Repaso de Recursión y primeros pasos con Tipos Abstractos de Datos

Algoritmos y Estructuras de Datos II

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

4 de abril de 2018

¿Qué era recursión?

- ¿Qué es una función recursiva?

¿Qué era recursión?

- ¿Qué es una función recursiva?

  Una función que en su definición se llama a sí misma...

¿Qué era recursión?

- ¿Qué es una función recursiva?

  Una función que en su definición se llama a sí misma...

- ¿Por qué no se cuelga esto?

¿Qué era recursión?

- ¿Qué es una función recursiva?

  Una función que en su definición se llama a sí misma...

- ¿Por qué no se cuelga esto?

  Tiene al menos un *caso base*, y ...

¿Qué era recursión?

- ¿Qué es una función recursiva?

  Una función que en su definición se llama a sí misma...

- ¿Por qué no se cuelga esto?

  Tiene al menos un *caso base*, y ...

  cada llamado recursivo "nos acerca" un poco al caso base.

¿Qué era recursión?

- ¿Qué es una función recursiva?

  Una función que en su definición se llama a sí misma...

- ¿Por qué no se cuelga esto?

  Tiene al menos un *caso base*, y ...

  cada llamado recursivo "nos acerca" un poco al caso base.

- En general, una función recursiva tiene,
  - uno o más *casos base*
  - uno o más llamados recursivos

¿Qué era recursión?

- ¿Qué es una función recursiva?

  Una función que en su definición se llama a sí misma...

- ¿Por qué no se cuelga esto?

  Tiene al menos un *caso base*, y ...

  cada llamado recursivo "nos acerca" un poco al caso base.

- En general, una función recursiva tiene,
  - uno o más *casos base*
  - uno o más llamados recursivos

- ¿Cómo podemos asegurarnos de que una función recursiva está bien escrita? Es decir, resuelve lo que queremos que resuelva...

¿Cómo llevarse bien con la recursión?

- Tenemos que asegurarnos de algunas cosas:

¿Cómo llevarse bien con la recursión?

- Tenemos que asegurarnos de algunas cosas:
  1. ... que existan uno o más *casos base* (bien resueltos)

¿Cómo llevarse bien con la recursión?

- Tenemos que asegurarnos de algunas cosas:
  1. ... que existan uno o más *casos base* (bien resueltos)
  2. ... que los llamados recursivos "simplifiquen" el problema a resolver (y "alcancen" a algún caso base).

¿Cómo llevarse bien con la recursión?

- Tenemos que asegurarnos de algunas cosas:
  1. ... que existan uno o más *casos base* (bien resueltos)
  2. ... que los llamados recursivos “simplifiquen” el problema a resolver (y “alcancen” a algún caso base).
     ¿Con eso alcanza?

¿Cómo llevarse bien con la recursión?

- Tenemos que asegurarnos de algunas cosas:
  1. ... que existan uno o más *casos base* (bien resueltos)
  2. ... que los llamados recursivos “simplifiquen” el problema a resolver (y “alcancen” a algún caso base).
     ¿Con eso alcanza?
     Ejemplo:

$$0! = 1$$
$$n! = n + (n-1)!$$

¿Cómo llevarse bien con la recursión?

- Tenemos que asegurarnos de algunas cosas:
  1. ... que existan uno o más *casos base* (bien resueltos)
  2. ... que los llamados recursivos "simplifiquen" el problema a resolver (y "alcancen" a algún caso base).
     ¿Con eso alcanza?
     Ejemplo:
     $$\begin{aligned} 0! &= 1 \\ n! &= n * (n-1)! \end{aligned}$$
  3. ... que si el llamado recursivo funciona bien, entonces nosotros también!

¿Cómo llevarse bien con la recursión?

- Tenemos que asegurarnos de algunas cosas:
  1. ... que existan uno o más *casos base* (bien resueltos)
  2. ... que los llamados recursivos "simplifiquen" el problema a resolver (y "alcancen" a algún caso base).
     ¿Con eso alcanza?
     Ejemplo:
     $$0! = 1$$
     $$n! = n * (n-1)!$$
  3. ... que si el llamado recursivo funciona bien, entonces nosotros también!

- Si se cumplen esas **tres cosas**, entonces nuestra función está bien escrita y hace lo que queremos...

  ¿Por qué? ¿Cómo nos convencemos de esto? ¿Qué se aplica para demostrar que es correcto?

¿Dónde vamos a usar recursión?

- Vamos a usar recursión para axiomatizar el comportamiento de las funciones de los TADs.
- Antes de empezar a practicar, repasemos los TADs básicos del apunte de la página...

Ejercicio: Reverso

Extender el tipo $\textsc{Secuencia}(\alpha)$ con la operación *reverso* que devuelve la misma secuencia en orden inverso.

Ejercicio: Reverso

Extender el tipo $\textsc{Secuencia}(\alpha)$ con la operación *reverso* que devuelve la misma secuencia en orden inverso.

$$\text{reverso} : \text{secu}(\alpha) \longrightarrow \text{secu}(\alpha) \qquad \{\ \}$$

$$\text{reverso}(<>) \equiv <>$$

$$\text{reverso}(a \bullet s) \equiv \text{reverso}(s) \circ a$$

Ejercicio: esPrefijo?

Extender el tipo $\textsc{Secuencia}(\alpha)$ con la operación *esPrefijo?(s, t)* que verifica si la secuencia s es prefijo de la secuencia t.

Ejercicio: esPrefijo?

Extender el tipo $\textsc{Secuencia}(\alpha)$ con la operación *esPrefijo?(s, t)* que verifica si la secuencia s es prefijo de la secuencia t.

esPrefijo? : $\text{secu}(\alpha) \times \text{secu}(\alpha) \longrightarrow \text{bool}$ $\{\ \}$

$\text{esPrefijo?}(<>, t) \equiv \textit{true}$

$\text{esPrefijo?}(a \bullet s, t) \equiv \neg\ \text{vacía?}(t) \wedge_{\mathrm{L}} \text{prim}(t) = a \wedge \text{esPrefijo?}(s, \text{fin}(t))$

Ejercicio: Reemplazar

Extender el tipo $\text{Secuencia}(\alpha)$ con la operación *reemplazar*(s, a, b) que reemplaza en la secuencia s todas las apariciones del elemento a por el elemento b.

