Algoritmos y Estructuras de datos II - Trabajo Práctico 1

Grupo BlackMesa

TAD posición es tupla(int $\times$ int)

1. TAD Sokoban

TAD SOKOBAN

géneros sokoban

exporta observadores, generadores

usa PARTIDA, CONJ, NIVEL, DIRECCION

igualdad observacional

$$(\forall s_1, s_2 : \text{sokoban}) \left(s_1 =_{\text{obs}} s_2 \iff \begin{pmatrix} \text{partidaActual}(s_1) =_{\text{obs}} \text{partidaActual}(s_2) \land \\ \text{niveles}(s_1) =_{\text{obs}} \text{niveles}(s_2) \land \\ \text{nivelesCompletados}(s_1) =_{\text{obs}} \text{nivelesCompletados}(s_2) \end{pmatrix} \right)$$

observadores básicos

partidaActual	: sokoban	$\longrightarrow$	partida
niveles	: sokoban	$\longrightarrow$	conj(nivel)
nivelesCompletados	: sokoban	$\longrightarrow$	conj(nivel)

generadores

sokoban	: conj(nivel)	$\longrightarrow$	sokoban	
mover	: sokoban s $\times$ direccion d	$\longrightarrow$	sokoban	$\left\{ \begin{array}{l} \text{esTransitable(partidaActual(s), siguientePosicion(d, posicionDelJugador(partidaActual(s)))) } \land \\ ((\neg \text{hayCaja(partidaActual(s), siguientePosicion(d, posicionDelJugador(partidaActual(s))))) } \lor \\ (\text{esTransitable(partidaActual(s), siguientePosicion(d, siguientePosicion(d, posicionDelJugador(partidaActual(s)))))) } \land \\ \neg \text{ hayCaja(partidaActual(s), siguientePosicion(d, siguientePosicion(d, posicionDelJugador(partidaActual(s))))))} \end{array} \right\}$
usarBomba	: sokoban s	$\longrightarrow$	sokoban	$\{\#\text{bombas(partidaActual(s))} \neq 0\}$
deshacer	: sokoban s	$\longrightarrow$	sokoban	$\{\neg \text{vacia(estadosAnteriores(partidaActual(s)))}\}$

axiomas $\forall s$: sokoban, $\forall d$: direccion, $\forall cn$: conj(nivel)

partidaActual(sokoban(cn))	≡	iniciarPartida(dameUno(cn))
partidaActual(mover(s, d))	≡	**if** victorioso?(moverJugador(partidaActual(s), d)) **then** iniciarPartida(dameUno(niveles(s)−nivelesCompletados(mover(s, d)))) **else** moverJugador(partidaActual(s), d) **fi**
partidaActual(usarBomba(s))	≡	ponerBomba(partidaActual(s))
partidaActual(deshacer(s))	≡	Undo(partidaActual(s)
niveles(sokoban(cn))	≡	cn
niveles(mover(s, d))	≡	niveles(s)
niveles(usarBomba(s))	≡	niveles(s)
niveles(deshacer(s))	≡	niveles(s)
nivelesCompletados(sokoban(cn))	≡	$\emptyset$
nivelesCompletados(mover(s, d))	≡	**if** victorioso?(moverJugador(partidaActual(s), d)) **then** Ag(nivelActual(partidaActual(s)), nivelesCompletados(s)) **else** nivelesCompletados(s) **fi**
nivelesCompletados(usarBomba(s))	≡	nivelesCompletados(s)
nivelesCompletados(deshacer(s))	≡	nivelesCompletados(s)

Fin TAD

2. TAD Partida

TAD PARTIDA

géneros partida

exporta observadores, generadores

usa BOOL, NIVEL, POSICION, DIRECCION, NAT, PILA

igualdad observacional

$$(\forall pa_1, pa_2 : \text{partida}) \left(pa_1 =_{\text{obs}} pa_2 \iff \begin{pmatrix} \text{nivelActual}(pa_1) =_{\text{obs}} \text{nivelActual}(pa_2) \wedge \\ \text{posicionDelJugador}(pa_1) =_{\text{obs}} \\ \text{posicionDelJugador}(pa_2) \wedge \\ (\forall p\text{: posicion}) \\ (\text{esTransitable}(pa_2,\ \text{p}) =_{\text{obs}} \text{esTransitable}(pa_1,\ \text{p}) \\ \wedge\ \text{hayCaja}(pa_1,\ \text{p}) =_{\text{obs}} \text{hayCaja}(pa_2,\ \text{p})) \wedge \\ \text{estadosAnteriores}(pa_1) =_{\text{obs}} \\ \text{estadosAnteriores}(pa_2) \end{pmatrix} \right)$$

