**Pseudo-códigos para funciones de comunicación de tablas SQL**

La función **distribuye\_tabla**( *Tab\_in*, *K* ) se ejecuta en paralelo en cada procesador y toma como entrada una tabla *Tab\_in* y la identificación de un campo llave *K* de la misma tabla. Sobre el campo *K* se aplica una función de hashing para determinar a cual procesador se envía cada tupla de la tabla *Tab\_in*. Las tuplas enviadas en la forma de mensajes a los distintos procesadores son posteriormente recolectadas en cada procesador y son almacenadas en una tabla de salida *Tab\_out* creada en cada procesador. Es decir, tanto la tabla *Tab\_in* como la tabla *Tab\_out* contienen sus tuplas distribuidas en cada procesador.

tabla **distribuye\_tabla**( tabla Tab\_in, llave K ) {

// Una instancia se ejecuta en paralelo en cada procesador. // Cada instancia actúa sobre las tuplas almacenadas en su // respectivo procesador.

// envía las tuplas del procesador actual al procesador destino foreach tupla t in Tab\_in do {

bsp\_send( hash( t.K ), message( t ) );

}

bsp\_sync(); // sincronización de procesadores (recepción mensajes)

// recibe las tuplas y las almacena en la tabla de salida Tab\_out= crea\_tabla();

while( bsp\_next\_message() != void() ) {

tupla= bsp\_get\_message(); // recibe mensaje conteniendo tupla Tab\_out.insert( tupla );

}

return Tab\_out;

}

Ejemplo de funcionamiento. Suponemos 2 procesadores.

EMPLEADOS (procesador 0)

| RUT | NOMBRE | DIR |
| --- | --- | --- |
| 1111-1 | N1 | D1 |
| 4444-4 | N4 | D4 |

EMPLEADOS (procesador 1)

| RUT | NOMBRE | DIR |
| --- | --- | --- |
| 2222-2 | N2 | D2 |
| 3333-3 | N3 | D3 |

Resultados que entrega la función *hash*(rut)

Hash(1111-1) --> procesador 1

Hash(2222-2) --> procesador 0

Hash(3333-3) --> procesador 1

Hash(4444-4) --> procesador 0

Si cada procesador ejecuta en paralelo

DEMP = **distribuye\_tabla**( EMPLEADOS, RUT );

Entonces el resultado es

DEMP (procesador 0)

| RUT | NOMBRE | DIR |
| --- | --- | --- |
| 2222-2 | N2 | D2 |
| 4444-4 | N4 | D4 |

DEMP (procesador 1)

| RUT | NOMBRE | DIR |
| --- | --- | --- |
| 1111-1 | N1 | D1 |
| 3333-3 | N3 | D3 |

La función **envia\_a\_todos**(*T\_in*) se ejecuta en paralelo en cada procesador. Cada instancia de la función toma como entrada una tabla *Tab\_in* y la envía a todos los otros procesadores. En cada procesador, cada instancia de la función entrega como resultado una tabla *Tab\_out* que es la unión de todas las tablas recibidas desde los otros procesadores (incluido a sí mismo). El supuesto es que la *Tab\_in* es una tabla que contiene sus tuplas distribuidas en los procesadores. Es decir, en cada procesador se tiene el mismo esquema de tablas y las tuplas de cada tabla están distribuidas en los procesadores.

tabla **envia\_a\_todos**( tabla T\_in ) {

// Una instancia se ejecuta en paralelo en cada procesador. // Cada instancia actúa sobre la tabla almacenada en su

// respectivo procesador.

// Envía la tabla del procesador actual a todos los procesadores. // Es un patrón de comunicación all-to-all.

for( int i= 0; i<P; i++ )

bsp\_send( i, new message( T\_in ) );

bsp\_sync(); // sincronización de procesadores (recepción mensajes)

Tab\_out= crea\_tabla();

while( bsp\_next\_message() != void() ) {

tabla= bsp\_get\_message(); // recibe cada tabla

Tab\_out.union( tabla ); // forma la union de tablas

}

return Tab\_out;

}

Ejemplo de funcionamiento. Suponemos 2 procesadores.

EMPLEADOS (procesador 0)

| RUT | NOMBRE | DIR |
| --- | --- | --- |
| 1111-1 | N1 | D1 |
| 4444-4 | N4 | D4 |

EMPLEADOS (procesador 1)

| RUT | NOMBRE | DIR |
| --- | --- | --- |
| 2222-2 | N2 | D2 |
| 3333-3 | N3 | D3 |

Si cada procesador ejecuta en paralelo

DEMP = **envia\_a\_todos**( EMPLEADOS ); Entonces el resultado es

DEMP (procesador 0)

| RUT | NOMBRE | DIR |
| --- | --- | --- |
| 1111-1 | N1 | D1 |
| 2222-2 | N2 | D2 |
| 3333-3 | N3 | D3 |
| 4444-4 | N4 | D4 |

DEMP (procesador 1)

| RUT | NOMBRE | DIR |
| --- | --- | --- |
| 1111-1 | N1 | D1 |
| 2222-2 | N2 | D2 |
| 3333-3 | N3 | D3 |
| 4444-4 | N4 | D4 |

La función **recibe\_resultados**(P) es ejecutada por un procesador maestro o broker para esperar por los resultados provenientes desde un conjunto de *P* procesadores esclavos que contienen la base de datos.

tabla **recibe\_resultados**( P ) {

Tab\_out= crea\_tabla();

for( int i= 0; i<P; i++ ) {

tabla= bsp\_get\_async\_message(); // espera por siguiente mensaje Tab\_out.union( tabla );

}

return Tab\_out;

}

El patrón de ejecución del procesador maestro para solicitar la ejecución de la solución de una consulta a *P* procesadores que mantienen la base de datos distribuida entre ellos (mismos esquemas de tablas en cada procesador con tuplas distribuidas uniformemente en los procesadores, es el siguiente.

*Procesador maestro* ::

for( int i= 0; i<P; i++ )

bsp\_send( i, new message( "ID consulta" ) );

resultado = recibe\_resultados(); // recibe tabla total de resultados *fin procesador maestro*