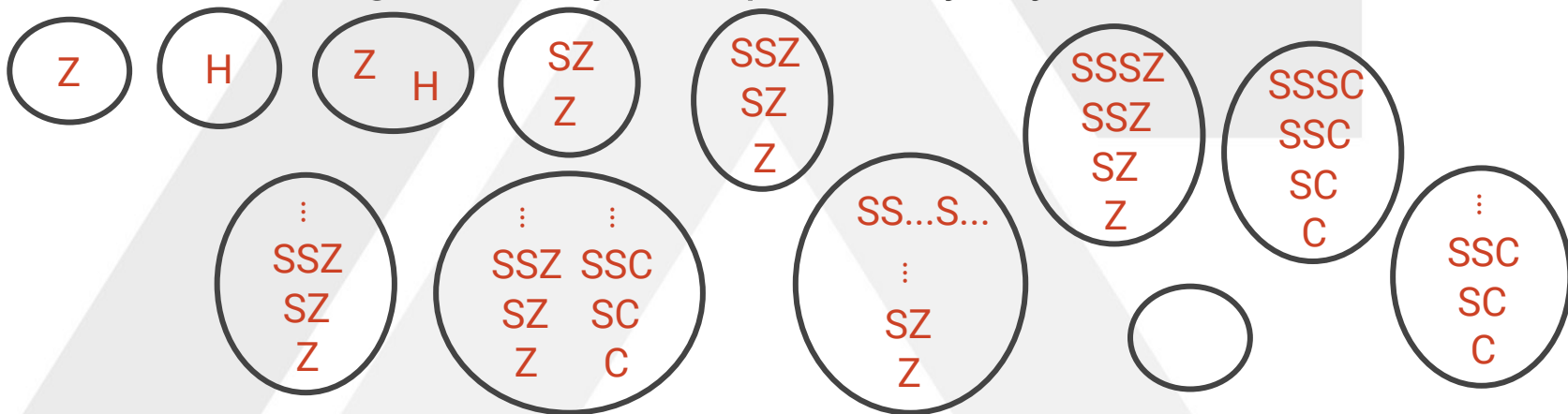
Inducción estructural

❏ ¿Cómo saber qué conjuntos satisfacen las condiciones?

→ ❏ **Condición 1:** Z tiene que estar en \mathcal{N}

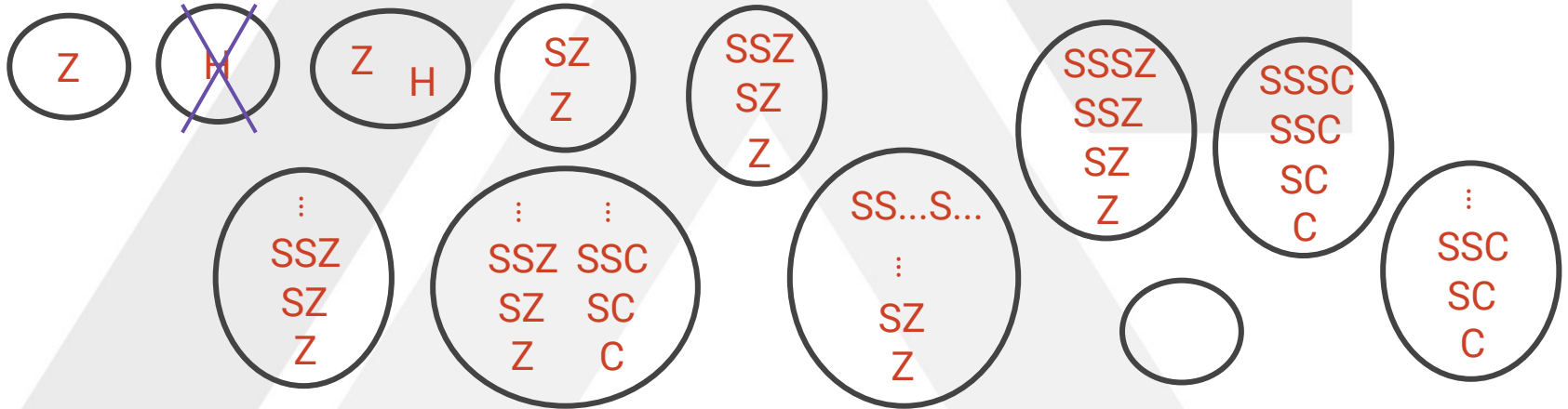
❏ **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}

❏ Analicemos algunos conjuntos posibles y vayamos descartando



Inducción estructural

- ¿Cómo saber qué conjuntos satisfacen las condiciones?
 - **Condición 1:** Z tiene que estar en \mathcal{N}
 - Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
- Analicemos algunos conjuntos posibles y vayamos descartando



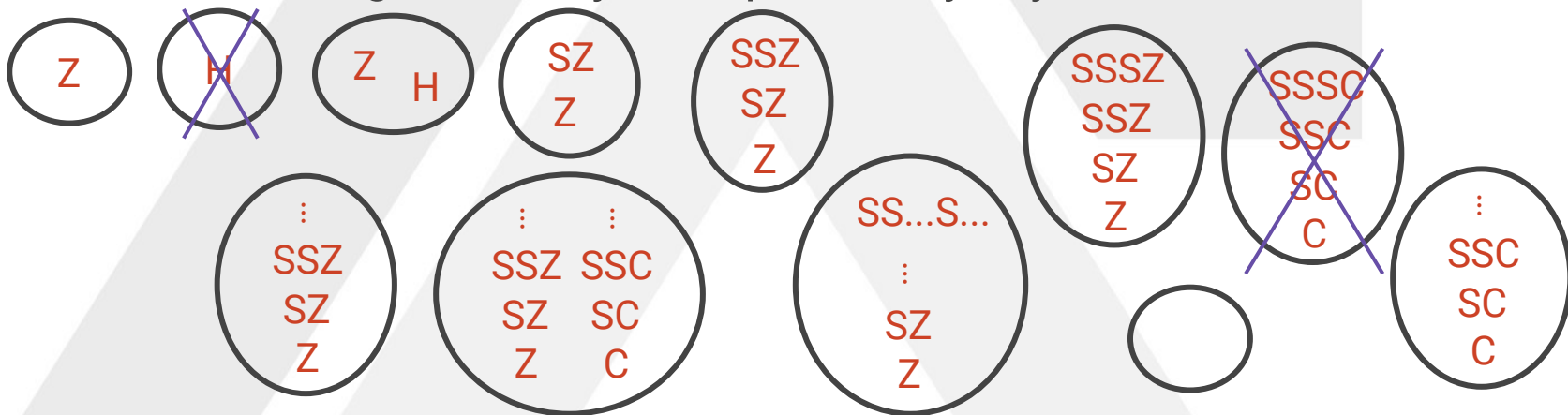
Inducción estructural

❑ ¿Cómo saber qué conjuntos satisfacen las condiciones?

➔ ❑ **Condición 1:** Z tiene que estar en \mathcal{N}

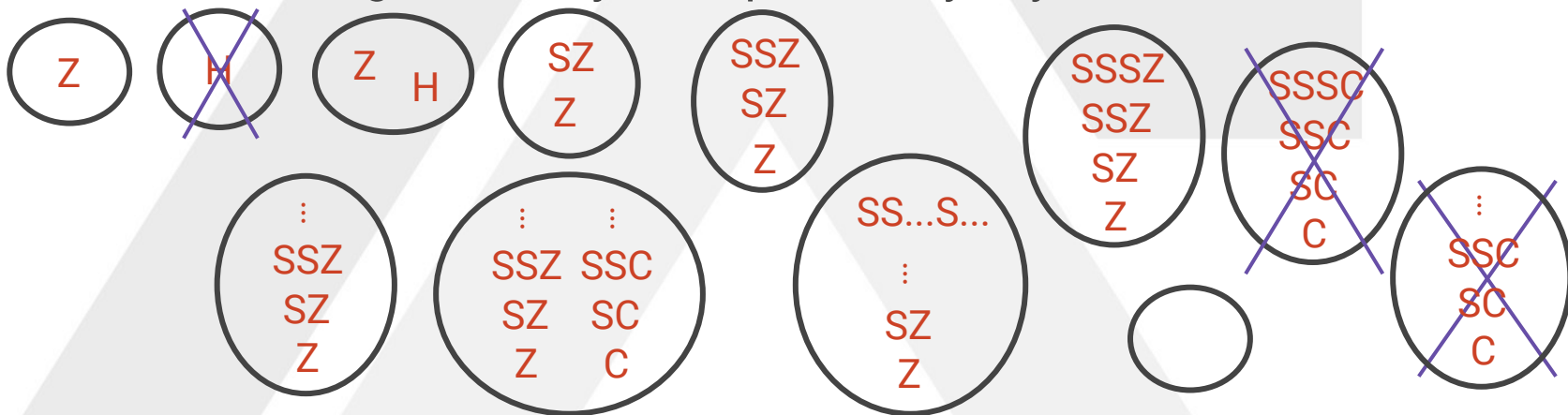
❑ **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}

❑ Analicemos algunos conjuntos posibles y vayamos descartando



Inducción estructural

- ¿Cómo saber qué conjuntos satisfacen las condiciones?
- **Condición 1:** z tiene que estar en \mathcal{N}
- Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
- Analicemos algunos conjuntos posibles y vayamos descartando



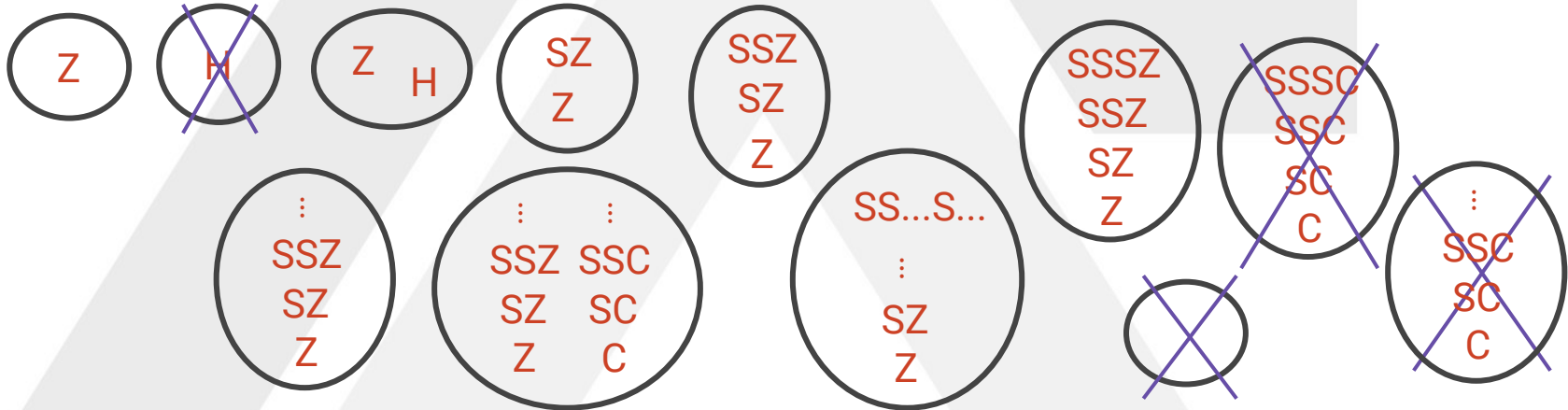
Inducción estructural

❑ ¿Cómo saber qué conjuntos satisfacen las condiciones?

➔ ❑ **Condición 1:** Z tiene que estar en \mathcal{N}

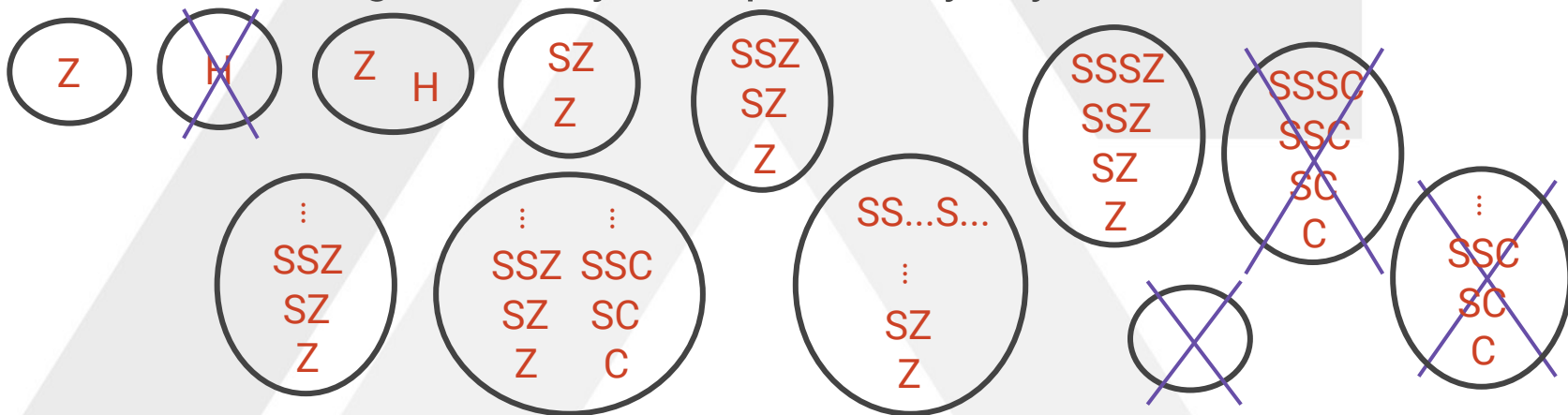
❑ **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}

❑ Analicemos algunos conjuntos posibles y vayamos descartando



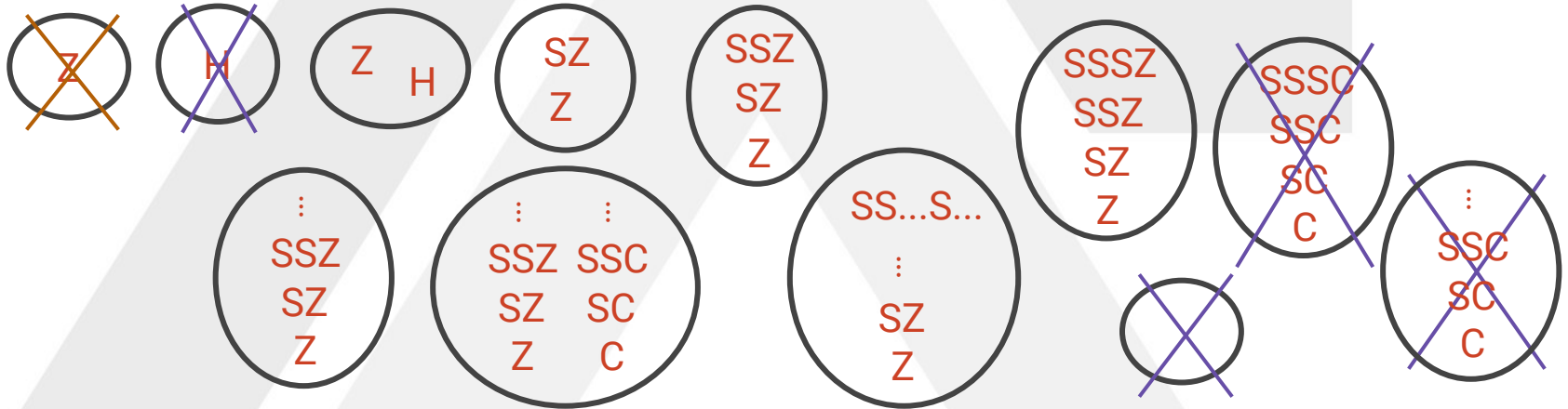
Inducción estructural

- ¿Cómo saber qué conjuntos satisfacen las condiciones?
 - Condición 1:** Z tiene que estar en \mathcal{N}
 - **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
- Analicemos algunos conjuntos posibles y vayamos descartando