observadores básicos

posicionDelJugador	:	partida	$\longrightarrow$	posicion
nivelActual	:	partida	$\longrightarrow$	nivel
#bombas	:	partida	$\longrightarrow$	nat
esTransitable	:	partida $\times$ posicion	$\longrightarrow$	bool
estadosAnteriores	:	partida	$\longrightarrow$	pila(partida)
hayCaja	:	partida $\times$ posicion	$\longrightarrow$	bool

generadores

iniciarPartida	:	nivel	$\longrightarrow$	partida
moverJugador	:	partida pa $\times$ direccion d	$\longrightarrow$	partida
ponerBomba	:	partida p	$\longrightarrow$	partida
Undo	:	partida	$\longrightarrow$	partida

otras operaciones

siguientePosicion	:	direccion d $\times$ posicion	$\longrightarrow$	posicion
revisarDepositos	:	conj(posicion)	$\longrightarrow$	bool
victorioso?	:	partida	$\longrightarrow$	bool

axiomas $\forall pa$: partida, $\forall n$: nivel, $\forall p$: posicion, $\forall d$: direccion, $\forall cd$: conj(posicion)

posicionDelJugador(iniciarPartida(n))	$\equiv$	posicionInicial(n)
posicionDelJugador(moverJugador(pa, d))	$\equiv$	siguientePosicion(d, posicionDelJugador(pa))
posicionDelJugador(ponerBomba(pa))	$\equiv$	posicionDelJugador(pa)
posicionDelJugador(Undo(pa))	$\equiv$	posicionDelJugador(tope(estadosAnteriores(pa)))
nivelActual(iniciarPartida(n))	$\equiv$	n
nivelActual(moverJugador(pa, d))	$\equiv$	nivelActual(pa)
nivelActual(ponerBomba(pa))	$\equiv$	nivelActual(pa)

nivelActual(Undo(pa))	$\equiv$	nivelActual(pa)
#bombas(iniciarPartida(n))	$\equiv$	bombasIniciales(n))
#bombas(moverJugador(pa, d))	$\equiv$	#bombas(pa)
#bombas(ponerBomba(pa))	$\equiv$	pred(#bombas(pa))
#bombas(Undo(pa))	$\equiv$	#bombas(tope(estadosAnteriores(pa)))
esTransitable(iniciarPartida(n), p)	$\equiv$	$\neg$tienePared(n, p)
esTransitable((moverJugador(pa, d), p))	$\equiv$	esTransitable(pa, p)
esTransitable(ponerBomba(pa), p)	$\equiv$	(π_1(p) $=_{obs}$ π_1(posicionDelJugador(pa))) $\vee$ (π_2(p) $=_{obs}$ π_2(posicionDelJugador(pa))) $\vee$ esTransitable(pa, p)
esTransitable(Undo(pa, p)	$\equiv$	esTransitable(tope(estadosAnteriores(pa)), p)
hayCaja(iniciarPartida(n), p)	$\equiv$	tieneCaja(n, p)
hayCaja(moverJugador(pa,d), p)	$\equiv$	**if** p $=_{obs}$ siguientePosicion(d, siguientePosicion(d, posicionDelJugador(pa)))) **then** hayCaja(pa, siguientePosicion(d, posicionDelJugador(pa))) **else** hayCaja(pa, p) **fi**
hayCaja(ponerBomba(pa), p)	$\equiv$	hayCaja(pa, p)
hayCaja(Undo(pa), p)	$\equiv$	hayCaja(tope(estadosAnteriores(pa)), p)
estadosAnteriores(iniciarPartida(n))	$\equiv$	vacia
estadosAnteriores(moverJugador(pa, d), p)	$\equiv$	apilar(pa, estadosAnteriores(pa))
estadosAnteriores(ponerBomba(pa))	$\equiv$	apilar(pa, estadosAnteriores(pa))
estadosAnteriores(Undo(pa))	$\equiv$	desapilar(estadosAnteriores(pa))
siguientePosicion(d, p)	$\equiv$	**if** char?(d) $=_{obs}$ 'n'$\vee$ char?(d) $=_{obs}$'s' **then** $< \pi_1(p)$, **if** char?(d) $=_{obs}$ 'n' **then** $\pi_2(p)+1$ **else** $\pi_2(p)-1$ **fi** $>$ **else** $<$ **if** char?(d) $=_{obs}$ 'e' **then** $\pi_1(p)+1$ **else** $\pi_1(p)-1$ **fi** ,$\pi_2(p)>$ **fi**
revisarDepositos(cd)	$\equiv$	$\emptyset$?(cd) $\vee$ (tieneCaja(dameUno(cd)) $\wedge$ revisarDepositos(sinUno(cd)))
victorioso?(pa)	$\equiv$	revisarDepositos(despositos(nivelActual(pa))))