Ejercicio: Reemplazar

Extender el tipo $\textsc{Secuencia}(\alpha)$ con la operación *reemplazar(s, a, b)* que reemplaza en la secuencia *s* todas las apariciones del elemento *a* por el elemento *b*.

$$\text{reemplazar} \;:\; \text{secu}(\alpha) \times \alpha \times \alpha \;\longrightarrow\; \text{secu}(\alpha) \qquad \{\ \}$$

$$\text{reemplazar}(<>, a, b) \;\equiv\; <>$$

$$\text{reemplazar}(t \bullet s, a, b) \;\equiv\; (\textbf{if}\ \mathrm{t} = a\ \ \textbf{then}\ \ b\ \ \textbf{else}\ \ \mathrm{t}\ \ \textbf{fi}) \bullet \text{reemplazar}(s, a, b)$$

Ejercicio: #Apariciones

Definir la operación *#Apariciones(ab, a)* sobre el TAD $\text{AB}(\alpha)$ (árboles binarios) que devuelve la cantidad de apariciones del elemento *a* en el árbol *ab*.

Ejercicio: #Apariciones

Definir la operación *#Apariciones(ab, a)* sobre el TAD $\text{AB}(\alpha)$ (árboles binarios) que devuelve la cantidad de apariciones del elemento *a* en el árbol *ab*.

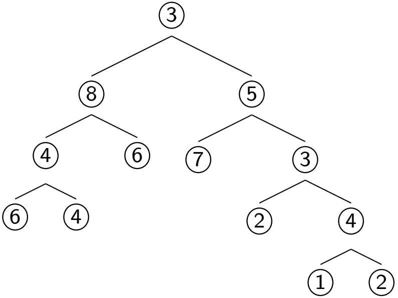
$$\text{\#Apariciones} \;:\; \text{ab}(\alpha) \times \alpha \;\longrightarrow\; \text{nat} \qquad \{\,\}$$

$$\text{\#Apariciones}(nil,\ a) \;\equiv\; 0$$

$$\begin{aligned}\text{\#Apariciones}(bin(i, r, d),\ a) \;\equiv\; & \textbf{if } r = a \textbf{ then } 1 \textbf{ else } 0 \textbf{ fi} + \\ & \text{\#Apariciones}(i, a) + \text{\#Apariciones}(d, a)\end{aligned}$$

Ejercicio: ultimoNivelCompleto

Definir la operación *ultimoNivelCompleto* sobre el TAD $\mathrm{AB}(\alpha)$ (árboles binarios) que devuelve el número del último nivel que está completo (es decir, aquél que tiene todos los nodos posibles).



Ejercicio: ultimoNivelCompleto

Definir la operación *ultimoNivelCompleto* sobre el TAD $\mathrm{AB}(\alpha)$ (árboles binarios) que devuelve el número del último nivel que está completo (es decir, aquél que tiene todos los nodos posibles).

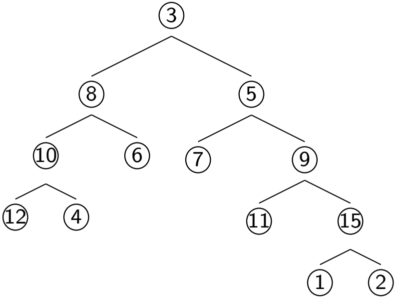
$$\text{ultimoNivelCompleto} : \text{ab}(\alpha) \longrightarrow \text{nat} \qquad \{\ \}$$

$$\text{ultimoNivelCompleto}(nil) \equiv 0$$

$$\text{ultimoNivelCompleto}(bin(i, r, d)) \equiv \text{min}(\text{ultimoNivelCompleto}(i), \text{ultimoNivelCompleto}(d)) + 1$$

Ejercicio: CaminoHasta

Definir la operación *CaminoHasta(ab, a)* sobre el TAD $\text{AB}(\alpha)$ (árboles binarios) que devuelve una secuencia que representa el camino desde la raíz del árbol *ab* hasta el elemento *a* en el mismo (asumir que el árbol no tiene elementos repetidos). Si el elemento *a* no aparece en el árbol, debe devolverse la secuencia vacía.



Ejercicio: CaminoHasta

Definir la operación *CaminoHasta(ab, a)* sobre el TAD $\text{AB}(\alpha)$ (árboles binarios) que devuelve una secuencia que representa el camino desde la raíz del árbol *ab* hasta el elemento *a* en el mismo (asumir que el árbol no tiene elementos repetidos). Si el elemento *a* no aparece en el árbol, debe devolverse la secuencia vacía.

$$\text{CaminoHasta} \; : \; \text{ab}(\alpha) \times \alpha \longrightarrow \text{secu}(\alpha) \qquad \{\ \}$$

$$\text{CaminoHasta}(nil,\ a) \equiv <>$$

$$\begin{array}{ll}
\text{CaminoHasta}(bin(i, r, d),\ a) \equiv & \textbf{if } r = a \textbf{ then} \\
& \quad a \bullet <> \\
& \textbf{else} \\
& \quad \textbf{if } \#\text{apariciones}(i, a) > 0 \textbf{ then} \\
& \quad\quad r \bullet \text{CaminoHasta}(i, a) \\
& \quad \textbf{else} \\
& \quad\quad \textbf{if } \#\text{apariciones}(d, a) > 0 \textbf{ then} \\
& \quad\quad\quad r \bullet \text{CaminoHasta}(d, a) \\
& \quad\quad \textbf{else} \\
& \quad\quad\quad <> \\
& \quad\quad \textbf{fi} \\
& \quad \textbf{fi} \\
& \textbf{fi}
\end{array}$$

Tipos Abstractos de Datos

- ¿Qué es un tipo de datos?

Tipos Abstractos de Datos

- ¿Qué es un tipo de datos?

  Conjunto de valores y operaciones...

Tipos Abstractos de Datos

- ¿Qué es un tipo de datos?

  Conjunto de valores y operaciones...

- ¿Qué es un tipo abstracto de datos?

Tipos Abstractos de Datos

- ¿Qué es un tipo de datos?

  Conjunto de valores y operaciones...

- ¿Qué es un tipo abstracto de datos?

  Es un tipo especificado por su comportamiento...

Tipos Abstractos de Datos

- ¿Qué es un tipo de datos?

  Conjunto de valores y operaciones...

- ¿Qué es un tipo abstracto de datos?

  Es un tipo especificado por su comportamiento...

- Es decir, en un TADs se definen funcionalidades y se explica **qué** hace cada una, sin especificar **cómo** lo hace... (el “cómo” lo dejamos para más adelante).

- ¿Qué es un tipo de datos?

  Conjunto de valores y operaciones...

- ¿Qué es un tipo abstracto de datos?

  Es un tipo especificado por su comportamiento...

- Es decir, en un TADs se definen funcionalidades y se explica **qué** hace cada una, sin especificar **cómo** lo hace... (el "cómo" lo dejamos para más adelante).

- Esto facilita la resolución de problemas "grandes" modularizando en problemas de menor complejidad.

