

Concurrent Vector Calculator

Ένα πρόγραμμα υπολογισμού διανυσμάτων με χρήση RPC.

Διονύσης Νικολόπουλος, ice18390126@uniwa.gr
Τμήμα Μηχανικών Πληροφορικής και Υπολογιστών
Πανεπιστήμιο Δυτικής Αττικής, Ελλάδα, Αθήνα
Απρίλιος - Μάιος 2022

Keywords: Γλώσσα προγραμματισμού C, Remote Procedure Calls, Κατανεμημένα Συστήματα

Θέμα και Λεπτομέρειες Υλοποίησης


Ζητούμενο της εργασίας ήταν η κατασκευή ενός προγράμματος τέλεσης 3 πιθανών διαδικασιών (Εύρεση Μέσου όρου, Εύρεση μικρότερου και μεγαλύτερου στοιχείου, Γινόμενο διανύσματος με έναν αριθμό).

Όλα τα ερωτήματα της εκφώνησης, όπως αυτή φαίνεται στον φάκελο `exc/` του repository της εργασίας, έχουν πραγματοποιηθεί.

Το πρόγραμμα δημιουργήθηκε και δοκιμάστηκε σε περιβάλλον Arch Linux με kernel 5.17.5-arch1-1, με gcc version 11.2.0, rpcgen (rpcsvc-proto) 1.4.3

Ο κώδικας που γράφθηκε υπόκειται στην άδεια GNU GPLv3.

Η Εργασία αυτή πραγματοποιήθηκε στο πλαίσιο του μαθήματος των Κατανεμημένων Συστημάτων του προγράμματος σπουδών Πανεπιστημίου Δυτικής Αττικής, τμήματος Μηχανικών και Πληροφορικής Υπολογιστών.

Κάθε αρχείο αυτής της εργασίας, συμπεριλαμβανομένου και του παρόντος εγχειριδίου βρίσκονται αποθηκευμένα σε ένα αποθετήριο στο  GitLab, στον υπερσύνδεσμο αυτό.

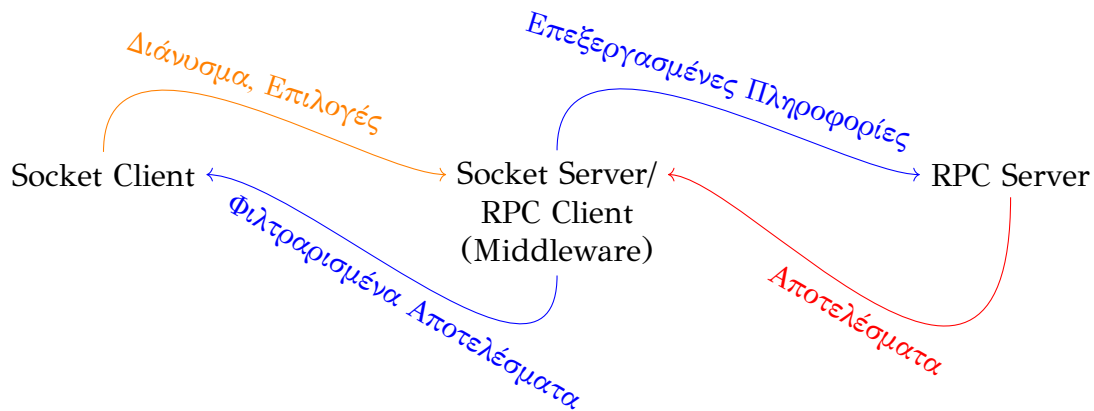
Εκεί φαίνεται αναλυτικά το πως και για πόσο έγινε η δουλειά πάνω στο πρόγραμμα.

1. Ανάλυση Μεθόδου

Το πρόγραμμα λειτουργεί με 3 βασικές οντώτητες:

- Τον **Socket Client**. Χρησιμοποιείται από τον τελικό χρήστη. Εδώ ο χρήστης εισάγει τα δεδομένα που είναι απαραίτητα, δηλαδή της πληροφορίες διανύσματος (μέγεθος και στοιχεία) και ένα αριθμό επιλογών για την επεξεργασία του διανύσματος (οι συγκεκριμένοι αριθμοί εισάγονται επαναληπτικά).

- Τον **Socket Server/ RPC Client ή Middleware**. Ουσιαστικά μεσολαβεί στην επικοινωνία των Socket Client και RPC Server, κάνοντας:
 - Τις απαραίτητες δευσμεύσεις μνήμης.
 - Τις επεξεργασίες μετάφρασης των δεδομένων του χρήστη προς τον RPC Server.
 - Την μεταβίβαση των πληροφοριών/αποτελεσμάτων από τον Socket Server προς τον Socket Client και το αντίθετο.
- Τον **RPC Server**. Εκτελεί τις μαθηματικές πράξεις πάνω στο διάνυσμα με χρήση των Remote Procedure Calls σε συνεργασία με τον RPC Client και στέλνει τα αποτελέσματα στον προαναφερόμενο.



