EXÀMENS LABORATORI EC

Creat per: Petru Rares Sincraian
Agraïments: Gerard Pradas, Xavier Font, Anselmo López
Data de creació: 20-12-2014
Darrera Modificació: 21-12-2014

Índex

Examen Laboratori Desembre 2013

Problema 1 Estructures de dades

Donat el programa en C:

```
int matrix[4][4] = { 2, 0, 4, 8, 1, 9, 3, 6, 7, 5, 2, 4, 0, 1, 5, 3 };
int max[4];

int main() {
        int i, j;

        for (i = 0; i < 4; i++) {
                max[i] = matrix[i][0];

                for (j = 1; j < 4; j++) {
                        if ( max[i] < matrix[i][j] )
                                max[i] = matrix[i][j];
                }
        }
}
```

Tradueix-lo a llenguatge assemblador del MIPS.

Comprova que al final de l'execució el contingut del vector `max` és: { 8, 9, 7, 5 }

Problema 2 Subrutines

Donat el següent codi en alt nivell:

```
int V[8] = {12, -3, 24, 41, -7, 5, 9, 1 };
int M[8][8];
int resultat;

void nofares(){};

main() {
        resultat = func(V, M, 8)
}

int func(int A[], intB[][8], int n){
        int i;
        int *p = &B[0][7];

        for ( i = 0; i < n; ++i) {
                if (A[i] > 0)
                        posar(p, &A[i]);
                p = p + 8;
        }

        return *(p-8);
}

void posar(int *p, int *x) {
        *p = *x;
        nofares();
        return;
}
```

Tradueix la funció `func` i la funció `posar` al llenguatge assemblador del MIPS en el fitxer examen2.s on ja trobaràs programada la funció `main` i `nofares`.

Comprova que al final de l'execució, el contingut de la columna 7 de la matriu M és: 12, 0, 24, 41, 0, 5, 9, 1 i que la variable resultat val 1.

Examen Laboratori Juny 2014 Torn 1

Problema 1 Estructura de dades

Donat el següent codi C:

```
int A[4][4] = { 0, 8, 0, 0, 0, 0, -2, 0, 0, 0, 4, 0, 5, 0, 0, 0 };
int Y[4] = { 1, 2, 2, 0 };
int sum = 0;

main() {
        int i, val;

        for (i = 0; i < 4; i++) {
                val = A[i][ Y[i] ];
                if (val > 0) {
                        sum += val;
                }
        }
}
```

Tradueix el programa principal a assemblador MIPS en el fitxer exam1.s

Comprova que al final de l'execució la variable sum = 17.

Examen Laboratori Juny 2014 Torn 2

Problema 1 Estructures de dades

Donat el següent codi C:

```
int A[4][4] = {3, -1, -1, -2, 27, -9, 0, 0, 10, 20, 12, 2, 0, 2, -3, -15}

void main() {
        int i, j;

        for (i = 0; i < 4; i++) {
                for (j = 0; j <= i; j++) {
                        if (A[i][j] < 0)
                                A[i][j] = -A[i][j]/3;
                        else
                                A[i][j] = A[i][j]/3;
                }
        }
}
```

Tradueix el programa principal a assemblador MIPS en el fitxer prob1.s

Comprova que al final de l'execució la matriu A conté els següent valors:

A[0] = { 1, -1, -1, -2 }
A[1] = { 9, 3, 0, 0 }

A[2] = { 3, 6, 4, 2 }
A[3] = { 0, 0, 1, 5 }

Problema 2 Subrutines

Donat el següent codi escrit en alt nivell:

```
int V1[8] = {2, -3, 6, 2, -8, 5, 6, 2};
int V2[8] = {0, 0, 0, 0, 0, 0, 0, 0};
int V3[8] = {0, 0, 0, 0, 0, 0, 0, 0};
int res1, res2;

main() {
        res1 = buscar(V1, V2, 2);
        res2 = buscar(V1, V3, 6);
}

int buscar(int *A, int *B, int c) {
        int i, cont = 0;

        for (i = 0; i < 8; ++i) {
                if ((*A) == c)
                        comptar(B, &cont);
                ++A;
                ++B;
        }

        return cont;
}

void comptar(int *x, int *num){
        (*num) = (*num) + 1;
        *x = -1;
}
```

Tradueix al fitxer exam2.s la subrutina `buscar`. Fixa't que ja està programat el codi del programa principal i de la subrutina `comptar`.

Comprova que al final de l'execució del programa:

res1 = 3
res2 = 2
v2 = {-1, 0, 0, -1, 0, 0, 0, -1}
v3 = {0, 0, -1, 0, 0, 0, -1, 0}

Problema 3 Memòria cache

Considera un sistema computador format per un processador MIPS, un a memòria principal i una memòria cache de dades amb la següent configuració:

- Correspondència directa
- Capacitat total: 16 blocs
- Mida del bloc: 32 bytes (8 words)
- Escriptura immediata amb assignació (és la que implementa el MARS)

Partim del següent programa en alt nivell (podeu trobar el programa equivalent en MIPS al fitxer pro3.s):

```
int V[16];
int M[8][16];

main() {
        int elem, i;

        for (i = 0; i < 8; ++i) {
                for (j = 0; j < 16; ++j) {
                        elem = M[i][j];
                        V[j] = V[j] + elem;
                }
        }
}
```

Considera que les variables globals que conté el programa estan emmagatzemades a partir de l'adreça 0x10010000. Considera també que la MC és inicialment buida.

Es demana)respon al fiter prob3_respostes.txt):

a) Quantes fallades es produeixen en cada iteració del bucle?
b) Si has contestat bé l'apartat anterior observa que en la iteració i=7 es produeixen moltes més fallades que en les altres iteracions. Descriu amb precisió quina és la causa que en aquesta iteració hi hagi moltes més fallades.
c) ¿Quin és el mínim nombre de fallades que podem aconseguir en aquest programa canviant el grau d'associativitat de la cache (però sense alterar la capacitat total ni la mida del bloc)? ¿Quin és el mínim grau d'associativitat amb el qual s'aconsegueix aquest resultat? Justifica perquè l'associativitat redueix el nombre de fallades en aquest programa.
d) ¿Quin és el mínim nombre de fallades que podem aconseguir en aquest programa canviant la mida del bloc i el grau d'associativitat (però sense alterar la capacitat total)? ¿Quins són el grau d'associativitat i la mida de la cache amb els quals s'aconsegueix aquest resultat? Justifica per què el canvi de la mida del bloc redueix el nombre de fallades en aquest programa.