Ejercicio: Agenda de compromisos

Necesitamos una agenda en la cual podamos registrar compromisos para un día. Por ejemplo, "Turno con el dentista de 16 a 17 hs", o "Reunión de cátedra de 18 a 20". No hay problema con que los compromisos registrados en la agenda se solapen. De hecho, nos interesa saber, dada una hora del día, qué compromisos tenemos en ese momento. Además, dado un intervalo de horas, quisiéramos poder saber qué hora del intervalo es la más ocupada en la agenda.

Resolución ...en teoría

¿¿Cómo empezamos??

1. Leer bien el enunciado e identificar qué cosas son importantes y qué cosas no lo son.
2. Definir los observadores y la igualdad observacional.
3. Definir los generadores.
4. Definir las otras operaciones.
5. Definir las restricciones donde corresponda
6. Incluir otras operaciones auxiliares, de haberlas.
7. Axiomatizar todo.

Resolución . . . en teoría

¿¿Cómo empezamos??

1. Leer bien el enunciado e identificar qué cosas son importantes y qué cosas no lo son.
2. Definir los observadores y la igualdad observacional.
3. Definir los generadores.
4. Definir las otras operaciones.
5. Definir las restricciones donde corresponda
6. Incluir otras operaciones auxiliares, de haberlas.
7. Axiomatizar todo.

Pero. . . ¿se puede hacer así, realmente, paso por paso?

En general conviene ir pensando algunas cosas en paralelo . . .

Repasemos el enunciado

Necesitamos una agenda en la cual podamos registrar compromisos para un día. Por ejemplo, “Turno con el dentista de 16 a 17 hs”, o “Reunión de cátedra de 18 a 20hs”. No hay problema con que los compromisos registrados en la agenda se solapen. De hecho, nos interesa saber, dada una hora del día, qué compromisos tenemos en ese momento. Además, dado un intervalo de horas, quisiéramos poder saber qué hora del intervalo es la más ocupada en la agenda.

Repasemos el enunciado

Necesitamos una agenda en la cual podamos registrar compromisos para un día. Por ejemplo, "Turno con el dentista de 16 a 17 hs", o "Reunión de cátedra de 18 a 20hs". No hay problema con que los compromisos registrados en la agenda se solapen. De hecho, nos interesa saber, dada una hora del día, qué compromisos tenemos en ese momento. Además, dado un intervalo de horas, quisiéramos poder saber qué hora del intervalo es la más ocupada en la agenda.

Repasemos el enunciado

Necesitamos una agenda en la cual podamos registrar compromisos para un día. Por ejemplo, "Turno con el dentista de 16 a 17 hs", o "Reunión de cátedra de 18 a 20hs". No hay problema con que los compromisos registrados en la agenda se solapen. De hecho, nos interesa saber, dada una hora del día, qué compromisos tenemos en ese momento. Además, dado un intervalo de horas, quisiéramos poder saber qué hora del intervalo es la más ocupada en la agenda.

Repasemos el enunciado

Necesitamos una agenda en la cual podamos registrar compromisos para un día. Por ejemplo, "Turno con el dentista de 16 a 17 hs", o "Reunión de cátedra de 18 a 20hs". No hay problema con que los compromisos registrados en la agenda se solapen. De hecho, nos interesa saber, dada una hora del día, qué compromisos tenemos en ese momento. Además, dado un intervalo de horas, quisiéramos poder saber qué hora del intervalo es la más ocupada en la agenda.

Repasemos el enunciado

Necesitamos una agenda en la cual podamos registrar compromisos para un día. Por ejemplo, "Turno con el dentista de 16 a 17 hs", o "Reunión de cátedra de 18 a 20hs". No hay problema con que los compromisos registrados en la agenda se solapen. De hecho, nos interesa saber, dada una hora del día, qué compromisos tenemos en ese momento. Además, dado un intervalo de horas, quiséramos poder saber qué hora del intervalo es la más ocupada en la agenda.

Repasemos el enunciado

Necesitamos una agenda en la cual podamos registrar compromisos para un día. Por ejemplo, "Turno con el dentista de 16 a 17 hs", o "Reunión de cátedra de 18 a 20hs". No hay problema con que los compromisos registrados en la agenda se solapen. De hecho, nos interesa saber, dada una hora del día, qué compromisos tenemos en ese momento. Además, dado un intervalo de horas, quisiéramos poder saber qué hora del intervalo es la más ocupada en la agenda.

- La agenda debe permitir registrar compromisos (String), con su hora de inicio y su hora de fin (Nats).

Repasemos el enunciado

Necesitamos una agenda en la cual podamos registrar compromisos para un día. Por ejemplo, “Turno con el dentista de 16 a 17 hs”, o “Reunión de cátedra de 18 a 20hs”. No hay problema con que los compromisos registrados en la agenda se solapen. De hecho, nos interesa saber, dada una hora del día, qué compromisos tenemos en ese momento. Además, dado un intervalo de horas, quiséramos poder saber qué hora del intervalo es la más ocupada en la agenda.

- La agenda debe permitir registrar compromisos (String), con su hora de inicio y su hora de fin (Nats).
- Deberíamos poder consultar los compromisos de un determinado momento.

Repasemos el enunciado

Necesitamos una agenda en la cual podamos registrar compromisos para un día. Por ejemplo, "Turno con el dentista de 16 a 17 hs", o "Reunión de cátedra de 18 a 20hs". No hay problema con que los compromisos registrados en la agenda se solapen. De hecho, nos interesa saber, dada una hora del día, qué compromisos tenemos en ese momento. Además, dado un intervalo de horas, quisiéramos poder saber qué hora del intervalo es la más ocupada en la agenda.

- La agenda debe permitir registrar compromisos (String), con su hora de inicio y su hora de fin (Nats).
- Deberíamos poder consultar los compromisos de un determinado momento.
- Saber qué hora de un intervalo es la más ocupada.

- ¿Observadores?

- ¿Observadores?

compromisos	:	agenda × nat	⟶	conj(compromiso)
horaMasOcupada	:	agenda × nat × nat	⟶	nat

Observadores, generadores e igualdad

- ¿Observadores?

$$\boxed{\text{compromisos} \;:\; \text{agenda} \times \text{nat} \longrightarrow \text{conj(compromiso)}}$$

- ¿Igualdad observacional?

Observadores, generadores e igualdad

- ¿Observadores?

compromisos : agenda $\times$ nat $\longrightarrow$ conj(compromiso)

- ¿Igualdad observacional?

$(\forall a, a'\colon \text{agenda})$
$\left(a =_{\text{obs}} a' \Leftrightarrow (\forall h\colon \text{nat})\ (\text{compromisos}(a, h) =_{\text{obs}} \text{compromisos}(a', h))\right)$

Observadores, generadores e igualdad

- ¿Observadores?

$$\text{compromisos} \; : \; \text{agenda} \times \text{nat} \longrightarrow \text{conj(compromiso)}$$

- ¿Igualdad observacional?

$$(\forall a, a' \text{: agenda})$$
$$(\; a =_{\text{obs}} a' \Leftrightarrow (\forall h \text{: nat}) \; (\text{compromisos}(a, h) =_{\text{obs}} \text{compromisos}(a', h)))$$

- ¿Generadores?