Σχήμα 1: Η αρχιτεκτονική του προγράμματος

Παραπάνω αρχεία που βρίσκονται στους φακέλους με τον πηγαίο κώδικα είναι είτε βοηθητικές συναρτήσεις είτε μέρος αυτών των 3 οντωτήτων αλλά χωρισμένες σε διαφορετικά αρχεία για ευκολία χειρισμού και κατανόησης.

Έχει γίνει προσπάθεια να γίνει πρόληψη για τα σφάλματα του χρήστη, για παράδειγμα η εσφαλμένη είσοδος δεδομένων με διαφορετικό τύπο από τον ζητούμενο κ.α.

2. Ενδεικτικά Τρεξίματα

Για το compilation του προγράμματος αρκεί η εντολή "make" τον αρχικό κατάλογο του προγράμματος:



```
On 12 main ~/uni/classes/Eksamino_6/Katanemimena_Systhmata/2022/concurrent-vector-calculator <2>
$ make
make -C src/rpc make_rpc_files
make[1]: Entering directory '/home/nns/uni/classes/Eksamino_6/Katanemimena_Systhmata/2022/concurrent-vector-calculator/src/rpc'
rpcgen ./rpcvec.x
make[1]: Leaving directory '/home/nns/uni/classes/Eksamino_6/Katanemimena_Systhmata/2022/concurrent-vector-calculator/src/rpc'
make -C src/ make_c_files
make[1]: Entering directory '/home/nns/uni/classes/Eksamino_6/Katanemimena_Systhmata/2022/concurrent-vector-calculator/src'
cc ./server/rpcvec_server.c ./rpc/rpcvec_svc.c ./rpc/rpcvec_xdr.c -o rpcvec_server -I /usr/include/tirpc/ -I ../include/ -ltirpc -lnsl
cc ./middleware/middleware.c ./middleware/checkalloc.c ./rpc/rpcvec_clnt.c ./rpc/rpcvec_xdr.c -o rpcvec_middleware -I /usr/include/tirpc/ -I ../include/ -ltirpc -lnsl
cc ./client/tui.c ./client/prompts.c ./client/sanitary.c -o rpcvec_user_client -I /usr/include/tirpc/ -I ../include/ -ltirpc -lnsl
mv rpcvec_user_client rpcvec_middleware rpcvec_server ../bin
make[1]: Leaving directory '/home/nns/uni/classes/Eksamino_6/Katanemimena_Systhmata/2022/concurrent-vector-calculator/src'
On 12 main ~/uni/classes/Eksamino_6/Katanemimena_Systhmata/2022/concurrent-vector-calculator
$
```

Για το τρέξιμο εκτελούμε τις εξής εντολές σε σειρά στον αρχικό κατάλογο της εργασίας (Σε τρία διαφορετικά τερματικά, με το port τις επιλογής μας)

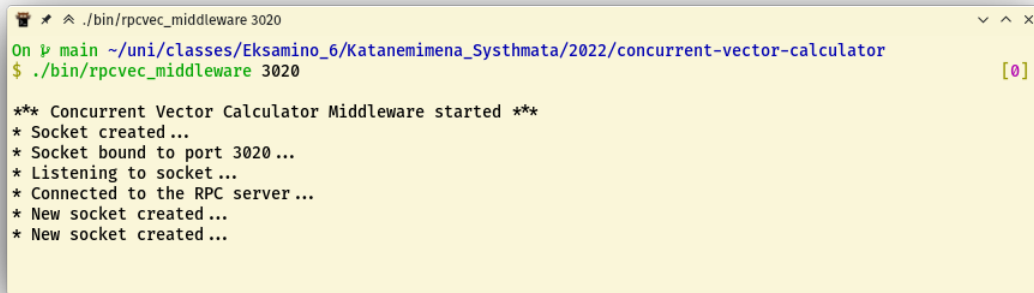
```
On p main ~/uni/classes/Eksamino_6/Katanemimena_Systhmata/2022/concurrent-vector-calculator
$ ./bin/rpcvec_server [0]
```

```
On p main ~/uni/classes/Eksamino_6/Katanemimena_Systhmata/2022/concurrent-vector-calculator
$ ./bin/rpcvec_middleware 3020 [0]

*** Concurrent Vector Calculator Middleware started ***
* Socket created...
* Socket bound to port 3020 ...
* Listening to socket ...
```

```
On p main ~/uni/classes/Eksamino_6/Katanemimena_Systhmata/2022/concurrent-vector-calculator
$ ./bin/rpcvec_user_client localhost 3020 [X-130]
*** Concurrent Vector Calculator ***
* Socket opened ...
* Connecting to server ...
* Connected to server!
Please provide number of elements for vector: 
```