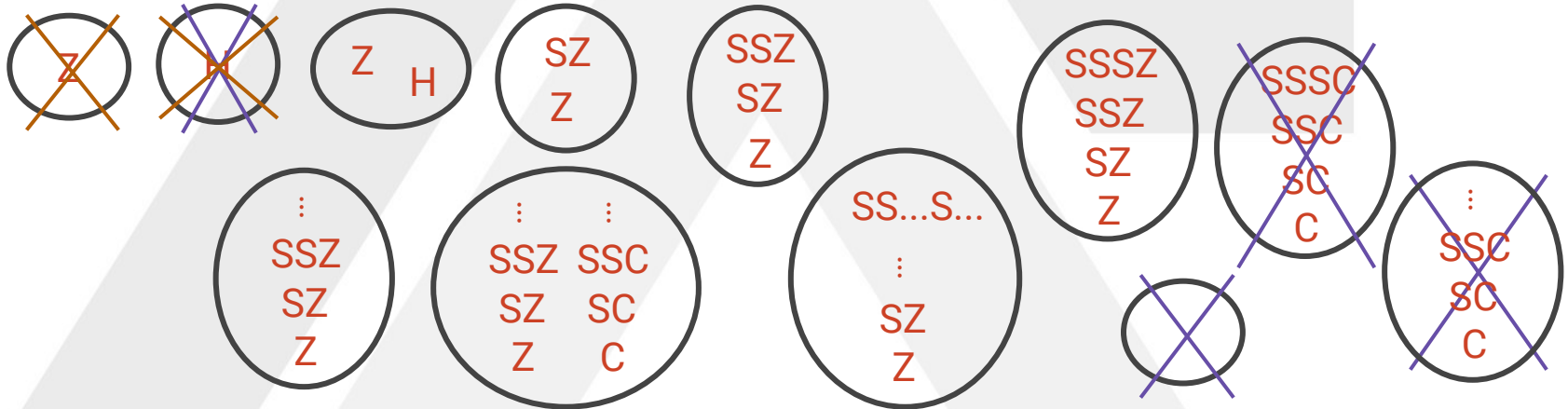
Inducción estructural

- ¿Cómo saber qué conjuntos satisfacen las condiciones?
 - Condición 1:** Z tiene que estar en \mathcal{N}
 - **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
- Analicemos algunos conjuntos posibles y vayamos descartando



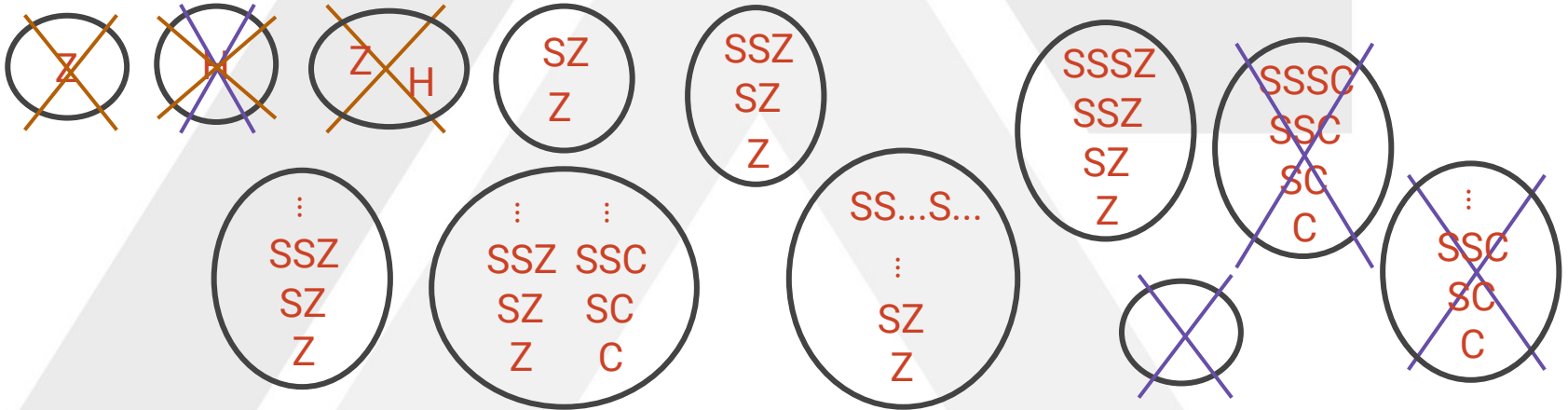
Inducción estructural

- ¿Cómo saber qué conjuntos satisfacen las condiciones?
 - Condición 1:** Z tiene que estar en \mathcal{N}
 - **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
- Analicemos algunos conjuntos posibles y vayamos descartando



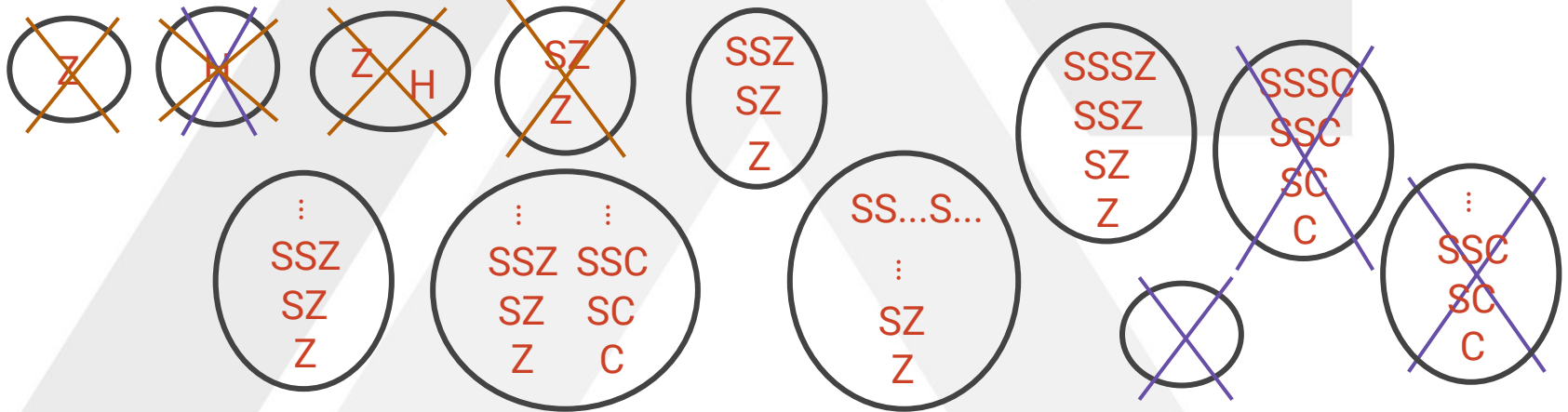
Inducción estructural

- ¿Cómo saber qué conjuntos satisfacen las condiciones?
 - Condición 1:** Z tiene que estar en \mathcal{N}
 - **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
- Analicemos algunos conjuntos posibles y vayamos descartando



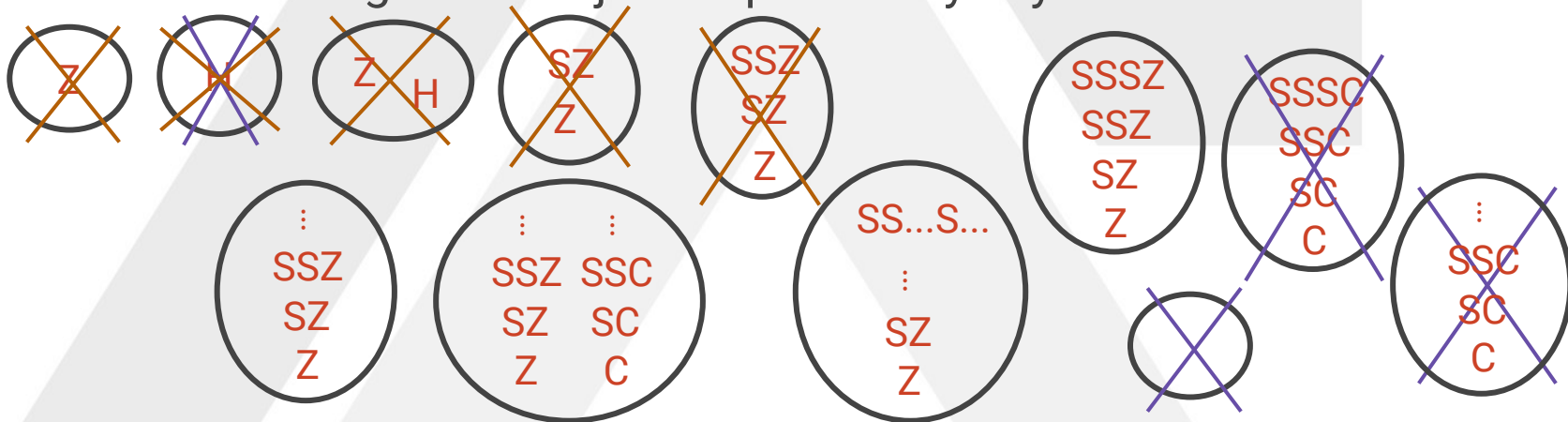
Inducción estructural

- ¿Cómo saber qué conjuntos satisfacen las condiciones?
 - Condición 1:** Z tiene que estar en \mathcal{N}
 - **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
- Analicemos algunos conjuntos posibles y vayamos descartando



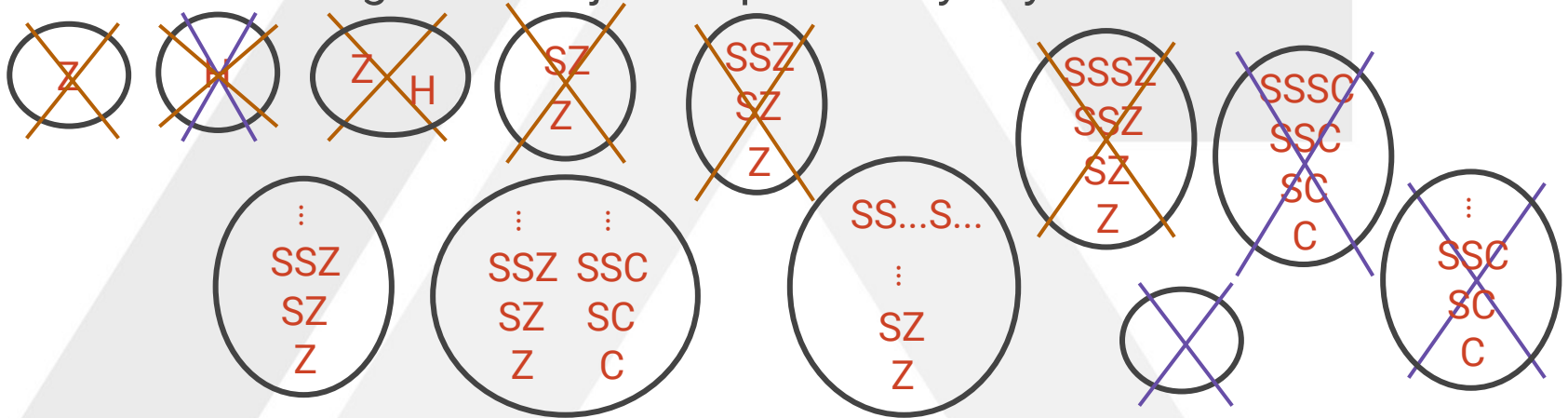
Inducción estructural

- ❑ ¿Cómo saber qué conjuntos satisfacen las condiciones?
 - ❑ **Condición 1:** Z tiene que estar en \mathcal{N}
 - ➔ ❑ **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
- ❑ Analicemos algunos conjuntos posibles y vayamos descartando



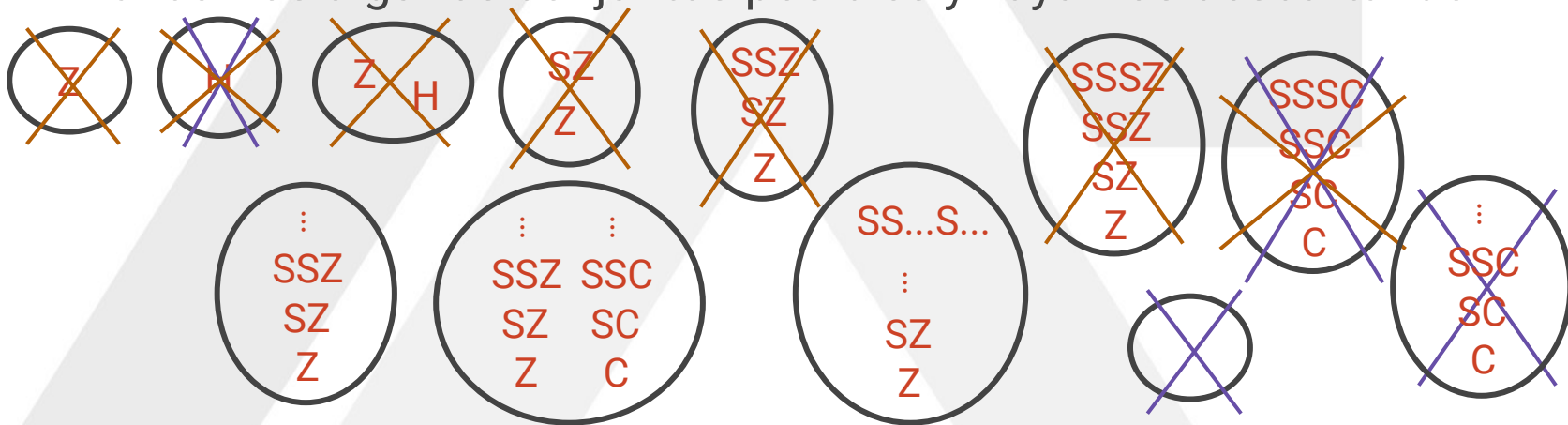
Inducción estructural

- ❑ ¿Cómo saber qué conjuntos satisfacen las condiciones?
 - ❑ **Condición 1:** Z tiene que estar en \mathcal{N}
 - ➔ ❑ **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
- ❑ Analicemos algunos conjuntos posibles y vayamos descartando



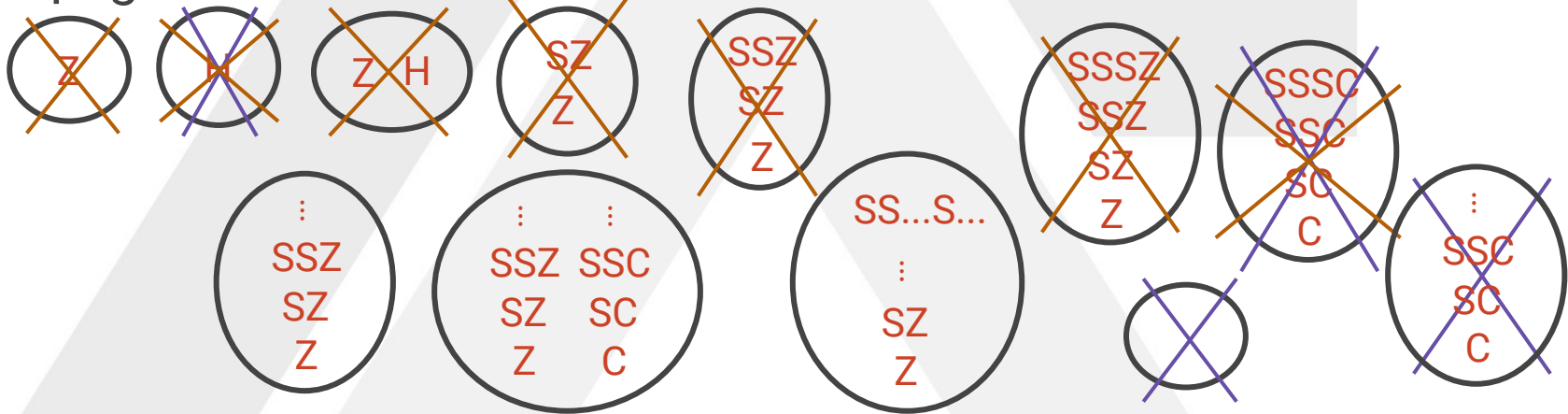
Inducción estructural

- ❑ ¿Cómo saber qué conjuntos satisfacen las condiciones?
 - ❑ **Condición 1:** Z tiene que estar en \mathcal{N}
 - ➔ ❑ **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
- ❑ Analicemos algunos conjuntos posibles y vayamos descartando



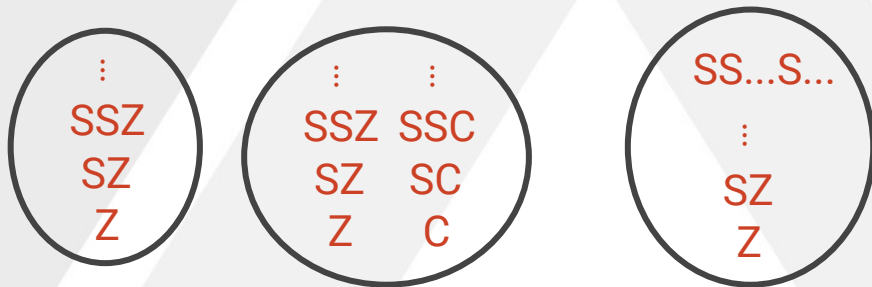
Inducción estructural

- ¿Cómo saber qué conjuntos satisfacen las condiciones?
 - Condición 1:** Z tiene que estar en \mathcal{N}
 - Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
- ¡Sigue habiendo muchos válidos!



