Reproducció en el laboratori: justificació de les operacions

26 de maig de 2014

A continuació es mostra les justificacions de totes les operacions recursives necessàries per poder aplicar un creixement a un organisme, és a dir, l'opció -1 del programa principal, i també de totes les operacions iteratives que s'utilitzen en aplicar una ronda de reproducció en l'experiment, és a dir, l'opció -3, menys l'actualització del ranking.

1 Creixement d'un organisme

1.1 L'operació `crece`

Aquesta operació acció recursiva, té com a únic paràmetre, passat per referència, un `Arbre<Celula>` que és el que volem aplicar el creixement. En ella s'utilitza funcions de les classes `Celula` i `Arbre`.

1.1.1 Implementació

```
void Organisme::crece(Arbre<Celula> &a)
// Pre: a = A
{
    if (not a.es_buit()) {
        Arbre<Celula> a1, a2;
        Celula c;
        c = a.arrel();
        a.fills(a1,a2);
        if (a1.es_buit() and a2.es_buit()) {
            Celula c1(c);
            Celula c2(c);
            ++maxid;
            c1.modificar_id(maxid);
```

```
            ++maxid;
            c2.modificar_id(maxid);
            tamano += 2;
            Arbre<Celula> aux1 = a1;
            Arbre<Celula> aux2 = a2;
            a1.plantar(c1,aux1,aux2);
            a2.plantar(c2,aux1,aux2);
            a.plantar(c,a1,a2);
        }
        // HI: crece provoca que el paràmetre pateixi un
        // creixement.
        else if (a1.es_buit()) {
            crece(a2);
            a.plantar(c,a1,a2);
        }
        else if (a2.es_buit()) {
            crece(a1);
            a.plantar(c,a1,a2);
        }
        else {
            crece(a1);
            crece(a2);
            a.plantar(c,a1,a2);
        }
    }
// Post: A ha patit un creixement
}
```

1.1.2 Justificació

- *Cas directe*: Si els dos fills del arbre passat com a paràmetre són buits, tant `a1` com `a2`, llavors entra dins el cas directe. Llavors compleix la postcondició, ja que el creixement d'un arbre consisteix en plantar dues cèl·lules en les fulles d'un arbre. La crida a les funcions de la creadora copiadora de la classe `Celula`, `modificar_id` i `plantar` de la classe `Arbre` són correctes ja que compleixen la seva precondició.
- *Cas recursiu:* Si un dels dos fills del arbre no és buit llavors es poden donar fins a tres opcions:
    - El fill esquerra és buit. Llavors s'ha de seguir buscant la fulla del arbre pel fill dret, ja que aquest no és buit. La crida recursiva es correcte ja que compleix la hipòtesi d'inducció (`HI`), i deixa el paràmetre `a2` amb un creixement aplicat. La crida a la funció `plantar` es correcte perquè compleix la seva precondició.
    - El fill dret és buit. Llavors s'ha de seguir buscant la fulla del arbre pel fill esquerra, ja que aquest no és buit. La crida recursiva es correcte ja que compleix la hipòtesi d'inducció (`HI`), i deixa el paràmetre `a1` amb un

creixement aplicat. La crida a la funció `plantar` es correcte perquè compleix la seva precondició.

- El fill esquerra i el fill dret no són buits. Llavors s'ha d'aplicar la recursivitat als dos fills en busca de les fulles de l'arbre. La crida recursiva es correcte ja que compleix la hipòtesi d'inducció (`HI`), i deixa el paràmetre `a1` i `a2` amb un creixement aplicat. La crida a la funció `plantar` es correcte perquè compleix la seva precondició.

- *Finalització:* La funció acabarà quan els dos fills siguin buits. Llavors com les crides sempre són a un dels fills del paràmetre, i els fills sempre tenen una mida més petita estrictament que el seu progenitor, llavors el paràmetre decreix en cada crida recursiva.