Με την εκτέλεση της τελευταίας εντολής βλέπουμε στο δεύτερο παράθυρό μας ότι πλέον συνδέθηκε το Middleware με τον Socket Client:



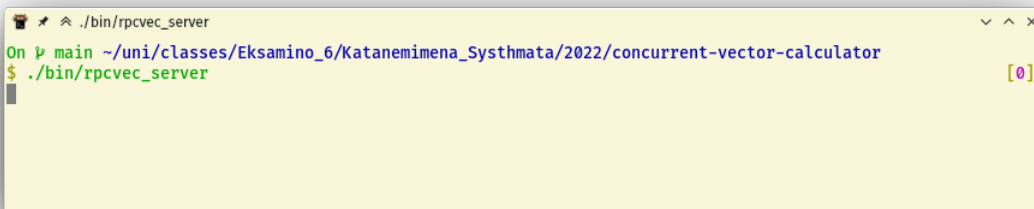
```
On p main ~/uni/classes/Eksamino_6/Katanemimena_Systhmata/2022/concurrent-vector-calculator
$ ./bin/rpcvec_middleware 3020 [0]

*** Concurrent Vector Calculator Middleware started ***
* Socket created...
* Socket bound to port 3020 ...
* Listening to socket...
* Connected to the RPC server...
* New socket created...
* New socket created...
```

Έπειτα, χρησιμοποιούμε το πρόγραμμα κανονικά με τις ενδείξεις στο τερματικό, τα τρία παράθυρα κατά την διάρκεια του ενδεικτικού τρεξίματος αυτού είναι σε κάθε βήμα τις διαδικασίες:

2.1 Βήμα Πρώτο

Εισάγουμε τον αριθμό των στοιχείων και τα στοιχεία:



```
On p main ~/uni/classes/Eksamino_6/Katanemimena_Systhmata/2022/concurrent-vector-calculator
$ ./bin/rpcvec_server [0]
```

```
On P main ~/uni/classes/Eksamino_6/Katanemimena_Systhmata/2022/concurrent-vector-calculator
$ ./bin/rpcvec_middleware 3020 [0]

*** Concurrent Vector Calculator Middleware started ***
* Socket created...
* Socket bound to port 3020 ...
* Listening to socket...
* Connected to the RPC server...
* New socket created...
* New socket created...
* Waiting for vector data...
=> Received size of vector from client side, size is 6
* Initialized vector with 6 elements!
=> Received value of vector from client side, value is 1.000000
=> Received value of vector from client side, value is 2.000000
=> Received value of vector from client side, value is 3.000000
=> Received value of vector from client side, value is 4.000000
=> Received value of vector from client side, value is 5.000000
```

```
./bin/rpcvec_user_client localhost 3020

* Connecting to server...
* Connected to server!
Please provide number of elements for vector: 6

Please provide element number 0 of vector: 1
Please provide element number 1 of vector: 2
Please provide element number 2 of vector: 3
Please provide element number 3 of vector: 4
Please provide element number 4 of vector: 5
Please provide element number 5 of vector: 6

Please choose calculation to make or exit:
0. Exit program
1. Average of vector
2. Minimum and Maximum element of vector
3. Product of vector with a real number
Choice:
```

2.2 Βήμα Δεύτερο

Εισάγουμε 0-3 για να ακολουθήσουμε την αντίστοιχη διαδικασία. Στο ενδεικτικό τρέξιμο αυτό θα τις επιλέξουμε με σειρά 1,2,3,0.