Observadores, generadores e igualdad

- ¿Observadores?

compromisos : agenda $\times$ nat $\longrightarrow$ conj(compromiso)

- ¿Igualdad observacional?

$$(\forall a, a'\text{: agenda})$$
$$(\ a =_{\mathrm{obs}} a' \Leftrightarrow (\forall h\text{: nat})\ (\text{compromisos}(a, h) =_{\mathrm{obs}} \text{compromisos}(a', h)))$$

- ¿Generadores?

crearAgenda : $\longrightarrow$ agenda
registrar : agenda $\times$ compromiso $\times$ nat $\times$ nat $\longrightarrow$ agenda

Observadores, generadores e igualdad

- ¿Observadores?

compromisos : agenda $\times$ nat $\longrightarrow$ conj(compromiso)

- ¿Igualdad observacional?

$$(\forall a, a'\text{: agenda})$$
$$(\ a =_{\text{obs}} a' \Leftrightarrow (\forall h\text{: nat})\ (\text{compromisos}(a, h) =_{\text{obs}} \text{compromisos}(a', h)))$$

- ¿Generadores?

crearAgenda : $\longrightarrow$ agenda
registrar : agenda $\times$ compromiso $\times$ nat d $\times$ nat h $\longrightarrow$ agenda $\{d < h\}$

Observadores, generadores e igualdad

- ¿Observadores?

compromisos : agenda $\times$ nat $\longrightarrow$ conj(compromiso)

- ¿Igualdad observacional?

$(\forall a, a'\text{: agenda})$ $(a =_{\text{obs}} a' \Leftrightarrow (\forall h\text{: nat})\ (\text{compromisos}(a, h) =_{\text{obs}} \text{compromisos}(a', h)))$

- ¿Generadores?

crearAgenda : $\longrightarrow$ agenda registrar : agenda $\times$ compromiso $\times$ nat d $\times$ nat h $\longrightarrow$ agenda $\{d < h\}$

- ¿Otras operaciones?

Observadores, generadores e igualdad

- ¿Observadores?

compromisos : agenda $\times$ nat $\longrightarrow$ conj(compromiso)

- ¿Igualdad observacional?

$(\forall a, a'\colon \text{agenda})$
$(\ a =_{\text{obs}} a' \Leftrightarrow (\forall h\colon \text{nat})\ (\text{compromisos}(a, h) =_{\text{obs}} \text{compromisos}(a', h)))$

- ¿Generadores?

crearAgenda : $\longrightarrow$ agenda
registrar : agenda $\times$ compromiso $\times$ nat d $\times$ nat h $\longrightarrow$ agenda $\{d < h\}$

- ¿Otras operaciones?

horaMasOcupada : agenda $\times$ nat $\times$ nat $\longrightarrow$ nat

- ¿Observadores?

compromisos : agenda $\times$ nat $\longrightarrow$ conj(compromiso)

- ¿Igualdad observacional?

$(\forall a, a'$: agenda)
$(\ a =_{\mathrm{obs}} a' \Leftrightarrow (\forall h$: nat) (compromisos$(a, h) =_{\mathrm{obs}}$ compromisos$(a', h)))$

- ¿Generadores?

crearAgenda	:	$\longrightarrow$ agenda	
registrar	:	agenda $\times$ compromiso $\times$ nat d $\times$ nat h $\longrightarrow$ agenda	$\{d < h\}$

- ¿Otras operaciones?

horaMasOcupada	:	agenda $\times$ nat d $\times$ nat h $\longrightarrow$ nat	$\{d \leq h\}$

AXIOMAS

$\text{compromisos} \ : \ \text{agenda} \times \text{nat} \ \longrightarrow \ \text{conj(compromiso)}$

$\text{compromisos(crearAgenda}, \ h) \ \equiv \ ???$

$\text{compromisos(registrar}(a, ini, fin, c), \ h) \ \equiv \ ???$

$\text{horaMasOcupada} \ : \ \text{agenda} \times \text{secu(nat)} \ s \ \longrightarrow \ \text{nat} \qquad \{\neg \ \text{vacía?}(s))\}$

$\text{horaMasOcupada(a, s)} \ \equiv \ ???$

Axiomas

$\text{compromisos} \;:\; \text{agenda} \times \text{nat} \;\longrightarrow\; \text{conj(compromiso)}$

$\text{compromisos}(\text{crearAgenda},\ h) \;\equiv\; \emptyset$

$$
\begin{aligned}
\text{compromisos}(\text{registrar}(a, ini, fin, c),\ h) \;\equiv\; & \textbf{if } ini < h < fin \ \textbf{ then} \\
& \quad \text{Ag}(c,\ \text{compromisos}(a,\ h)) \\
& \textbf{else} \\
& \quad \text{compromisos}(a,\ h) \\
& \textbf{fi}
\end{aligned}
$$

$\text{horaMasOcupada} \;:\; \text{agenda} \times \text{secu(nat)}\ s \;\longrightarrow\; \text{nat} \qquad \{\neg\ \text{vacía?}(s))\}$

horaMasOcupada(a, s) $\equiv$ queda de tarea para practicar recursión...

Ejercicio: Insoportables

Insoportables es un programa televisivo muy exitoso que sale al aire todas las noches; en él se debate acerca de las relaciones entre los personajes de la farándula (los "famosos"). Debido a la gran cantidad de peleas y reconciliaciones, los productores nos encargaron el desarrollo de un sistema que permita saber en todo momento quiénes están peleados y quiénes no.
Además, los productores quieren poder determinar quién es el famoso que actualmente está involucrado en la mayor cantidad de peleas. Las peleas del pasado no interesan.
Otra premisa de los productores es que una vez que una persona es famosa, sigue siendo famosa para siempre.

Resolución ...en teoría

Ver qué tenemos que especificar, y en base a esto:

1. Definir los observadores y la igualdad observacional.
2. Definir los generadores.
3. Definir las otras operaciones.
4. Definir las restricciones donde corresponda
5. Incluir otras operaciones auxiliares, de haberlas.
6. Axiomatizar todo.

Enunciado

Insoportables es un programa televisivo muy exitoso que sale al aire todas las noches; en él se debate acerca de las relaciones entre los personajes de la farándula (los "famosos"). Debido a la gran cantidad de peleas y reconciliaciones, los productores nos encargaron el desarrollo de un sistema que permita saber en todo momento quiénes están peleados y quiénes no.
Además, los productores quieren poder determinar quién es el famoso que actualmente está involucrado en la mayor cantidad de peleas. Las peleas del pasado no interesan.
Otra premisa de los productores es que una vez que una persona es famosa, sigue siendo famosa para siempre.