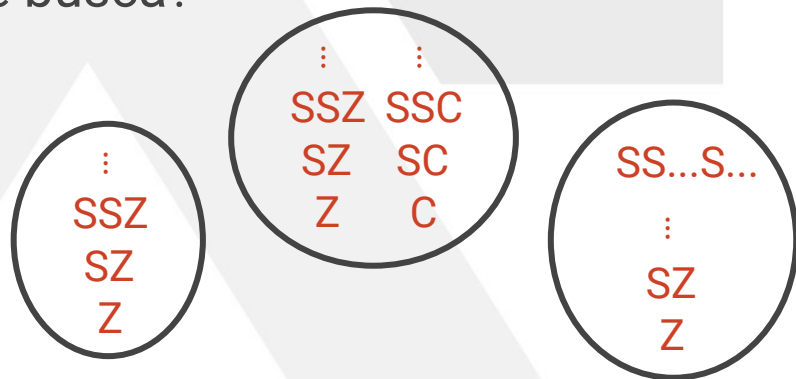
Inducción estructural

- ❏ ¿Cómo saber qué conjuntos satisfacen las condiciones?
 - ❏ **Condición 1:** Z tiene que estar en \mathcal{N}
 - ❏ **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
- ❏ ¡Sigue habiendo muchos válidos!



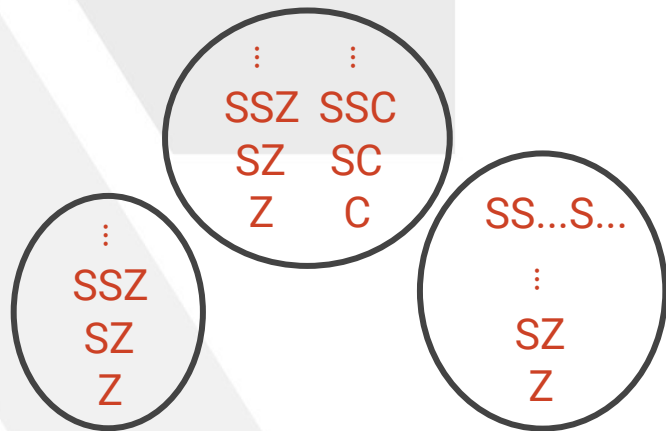
Inducción estructural

- ❑ ¿Cómo saber qué conjuntos satisfacen las condiciones?
 - ❑ **Condición 1:** Z tiene que estar en \mathcal{N}
 - ❑ **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
- ❑ ¡Sigue habiendo muchos válidos!
 - ❑ ¿Qué diferencia al que se busca?



Inducción estructural

- ❑ ¿Cómo saber qué conjuntos satisfacen las condiciones?
 - ❑ **Condición 1:** Z tiene que estar en \mathcal{N}
 - ❑ **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
- ❑ ¡Sigue habiendo muchos válidos!
 - ❑ ¿Qué diferencia al que se busca?



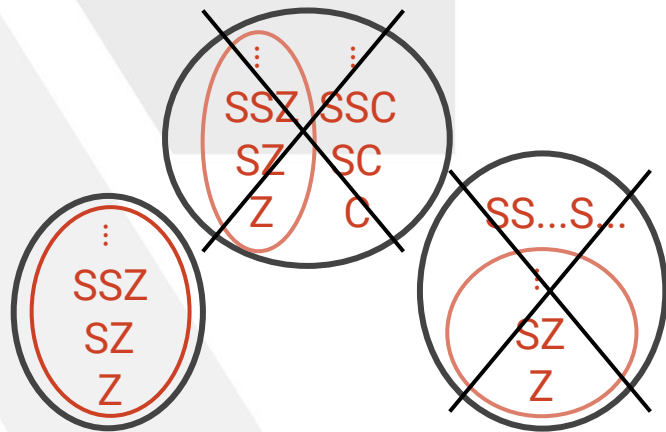
Inducción estructural

- ❑ ¿Cómo saber qué conjuntos satisfacen las condiciones?
 - ❑ **Condición 1:** Z tiene que estar en \mathcal{N}
 - ❑ **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
- ❑ ¡Sigue habiendo muchos válidos!
 - ❑ ¿Qué diferencia al que se busca?
 - ❑ ¡Está contenido en todos los que cumplen las propiedades!



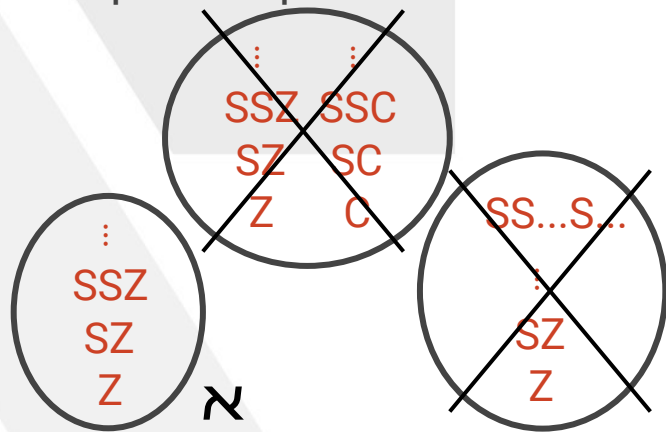
Inducción estructural

- ❑ ¿Cómo saber qué conjuntos satisfacen las condiciones?
 - ❑ **Condición 1:** Z tiene que estar en \mathcal{N}
 - ❑ **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
- ❑ ¡Sigue habiendo muchos válidos!
 - ❑ ¿Qué diferencia al que se busca?
 - ❑ ¡Está contenido en todos los que cumplen las propiedades!
 - ❑ **Condición adicional:** es el *menor* de los que cumplen las anteriores



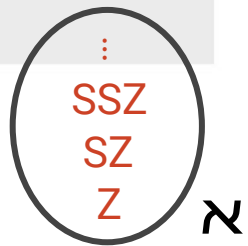
Inducción estructural

- Definición del conjunto \mathcal{N} buscado
 - Condición 1:** Z tiene que estar en \mathcal{N}
 - Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
 - Condición adicional:** es el *menor* de los que cumplen



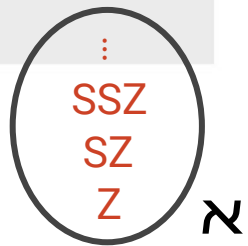
Inducción estructural

- Definición del conjunto \mathcal{N} buscado
 - Condición 1:** Z tiene que estar en \mathcal{N}
 - Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
 - Condición adicional:** es el *menor* de los que cumplen



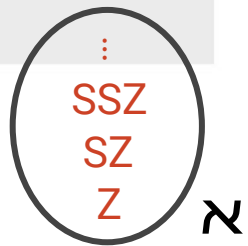
Inducción estructural

- Definición del conjunto \mathcal{N} buscado
 - Condición 1:** Z tiene que estar en \mathcal{N}
 - Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
 - Condición adicional:** es el *menor* de los que cumplen
- ¿Qué características tiene esta definición?



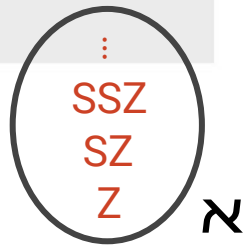
Inducción estructural

- Definición del conjunto \mathcal{N} buscado
 - Condición 1:** Z tiene que estar en \mathcal{N}
 - Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
 - Condición adicional:** es el *menor* de los que cumplen
- ¿Qué características tiene esta definición?
 - Una condición es una *afirmación*



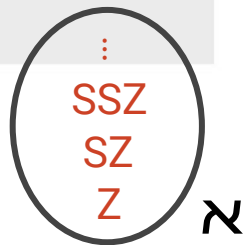
Inducción estructural

- Definición del conjunto \mathcal{N} buscado
 - Condición 1:** Z tiene que estar en \mathcal{N}
 - Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
 - Condición adicional:** es el *menor* de los que cumplen
- ¿Qué características tiene esta definición?
 - Una condición es una *afirmación*
 - Una condición es una *implicación*



Inducción estructural

- Definición del conjunto \mathcal{N} buscado
 - Condición 1:** Z tiene que estar en \mathcal{N}
 - Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
 - Condición adicional:** es el *menor* de los que cumplen
- ¿Qué características tiene esta definición?
 - Una condición es una *afirmación*
 - Una condición es una *implicación*
 - La condición extra provee *unicidad*




Inducción estructural

- ❑ ¿Cualquier conjunto de condiciones sirve?
- ❑ Intento de definición “inductiva” del conjunto \mathcal{N}
 - ❑ **Condición 1:** z tiene que estar en \mathcal{N}
 - ❑ **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
 - ❑ **Condición 3:** si c está en \mathcal{N} , entonces SSc tiene que estar en \mathcal{N}

Inducción estructural

- ❑ ¿Cualquier conjunto de condiciones sirve?
- ❑ Intento de definición “inductiva” del conjunto \mathcal{N}
 - ❑ **Condición 1:** Z tiene que estar en \mathcal{N}
 - ❑ **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
 - ❑ **Condición 3:** si c está en \mathcal{N} , entonces SSc tiene que estar en \mathcal{N}



TODOS los conjuntos que puede descartar la condición 3, también son descartados por la condición 2

Inducción estructural

- ❑ ¿Cualquier conjunto de condiciones sirve?
- ❑ Intento de definición “inductiva” del conjunto \mathcal{N}
 - ❑ **Condición 1:** Z tiene que estar en \mathcal{N}
 - ❑ **Condición 2:** si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
 - ❑ **Condición 3:** si c está en \mathcal{N} , entonces SSc tiene que estar en \mathcal{N}
- ❑ Observar que la condición 3 no aporta información nueva
 - ❑ No es *productiva*
 - ❑ Una definición por inducción necesita ***reglas productivas***

Inducción estructural

- ❏ La definición de un conjunto por ***inducción estructural*** se compone de condiciones (llamadas **reglas**)

Inducción estructural

- ❑ La definición de un conjunto por ***inducción estructural*** se compone de condiciones (llamadas ***reglas***)
 - ❑ Una o varias ***reglas base***: afirmaciones directas

Inducción estructural

- ❑ La definición de un conjunto por ***inducción estructural*** se compone de condiciones (llamadas **reglas**)
 - ❑ Una o varias ***reglas base***: afirmaciones directas
 - ❑ Una o varias ***reglas inductivas***: implicaciones (productivas)

Inducción estructural

- ❑ La definición de un conjunto por ***inducción estructural*** se compone de condiciones (llamadas **reglas**)
 - ❑ Una o varias **reglas base**: afirmaciones directas
 - ❑ Una o varias **reglas inductivas**: implicaciones (productivas)
 - ❑ Una única **condición adicional**: pedir que sea el *menor*

Inducción estructural

- ❑ La definición de un conjunto por **inducción estructural** se compone de condiciones (llamadas **reglas**)
 - ❑ Una o varias **reglas base**: afirmaciones directas
 - ❑ e.g. Z tiene que estar en \mathcal{N}
 - ❑ Una o varias **reglas inductivas**: implicaciones (productivas)
 - ❑ Una única **condición adicional**: pedir que sea el *menor*

Inducción estructural

- ❑ La definición de un conjunto por **inducción estructural** se compone de condiciones (llamadas **reglas**)
 - ❑ Una o varias **reglas base**: afirmaciones directas
 - ❑ e.g. **Z** tiene que estar en \mathcal{N}
 - ❑ Una o varias **reglas inductivas**: implicaciones (productivas)
 - ❑ e.g. si **c** está en \mathcal{N} , entonces **Sc** tiene que estar en \mathcal{N}
 - ❑ Una única **condición adicional**: pedir que sea el *menor*

Inducción estructural

- ❑ La definición de un conjunto por **inducción estructural** se compone de condiciones (llamadas **reglas**)
 - ❑ Una o varias **reglas base**: afirmaciones directas
 - ❑ e.g. Z tiene que estar en \mathcal{N}
 - ❑ Una o varias **reglas inductivas**: implicaciones (productivas)
 - ❑ e.g. si c está en \mathcal{N} , entonces Sc tiene que estar en \mathcal{N}
 - ❑ Una única **condición adicional**: pedir que sea el *menor*
 - ❑ el menor que cumple todas las condiciones anteriores
 - ❑ provee *unicidad* al conjunto buscado

Inducción estructural

Definición de un conjunto por ***inducción estructural***

- ❑ Una o varias ***reglas base*** (afirmaciones directas)
- ❑ Una o varias ***reglas inductivas*** (implicaciones productivas)
- ❑ Una única ***condición adicional*** (pedir que sea el *menor*)

Inducción estructural

Definición de un conjunto S por *inducción estructural*

- ❏ **Reglas base:** z_1 está en S
...
 z_n está en S
- ❏ **Reglas inductivas:** si e_{11}, \dots, e_{i1} están en S , entonces e_1 está en S
...
si e_{1k}, \dots, e_{ik} están en S , entonces e_k está en S
- ❏ El *MENOR* conjunto que cumple todas las reglas (en el sentido de la inclusión)

Inducción estructural

- ❑ Conjunto definido por ***inducción estructural***
 - ❑ También llamado ***conjunto inductivo***
 - ❑ Se define mediante *reglas básicas y reglas inductivas*
 - ❑ Siempre está la condición de ser el *menor* conjunto que cumple todas las reglas (en el sentido de la inclusión)
 - ❑ Las reglas pueden usarse de diversas formas
 - ❑ Como condiciones
 - ❑ Como formas de agregar elementos
 - ❑ Para verificar si un elemento es miembro del conjunto

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)

Inducción estructural

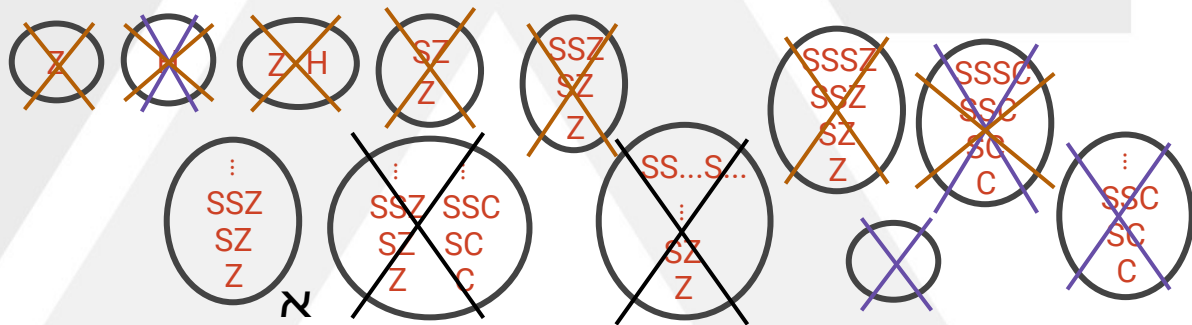
Definición *inductiva* de \mathcal{N}

Regla base: Z está en \mathcal{N}

Regla inductiva: si c está en \mathcal{N} , entonces Sc está en \mathcal{N}

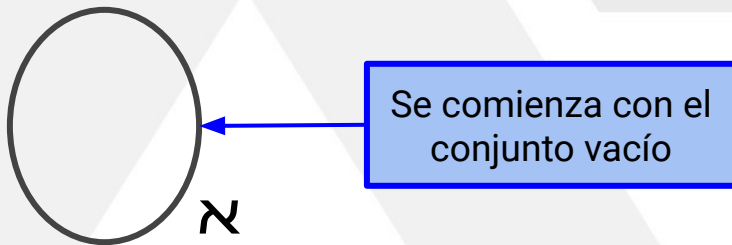
(La regla de ser el menor es implícita en el adjetivo “inductivo”)

