

1. Módulo Ciudad Robotica

Interfaz

se explica con: SECUENCIA(α), ITERADOR BIDIRECCIONAL(α).

géneros: ciudad, itLista(α).

usa:

Operaciones básicas de ciudad

CREAR(in m : mapa) $\rightarrow res$: ciudad

Pre $\equiv \{\text{true}\}$

Post $\equiv \{res =_{\text{obs}} \text{crear}(m)\}$

Complejidad: $\Theta(1)$

Descripción: genera una nueva Ciudad.

Aliasing: lo ideal seria no copiar ese mapa no?

ENTRAR(in ts : conj(tag), in e : estacion, in/out c : ciudad)

Pre $\equiv \{c =_{\text{obs}} c_0 \wedge e \in \text{estaciones}(c)\}$

Post $\equiv \{c =_{\text{obs}} \text{entrar}(ts, e, c_0)\}$

Complejidad: $\Theta(\text{copy}(a))..$

MOVER(in u : rur, in e : estacion, in/out c : ciudad)

Pre $\equiv \{c =_{\text{obs}} c_0 \wedge e \in \text{estaciones}(c) \wedge e \in \text{robots}(c)\}$

Post $\equiv \{c =_{\text{obs}} \text{mover}(u, e, c_0)\}$

Complejidad: $\Theta(\text{copy}(a))..$

INSPECCION(in e : estacion, in/out c : ciudad)

Pre $\equiv \{c =_{\text{obs}} c_0 \wedge e \in \text{estaciones}(c)\}$

Post $\equiv \{c =_{\text{obs}} \text{inspeccion}(e, c_0)\}$

Complejidad: $\Theta(\text{copy}(a))..$

PROXIMORUR(in c : ciudad) $\rightarrow res$: rur

Pre $\equiv \{\text{true}\}$

Post $\equiv \{res =_{\text{obs}} \text{proximoRUR}(c)\}$

Complejidad: $O(1)..$

MAPA(in c : ciudad) $\rightarrow res$: mapa

Pre $\equiv \{\text{true}\}$

Post $\equiv \{res =_{\text{obs}} \text{mapa}(c)\}$

Complejidad: $\Theta(\text{copy}(a))..$

ROBOTS(in c : ciudad) $\rightarrow res$: it(conj(rur))

Pre $\equiv \{\text{true}\}$

Post $\equiv \{res =_{\text{obs}} \text{CrearIt}(\text{robots}(c))\}$

Complejidad: $O(1)..$

ESTACION(in u : rur, in c : ciudad) $\rightarrow res$: estacion

Pre $\equiv \{u \in \text{robots}(c)\}$

Post $\equiv \{res =_{\text{obs}} \text{estacion}(u, c)\}$

Complejidad: $O(1)..$

TAGS(in u : rur, in c : ciudad) $\rightarrow res$: conj(tag)

Pre $\equiv \{u \in \text{robots}(c)\}$

Post $\equiv \{c =_{\text{obs}} \text{inspeccion}(e, c_0)\}$

Complejidad: $O(1)..$

#INFRACCIONES(in u : rur, in c : ciudad) $\rightarrow res$: nat

Pre $\equiv \{u \in \text{robots}(c)\}$

Post $\equiv \{c =_{\text{obs}} \text{inspeccion}(e, c_0)\}$

Complejidad: $\Theta(\text{copy}(a))..$

ESTACIONES(**in** c : ciudad) $\rightarrow res$: it(conj(estaciones))

Pre $\equiv \{\text{true}\}$

Post $\equiv \{res =_{\text{obs}} \text{CrearIt}(\text{estaciones}(c))\}$

Complejidad: $O(1)..$

Representación

Representación de la ciudad

ciudad se representa con str

donde str es tupla($robRUR$: DiccArreglo(rur, datosRobot),
 $robEstacion$: DiccTrie(estacion, datosEstacion),
 $mapa$: DiccTrie(estacion, DiccTrie(estacion, Restriccion))
donde datosRobot es tupla($presente?$: bool,
 est : estacion,
 $infr$: nat,
 $tags$: conjTrie(tags),
 $sendas$: DiccTrie(estacion, DiccTrie(estacion, bool),
 $itEst$: it(colaPrio(rur)))
donde datosEstacion es tupla($robots$: colaPrio(rur), $ultimo$: puntero(nodo))

Rep : lst \rightarrow bool

Rep(l) $\equiv \text{true} \iff (l.\text{primero} = \text{NULL}) = (l.\text{longitud} = 0) \wedge_L (l.\text{longitud} \neq 0 \Rightarrow_L$
 $\text{Nodo}(l, l.\text{longitud}) = l.\text{primero} \wedge$
 $(\forall i: \text{nat})(\text{Nodo}(l, i) \rightarrow \text{siguiente} = \text{Nodo}(l, i + 1) \rightarrow \text{anterior}) \wedge$
 $(\forall i: \text{nat})(1 \leq i < l.\text{longitud} \Rightarrow \text{Nodo}(l, i) \neq l.\text{primero})$

Nodo : lst $l \times \text{nat} \rightarrow$ puntero(nodo)