Insoportables es un programa televisivo muy exitoso que sale al aire todas las noches; en él se debate acerca de las relaciones entre los personajes de la farándula (los "famosos"). Debido a la gran cantidad de **peleas** y **reconciliaciones**, los productores nos encargaron el desarrollo de un sistema que permita saber en todo momento quiénes están peleados y quiénes no.
Además, los productores quieren poder determinar quién es el famoso que actualmente está involucrado en la mayor cantidad de peleas. Las peleas del pasado no interesan.
Otra premisa de los productores es que una vez que una persona es famosa, sigue siendo famosa para siempre.

¿Qué tenemos que especificar?

¿Qué tenemos que especificar?

- El sistema debería permitir registrar nuevos famosos, nuevas peleas y nuevas reconciliaciones.
- Determinar quiénes son famosos.
- Qué famosos están peleados. (¿La relación “estar peleado con” siempre es simétrica?)
- Y quién es el famoso involucrado en la mayor cantidad de peleas. (¿Siempre hay uno?)

Definir los observadores

Definir los observadores

- Los observadores deben permitirnos distinguir todas las instancias.

Definir los observadores

- Los observadores deben permitirnos distinguir todas las instancias.
- Es decir, deberíamos poder definir todas las operaciones a partir de los observadores.

Definir los observadores

- Los observadores deben permitirnos distinguir todas las instancias.
- Es decir, deberíamos poder definir todas las operaciones a partir de los observadores.

$$\boxed{\begin{array}{llll} \text{famosos} : \text{bdf} & \longrightarrow \text{conj(famoso)} & & \\ \text{enemigos} : \text{bdf } b \times \text{famoso } f & \longrightarrow \text{conj(famoso)} & & \{f \in \text{famosos}(b)\} \end{array}}$$

Definir los observadores

- Los observadores deben permitirnos distinguir todas las instancias.
- Es decir, deberíamos poder definir todas las operaciones a partir de los observadores.

$$\begin{array}{lllll} \text{famosos} & : & \text{bdf} & \longrightarrow & \text{conj(famoso)} \\ \text{enemigos} & : & \text{bdf } b \times \text{famoso } f & \longrightarrow & \text{conj(famoso)} \qquad \{f \in \text{famosos}(b)\} \end{array}$$

- ¿Es posible responder todas las preguntas en base a esta información?

Escribir la igualdad observacional

- Ya podemos escribir la igualdad observacional.

- Ya podemos escribir la igualdad observacional.

igualdad observacional

$$(\forall b, b' : \text{bdf})\ \left(b =_{\text{obs}} b' \Longleftrightarrow \begin{pmatrix} \text{famosos}(b) =_{\text{obs}} \text{famosos}(b') \wedge_{\text{L}} \\ (\forall f\colon\ \text{famoso})(f \in \text{famosos}(b) \Rightarrow_{\text{L}} \\ \text{enemigos}(b, f) =_{\text{obs}} \text{enemigos}(b', f)) \end{pmatrix} \right)$$

Definir los generadores

Definir los generadores

- Los generadores deben permitirnos construir todas las instancias observacionalmente distintas.

Definir los generadores

- Los generadores deben permitirnos construir todas las instancias observacionalmente distintas.

$$\text{crearBD} \; : \; \longrightarrow \; \text{bdf}$$

$$\text{nuevoFamoso} \; : \; \text{bdf } b \times \text{famoso } f \;\longrightarrow\; \text{bdf} \qquad \{f \notin \text{famosos}(b)\}$$

$$\text{pelear} \; : \; \text{bdf } b \times \text{famoso } f \times \text{famoso } f' \;\longrightarrow\; \text{bdf}$$
$$\{\{f, f'\} \subseteq \text{famosos}(b) \wedge_{\mathrm{L}} f \notin \text{enemigos}(b, f') \wedge f \neq f'\}$$

Definir los generadores

- Los generadores deben permitirnos construir todas las instancias observacionalmente distintas.

crearBD : $\longrightarrow$ bdf
nuevoFamoso : bdf b $\times$ famoso f $\longrightarrow$ bdf $\{f \notin \text{famosos}(b)\}$
pelear : bdf b $\times$ famoso f $\times$ famoso f' $\longrightarrow$ bdf $\{\{f, f'\} \subseteq \text{famosos}(b) \wedge_{\mathrm{L}} f \notin \text{enemigos}(b, f') \wedge f \neq f'\}$

- ¿Podemos generar todas las instancias?

Definir las otras operaciones

Definir las otras operaciones

- Las otras operaciones tienen que ser suficientes para permitir utilizar el TAD fácilmente.

- Las otras operaciones tienen que ser suficientes para permitir utilizar el TAD fácilmente.

$$\mathsf{reconciliar} : \mathsf{bdf}\ b \times \mathsf{famoso}\ f \times \mathsf{famoso}\ f' \longrightarrow \mathsf{bdf}$$
$$\{\{f, f'\} \subseteq \mathsf{famosos}(b) \wedge_{\mathrm{L}} (f \in \mathsf{enemigos}(b, f'))\}$$

$$\mathsf{másPeleador} : \mathsf{bdf}\ b \longrightarrow \mathsf{famoso} \qquad \{\mathsf{famosos}(b) \neq \emptyset\}$$

Definir las otras operaciones

- Las otras operaciones tienen que ser suficientes para permitir utilizar el TAD fácilmente.

$$\text{reconciliar} : \text{bdf } b \times \text{famoso } f \times \text{famoso } f' \longrightarrow \text{bdf}$$
$$\{\{f, f'\} \subseteq \text{famosos}(b) \wedge_{\mathrm{L}} (f \in \text{enemigos}(b, f'))\}$$

$$\text{másPeleador} : \text{bdf } b \longrightarrow \text{famoso} \qquad \{\text{famosos}(b) \neq \emptyset\}$$

- ¿Se pueden definir solamente en base a los observadores y aplicación de generadores?