Reglas como condiciones (ya visto)



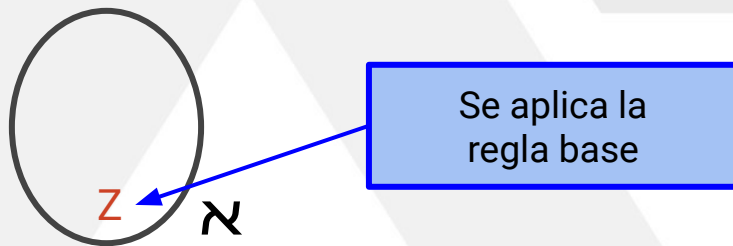
Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas como forma de agregar elementos



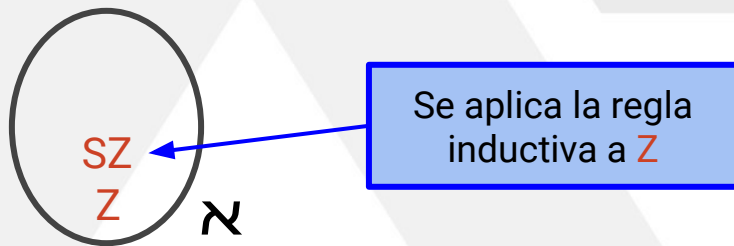
Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas como forma de agregar elementos



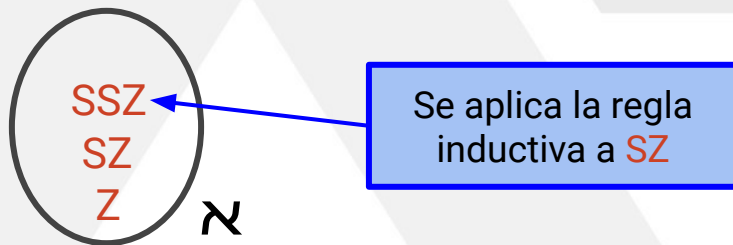
Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas como forma de agregar elementos



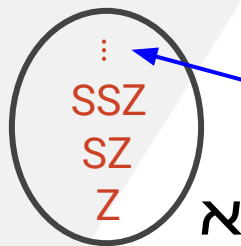
Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas como forma de agregar elementos



Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas como forma de agregar elementos



Y así para el resto...

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas para ver si un elemento es parte del conjunto

¿ $SSSSZ$ está en \mathcal{N} ?

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas para ver si un elemento es parte del conjunto

¿ $SSSSZ^c$ está en \mathcal{N} ?

¿ $SSSZ$ está en \mathcal{N} ?

Por la regla inductiva,
debería también valer que...

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas para ver si un elemento es parte del conjunto

¿ $SSSSZ$ está en \mathcal{N} ?

¿ $SSSZ^c$ está en \mathcal{N} ?

¿ SSZ está en \mathcal{N} ?

Por la regla inductiva,
debería también valer que...

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas para ver si un elemento es parte del conjunto

¿ $SSSSZ$ está en \mathcal{N} ?

¿ $SSSZ$ está en \mathcal{N} ?

¿ SSZ^c está en \mathcal{N} ?

¿ SZ está en \mathcal{N} ?

Por la regla inductiva,
debería también valer que...

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas para ver si un elemento es parte del conjunto

¿ $SSSSZ$ está en \mathcal{N} ?

¿ $SSSZ$ está en \mathcal{N} ?

¿ SSZ está en \mathcal{N} ?

¿ $S\boxed{Z}^c$ está en \mathcal{N} ?

¿ Z está en \mathcal{N} ?

Por la regla inductiva,
debería también valer que...

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas para ver si un elemento es parte del conjunto

¿ $SSSSZ$ está en \mathcal{N} ?

¿ $SSSZ$ está en \mathcal{N} ?

¿ SSZ está en \mathcal{N} ?

¿ SZ está en \mathcal{N} ?

¡ Z está en \mathcal{N} !

Por la regla base

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas para ver si un elemento es parte del conjunto

¿ $SSSSZ$ está en \mathcal{N} ?

¿ $SSSZ$ está en \mathcal{N} ?

¿ SSZ está en \mathcal{N} ?

¡ SZ está en \mathcal{N} !

¡ Z está en \mathcal{N} !

Por la regla inductiva

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas para ver si un elemento es parte del conjunto

¿ $SSSSZ$ está en \mathcal{N} ?

¿ $SSSZ$ está en \mathcal{N} ?

¡ SSZ está en \mathcal{N} !

¡ SZ está en \mathcal{N} !

¡ Z está en \mathcal{N} !

Por la regla inductiva

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas para ver si un elemento es parte del conjunto

¿ $SSSSZ$ está en \mathcal{N} ?

¡ $SSSZ$ está en \mathcal{N} !

¡ SSZ está en \mathcal{N} !

¡ SZ está en \mathcal{N} !

¡ Z está en \mathcal{N} !

Por la regla inductiva

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas para ver si un elemento es parte del conjunto

¡ $SSSSZ$ está en \mathcal{N} !

¡ $SSSZ$ está en \mathcal{N} !

¡ SSZ está en \mathcal{N} !

¡ SZ está en \mathcal{N} !

¡ Z está en \mathcal{N} !

Por la regla inductiva

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas para ver si un elemento es parte del conjunto
 - ¡ $SSSSZ$ está en \mathcal{N} !

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas para ver si un elemento es parte del conjunto
 - ¿ $SSSSZ$ está en \mathcal{N} ! ¿ $SSSSH$ está en \mathcal{N} ?

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas para ver si un elemento es parte del conjunto
 - ¡ $SSSSZ$ está en \mathcal{N} !
 - ¿ $SSSSH$ está en \mathcal{N} ?
 - ¿ $SSSH$ está en \mathcal{N} ?

Por la regla inductiva,
debería también valer que...

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas para ver si un elemento es parte del conjunto

¿ $SSSSZ$ está en \mathcal{N} !

¿ $SSSSH$ está en \mathcal{N} ?

¿ $SSSH$ está en \mathcal{N} ?

¿ SSH está en \mathcal{N} ?

Por la regla inductiva,
debería también valer que...

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas para ver si un elemento es parte del conjunto
 - ¡ $SSSSZ$ está en \mathcal{N} !
 - ¿ $SSSSH$ está en \mathcal{N} ?
 - ¿ $SSSH$ está en \mathcal{N} ?
 - ¿ SSH^c está en \mathcal{N} ?
 - ¿ SH está en \mathcal{N} ?

Por la regla inductiva,
debería también valer que...

Inducción estructural

- Definición *inductiva* de \mathcal{N}

- Regla base:** Z está en \mathcal{N}

- Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}

- (La regla de ser el menor es implícita en el adjetivo “inductivo”)

- Reglas para ver si un elemento es parte del conjunto

¡ $SSSSZ$ está en \mathcal{N} !

¿ $SSSSH$ está en \mathcal{N} ?

¿ $SSSH$ está en \mathcal{N} ?

¿ SSH está en \mathcal{N} ?

¿ $S\boxed{H}^c$ está en \mathcal{N} ?

¿ H está en \mathcal{N} ?

Por la regla inductiva,
debería también valer que...

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas para ver si un elemento es parte del conjunto
 - ¡ $SSSSZ$ está en \mathcal{N} !
 - ¿ $SSSSH$ está en \mathcal{N} ?
 - ¿ $SSSH$ está en \mathcal{N} ?
 - ¿ SSH está en \mathcal{N} ?
 - ¿ SH está en \mathcal{N} ?
 - ¡ H NO está en \mathcal{N} !

Ninguna regla aplica

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)

- Reglas para ver si un elemento es parte del conjunto

¡ $SSSSZ$ está en \mathcal{N} !

¿ $SSSSH$ está en \mathcal{N} ?

¿ $SSSH$ está en \mathcal{N} ?

¿ SSH está en \mathcal{N} ?

¡ SH NO está en \mathcal{N} !

¡ H NO está en \mathcal{N} !

No se cumple el
antecedente de la regla
inductiva

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas para ver si un elemento es parte del conjunto
 - ¡ $SSSSZ$ está en \mathcal{N} !
 - ¿ $SSSSH$ está en \mathcal{N} ?
 - ¿ $SSSH$ está en \mathcal{N} ?
 - ¡ SSH NO está en \mathcal{N} !
 - ¡ SH NO está en \mathcal{N} !
 - ¡ H NO está en \mathcal{N} !

No se cumple el
antecedente de la regla
inductiva

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas para ver si un elemento es parte del conjunto
 - ¡ $SSSSZ$ está en \mathcal{N} !
 - ¿ $SSSSH$ está en \mathcal{N} ?
 - ¡ $SSSH$ NO está en \mathcal{N} !
 - ¡ SSH NO está en \mathcal{N} !
 - ¡ SH NO está en \mathcal{N} !
 - ¡ H NO está en \mathcal{N} !

No se cumple el
antecedente de la regla
inductiva

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)

- Reglas para ver si un elemento es parte del conjunto

¡ $SSSSZ$ está en \mathcal{N} !

¡ $SSSSH$ NO está en \mathcal{N} !

¡ $SSSH$ NO está en \mathcal{N} !

¡ SSH NO está en \mathcal{N} !

¡ SH NO está en \mathcal{N} !

¡ H NO está en \mathcal{N} !

No se cumple el
antecedente de la regla
inductiva

Inducción estructural

- Definición *inductiva* de \mathcal{N}
 - Regla base:** Z está en \mathcal{N}
 - Regla inductiva:** si c está en \mathcal{N} , entonces Sc está en \mathcal{N}
 - (La regla de ser el menor es implícita en el adjetivo “inductivo”)
- Reglas para ver si un elemento es parte del conjunto
 - ¡ $SSSSZ$ está en \mathcal{N} ! ¡ $SSSSH$ NO está en \mathcal{N} !

Inducción estructural

- ❑ Conjunto definido por *inducción estructural*
- ❑ ¿Qué propiedades tiene?
 - ❑ ¿Cuántos elementos tiene?
 - ❑ ¿Cuántas reglas satisface un elemento exactamente?
 - ❑ ¿Se pueden usar las reglas para ordenar los elementos de alguna forma?
 - ❑ ¿Puede haber elementos de tamaño infinito?
 - ❑ Analicemos cada una de ellas por separado

Inducción estructural

- ❑ Conjunto definido por *inducción estructural*
 - ❑ ¿Cuántos elementos tiene?

Inducción estructural

- ❑ Conjunto definido por ***inducción estructural***
 - ❑ ¿Cuántos elementos tiene?
 - ❑ No puede estar vacío (por las reglas base)
 - ❑ Por cada elemento diferente, cada regla inductiva exige que haya otro elemento... (pues son *productivas*)

Inducción estructural

- ❑ Conjunto definido por ***inducción estructural***
 - ❑ ¿Cuántos elementos tiene?
 - ❑ No puede estar vacío (por las reglas base)
 - ❑ Por cada elemento diferente, cada regla inductiva exige que haya otro elemento... (pues son *productivas*)
 - ❑ ¡Debe tener infinitos elementos!

Inducción estructural

- ❑ Conjunto definido por *inducción estructural*
 - ❑ ¿Cuántas reglas satisface cada elemento del conjunto?

Inducción estructural

- ❏ Conjunto definido por *inducción estructural*
 - ❏ ¿Cuántas reglas satisface cada elemento del conjunto?
 - ❏ ¿Puede un elemento no satisfacer ninguna?

Inducción estructural

- ❑ Conjunto definido por ***inducción estructural***
 - ❑ ¿Cuántas reglas satisface cada elemento del conjunto?
 - ❑ ¿Puede un elemento no satisfacer ninguna?
 - ❑ ¿Puede satisfacer más de una regla?

Inducción estructural

- ❑ Conjunto definido por ***inducción estructural***
 - ❑ ¿Cuántas reglas satisface cada elemento del conjunto?
 - ❑ ¿Puede un elemento no satisfacer ninguna? No, no sería el *menor*
 - ❑ ¿Puede satisfacer más de una regla? No, son reglas *productivas*

Inducción estructural

- ❑ Conjunto definido por **inducción estructural**
 - ❑ ¿Cuántas reglas satisface cada elemento del conjunto?
 - ❑ ¿Puede un elemento no satisfacer ninguna? No, no sería el *menor*
 - ❑ ¿Puede satisfacer más de una regla? No, son reglas *productivas*
 - ❑ ¡Cada elemento satisface exactamente UNA regla!
 - ❑ O bien una regla base (**elemento base**)
 - ❑ O bien una regla inductiva (**elemento inductivo**)
(y en ese caso hay otros elementos que lo “justifican”:
sus **partes**; e.g. **c** es parte de **Sc**, por la regla inductiva)

Inducción estructural

- ❑ Conjunto definido por ***inducción estructural***
 - ❑ ¿Se pueden usar las reglas para ordenar elementos?

Inducción estructural

- ❑ Conjunto definido por ***inducción estructural***
 - ❑ ¿Se pueden usar las reglas para ordenar elementos?
 - ❑ Los **elementos base** son los *más chicos*

Inducción estructural

- ❏ Conjunto definido por ***inducción estructural***
 - ❏ ¿Se pueden usar las reglas para ordenar elementos?
 - ❏ Los **elementos base** son los *más chicos*
 - ❏ Un **elemento inductivo** es *más grande* que sus **partes**

Inducción estructural

- ❑ Conjunto definido por ***inducción estructural***
 - ❑ ¿Se pueden usar las reglas para ordenar elementos?
 - ❑ Los **elementos base** son los *más chicos*
 - ❑ Un **elemento inductivo** es *más grande* que sus partes
 - ❑ Orden ***“es parte de”***
 - ❑ **Z** es parte de **SZ**
 - ❑ **SZ** es parte de **SSZ**
 - ❑ **SSZ** es parte de **SSSZ**
 - ❑ etc.

Inducción estructural

- ❑ Conjunto definido por *inducción estructural*
 - ❑ ¿Puede haber elementos de tamaño infinito?

Inducción estructural

- ❑ Conjunto definido por ***inducción estructural***
 - ❑ ¿Puede haber elementos de tamaño infinito?
 - ❑ O sea, con infinitas partes
 - ❑ Suponiendo que sí, ¿qué sucede si se remueven?

Inducción estructural

- ❑ Conjunto definido por ***inducción estructural***
 - ❑ ¿Puede haber elementos de tamaño infinito?
 - ❑ O sea, con infinitas partes
 - ❑ Suponiendo que sí, ¿qué sucede si se remueven?
 - ❑ ¡Siguen valiendo todas las reglas!

