Instituto Tecnológico de Costa Rica

Escuela de Computación

Bachillerato en Ingeniería en Computación

Curso: Inteligencia Artificial IC-6200

Profesora:

María Auxiliadora Mora Cross

T03-Documentación Práctica de Racket

Estudiantes:

Jose Pablo Fernández Cubillo

Roberto Vidal Patiño

Grupo 20

I Semestre, 2022

Crear el árbol

Para crear el árbol lo que hacemos es llamar a la función node. Esta función tiene los parámetros id, name y value (en este orden). Cada nodo es representado por una lista con dos listas dentro, la primera tiene la información del nodo (id, nombre y valor) y la segunda contiene a los nodos hijos (al inicio es una lista vacía).

```
Welcome to DrRacket, version 8.3 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> (node 1 "a" "a")
'((1 "a" "a") ())
>
```

En la imagen vemos que se llama a la función node con el id 1, nombre “a” y valor “a”.

Insertar nodo en el árbol

Para insertar un nodo en el árbol se usa la función insert-node con los parámetros tree, parent, id, name, value, weight (en ese orden). Lo que devuelve es un árbol que tenga una nueva arista dentro de la lista de hijos del nodo escogido según el parámetro parent.

```
Welcome to DrRacket, version 8.3 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> (insert-node (node 1 "a" "a") 1 4 "d" "d" 10)
'((1 "a" "a") ((((4 "d" "d") () 10)))
>
```

En la imagen vemos que se llama a la función insert-node con el árbol igual al que usamos en el ejemplo anterior, en este caso escogemos al nodo padre al nodo con id 1, insertamos un nuevo nodo con id 4, nombre “d” y valor “d”, y a la arista se le puso un peso de 10.

Se puede usar la misma función múltiples veces para crear un árbol más grande. Para hacer esto simplemente colocamos el resultado de la función insert-node dentro del parámetro tree (ya que esta función devuelve un árbol).

```
Welcome to DrRacket, version 8.3 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> (insert-node (insert-node (insert-node (node 1 "a" "a") 1 2 "b" "b" 10) 1 3 "c" "c" 20) 3 4 "d" "d" 30)
'((1 "a" "a") ((((2 "b" "b") () 10) (((3 "c" "c") ((((4 "d" "d") () 30))) 20)))
>
```

Listar todos los nodos del árbol

Para listar todos los nodos del árbol lo que hacemos es llamar a la función list-all-nodes. Esta función tiene el parámetro tree. Lo que hace es desplegar en consola el árbol.

```
Welcome to DrRacket, version 8.3 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> (list-all-nodes (insert-node (insert-node (insert-node (node 1 "a" "a") 1 2 "b" "b" 10) 1 3 "c" "c" 20) 3 4 "d" "d" 30))
((1 a a) ((((2 b b) () ()) 10) (((3 c c) ((((4 d d) () ()) 30))) 20)))
>
```

Borrar nodo en el árbol

Para borrar un nodo en el árbol lo que hacemos es llamar a la función delete-node. Esta función tiene los parámetros tree y también id (en este orden). Devuelve el árbol sin el nodo con el id suministrado.

```
Welcome to DrRacket, version 8.3 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> (delete-node (node 1 "a" "a") 1)
'()
>
```

En la imagen vemos que se eliminó el único nodo que existía en el árbol, por tanto, devuelve una lista vacía. También se puede hacer con árboles más grandes.

```
Welcome to DrRacket, version 8.3 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> (delete-node (insert-node (insert-node (insert-node (node 1 "a" "a") 1 2 "b" "b" 10) 1 3 "c" "c" 20) 3 4 "d" "d" 30) 3)
'((1 "a" "a") ((((2 "b" "b") () ()) 10) (((4 "d" "d") () ()) 20)))
>
```

Es importante notar que si se elimina un nodo con hijos el primer hijo lo remplazará.

Encontrar nodo en el árbol

Para encontrar un nodo en el árbol lo que hacemos es llamar a la función find-node. Esta función tiene los parámetros tree y también id (en este orden). Lo que hace es devolver la lista con la información del nodo que tenga el mismo id.

```
Welcome to DrRacket, version 8.3 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> (find-node (insert-node (node 1 "a" "a") 1 2 "b" "b" 30) 2)
'(2 "b" "b")
>
```

En la imagen vemos que solo se devuelve la información del nodo. También se puede usar con árboles más grandes.

```
Welcome to DrRacket, version 8.3 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> (find-node (insert-node (insert-node (insert-node (node 1 "a" "a") 1 2 "b" "b" 10) 1 3 "c" "c" 20) 3 4 "d" "d" 30) 4)
'(4 "d" "d")
>
```

Buscar ancestro en el árbol

Para buscar el ancestro de un nodo en el árbol lo que hacemos es llamar a la función ancestor. Esta función tiene los parámetros tree y también id (en este

orden). Devuelve el nodo del node que tenga como hijo a un nodo con el id suministrado.

```
Welcome to DrRacket, version 8.3 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> (ancestor (insert-node (node 1 "a" "a") 1 2 "b" "b" 10) 2)
'(1 "a" "a")
>
```

En la imagen vemos que se devuelve el nodo con id 1, que es el padre del nodo con id 2. También se puede hacer con árboles más grandes.

```
Welcome to DrRacket, version 8.3 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> (ancestor (insert-node (insert-node (insert-node (node 1 "a" "a") 1 2 "b" "b" 10) 1 3 "c" "c" 20) 3 4 "d" "d" 30) 4)
'(3 "c" "c")
>
```