Axiomatización I

- Encuentre el/los error/es:

$$\text{enemigos} \;:\; \text{bdf } b \times \text{famoso } f \longrightarrow \text{conj(famoso)} \qquad \{f \in \text{famosos}(b)\}$$

$$\text{enemigos(crearBD}, f) \equiv \emptyset$$

$$\text{enemigos(nuevoFamoso}(b, g), f) \equiv \text{enemigos}(b, f)$$

$$\text{enemigos(pelear}(b, g, g'), f) \equiv \begin{array}{l} \textbf{if } f \in \{g, g'\} \textbf{ then} \\ \quad \{g, g'\} \setminus \{f\} \\ \textbf{else} \\ \quad \emptyset \\ \textbf{fi} \cup \text{enemigos}(b, f) \end{array}$$

Axiomatización I

- Encuentre el/los error/es:

$$\text{enemigos} \ : \ \text{bdf } b \times \text{famoso } f \longrightarrow \text{conj(famoso)} \qquad \{f \in \text{famosos}(b)\}$$

$$\text{enemigos(crearBD}, f) \equiv \emptyset$$

$$\text{enemigos(nuevoFamoso}(b, g), f) \equiv \text{enemigos}(b, f)$$

$$\begin{aligned}\text{enemigos(pelear}(b, g, g'), f) \equiv\ & \textbf{if } f \in \{g, g'\} \textbf{ then} \\ & \quad \{g, g'\} \setminus \{f\} \\ & \textbf{else} \\ & \quad \emptyset \\ & \textbf{fi} \cup \text{enemigos}(b, f)\end{aligned}$$

- ¿Qué pasa con las restricciones?

$$\text{enemigos} : \text{bdf } b \times \text{famoso } f \longrightarrow \text{conj(famoso)} \qquad \{f \in \text{famosos}(b)\}$$

$$\begin{aligned}
\text{enemigos}(\text{nuevoFamoso}(b, g),\ f) \equiv\ & \textbf{if } g = f \textbf{ then} \\
& \quad \emptyset \\
& \textbf{else} \\
& \quad \text{enemigos}(b,\ f) \\
& \textbf{fi}
\end{aligned}$$

$$\begin{aligned}
\text{enemigos}(\text{pelear}(b,\ g,\ g'),\ f) \equiv\ & \textbf{if } f \in \{g, g'\} \textbf{ then} \\
& \quad \{g, g'\} \setminus \{f\} \\
& \textbf{else} \\
& \quad \emptyset \\
& \textbf{fi} \cup \text{enemigos}(b,\ f)
\end{aligned}$$

- ¿Qué hay de raro acá?

$$
\begin{array}{lcl}
\text{reconciliar} \;:\; \text{bdf } b \times \text{famoso } f \times \text{famoso } f' & \longrightarrow & \text{bdf} \\
 & & \{\{f, f'\} \subseteq \text{famosos}(b) \wedge f \in \text{enemigos}(b, f')\} \\
\text{reconciliar}(\text{pelear}(b, g, g'),\ f,\ f') & \equiv & \textbf{if } \{g, g'\} = \{f, f'\} \textbf{ then} \\
 & & \quad b \\
 & & \textbf{else} \\
 & & \quad \text{pelear}(\text{reconciliar}(b,\ f,\ f'),\ g,\ g') \\
 & & \textbf{fi} \\
\text{reconciliar}(\text{nuevoFamoso}(b, g),\ f,\ f') & \equiv & \textbf{if } g \in \{f, f'\} \textbf{ then} \\
 & & \quad b \\
 & & \textbf{else} \\
 & & \quad \text{nuevoFamoso}(\text{reconciliar}(b,\ f,\ f'),\ g) \\
 & & \textbf{fi}
\end{array}
$$

Axiomatización I

- ¿Qué hay de raro acá?

$$\begin{array}{lcl}
\text{reconciliar} & : & \text{bdf } b \times \text{famoso } f \times \text{famoso } f' \longrightarrow \text{bdf} \\
& & \{\{f, f'\} \subseteq \text{famosos}(b) \wedge f \in \text{enemigos}(b, f')\} \\
\text{reconciliar}(\text{pelear}(b, g, g'),\ f,\ f') & \equiv & \textbf{if } \{g, g'\} = \{f, f'\} \textbf{ then} \\
& & \quad b \\
& & \textbf{else} \\
& & \quad \text{pelear}(\text{reconciliar}(b,\ f,\ f'),\ g,\ g') \\
& & \textbf{fi} \\
\text{reconciliar}(\text{nuevoFamoso}(b, g),\ f,\ f') & \equiv & \textbf{if } g \in \{f, f'\} \textbf{ then} \\
& & \quad b \\
& & \textbf{else} \\
& & \quad \text{nuevoFamoso}(\text{reconciliar}(b,\ f,\ f'),\ g) \\
& & \textbf{fi}
\end{array}$$

- ¿Es incorrecto chequear que $g \in \{f, f'\}$?

Axiomatización I

- ¿Qué hay de raro acá?

$$
\begin{array}{lcl}
\text{reconciliar} : \text{bdf } b \times \text{famoso } f \times \text{famoso } f' & \longrightarrow & \text{bdf} \\
 & & \{\{f, f'\} \subseteq \text{famosos}(b) \wedge f \in \text{enemigos}(b, f')\} \\
\text{reconciliar}(\text{pelear}(b, g, g'),\ f,\ f') & \equiv & \textbf{if } \{g, g'\} = \{f, f'\} \textbf{ then} \\
 & & \quad b \\
 & & \textbf{else} \\
 & & \quad \text{pelear}(\text{reconciliar}(b,\ f,\ f'),\ g,\ g') \\
 & & \textbf{fi} \\
\text{reconciliar}(\text{nuevoFamoso}(b, g),\ f,\ f') & \equiv & \textbf{if } g \in \{f, f'\} \textbf{ then} \\
 & & \quad b \\
 & & \textbf{else} \\
 & & \quad \text{nuevoFamoso}(\text{reconciliar}(b,\ f,\ f'),\ g) \\
 & & \textbf{fi}
\end{array}
$$

- ¿Es incorrecto chequear que $g \in \{f, f'\}$?
- ¿Qué pasa con las restricciones?

Axiomatización I

$$
\begin{array}{lcl}
\mathsf{reconciliar} : \mathsf{bdf}\ b \times \mathsf{famoso}\ f \times \mathsf{famoso}\ f' & \longrightarrow & \mathsf{bdf} \\
 & & \{\{f, f'\} \subseteq \mathsf{famosos}(b) \wedge f \in \mathsf{enemigos}(b, f')\} \\
\mathsf{reconciliar}(\mathsf{pelear}(b, g, g'),\ f,\ f') & \equiv & \textbf{if } \{g, g'\} = \{f, f'\} \textbf{ then} \\
 & & \quad b \\
 & & \textbf{else} \\
 & & \quad \mathsf{pelear}(\mathsf{reconciliar}(b,\ f,\ f'),\ g,\ g') \\
 & & \textbf{fi} \\
\mathsf{reconciliar}(\mathsf{nuevoFamoso}(b, g),\ f,\ f') & \equiv & \mathsf{nuevoFamoso}(\mathsf{reconciliar}(b,\ f,\ f'),\ g)
\end{array}
$$

- Encuentre el/los posible/s error/es.

$$\text{másPeleador}(b) \equiv \text{prim}(\text{másPeleadores}(b))$$

donde

$$\text{másPeleadores} : \text{bdf} \longrightarrow \text{secu}(\text{famoso})$$

Axiomatización II

- Encuentre el/los posible/s error/es.

$$\text{másPeleador}(b) \equiv \text{prim}(\text{másPeleadores}(b))$$

donde

$$\text{másPeleadores} : \text{bdf} \longrightarrow \text{secu}(\text{famoso})$$

- ¿Qué hay de raro acá?