Inducción estructural

- ❑ Conjunto definido por *inducción estructural*
 - ❑ ¿Puede haber elementos de tamaño infinito?
 - ❑ O sea, con infinitas partes
 - ❑ Suponiendo que sí, ¿qué sucede si se remueven?
 - ❑ ¡Siguen valiendo todas las reglas!
 - ❑ ¡No hay elementos de tamaño infinito!
 - ❑ O sea, ningún elemento tiene infinitas partes
 - ❑ El orden “es parte de” es *bien fundado*

Inducción estructural

- ❑ Conjunto definido por ***inducción estructural***
- ❑ ¿Qué propiedades tiene?
 - ❑ Tiene infinitos elementos
 - ❑ Cada elemento satisface *exactamente* una regla
 - ❑ Se puede definir el orden “*es parte de*”
 - ❑ Y es **bien fundado** (no hay cadenas descendentes infinitas)
 - ❑ No puede haber elementos de tamaño infinito
 - ❑ Se puede desarmar un elemento en todas sus partes y subpartes y terminar en algún momento

Inducción estructural

- Definición *inductiva* de XBE
 - Regla base 1: \triangle está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE

Inducción estructural

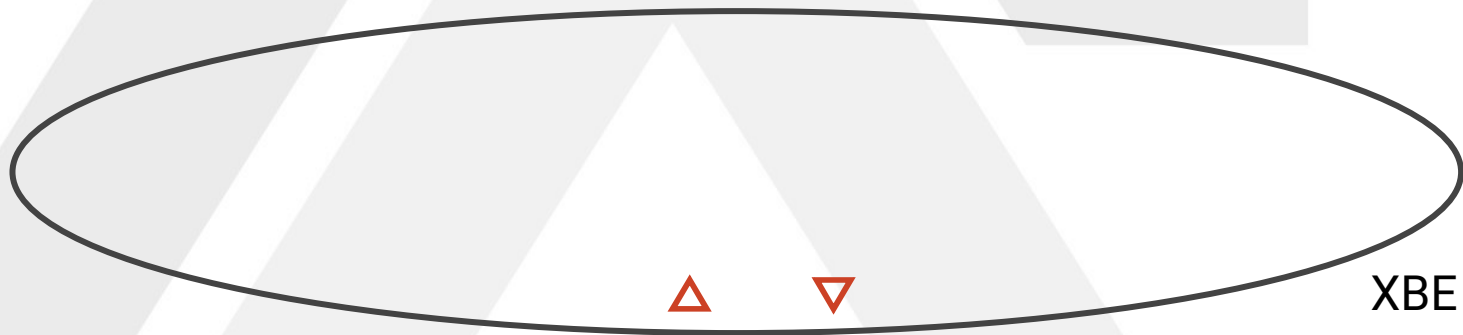
- Definición *inductiva* de XBE
 - Regla base 1: \triangle está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE



XBE

Inducción estructural

- Definición *inductiva* de XBE
 - Regla base 1: \triangle está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE



Inducción estructural

- Definición *inductiva* de XBE
 - Regla base 1: \triangle está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE

$\# \$ \triangle //$



XBE

Inducción estructural

- Definición *inductiva* de XBE
 - Regla base 1: \triangle está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE

$\# \$ \triangle //$

\triangle

∇

$\# \$ \nabla //$

XBE

Inducción estructural

- Definición *inductiva* de XBE
 - Regla base 1: \triangle está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE

$\# \$ \$ \triangle ///$

$\# \$ \triangle //$

\triangle

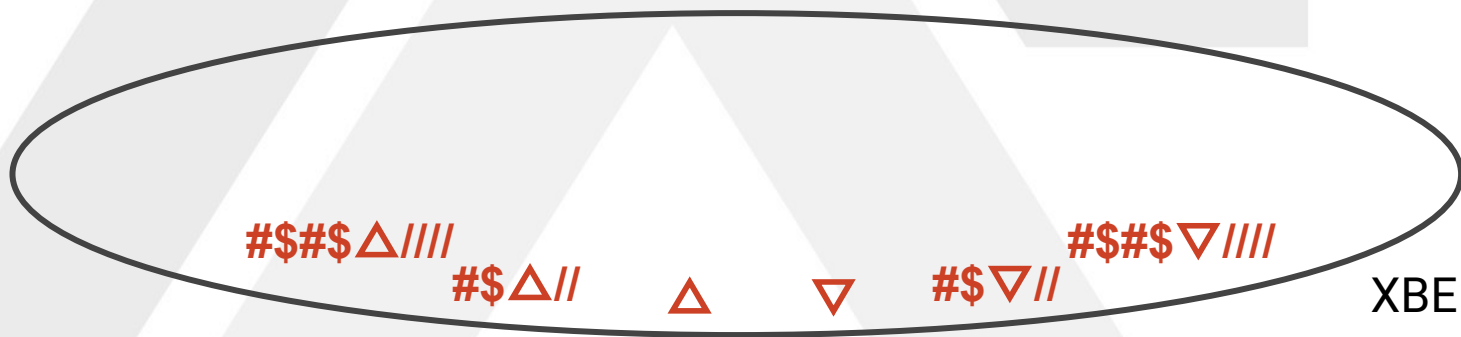
∇

$\# \$ \nabla //$

XBE

Inducción estructural

- Definición *inductiva* de XBE
 - Regla base 1: \triangle está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE



Inducción estructural

Definición *inductiva* de XBE

- Regla base 1: \triangle está en XBE
- Regla base 2: ∇ está en XBE
- Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
- Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE

#####

▽////////

####Δ////

▽////

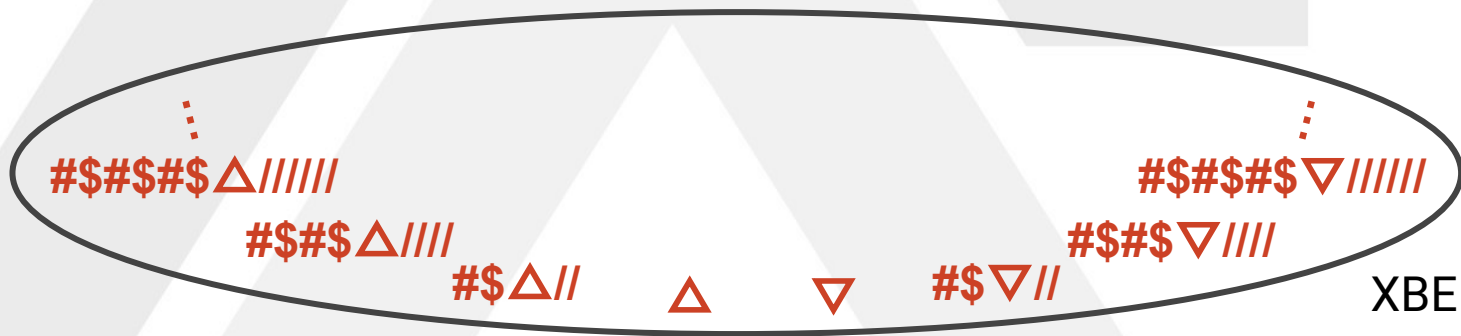
\$ Δ //

#\$▽//

XBE

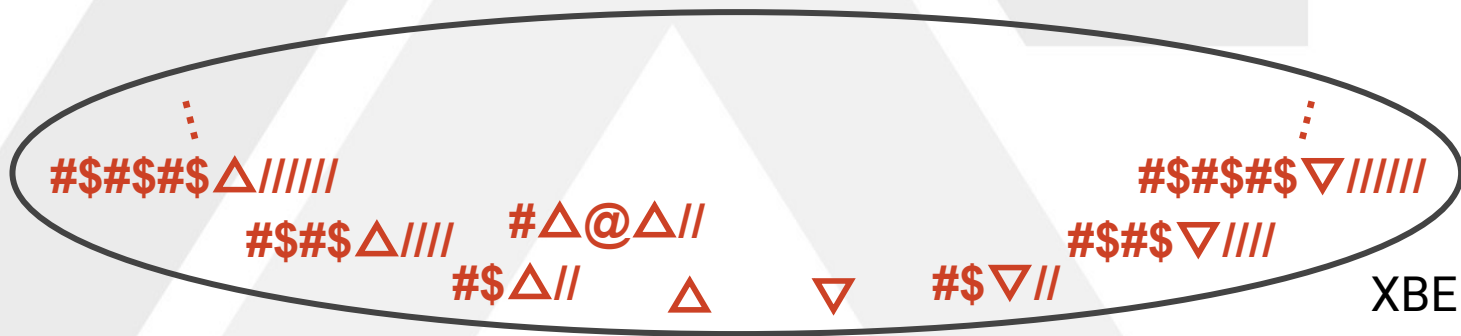
Inducción estructural

- Definición *inductiva* de XBE
 - Regla base 1: \triangle está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE



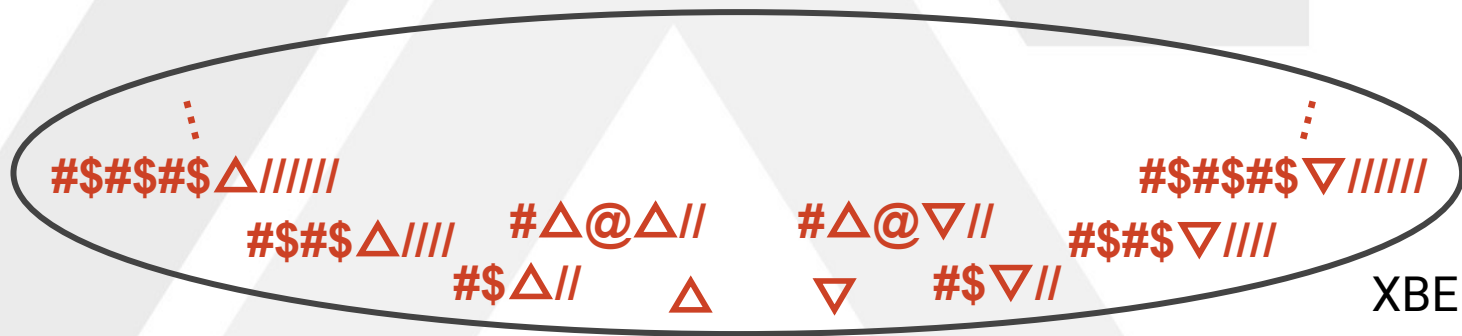
Inducción estructural

- Definición *inductiva* de XBE
 - Regla base 1: \triangle está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE



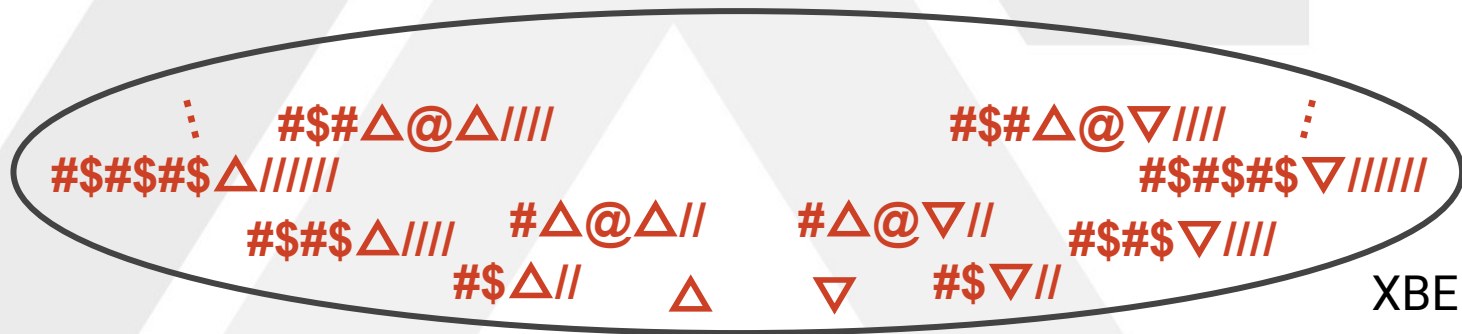
Inducción estructural

- Definición *inductiva* de XBE
 - Regla base 1: Δ está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE



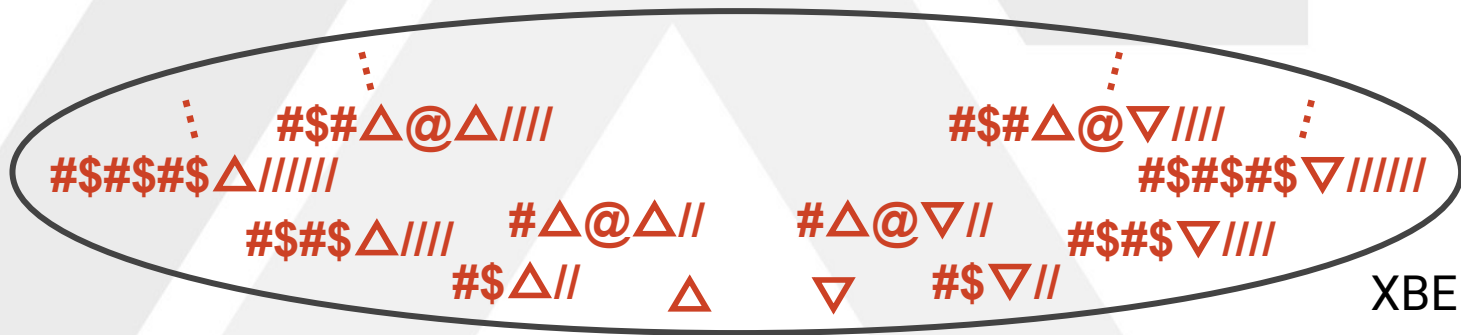
Inducción estructural

- ❑ Definición *inductiva* de XBE
 - ❑ Regla base 1: Δ está en XBE
 - ❑ Regla base 2: ∇ está en XBE
 - ❑ Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - ❑ Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE



Inducción estructural

- ❑ Definición *inductiva* de XBE
 - ❑ Regla base 1: Δ está en XBE
 - ❑ Regla base 2: ∇ está en XBE
 - ❑ Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - ❑ Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE



Inducción estructural

- Definición *inductiva* de XBE
 - Regla base 1: \triangle está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE

Diagram illustrating a 2D hexagonal lattice structure. The central hexagon is labeled **XBE**. It is surrounded by six other hexagons, each containing a different symbol: a triangle, a circle, a square, a diamond, a plus sign, and a minus sign. These six hexagons are further surrounded by a larger set of hexagons, some of which are also highlighted in red. The diagram illustrates the connectivity and potential movement paths between different states or configurations in a 2D lattice.

Inducción estructural

- Definición *inductiva* de XBE
 - Regla base 1: \triangle está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE

Diagram illustrating a complex arrangement of symbols (hash, dollar, at, triangle, and vertical ellipsis) enclosed within a large oval, labeled XBE.

Inducción estructural

- Definición *inductiva* de XBE
 - Regla base 1: Δ está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE



Inducción estructural

- Definición *inductiva* de XBE
 - Regla base 1: \triangle está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE
- ¿ $\# \# \# \$ \triangle // @ \# \$ \nabla // // @ \nabla //$ está en XBE?

Inducción estructural

- Definición *inductiva* de XBE
 - Regla base 1: Δ está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE

¿Qué regla debe aplicarse?

¿ $\# \# \# \$ \Delta // @ \# \$ \nabla // // @ \nabla //$ está en XBE?

Inducción estructural

- Definición *inductiva* de XBE
 - Regla base 1: Δ está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE

¿Qué regla debe aplicarse?
Ri2

¿ $\# \# \$ \overset{e_1}{\Delta} // @ \# \$ \overset{e_2}{\nabla} ///$ está en XBE?
¿ $\# \# \$ \Delta // @ \# \$ \nabla ///$ está en XBE? ¿ ∇ está en XBE?

Inducción estructural

- Definición *inductiva* de XBE
 - Regla base 1: Δ está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE

¿Qué regla debe aplicarse?
Ri2 y Rb2

¿ $\# \# \$ \Delta // @ \# \$ \nabla // //$ está en XBE?

¿ $\# \# \$ \Delta //$ está en XBE? ¿ $\# \$ \nabla //$ está en XBE?

¿ ∇ está en XBE!

Inducción estructural

- Definición *inductiva* de XBE
 - Regla base 1: Δ está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE

¿Qué regla debe aplicarse?
Ri1 (x2)

¿ $\# \# \# \$ \Delta // @ \# \$ \nabla // // @ \nabla //$ está en XBE?

¿ $\# \# \$ \Delta // @ \# \$ \nabla // //$ está en XBE?

¿ ∇ está en XBE!

¿ $\# \$ \overset{e}{\Delta} //$ está en XBE?
¿ Δ está en XBE?

¿ $\# \$ \overset{e'}{\nabla} //$ está en XBE?
¿ ∇ está en XBE?

Inducción estructural

- Definición *inductiva* de XBE
 - Regla base 1: Δ está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE

¿Qué regla debe aplicarse?
Rb1 y Rb2

¿ $\# \# \# \$ \Delta // @ \# \$ \nabla // @ \nabla //$ está en XBE?