$\{l.\text{primero} \neq \text{NULL}\}$

Nodo(l, i) $\equiv \text{if } i = 0 \text{ then } l.\text{primero} \text{ else } \text{Nodo}(\text{FinLst}(l), i - 1) \text{ fi}$

FinLst : lst \rightarrow lst

FinLst(l) $\equiv \text{Lst}(l.\text{primero} \rightarrow \text{siguiente}, l.\text{longitud} - \min\{l.\text{longitud}, 1\})$

Lst : puntero(nodo) $\times \text{nat} \rightarrow$ lst

Lst(p, n) $\equiv \langle p, n \rangle$

Abs : lst $l \rightarrow \text{secu}(\alpha)$

$\{\text{Rep}(l)\}$

Abs(l) $\equiv \text{if } l.\text{longitud} = 0 \text{ then } <> \text{ else } l.\text{primero} \rightarrow \text{dato} \bullet \text{Abs}(\text{FinLst}(l)) \text{ fi}$

Representación del iterador

itLista(α) se representa con iter

donde iter es tupla($siguiente$: puntero(nodo), $lista$: puntero(lst))

Rep : iter \rightarrow bool

Rep(it) $\equiv \text{true} \iff \text{Rep}(*(\text{it}.lista)) \wedge_L (\text{it}.siguiente = \text{NULL} \vee_L (\exists i: \text{nat})(\text{Nodo}(*\text{it}.lista, i) = \text{it}.siguiente))$

Abs : iter $it \rightarrow \text{itBi}(\alpha)$

$\{\text{Rep}(it)\}$

Abs(it) $=_{\text{obs}} b: \text{itBi}(\alpha) \mid \text{Siguientes}(b) = \text{Abs}(\text{Sig}(\text{it}.lista, \text{it}.siguiente)) \wedge$
 $\text{Anteriores}(b) = \text{Abs}(\text{Ant}(\text{it}.lista, \text{it}.siguiente))$

Sig : puntero(lst) $l \times \text{puntero(nodo)} p \rightarrow$ lst

$\{\text{Rep}(\langle l, p \rangle)\}$

Sig(i, p) $\equiv \text{Lst}(p, l \rightarrow \text{longitud} - \text{Pos}(*l, p))$

Ant : puntero(lst) $l \times \text{puntero(nodo)} p \rightarrow$ lst

$\{\text{Rep}(\langle l, p \rangle)\}$

Ant(i, p) $\equiv \text{Lst}(\text{if } p = l \rightarrow \text{primero} \text{ then } \text{NULL} \text{ else } l \rightarrow \text{primero} \text{ fi}, \text{Pos}(*l, p))$

Nota: cuando $p = \text{NULL}$, Pos devuelve la longitud de la lista, lo cual está bien, porque significa que el iterador no tiene siguiente.

Pos : lst $l \times \text{puntero}(\text{nodo})\ p \longrightarrow \text{puntero}(\text{nodo})$ {Rep($\langle l, p \rangle$)}
 Pos(l, p) \equiv if $l.\text{primero} = p \vee l.\text{longitud} = 0$ then 0 else 1 + Pos(FinLst(l), p) fi

Algoritmos

2. Modulo Restriccion

Interfaz

se explica con: RESTRICCION

géneros: restriccion

usa:

$\langle \bullet \rangle (\text{in } t : \text{tag}) \rightarrow res : \text{restriccion}$

Pre $\equiv \{\text{true}\}$

Post $\equiv \{res =_{\text{obs}} \langle t \rangle\}$

Complejidad: $\Theta(1)$

Descripción: genera una restriccion de un solo tag.

NOT $\bullet (\text{in } r : \text{restriccion}) \rightarrow res : \text{restriccion}$

Pre $\equiv \{\text{true}\}$

Post $\equiv \{res =_{\text{obs}} \text{NOT } r\}$

Complejidad: $\Theta(1)_{..}$

\bullet AND $\bullet (\text{in } r1 : \text{restriccion}, \text{in } r2 : \text{restriccion}) \rightarrow res : \text{restriccion}$

Pre $\equiv \{\text{true}\}$

Post $\equiv \{res =_{\text{obs}} r1 \text{ AND } r2\}$

Complejidad: $\Theta(1)_{..}$

\bullet OR $\bullet (\text{in } r1 : \text{restriccion}, \text{in } r2 : \text{restriccion}) \rightarrow res : \text{restriccion}$

Pre $\equiv \{\text{true}\}$

Post $\equiv \{res =_{\text{obs}} r1 \text{ OR } r2\}$

Complejidad: $\Theta(1)_{..}$

VERIFICA? $(\text{in } ts : \text{conj}(\text{tag}), \text{in } r : \text{restriccion}) \rightarrow res : \text{bool}$

Pre $\equiv \{\text{true}\}$

Post $\equiv \{res =_{\text{obs}} \text{verifica?}(ts, r)\}$

Complejidad: $O(R)_{..}$

Representación

Representacion de la restriccion

restriccion se representa con rtr

donde rtr es tupla($raiz$: puntero(Nodo))

donde Nodo es tupla(tag : tag ,

$tipo$: log ,

$negado?$: bool ,

izq : puntero(Nodo) ,

der : puntero(Nodo))

donde log es enum{AND, OR, CAR}