Axiomatización II

- Encuentre el/los posible/s error/es.

$$\text{másPeleador}(b) \equiv \text{prim}(\text{másPeleadores}(b))$$

donde

$$\text{másPeleadores} : \text{bdf} \longrightarrow \text{secu}(\text{famoso})$$

- ¿Qué hay de raro acá?
- ¿Qué pasa con las instancias generadas distintas pero que deberían ser iguales?

Axiomatización II: congruencia

- Recordemos que una operación f es congruente (con la igualdad observacional) si y sólo si para cada par $i =_{obs} i'$ también vale $f(i) =_{obs} f(i')$.

Axiomatización II: congruencia

- Recordemos que una operación f es congruente (con la igualdad observacional) si y sólo si para cada par $i =_{obs} i'$ también vale $f(i) =_{obs} f(i')$.

- Una solución correcta:

$$\text{másPeleador}(b) \equiv \text{dameUno}(\text{másPeleadores}(b))$$

donde

$$\text{másPeleadores} : \text{bdf} \longrightarrow \text{conj}(\text{famoso})$$

AXIOMATIZACIÓN II: CONGRUENCIA

- Recordemos que una operación f es congruente (con la igualdad observacional) si y sólo si para cada par $i =_{obs} i'$ también vale $f(i) =_{obs} f(i')$.

- Una solución correcta:

$$\text{másPeleador}(b) \equiv \text{dameUno}(\text{másPeleadores}(b))$$

donde

$$\text{másPeleadores} : \text{bdf} \longrightarrow \text{conj(famoso)}$$

- Otra solución:

$$\text{másPeleador}(b) \equiv \text{elegirUnMásPeleador}(b, \text{famosos}(b))$$

donde

$$\text{elegirUnMásPeleador} : \text{bdf} \times \text{conj(famoso)}\ cf \longrightarrow \text{famoso} \qquad \{\neg\emptyset(cf)\}$$

Minimalidad

- ¿Sería buena idea agregar el siguiente observador?

sonEnemigos? : bdf b $\times$ famoso f $\times$ famoso f' $\longrightarrow$ bool $\{\ldots\}$

Minimalidad

- ¿Sería buena idea agregar el siguiente observador?

sonEnemigos?	:	bdf b $\times$ famoso f $\times$ famoso f'	$\longrightarrow$	bool	$\{\dots\}$

- ¿Sería buena idea que "reconciliar" fuese un generador?

reconciliar	:	bdf b $\times$ famoso f $\times$ famoso f'	$\longrightarrow$	bdf	$\{\dots\}$

Minimalidad

- ¿Sería buena idea agregar el siguiente observador?

sonEnemigos? : bdf b × famoso f × famoso f' $\longrightarrow$ bool		$\{\dots\}$

- ¿Sería buena idea que "reconciliar" fuese un generador?

reconciliar : bdf b × famoso f × famoso f' $\longrightarrow$ bdf		$\{\dots\}$

En el contexto de Algo 2 ...

Minimalidad

- ¿Sería buena idea agregar el siguiente observador?

sonEnemigos? :	bdf b × famoso f × famoso f'	$\longrightarrow$ bool	$\{\ldots\}$

- ¿Sería buena idea que "reconciliar" fuese un generador?

reconciliar :	bdf b × famoso f × famoso f'	$\longrightarrow$ bdf	$\{\ldots\}$

En el contexto de Algo 2 ...

- El conjunto de observadores debe ser minimal.
- Es *deseable* que el conjunto de generadores sea minimal.
- De lo contrario, la especificación se torna más difícil, menos clara y más propensa a errores.

Enunciado (bis)

Supongamos que los productores de *Insoportables* están contentos con la especificación entregada, pero que ahora también quieren poder determinar cuáles son los famosos que más se pelearon en su vida.

¿Qué modificaciones hay que hacerle al TAD?

¿Qué tenemos que especificar? (bis)

¿Qué tenemos que especificar? (bis)

- Quiénes son famosos.
- Qué famosos están peleados.
- Quién es el famoso involucrado en la mayor cantidad de peleas.
- El sistema debería permitir la definición de nuevos famosos, nuevas peleas y nuevas reconciliaciones.
- Quiénes son los famosos que más veces se pelearon en su vida.

Definir los observadores (bis)

Definir los observadores (bis)

- Necesitamos una operación que nos permita determinar quiénes son los más peleadores históricos.

$\boxed{\text{másPeleadoresHistóricos} \;:\; \text{bdf } b \;\longrightarrow\; \text{conj(famoso)}}$

Definir los observadores (bis)

- Necesitamos una operación que nos permita determinar quiénes son los más peleadores históricos.

$\boxed{\text{másPeleadoresHistóricos} \ : \ \text{bdf } b \ \longrightarrow \ \text{conj(famoso)}}$

- ¿Estaría bien incluir esta operación como otra operación?
- Cuidado con la congruencia.

Definir los observadores (bis)

- Necesitamos una operación que nos permita determinar quiénes son los más peleadores históricos.

másPeleadoresHistóricos : bdf b $\longrightarrow$ conj(famoso)

- ¿Estaría bien incluir esta operación como otra operación?
- Cuidado con la congruencia.
- ¿Estaría bien incluir esta operación como observador?

Definir los observadores (bis)

- Necesitamos una operación que nos permita determinar quiénes son los más peleadores históricos.

másPeleadoresHistóricos : bdf b $\longrightarrow$ conj(famoso)

- ¿Estaría bien incluir esta operación como otra operación?
- Cuidado con la congruencia.
- ¿Estaría bien incluir esta operación como observador?
- Cuidado con la congruencia (más sutil; pensarlo).

Definir los observadores (bis)

- Una forma de resolver esto correctamente sería agregar como observador básico una función que permita determinar la cantidad de peleas (incluyendo historia) de un famoso.

$$\boxed{\#\text{peleasHistórico} \;:\; \text{bdf } b \times \text{famoso } f \;\longrightarrow\; \text{nat} \qquad \{f \in \text{famosos}(b)\}}$$

Definir los observadores (bis)

- Una forma de resolver esto correctamente sería agregar como observador básico una función que permita determinar la cantidad de peleas (incluyendo historia) de un famoso.

$$\#\text{peleasHistórico} \;:\; \text{bdf } b \times \text{famoso } f \longrightarrow \text{nat} \qquad \{f \in \text{famosos}(b)\}$$

- Notar que este observador capturaría un *poco* más de detalle del que el enunciado pedía.