¿ $\# \# \$ \Delta // @ \# \$ \nabla //$ está en XBE?

¿ ∇ está en XBE!

¿ $\# \$ \Delta //$ está en XBE?

¿ $\# \$ \nabla //$ está en XBE?

¿ Δ está en XBE!

¿ ∇ está en XBE!

Inducción estructural

- Definición *inductiva* de XBE
 - Regla base 1: Δ está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE

$\text{¡} \# \# \# \$ \Delta // @ \# \$ \nabla // // @ \nabla // \text{!}$ está en XBE!

$\text{¡} \# \# \$ \Delta // @ \# \$ \nabla // // \text{!}$ está en XBE!

$\text{¡} \nabla \text{!}$ está en XBE!

$\text{¡} \# \$ \Delta // \text{!}$ está en XBE!

$\text{¡} \# \$ \nabla // \text{!}$ está en XBE!

$\text{¡} \Delta \text{!}$ está en XBE!

$\text{¡} \nabla \text{!}$ está en XBE!

Inducción estructural

Definición *inductiva* de XBE

¿Qué significan los elementos de XBE?

- Definición *inductiva* de XBE
 - Regla base 1:** \triangle está en XBE
 - Regla base 2:** ∇ está en XBE
 - Regla inductiva 1:** si e está en XBE, entonces $\#e//$ está en XBE
 - Regla inductiva 2:** si e_1 está en XBE y e_2 está en XBE, entonces $\#e_1@e_2//$ está en XBE

¿Qué significa este elemento:

$\#\#\$\triangle//@#\$\nabla////@ \nabla//?$

Inducción estructural

Definición *inductiva* de XBE

- Definición *inductiva* de XBE
 - Regla base 1: \triangle está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\#e//$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\#e_1@e_2//$ está en XBE

¿Qué significan los elementos de XBE?

¡NADA!

Solamente siguen la aplicación de las reglas

¿Qué significa este elemento:

$\#\#\$\triangle//@#\$\nabla////@ \nabla//?$

Inducción estructural

Definición *inductiva* de XBE

- Definición *inductiva* de XBE
 - Regla base 1:** \triangle está en XBE
 - Regla base 2:** ∇ está en XBE
 - Regla inductiva 1:** si e está en XBE,
entonces $\#e//$ está en XBE
 - Regla inductiva 2:** si e_1 está en XBE
y e_2 está en XBE,
entonces $\#e_1@e_2//$ está en XBE

¿Qué significan los elementos de XBE?

¡NADA!

Solamente siguen la aplicación de las reglas

Y eso no es poca cosa...

Inducción estructural

- Definición *inductiva* de BE
 - Regla base 1: **T** está en BE
 - Regla base 2: **F** está en BE
 - Regla inductiva 1: si **e** está en BE, entonces **(~e)** está en BE
 - Regla inductiva 2: si **e₁** está en BE y **e₂** está en BE, entonces **(e₁ ^ e₂)** está en BE

Inducción estructural

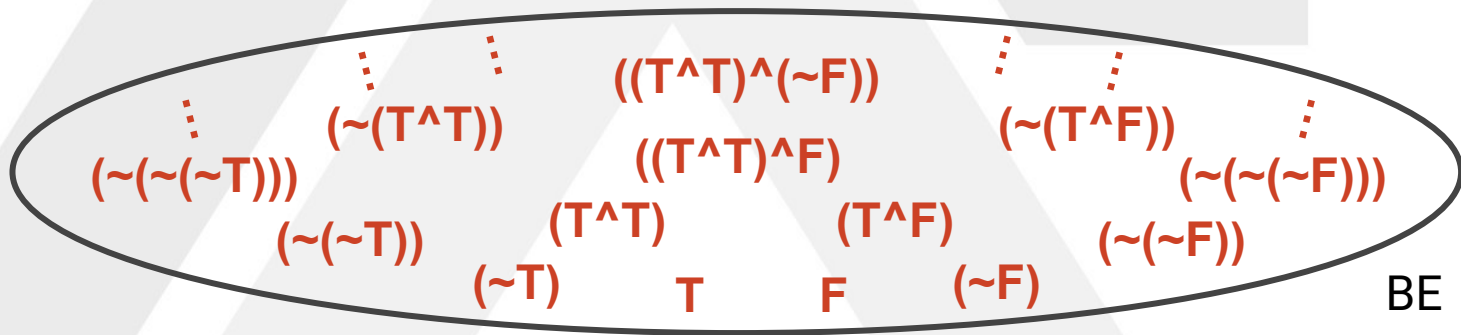
Definición *inductiva* de BE

Regla base 1: **T** está en BE

Regla base 2: **F** está en BE

Regla inductiva 1: si **e** está en BE, entonces **(~e)** está en BE

Regla inductiva 2: si **e₁** está en BE y **e₂** está en BE,
entonces **(e₁ ^ e₂)** está en BE



Inducción estructural

Definición *inductiva* de BE

Regla base 1: **T** está en BE

Regla base 2: **F** está en BE

Regla inductiva 1: si **e** está en BE, entonces **(~e)** está en BE

Regla inductiva 2: si **e₁** está en BE y **e₂** está en BE,
entonces **(e₁ ^ e₂)** está en BE

i(((~T)^(~F))^F) está en BE!

i((~T)^(~F)) está en BE!

iF está en BE!

i(~T) está en BE!

i(~F) está en BE!

iT está en BE!

iF está en BE!

Inducción estructural

Definición *inductiva* de BE

¿Qué significan los elementos de BE?

- Definición *inductiva* de BE
 - Regla base 1:** **T** está en BE
 - Regla base 2:** **F** está en BE
 - Regla inductiva 1:** si **e** está en BE, entonces **(~e)** está en BE
 - Regla inductiva 2:** si **e₁** está en BE y **e₂** está en BE, entonces **(e₁ ^ e₂)** está en BE

¿Qué significa este elemento:

$((\sim T) \wedge (\sim F)) \wedge F$?

Inducción estructural

Definición *inductiva* de BE

- ¿Qué significan los elementos de BE?
- ¡NADA!

- Definición *inductiva* de BE
 - Regla base 1:** **T** está en BE
 - Regla base 2:** **F** está en BE
 - Regla inductiva 1:** si **e** está en BE, entonces **(~e)** está en BE
 - Regla inductiva 2:** si **e₁** está en BE y **e₂** está en BE, entonces **(e₁ ^ e₂)** está en BE

¿Qué significa este elemento:
 $((\sim T) \wedge (\sim F)) \wedge F$?

Inducción estructural

Definición *inductiva* de BE

¿Qué significan los elementos de BE?

¡NADA!

Pero se parecen a cosas que conozco...

- Definición *inductiva* de BE
 - Regla base 1:** **T** está en BE
 - Regla base 2:** **F** está en BE
 - Regla inductiva 1:** si **e** está en BE, entonces **(~e)** está en BE
 - Regla inductiva 2:** si **e₁** está en BE y **e₂** está en BE, entonces **(e₁ ^ e₂)** está en BE

¿Qué significa este elemento:

$((\sim T) \wedge (\sim F)) \wedge F$?

Inducción estructural

Definición *inductiva* de BE

¿Qué significan los elementos de BE?

¡NADA!

Pero se parecen a cosas que conozco...

... Sin embargo, NO hay nada (aún) que implique eso.

- Definición *inductiva* de BE
 - Regla base 1:** **T** está en BE
 - Regla base 2:** **F** está en BE
 - Regla inductiva 1:** si **e** está en BE,
entonces **(~e)** está en BE
 - Regla inductiva 2:** si **e₁** está en BE
y **e₂** está en BE,
entonces **(e₁ ^ e₂)** está en BE

¿Qué significa este elemento:

$((\sim T) \wedge (\sim F)) \wedge F$?



Recursión estructural

Recursión estructural

■ Comparar las definiciones de XBE y de BE

■ ¿Qué características comparten?

- Regla base 1: Δ está en XBE
- Regla base 2: ∇ está en XBE
- Regla inductiva 1: si e está en XBE, entonces $\#e//$ está en XBE
- Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\#e_1@e_2//$ está en XBE
- Regla base 1: T está en BE
- Regla base 2: F está en BE
- Regla inductiva 1: si e está en BE, entonces $(\sim e)$ está en BE
- Regla inductiva 2: si e_1 está en BE y e_2 está en BE, entonces $(e_1 \wedge e_2)$ está en BE

Inducción estructural

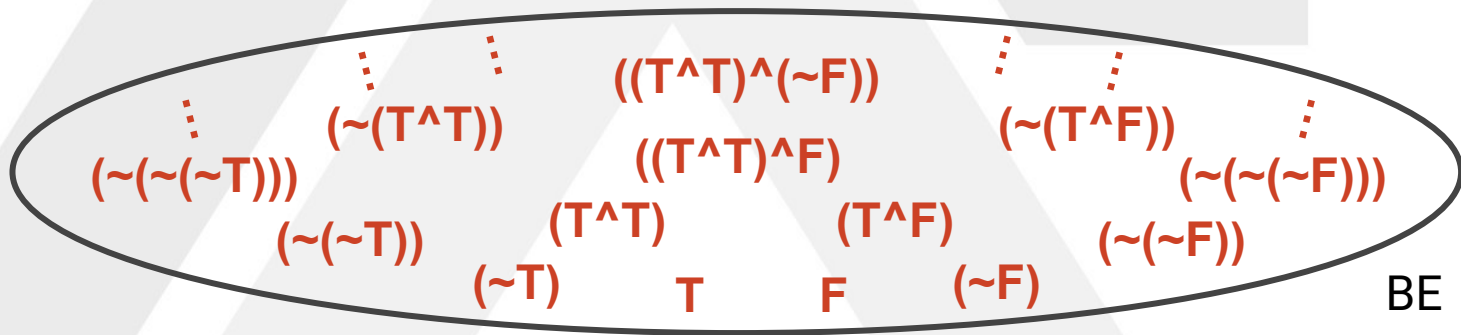
Definición *inductiva* de BE

Regla base 1: **T** está en BE

Regla base 2: **F** está en BE

Regla inductiva 1: si **e** está en BE, entonces **(~e)** está en BE

Regla inductiva 2: si **e₁** está en BE y **e₂** está en BE,
entonces **(e₁ ^ e₂)** está en BE



Inducción estructural

- Definición *inductiva* de XBE
 - Regla base 1: Δ está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE



Recursión estructural

- ❑ Comparar las definiciones de XBE y de BE
 - ❑ ¿Qué características comparten?
 - ❑ Misma cantidad de reglas base y misma cantidad de reglas inductivas con la misma cantidad de antecedentes cada una
 - ❑ ¿Qué implica esto?
 - ❑ Los conjuntos tienen la misma “cantidad” de elementos
 - ❑ Los órdenes “es parte de” de ambos son similares
 - ❑ Los conjuntos tienen la misma **estructura**
 - ❑ Los símbolos particulares NO son importantes

Recursión estructural

- ❏ ¿Cómo definir funciones sobre los elementos de un conjunto inductivo **S**?

Recursión estructural

- ❏ ¿Cómo definir funciones sobre los elementos de un conjunto inductivo **S**?
- ❏ Aprovechar la *estructura*

Recursión estructural

❏ ¿Cómo definir funciones sobre los elementos de un conjunto inductivo **S**?

❏ Aprovechar la *estructura*

$f :: S \rightarrow T$

$f \mathbf{z}_1 = \dots$

\dots

$f \mathbf{z}_n = \dots$

$f \mathbf{e}_1 = \dots f \mathbf{e}_{11} \dots f \mathbf{e}_{i1} \dots$

\dots

$f \mathbf{e}_k = \dots f \mathbf{e}_{1k} \dots f \mathbf{e}_{ik} \dots$

Recursión estructural

Definición de una función por ***recursión estructural***

- ❑ Por cada **elemento base**, dar el resultado directamente
- ❑ Por cada **elemento inductivo**, dar el resultado usando el valor de transformar las *partes* con la *misma función que se define*

Recursión estructural

Definición de $f : S \rightarrow T$ por *recursión estructural*

$f :: S \rightarrow T$

$f \ z_1 = \dots$

\dots

$f \ z_n = \dots$

$f \ e_1 = \dots f \ e_{11} \dots f \ e_{i1} \dots$

\dots

$f \ e_k = \dots f \ e_{1k} \dots f \ e_{ik} \dots$

Recursión estructural

- ❑ Recordar la definición *inductiva* de XBE
 - ❑ **Regla base 1:** \triangle está en XBE
 - ❑ **Regla base 2:** ∇ está en XBE
 - ❑ **Regla inductiva 1:** si e está en XBE, entonces $\# \$ e //$ está en XBE
 - ❑ **Regla inductiva 2:** si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE
- ❑ Definir $nhash, nbar :: XBE \rightarrow Int$, el número de $\#$ y el de $/$
 - ❑ ¿Por qué se precisa recursión estructural?
 - ❑ ¿Cómo saber cuántos $\#$ hay en $\# \$ e //$ sin saber cuántos hay en e ?

Recursión estructural

■ Definir `nhash, nbar :: XBE -> Int`, el número de `#` y el de `/`

Recursión estructural

Definir $\text{nhash}, \text{nbar} :: \text{XBE} \rightarrow$

Por recursión estructural

- Definición *inductiva* de XBE
 - Regla base 1: \triangle está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\#e//$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\#e_1@e_2//$ está en XBE

Se decide usar
recursión
estructural

Recursión estructural

Definir $\text{nhash}, \text{nbar} :: \text{XBE} \rightarrow$

Por recursión estructural

$\text{nhash} :: \text{XBE} \rightarrow \text{Int}$

$\text{nhash } \triangle = \dots$

$\text{nhash } \nabla = \dots$

$\text{nhash } \$e = \dots \text{nhash } e \dots$

$\text{nhash } e_1 @ e_2 = \dots \text{nhash } e_1 \dots \text{nhash } e_2 \dots$

$\text{nbar} :: \text{XBE} \rightarrow \text{Int}$

$\text{nbar } \triangle = \dots$

$\text{nbar } \nabla = \dots$

$\text{nbar } \$e = \dots \text{nbar } e \dots$

$\text{nbar } e_1 @ e_2 = \dots \text{nbar } e_1 \dots \text{nbar } e_2 \dots$

- Definición *inductiva* de XBE
 - Regla base 1: \triangle está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\$e$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $e_1 @ e_2$ está en XBE

Se plantea la estructura en base a las reglas

Recursión estructural

Definir $\text{nhash}, \text{nbar} :: \text{XBE} \rightarrow$

Por recursión estructural

$\text{nhash} :: \text{XBE} \rightarrow \text{Int}$

$\text{nhash } \triangle = \dots$

$\text{nhash } \nabla = \dots$

$\text{nhash } \$e = 1 + \text{nhash } e$

$\text{nhash } e_1 @ e_2 = 1 + \text{nhash } e_1 + \text{nhash } e_2$

$\text{nbar} :: \text{XBE} \rightarrow \text{Int}$

$\text{nbar } \triangle = \dots$

$\text{nbar } \nabla = \dots$

$\text{nbar } \$e = 2 + \text{nbar } e$

$\text{nbar } e_1 @ e_2 = 2 + \text{nbar } e_1 + \text{nbar } e_2$

- Definición *inductiva* de XBE
 - Regla base 1: \triangle está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\$e$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $e_1 @ e_2$ está en XBE

Se definen los casos inductivos

Recursión estructural

Definir $\text{nhash}, \text{nbar} :: \text{XBE} \rightarrow$

Por recursión estructural

$\text{nhash} :: \text{XBE} \rightarrow \text{Int}$

$\text{nhash } \triangle = 0$

$\text{nhash } \nabla = 0$

$\text{nhash } \$e = 1 + \text{nhash } e$

$\text{nhash } e_1 @ e_2 = 1 + \text{nhash } e_1 + \text{nhash } e_2$

$\text{nbar} :: \text{XBE} \rightarrow \text{Int}$

$\text{nbar } \triangle = 0$

$\text{nbar } \nabla = 0$

$\text{nbar } \$e = 2 + \text{nbar } e$

$\text{nbar } e_1 @ e_2 = 2 + \text{nbar } e_1 + \text{nbar } e_2$

- Definición *inductiva* de XBE
 - Regla base 1: \triangle está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\$e$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $e_1 @ e_2$ está en XBE

Se completa con los casos base

Recursión estructural

- ❑ ¿Y por qué funciona la *recursión estructural*?
- ❑ Si usa la misma función, ¿por qué termina (y no da **⌋**)?