2 Ronda de reproducció

2.1 L'operació `reproduccion`

Aquesta operació funció iterativa, que té com a paràmetre implícit un `Experiment` i com a paràmetre explícit passat per referència un `Ranking`, modifica el paràmetre implícit després d'haver realitzat una ronda de reproducció, actualitza el ranking corresponent després d'haver passat per la ronda esmentada, i retorna un enter que indica el nombre d'organismes fills produïts. En ella s'utilitza funcions de les classes `Arbre`, `Ranking` i `Organisme`.

2.1.1 Implementació

```
int Experiment::reproduccion(Ranking &R)
// Pre: cert
// Post: El resultat és el nombre de fills produïts en la ronda
// de reproducció modificant el paràmetre implícit i
// actualitzant el ranking de l'experiment passat per paràmetre
// per referència.

{
    int fills = 0;
    int final = mida;
    int i = 0;
    int j = 1;
    vector<bool> aparicions(final,false);

    // Inv: 1 <= j <= final <= mida <= MAXORGANISMES
    // Cota: (final - j) or (MAXORGANISMES - mida)
    while (j < final and mida != MAXORGANISMES) {

        // Inv1: 1 <= i < final
        // Cota1: final - 1 - i
```

```
        while (i < final - 1 and (aparicions[i] or not
EXP[i].esta_vivo())) ++i;
        // Post1: i indica la posició del pare en el vector de
        // l'experiment.

        j = i + 1;

        // Inv2: i+1 <= j <= final
        // Cota2: final - j
        while (j < final and (aparicions[j] or emparellats[i][j]
or not EXP[j].esta_vivo())) ++j;
        // Post2: j indica la posició de la mare en el vector de
        // l'experiment.

        if (j < final and i < final - 1) {
            Organisme h;
            h.reproduccion(EXP[i],EXP[j]);
            aparicions[i] = true;
            aparicions[j] = true;
            emparellats[i][j] = true;
            emparellats[j][i] = true;
            if (h.esta_vivo()) {
                EXP[mida] = h;
                ++mida;
                R.actualizar_ranking(i,j,mida);
                ++fills;
                ++vius;
            }
        }
        else if (i < final - 1) {
            ++i;
            j = i + 1;
        }
    }
    return fills;
}
```

2.1.2 Justificació

- *Inicialitzacions:* `fills` es el nombre de fills reproduïts en la ronda, i es zero inicialment ja que al principi de una ronda de reproducció no s'ha reproduït encara cap organisme. La paràmetre `final` és constant i indica el nombre d'organismes existents al començar la ronda. I les variables `i` i `j` indiquen les posicions del pare i la mare respectivament en el vector de l'experiment i valen zero i u ja que es el cas inicial on s'avaluarà el primer i segon organisme de l'experiment.
- *Condició de sortida:* Amb la negació de la condició del bucle i el compliment de l'`Inv`, es dona correctament la postcondició.
- *Cos del bucle:* primer es calcula la posició en el vector de l'experiment del pare (`i`) i de la mare (`j`) amb l'ajuda de dos `while`, en els quals en el primer va augmentant la variable `i` un a un fins incomplir la condició (`++i`), i en el segon el

mateix però per la variable `j` (`++j`). I una vegada ja calculats, si aquest `i` i `j` estan dins el rang, s'intenten reproduir els organismes `EXP[i]` i `EXP[j]`. Si no es així i la variable `i` encara està dins el rang, s'augmenta en un i (`++i`) i s'aplica a `j` el següent de `i` (`j = i + 1`). La funció creadora del `Organisme`, `reproduccion` i `esta_vivo` de la classe `Organisme` i `actualizar_ranking` de la classe `Ranking` son correctes ja que compleixen la seva precondició.

- *Finalització:* `(final - j)` o bé `(MAXORGANISMES - mida)`. Ja que com `j` i `mida` van augmentant, al final la funció de fita serà igual a zero.
  I en els dos bucles interiors:
  - `final - 1 - i`: com la `i` va augmentant, la funció de fita va decreixent.
  - `final - j`: com la `j` va augmentant, la funció de fita va decreixent.