Definir los observadores (bis)

- Una forma de resolver esto correctamente sería agregar como observador básico una función que permita determinar la cantidad de peleas (incluyendo historia) de un famoso.

#peleasHistórico	:	bdf b $\times$ famoso f	$\longrightarrow$	nat	$\{f \in$ famosos$(b)\}$

- Notar que este observador capturaría un *poco* más de detalle del que el enunciado pedía.
- Si tuviéramos un observador que devuelva el historial completo de peleas de un famoso f dado, también podríamos usarlo para calcular #peleasHistórico. Sin embargo, esto capturaría **mucho** más detalle del que el enunciado pedía (¡y distinguiría instancias que no nos pidieron distinguir!).

Definir los observadores (bis)

- Una forma de resolver esto correctamente sería agregar como observador básico una función que permita determinar la cantidad de peleas (incluyendo historia) de un famoso.

$$\boxed{\#\text{peleasHistórico} \;:\; \text{bdf } b \times \text{famoso } f \longrightarrow \text{nat} \qquad \{f \in \text{famosos}(b)\}}$$

- Notar que este observador capturaría un *poco* más de detalle del que el enunciado pedía.
- Si tuviéramos un observador que devuelva el historial completo de peleas de un famoso f dado, también podríamos usarlo para calcular #peleasHistórico. Sin embargo, esto capturaría **mucho** más detalle del que el enunciado pedía (¡y distinguiría instancias que no nos pidieron distinguir!).
- ... esto sería **sobreespecificar**.

Definir los generadores (bis)

- ¿Podemos construir todas las instancias observacionalmente distintas con los generadores que tenemos?
  (crearBD, nuevoFamoso, pelear)

Definir los generadores (bis)

- ¿Podemos construir todas las instancias observacionalmente distintas con los generadores que tenemos? (crearBD, nuevoFamoso, pelear)
- Tomarse un momento para actualizar la igualdad observacional y los axiomas.

Definir los generadores (bis)

- ¿Podemos construir todas las instancias observacionalmente distintas con los generadores que tenemos? (crearBD, nuevoFamoso, pelear)
- Tomarse un momento para actualizar la igualdad observacional y los axiomas.
- ¡Ya no, porque ahora (parte de) la historia es observable!

Definir los generadores (bis)

- ¿Podemos construir todas las instancias observacionalmente distintas con los generadores que tenemos? (crearBD, nuevoFamoso, pelear)
- Tomarse un momento para actualizar la igualdad observacional y los axiomas.
- ¡Ya no, porque ahora (parte de) la historia es observable!
- Solución:

Definir los generadores (bis)

- ¿Podemos construir todas las instancias observacionalmente distintas con los generadores que tenemos?
  (crearBD, nuevoFamoso, pelear)
- Tomarse un momento para actualizar la igualdad observacional y los axiomas.
- ¡Ya no, porque ahora (parte de) la historia es observable!
- Solución:

Definir los generadores (bis)

- ¿Podemos construir todas las instancias observacionalmente distintas con los generadores que tenemos? (crearBD, nuevoFamoso, pelear)
- Tomarse un momento para actualizar la igualdad observacional y los axiomas.
- ¡Ya no, porque ahora (parte de) la historia es observable!
- Solución: agregar “reconciliar” como un generador.

Definir los generadores (bis)

- ¿Podemos construir todas las instancias observacionalmente distintas con los generadores que tenemos? (crearBD, nuevoFamoso, pelear)
- Tomarse un momento para actualizar la igualdad observacional y los axiomas.
- ¡Ya no, porque ahora (parte de) la historia es observable!
- Solución: agregar "reconciliar" como un generador.

$$\boxed{\begin{array}{l} \text{reconciliar} \;:\; \text{bdf } b \times \text{famoso } f \times \text{famoso } f' \;\longrightarrow\; \text{bdf} \\ \hfill \{\{f, f'\} \subseteq \text{famosos}(b) \wedge f \in \text{enemigos}(b, f')\} \end{array}}$$

- ¿Podemos generar ahora todas las instancias?

Definir los generadores (bis)

- ¿Podemos construir todas las instancias observacionalmente distintas con los generadores que tenemos? (crearBD, nuevoFamoso, pelear)
- Tomarse un momento para actualizar la igualdad observacional y los axiomas.
- ¡Ya no, porque ahora (parte de) la historia es observable!
- Solución: agregar "reconciliar" como un generador.

$$\text{reconciliar} : \text{bdf } b \times \text{famoso } f \times \text{famoso } f' \longrightarrow \text{bdf}$$
$$\{\{f, f'\} \subseteq \text{famosos}(b) \wedge f \in \text{enemigos}(b, f')\}$$

- ¿Podemos generar ahora todas las instancias?
- Con esta función, tenemos memoria de todas las peleas históricas.

Axiomatización (bis)

- Ahora que "reconciliar" es un generador, no se axiomatiza.
- Sin embargo, hay que axiomatizar todas las funciones para el caso del generador "reconciliar".

Axiomatización (bis)

- Ahora que "reconciliar" es un generador, no se axiomatiza.
- Sin embargo, hay que axiomatizar todas las funciones para el caso del generador "reconciliar".

$$\boxed{\#\text{peleasHistorico}(\text{reconciliar}(b,\ g,\ g'),\ f)\ \equiv\ \#\text{peleasHistorico}(b,\ f)}$$

Conclusiones

- Proceso de construcción de un TAD.
  - Determinar qué hay que especificar.
  - Observadores básicos e igualdad observacional
  - Generadores
  - Otras operaciones
  - Axiomas
  - Operaciones auxiliares

Conclusiones

- Proceso de construcción de un TAD.
  - Determinar qué hay que especificar.
  - Observadores básicos e igualdad observacional
  - Generadores
  - Otras operaciones
  - Axiomas
  - Operaciones auxiliares
- ¿Es posible escribir por completo los generadores antes de ponerse a pensar siquiera en los observadores?

Conclusiones

- Proceso de construcción de un TAD.
  - Determinar qué hay que especificar.
  - Observadores básicos e igualdad observacional
  - Generadores
  - Otras operaciones
  - Axiomas
  - Operaciones auxiliares
- ¿Es posible escribir por completo los generadores antes de ponerse a pensar siquiera en los observadores?
- ¿Es posible escribir por completo los observadores antes de ponerse a pensar siquiera en los generadores?

Conclusiones

- Proceso de construcción de un TAD.
  - Determinar qué hay que especificar.
  - Observadores básicos e igualdad observacional
  - Generadores
  - Otras operaciones
  - Axiomas
  - Operaciones auxiliares
- ¿Es posible escribir por completo los generadores antes de ponerse a pensar siquiera en los observadores?
- ¿Es posible escribir por completo los observadores antes de ponerse a pensar siquiera en los generadores?
- Usualmente todo esto termina siendo un proceso iterativo.