Recursión estructural

- ❑ ¿Y por qué funciona la *recursión estructural*?
- ❑ Si usa la misma función, ¿por qué termina (y no da \perp)?
- ❑ ¡El orden “**es parte de**” es *bien fundado*!
 - ❑ O sea, nunca se puede descomponer un elemento en sus partes de forma infinita...
 - ❑ De esta forma, la reducción *tiene que terminar*

Recursión estructural

- ❑ ¿Y por qué funciona la *recursión estructural*?
- ❑ Si usa la misma función, ¿por qué termina (y no da \perp)?
- ❑ ¡El orden “**es parte de**” es *bien fundado*!
 - ❑ O sea, nunca se puede descomponer un elemento en sus partes de forma infinita...
 - ❑ De esta forma, la reducción *debe terminar*
- ❑ La condición de ser el **menor** es esencial para esto

Recursión estructural

- ❑ ¿Y por qué funciona la *recursión estructural*?
- ❑ Si usa la misma función, ¿por qué termina (y no da \perp)?
- ❑ ¡El orden “**es parte de**” es *bien fundado*!
 - ❑ O sea, nunca se puede descomponer un elemento en sus partes de forma infinita...
 - ❑ De esta forma, la reducción *debe terminar*
- ❑ La condición de ser el **menor** es esencial para esto
- ❑ Una función recursiva estructural **termina** de reducir para todos los elementos inductivos

Recursión estructural

Una función recursiva estructural *termina* de reducir para todos los elementos inductivos

Inducción estructural

- ❑ ¿Y por qué funciona la *recursión estructural*?
- ❑ Si usa la misma función, ¿por qué termina (y no da \perp)?
- ❑ Las reglas dan una forma de desarmar un elemento en sus partes
- ❑ Esa forma es aprovechada por la reducción

nbar $###\$ \Delta // @ \# \$ \nabla // @ \nabla // = ??$

Inducción estructural

¿Y por qué funciona la *reducción*?

Si usa la misma función, ¿por qué termina (y no da \perp)?

Las reglas dan una forma de desarmar un elemento en sus partes

Esa forma es aprovechada por la reducción

- Definición *inductiva* de XBE
 - Regla base 1: Δ está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\# \$ e //$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\# e_1 @ e_2 //$ está en XBE

nbar $### \$ \Delta // @ \$ \nabla // @ \nabla //$ = ??

nbar $### \$ \Delta // @ \$ \nabla //$ = ??

nbar ∇ = ??

nbar $\# \$ \Delta //$ = ??

nbar $\# \$ \nabla //$ = ??

nbar Δ = ??

nbar ∇ = ??

Inducción estructural

¿Y por qué funciona la *recursión*?

Si usa la misma función, ¿por qué termina (y no da \perp)?

Las reglas dan una forma de desarmar un elemento en sus partes

Esa forma es aprovechada por la reducción

- Definición *inductiva* de XBE
 - Regla base 1: Δ está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\#e//$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\#e_1@e_2//$ está en XBE

$$\text{nbar } \#\#\$\Delta//@#\$\nabla////@ \nabla// = 2+6+0$$

$$\text{nbar } \#\#\$\Delta//@#\$\nabla//// = 2+2+2$$

$$\text{nbar } \nabla = 0$$

$$\text{nbar } \#\$\Delta// = 2+0$$

$$\text{nbar } \#\$\nabla// = 2+0$$

$$\text{nbar } \Delta = 0$$

$$\text{nbar } \nabla = 0$$

Inducción estructural

¿Y por qué funciona la *recursión*?

Si usa la misma función, ¿por qué termina (y no da \perp)?

Las reglas dan una forma de desarmar un elemento en sus partes

Esa forma es aprovechada por la reducción

- Definición *inductiva* de XBE
 - Regla base 1: Δ está en XBE
 - Regla base 2: ∇ está en XBE
 - Regla inductiva 1: si e está en XBE, entonces $\#e//$ está en XBE
 - Regla inductiva 2: si e_1 está en XBE y e_2 está en XBE, entonces $\#e_1@e_2//$ está en XBE

$$\text{nbar } \#\#\$\Delta//@#\$\nabla////@ \nabla// = 8$$

$$\text{nbar } \#\#\$\Delta//@#\$\nabla//// = 6$$

$$\text{nbar } \nabla = 0$$

$$\text{nbar } \#\$\Delta// = 2$$

$$\text{nbar } \#\$\nabla// = 2$$

$$\text{nbar } \Delta = 0$$

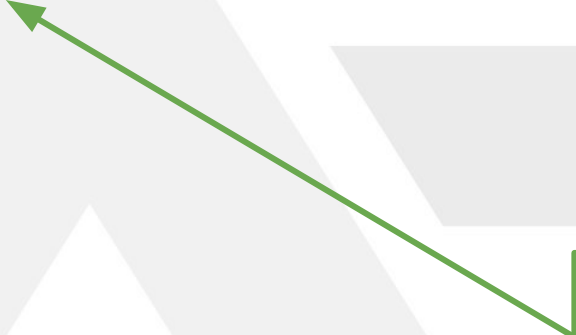
$$\text{nbar } \nabla = 0$$

Recursión estructural

- Definir **$\text{xbe2be} :: \text{XBE} \rightarrow \text{BE}$** y **$\text{be2xbe} :: \text{BE} \rightarrow \text{XBE}$** que transformen elementos de un conjunto a otro y de vuelta

Recursión estructural

- Definir $\mathbf{xbe2be} :: \mathbf{XBE} \rightarrow \mathbf{BE}$ y $\mathbf{be2xbe} :: \mathbf{BE} \rightarrow \mathbf{XBE}$ que transformen elementos de un conjunto a otro y de vuelta
 - Por recursión estructural



Se decide usar
recursión
estructural

Recursión estructural

- Definir $\text{xbe2be} :: \text{XBE} \rightarrow \text{BE}$ y $\text{be2xbe} :: \text{BE} \rightarrow \text{XBE}$ que transformen elementos de un conjunto a otro y de vuelta

- Por recursión estructural

$\text{xbe2be} :: \text{XBE} \rightarrow \text{BE}$

$\text{xbe2be} \triangle = \dots$

$\text{xbe2be} \nabla = \dots$

$\text{xbe2be} \# \$ e // = \dots \text{xbe2be } e \dots$

$\text{xbe2be} \# e_1 @ e_2 // = \dots \text{xbe2be } e_1 \dots \text{xbe2be } e_2 \dots$

$\text{be2xbe} :: \text{BE} \rightarrow \text{XBE}$

$\text{be2xbe} T = \dots$

$\text{be2xbe} F = \dots$

$\text{be2xbe} (\sim e) = \dots \text{be2xbe } e \dots$

$\text{be2xbe} (e_1 \wedge e_2) = \dots \text{be2xbe } e_1 \dots \text{be2xbe } e_2 \dots$

Se plantea la estructura en base a las reglas

Recursión estructural

- Definir $\text{xbe2be} :: \text{XBE} \rightarrow \text{BE}$ y $\text{be2xbe} :: \text{BE} \rightarrow \text{XBE}$ que transformen elementos de un conjunto a otro y de vuelta

- Por recursión estructural

```
xbe2be :: XBE -> BE
xbe2be  $\Delta$  = ...
xbe2be  $\nabla$  = ...
xbe2be  $\# \$ e //$  =  $(\sim \text{xbe2be } e)$ 
xbe2be  $\# e_1 @ e_2 //$  =  $(\text{xbe2be } e_1 \wedge \text{xbe2be } e_2)$ 

be2xbe :: BE -> XBE
be2xbe  $T$  = ...
be2xbe  $F$  = ...
be2xbe  $(\sim e)$  =  $\# \$ \text{be2xbe } e //$ 
be2xbe  $(e_1 \wedge e_2)$  =  $\# \text{be2xbe } e_1 @ \text{be2xbe } e_2 //$ 
```

Se definen los
casos inductivos

Recursión estructural

- Definir $\text{xbe2be} :: \text{XBE} \rightarrow \text{BE}$ y $\text{be2xbe} :: \text{BE} \rightarrow \text{XBE}$ que transformen elementos de un conjunto a otro y de vuelta

- Por recursión estructural

```
xbe2be :: XBE -> BE
xbe2be  $\Delta$       = T
xbe2be  $\nabla$       = F
xbe2be  $\# \$ e //$   = ( $\sim$  xbe2be e)
xbe2be  $\# e_1 @ e_2 //$  = (xbe2be  $e_1$  ^ xbe2be  $e_2$ )

be2xbe :: BE -> XBE
be2xbe T      =  $\Delta$ 
be2xbe F      =  $\nabla$ 
be2xbe ( $\sim$  e) =  $\# \$$  be2xbe e //
be2xbe ( $e_1$  ^  $e_2$ ) =  $\#$  be2xbe  $e_1$  @ be2xbe  $e_2$  //
```

Se completa con
los casos base



Ejemplo de aplicación

Uso de la inducción/recursión estructural

- Definición *inductiva* de \mathbb{N}
 - **Regla base:** 0 está en \mathbb{N}
 - **Regla inductiva:** si n está en \mathbb{N} , entonces $n+1$ está en \mathbb{N}
- Observar que las reglas son productivas
- Se cumplen las propiedades de conjuntos inductivos
 - Tiene infinitos elementos
 - Todos sus elementos son finitos
 - El orden usual de “menor” ($<$) es “es (sub)parte de”

Uso de la inducción/recursión estructural

- Definición *inductiva* de \mathbb{N}
 - Regla base:** 0 está en \mathbb{N}
 - Regla inductiva:** si n está en \mathbb{N} , entonces $n+1$ está en \mathbb{N}
- ¿Cómo sería una definición de función sobre \mathbb{N} ?

Uso de la inducción/recursión estructural

Definición *inductiva* de \mathbb{N}

Regla base: 0 está en \mathbb{N}

Regla inductiva: si n está en \mathbb{N} , entonces $n+1$ está en \mathbb{N}

¿Cómo sería una definición de función sobre \mathbb{N} ?

$f : \mathbb{N} \rightarrow T$

$f \ 0 = \dots$

$f \ (n+1) = \dots f \ n \dots$

Uso de la inducción/recursión estructural

Definición *inductiva* de \mathbb{N}

Regla base: 0 está en \mathbb{N}

Regla inductiva: si n está en \mathbb{N} , entonces $n+1$ está en \mathbb{N}

¿Cómo sería una definición de función sobre \mathbb{N} ?

$f :: \mathbb{N} \rightarrow T$

$f \ 0 = \dots$

$f \ (n+1) = \dots f \ n \dots$

Usualmente se escribe
de esta otra forma

$f :: \mathbb{N} \rightarrow T$

$f \ 0 = \dots$

$f \ m = \dots f \ (m-1) \dots$

Uso de la inducción/recursión estructural

- Definición *inductiva* de \mathbb{N}

- Regla base: 0 está en \mathbb{N}

- Regla inductiva: si n está en \mathbb{N} , entonces $n+1$ está en \mathbb{N}

- $f :: \mathbb{N} \rightarrow T$

- $f\ 0 = \dots$

- $f\ n = \dots f\ (n-1) \dots$

- ¡Los números naturales son un conjunto inductivo estructural!

- Funciones como **factorial** funcionan por su estructura

Inducción y recursión en Haskell

Inducción y recursión en Haskell

- ¿Cómo usar todas estas técnicas en Haskell?
 - ¿Definición inductiva de estructuras?
 - ¿Definición de funciones por recursión estructural?
 - ¿Y la 3era pregunta?
- ¿Qué cosas interesantes se pueden expresar mediante la utilización de estas técnicas?

Inducción y recursión en Haskell

- ¿Cómo usar todas estas técnicas en Haskell?
 - ¿Definición inductiva de estructuras?

Inducción y recursión en Haskell

- ¿Cómo usar todas estas técnicas en Haskell?
 - ¿Definición inductiva de estructuras?
 - Usando tipos algebraicos

Inducción y recursión en Haskell

- ¿Cómo usar todas estas técnicas en Haskell?
 - ¿Definición inductiva de estructuras?
 - Usando tipos algebraicos
 - ¿Cómo expresar casos base?
 - ¿Cómo expresar casos recursivos?

Inducción y recursión en Haskell

- ❑ ¿Cómo usar todas estas técnicas en Haskell?
 - ❑ ¿Definición inductiva de estructuras?
 - ❑ Usando tipos algebraicos
 - ❑ ¿Cómo expresar casos base?
 - ❑ Usando constructores con o sin argumentos
 - ❑ ¿Cómo expresar casos recursivos?
 - ❑ ¡Usando constructores que usen el mismo tipo definido como argumento!

Inducción y recursión en Haskell

- ¿Cómo usar todas estas técnicas en Haskell?
 - ¿Definición inductiva de estructuras?

Inducción y recursión en Haskell

■ ¿Cómo usar todas estas técnicas en Haskell?

■ ¿Definición inductiva de estructuras?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa
data Cofre = Monedas Int | CorazonDe Pirata | Joyas Int | Nada
data Pirata = DavyJones | WillTurner
data Dir = Izq | Der
```

Inducción y recursión en Haskell

- ❑ ¿Cómo usar todas estas técnicas en Haskell?
- ❑ ¿Definición inductiva de estructuras?

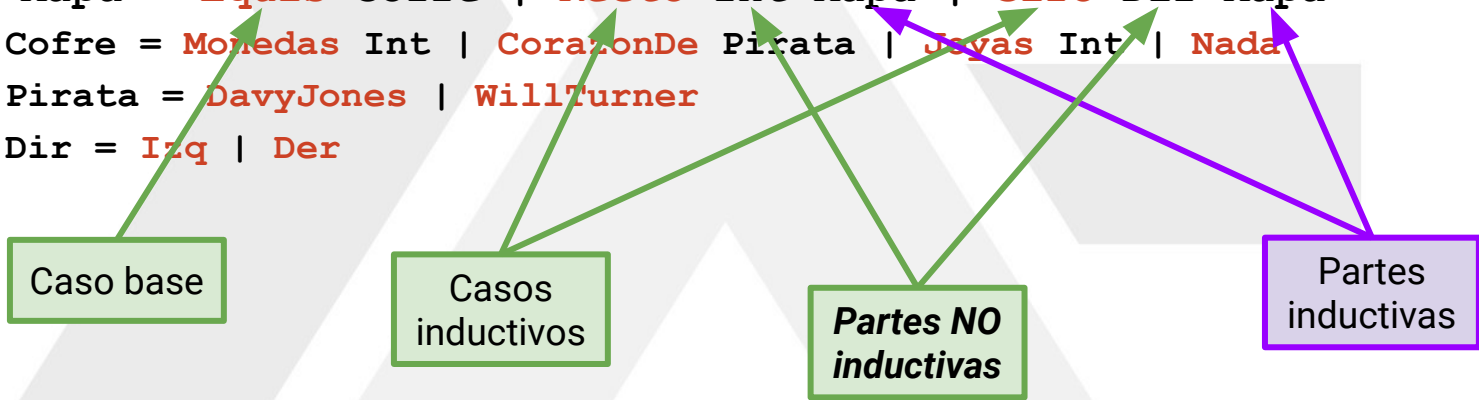
```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa
data Cofre = Monedas Int | CorazonDe Pirata | Joyas Int | Nada
data Pirata = DavyJones | WillTurner
data Dir = Izq | Der
```

Caso base

Casos
inductivos

**Partes NO
inductivas**

Partes
inductivas



Inducción y recursión en Haskell

- ¿Cómo usar todas estas técnicas en Haskell?
 - ¿Definición inductiva de estructuras?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa  
data Cofre = Monedas Int | CorazonDe Pirata | Joyas Int | Nada  
data Pirata = DavyJones | WillTurner  
data Dir = Izq | Der
```

?