Fin TAD

3. TAD Nivel

TAD NIVEL

géneros nivel

exporta observadores, generadores

usa BOOL, POSICION, CONJ

igualdad observacional

$$(\forall n_1, n_2 : \text{nivel}) \left(n_1 =_{\text{obs}} n_2 \iff \left(\begin{array}{l} (\forall p\text{: posicion})(\\ \{\text{tienePared}(n_1, \text{p}) =_{\text{obs}} \text{tienePared}(n_2, \text{p}) \wedge \\ \text{tieneCaja}(n_1, \text{p}) =_{\text{obs}} \text{tieneCaja}(n_2, \text{p})\} \wedge \\ \text{depositos}(n_1) =_{\text{obs}} \text{depositos}(n_2) \wedge \\ \text{posicionInicial}(n_1) =_{\text{obs}} \text{posicionInicial}(n_2) \wedge \\ \text{bombasIniciales}(n_1) =_{\text{obs}} \text{bombasIniciales}(n_2)) \end{array} \right) \right)$$

observadores básicos

tienePared	: nivel × posicion	$\longrightarrow$	bool	
depositos	: nivel	$\longrightarrow$	conj(posicion)	
tieneCaja	: nivel × posicion	$\longrightarrow$	bool	
posicionInicial	: nivel	$\longrightarrow$	posicion	
bombasIniciales	: nivel	$\longrightarrow$	nat	

generadores

nivel	: posicion × nat	$\longrightarrow$	nivel	
ponerDeposito	: nivel × posicion p	$\longrightarrow$	nivel	$\{\neg\text{tienePared(p)}\}$
ponerPared	: nivel × posicion p	$\longrightarrow$	nivel	$\{\neg\text{esDeposito(p)} \wedge \neg\text{tieneCaja(p)} \wedge \neg\text{tienePared(p)} \wedge \text{p} \neq \text{posicionInicial(n)}\}$
ponerCaja	: nivel × posicion p	$\longrightarrow$	nivel	$\{\neg\text{tienePared(p)} \wedge \neg\text{tieneCaja(p)} \wedge \text{p} \neq \text{posicionInicial(n)}\}$

axiomas $\forall n$: nivel, $\forall p, p'$: posicion, $\forall b$: nat

tienePared(nivel(p, b), p')	$\equiv$	False
tienePared(ponerDeposito(n, p), p')	$\equiv$	$\neg((\text{p} = \text{p'}) \wedge \text{tienePared(n, p')})$
tienePared(ponerPared(n, p), p')	$\equiv$	$(\text{p} = \text{p'}) \vee \text{tienePared(n, p')}$
tienePared(ponerCaja(n, p), p')	$\equiv$	$\neg((\text{p} = \text{p'}) \wedge \text{tienePared(n, p')})$
depositos(nivel(p, b))	$\equiv$	$\emptyset$
depositos(ponerDeposito(n, p)	$\equiv$	Ag(p, depositos(n))
depositos(ponerPared(n, p)	$\equiv$	depositos(n)
deposito(ponerCaja(n, p)	$\equiv$	deposito(n)
tieneCaja(nivel(p, b), p')	$\equiv$	false
tieneCaja(ponerDeposito(n, p), p')	$\equiv$	tieneCaja(n, p')
tieneCaja(ponerPared(n, p), p')	$\equiv$	$\neg((\text{p} = \text{p'}) \wedge \text{tieneCaja(n, p')})$
tieneCaja(ponerCaja(n, p), p')	$\equiv$	$(\text{p} = \text{p'}) \vee \text{tieneCaja(n, p')}$

bombasIniciales(nivel(p ,b)) $\equiv$ b

bombasIniciales(ponerDeposito(n, p)) $\equiv$ bombasIniciales(n)

bombasIniciales(ponerPared(n , p)) $\equiv$ bombasIniciales(n)

bombasIniciales(ponerCaja(n, p)) $\equiv$ bombasIniciales(n)

posicionInicial(nivel(p, b)) $\equiv$ p

posicionInicial(ponerDeposito(n, p)) $\equiv$ posicionInicial(n)

posicionInicial(ponerPared(n, p)) $\equiv$ posicionInicial(n)

posicionInicial(ponerCaja(n, p)) $\equiv$ posicionInicial(n)

Fin TAD

4. TAD Direccion

TAD Direccion

géneros dir

usa CHAR

exporta observadores, generadores

igualdad observacional

$(\forall dir_1, dir_2 : \text{dir})\ (dir_1 =_{\text{obs}} dir_2 \iff (\text{char?}(dir_1) =_{\text{obs}} \text{char?}(dir_2)))$

observadores básicos

char? : dir $\longrightarrow$ char

generadores

Norte : $\longrightarrow$ dir

Sur : $\longrightarrow$ dir

Este : $\longrightarrow$ dir

Oeste : $\longrightarrow$ dir

axiomas $\forall d$: dir

char?(Norte) $\equiv$ 'n'

char?(Sur) $\equiv$'s'

char?(Este) $\equiv$ 'e'

char?(Oeste) $\equiv$ 'o'

Fin TAD