Mapa

Inducción y recursión en Haskell

- ¿Cómo usar todas estas técnicas en Haskell?
- ¿Definición inductiva de estructuras?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa  
data Cofre = Monedas Int | CorazonDe Pirata | Joyas Int | Nada  
data Pirata = DavyJones | WillTurner  
data Dir = Izq | Der
```

...

Equis (Monedas 100)

Equis (CorazonDe DavyJones)

Equis Nada

...

Mapa

Inducción y recursión en Haskell

- ❑ ¿Cómo usar todas estas técnicas en Haskell?
- ❑ ¿Definición inductiva de estructuras?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa
data Cofre = Monedas Int | CorazonDe Pirata | Joyas Int | Nada
data Pirata = DavyJones | WillTurner
data Dir = Izq | Der
```

Diagram illustrating a nested structure of constructors and values, enclosed in a large black oval:

- Top level: `Recto 30 (Giro Izq (Recto 10 (Equis (Monedas 250))))`
- Second level (left): `Giro Izq (Recto 10 (Equis (Monedas 250)))`
- Second level (right): `Giro Der (Recto 30 (Equis Nada))`
- Third level (left): `Equis (Monedas 100)`
- Third level (middle): `Recto 10 (Equis (Monedas 250))`
- Third level (right): `Recto 30 (Equis Nada)`
- Bottom level (left): `Equis (Corazon DavyJones)`
- Bottom level (right): `Equis Nada`

Mapa

Inducción y recursión en Haskell

- ❑ ¿Cómo usar todas estas técnicas en Haskell?
- ❑ ¿Definición inductiva de estructuras?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa
data Cofre = Monedas Int | CorazonDe Pirata | Joyas Int | Nada
data Pirata = DavyJones | WillTurner
data Dir = Izq | Der
```

...

Giro Izq (Recto 1

...

Equis (Monedas 100)

Equis (Corazon DavyJones)

Equis Nada

...

¿Estos son todos los elementos?

Mapa

Inducción y recursión en Haskell

- ❑ ¿Cómo usar todas estas técnicas en Haskell?
- ❑ ¿Definición inductiva de estructuras?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa
data Cofre = Monedas Int | CorazonDe Pirata | Joyas Int | Nada
data Pirata = DavyJones | WillTurner
data Dir = Izq | Der
```

Diagram illustrating a nested expression tree structure for a `Mapa` value, enclosed in a large black oval:

- Top level: `Recto 30 (Giro Izq (Recto 10 (Equis (Monedas 250))))`
- Second level (under `Recto 30`):
 - `Giro Izq (Recto 10 (Equis (Monedas 250)))`
 - `Giro Der (Recto 30 (Equis Nada))`
 - `Recto 30 (Equis Nada)`
- Third level (under `Recto 10` and `Recto 30`):
 - `Equis (Monedas 100)`
 - `Equis (Corazon DavyJones)`
 - `Equis Nada`

Vertical ellipses (`⋮`) indicate further nesting or repetition of structure.



Mapa

Inducción y recursión en Haskell

❏ ¿Cómo usar todas estas técnicas en Haskell?

❏ ¿Definición inductiva de estructuras?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa
```

```
data Cofre = Monedas Int | CorazonDe Pirata | Joyas Int | Nada
```

```
data Pirata = DavyJones | WillTurner
```

```
data Dir = Izq | Der
```



Inducción y recursión en Haskell

- ❑ ¿Cómo usar todas estas técnicas en Haskell?
- ❑ ¿Definición inductiva de estructuras?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa
data Cofre = Monedas Int | CorazonDe Pirata | Joyas Int | Nada
data Pirata = DavyJones | WillTurner
data Dir = Izq | Der
```



Inducción y recursión en Haskell

■ ¿Cómo usar todas estas técnicas en Haskell?

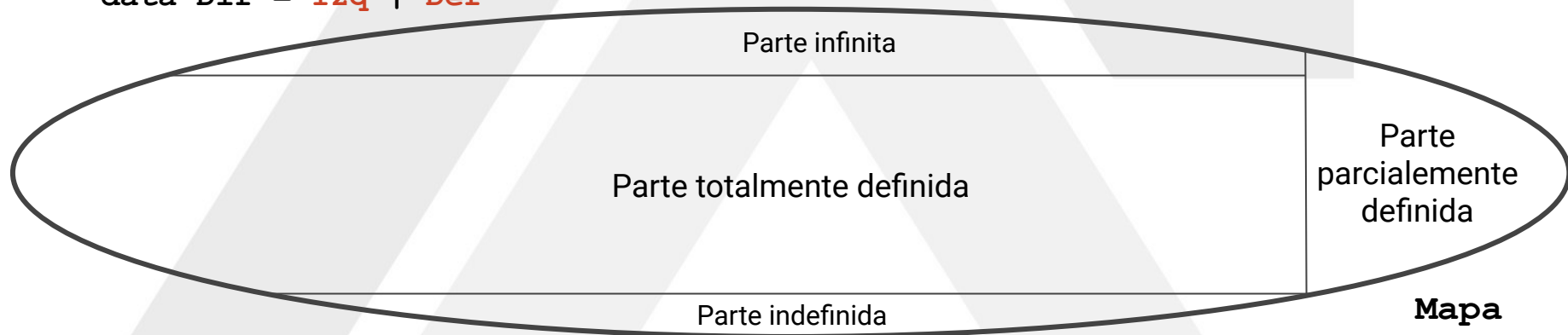
■ ¿Definición inductiva de estructuras?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa
```

```
data Cofre = Monedas Int | CorazonDe Pirata | Joyas Int | Nada
```

```
data Pirata = DavyJones | WillTurner
```

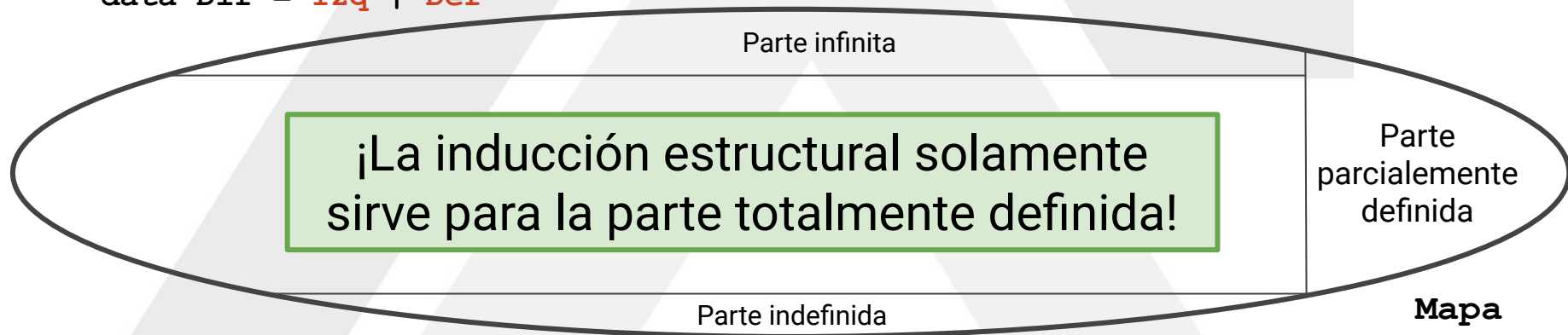
```
data Dir = Izq | Der
```



Inducción y recursión en Haskell

- ❑ ¿Cómo usar todas estas técnicas en Haskell?
- ❑ ¿Definición inductiva de estructuras?

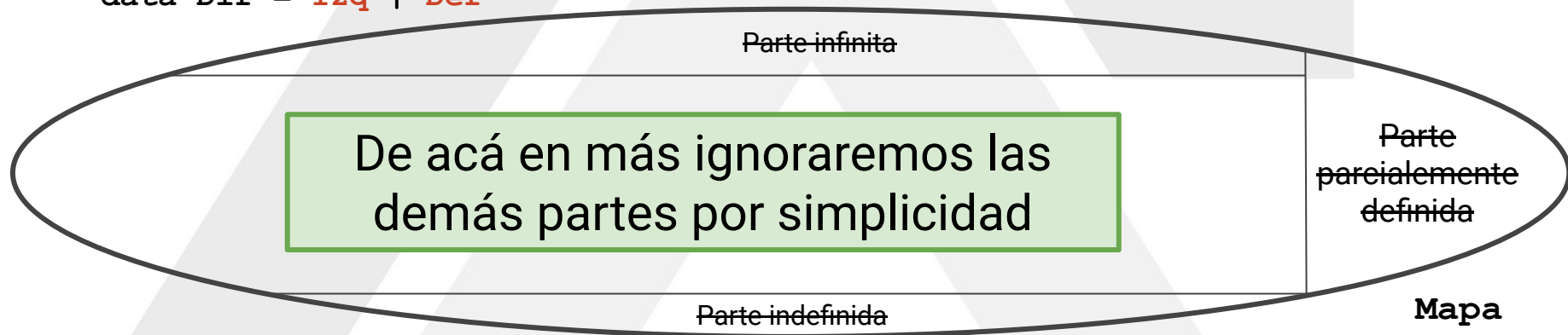
```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa  
data Cofre = Monedas Int | CorazonDe Pirata | Joyas Int | Nada  
data Pirata = DavyJones | WillTurner  
data Dir = Izq | Der
```



Inducción y recursión en Haskell

- ❑ ¿Cómo usar todas estas técnicas en Haskell?
- ❑ ¿Definición inductiva de estructuras?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa  
data Cofre = Monedas Int | CorazonDe Pirata | Joyas Int | Nada  
data Pirata = DavyJones | WillTurner  
data Dir = Izq | Der
```



Inducción y recursión en Haskell

- ¿Cómo usar todas estas técnicas en Haskell?
- ¿Definición inductiva de estructuras?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa
data Cofre = Monedas Int | CorazonDe Pirata | Joyas Int | Nada
data Pirata = DavyJones | WillTurner
data Dir = Izq | Der
```



Inducción y recursión en Haskell

- ¿Cómo usar todas estas técnicas en Haskell?
 - ¿Definición de funciones por recursión estructural?

Inducción y recursión en Haskell

- ❑ ¿Cómo usar todas estas técnicas en Haskell?
 - ❑ ¿Definición de funciones por recursión estructural?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa
```

```
tesoroEn :: Mapa -> Cofre
```

```
tesoroEn ...
```

Inducción y recursión en Haskell

- ¿Cómo usar todas estas técnicas en Haskell?
 - ¿Definición de funciones por recursión estructural?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa
```

```
tesoroEn :: Mapa -> Cofre
```

```
tesoroEn ...
```

¿Cómo definirla?

Inducción y recursión en Haskell

- ❑ ¿Cómo usar todas estas técnicas en Haskell?
 - ❑ ¿Definición de funciones por recursión estructural?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa
```

```
tesoroEn :: Mapa -> Cofre
```

```
tesoroEn ...
```

¡Por recursión en
la estructura de
Mapa!

¿Cómo definirla?

Inducción y recursión en Haskell

- ❑ ¿Cómo usar todas estas técnicas en Haskell?
 - ❑ ¿Definición de funciones por recursión estructural?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa
```

```
tesoroEn :: Mapa -> Cofre
```

```
tesoroEn (Equis c) = ... c ...
```

```
tesoroEn (Recto n m) = ... n ... tesoroEn m ...
```

```
tesoroEn (Giro d m) = ... d ... tesoroEn m ...
```

¡Por recursión en
la estructura de
Mapa!

Primero se
plantea el
esquema

Inducción y recursión en Haskell

- ❑ ¿Cómo usar todas estas técnicas en Haskell?
 - ❑ ¿Definición de funciones por recursión estructural?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa
```

```
tesoroEn :: Mapa -> Cofre
```

```
tesoroEn (Equis c) = ... c ...
```

```
tesoroEn (Recto n m) = caminarYDar n (tesoroEn m)
```

```
tesoroEn (Giro d m) = girarYDar d (tesoroEn m)
```

¡Se definen los
casos inductivos!

Inducción y recursión en Haskell

- ❑ ¿Cómo usar todas estas técnicas en Haskell?
 - ❑ ¿Definición de funciones por recursión estructural?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa
```

```
tesoroEn :: Mapa -> Cofre
```

```
tesoroEn (Equis c) = cavarYObtener c
```

```
tesoroEn (Recto n m) = caminarYDar n (tesoroEn m)
```

```
tesoroEn (Giro d m) = girarYDar d (tesoroEn m)
```

Se definen los
casos base

Inducción y recursión en Haskell

- ❑ ¿Cómo usar todas estas técnicas en Haskell?
 - ❑ ¿Definición de funciones por recursión estructural?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa
```

```
tesoroEn :: Mapa -> Cofre
```

```
tesoroEn (Equis c)    = cavarYObtener c
```

```
tesoroEn (Recto n m) = caminarYDar n (tesoroEn m)
```

```
tesoroEn (Giro d m)  = girarDar d (tesoroEn m)
```

```
cavarYObtener c = c    -- Implementaciones triviales
```

```
caminarYDar n c = c    -- Podrían complicarse con un
```

```
girarYDar    d c = c    -- terreno y verificaciones...
```

Se completa con
las funciones
auxiliares

Inducción y recursión en Haskell

- ❑ ¿Cómo usar todas estas técnicas en Haskell?
 - ❑ ¿Definición de funciones por recursión estructural?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa
```

```
comprimido :: Mapa -> Mapa  -- Comprime tramos rectos consecutivos
```

```
comprimido (Equis c)      = ... c ...
```

```
comprimido (Recto n m)    = ... n ... comprimido m ...
```

```
comprimido (Giro d m)     = ... d ... comprimido m ...
```


Inducción y recursión en Haskell

- ❑ ¿Cómo usar todas estas técnicas en Haskell?
- ❑ ¿Definición de funciones por recursión estructural?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa
```

```
comprimido :: Mapa -> Mapa  -- Comprime tramos rectos consecutivos
```

```
comprimido (Equis c)      = ... c ...
```

```
comprimido (Recto n m)    = compRecto n (comprimido m)
```

```
comprimido (Giro d m)     = Giro d (comprimido m)
```

Inducción y recursión en Haskell

- ❑ ¿Cómo usar todas estas técnicas en Haskell?
- ❑ ¿Definición de funciones por recursión estructural?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa
```

```
comprimido :: Mapa -> Mapa  -- Comprime tramos rectos consecutivos
```

```
comprimido (Equis c)      = Equis c
```

```
comprimido (Recto n m) = compRecto n (comprimido m)
```

```
comprimido (Giro d m)  = Giro d (comprimido m)
```

Inducción y recursión en Haskell

- ❑ ¿Cómo usar todas estas técnicas en Haskell?
- ❑ ¿Definición de funciones por recursión estructural?

```
data Mapa = Equis Cofre | Recto Int Mapa | Giro Dir Mapa
```

```
comprimido :: Mapa -> Mapa  -- Comprime tramos rectos consecutivos
```

```
comprimido (Equis c)      = Equis c
```

```
comprimido (Recto n m) = compRecto n (comprimido m)
```

```
comprimido (Giro d m)    = Giro d (comprimido m)
```

```
compRecto n (Recto n' m) = Recto (n+n') m
```

```
compRecto n m              = Recto n m
```

Inducción y recursión en Haskell

- ❏ ¿Qué cosas interesantes se pueden expresar?
- ❏ ¿Y qué pasó con la 3era pregunta?

Inc



Inc



La Programación Desconocida

Inducción y recursión en Haskell

- ❏ ¿Qué cosas interesantes se pueden expresar?
- ❏ ¿Y qué pasó con la 3era pregunta?
- ❏ Quizás pueda conocerse algo de esto en...

La Programación Desconocida

Nah. En la próxima clase. :)



Resumen

Resumen

- ❑ **Definición** de conjuntos *por inducción estructural*
 - ❑ Reglas base, reglas inductivas, el *MENOR*
- ❑ **Definición** de funciones *por recursión estructural*
 - ❑ Una ecuación por cada regla base e inductiva
 - ❑ Utilizar la misma función en las partes inductivas (y solo ahí)
 - ❑ PROPIEDAD: toda función recursiva estructural termina
- ❑ Aplicación de las ideas en Haskell
 - ❑ Tipos algebraicos recursivos
 - ❑ Funciones por recursión estructural sobre ellos