Επιλέγουμε την 1:

```
./bin/rpcvec_server
$ ./bin/rpcvec_server
Checking vec_val[1] = 2.000000
⇒ Call for function average_1_svc with rq_proc 1
* Iteration: 0:
⇒ input_vec_val[0] = 1.000000

⇒ Sum = 1.000000
* Iteration: 1:
⇒ input_vec_val[1] = 2.000000

⇒ Sum = 3.000000
* Iteration: 2:
⇒ input_vec_val[2] = 3.000000

⇒ Sum = 6.000000
* Iteration: 3:
⇒ input_vec_val[3] = 4.000000

⇒ Sum = 10.000000
* Iteration: 4:
⇒ input_vec_val[4] = 5.000000

⇒ Sum = 15.000000
* Iteration: 5:
⇒ input_vec_val[5] = 6.000000

⇒ Sum = 21.000000
⇒ Average is 3.500000
```

```
./bin/rpcvec_middleware 3020
On p main ~/uni/classes/Eksamino_6/Katanemimena_Sythmata/2022/concurrent-vector-calculator
$ ./bin/rpcvec_middleware 3020

*** Concurrent Vector Calculator Middleware started ***
* Socket created...
* Socket bound to port 3020...
* Listening to socket...
* Connected to the RPC server...
* New socket created...
* New socket created...
* Waiting for vector data...
⇒ Received size of vector from client side, size is 6
* Initialized vector with 6 elements!
⇒ Received value of vector from client side, value is 1.000000
⇒ Received value of vector from client side, value is 2.000000
⇒ Received value of vector from client side, value is 3.000000
⇒ Received value of vector from client side, value is 4.000000
⇒ Received value of vector from client side, value is 5.000000
⇒ Received value of vector from client side, value is 6.000000
⇒ Received rcv_choice number from client side, rcv_choice is 1
```

```
./bin/rpcvec_user_client localhost 3020

Please provide element number 3 of vector: 4
Please provide element number 4 of vector: 5
Please provide element number 5 of vector: 6

Please choose calculation to make or exit:
0. Exit program
1. Average of vector
2. Minimum and Maximum element of vector
3. Product of vector with a real number
Choice: 1

You chose to calculate the average of the vector.
← Sending info to server...
⇒ Average of vector is 3.500000
Please choose calculation to make or exit:
0. Exit program
1. Average of vector
2. Minimum and Maximum element of vector
3. Product of vector with a real number
Choice: █
```

Η μέση τιμή του διανύσματος βρέθηκε και εμφανίστηκε σωστά! Στην συνέχεια, επιλέγουμε την 2:

```
./bin/rpcvec_server

⇒ Sum = 3.000000
* Iteration: 2:
⇒ input_vec_val[2] = 3.000000

⇒ Sum = 6.000000
* Iteration: 3:
⇒ input_vec_val[3] = 4.000000

⇒ Sum = 10.000000
* Iteration: 4:
⇒ input_vec_val[4] = 5.000000

⇒ Sum = 15.000000
* Iteration: 5:
⇒ input_vec_val[5] = 6.000000

⇒ Sum = 21.000000
⇒ Average is 3.500000
← Return results...
⇒ Call for function minmax_1_svc with rq_proc 2
Current is 2.000000
Checking vec_val[1] = 2.000000
Current is 3.000000
Checking vec_val[2] = 3.000000
Current is 4.000000
Checking vec_val[3] = 4.000000
Current is 5.000000
Checking vec_val[4] = 5.000000
Current is 6.000000
Checking vec_val[5] = 6.000000
Max is 6.000000
Min is 1.000000
```



```
./bin/rpcvec_middleware 3020

*** Concurrent Vector Calculator Middleware started ***
* Socket created...
* Socket bound to port 3020...
* Listening to socket...
* Connected to the RPC server...
* New socket created...
* New socket created...
* Waiting for vector data...
⇒ Received size of vector from client side, size is 6
* Initialized vector with 6 elements!
⇒ Received value of vector from client side, value is 1.000000
⇒ Received value of vector from client side, value is 2.000000
⇒ Received value of vector from client side, value is 3.000000
⇒ Received value of vector from client side, value is 4.000000
⇒ Received value of vector from client side, value is 5.000000
⇒ Received value of vector from client side, value is 6.000000
⇒ Received rcv_choice number from client side, rcv_choice is 1
⇒ Average of vector is: 3.500000
⇒ Received rcv_choice number from client side, rcv_choice is 2
⇒ The minimum of the vector is: 1.000000
```

```
./bin/rpcvec_user_client localhost 3020

Choice: 1

You chose to calculate the average of the vector.
⇐ Sending info to server...
⇒ Average of vector is 3.500000
Please choose calculation to make or exit:
0. Exit program
1. Average of vector
2. Minimum and Maximum element of vector
3. Product of vector with a real number
Choice: 2

You chose to calculate the minimum and maximum of the vector.
⇐ Sending info to server...
⇒ Minimum of vector is 1.000000
⇒ Maximum of vector is 6.000000
Please choose calculation to make or exit:
0. Exit program
1. Average of vector
2. Minimum and Maximum element of vector
3. Product of vector with a real number
Choice:
```

Η μέγιστη και ελάχιστη τιμή του διανύσματος βρέθηκε και εμφανίστηκε σωστά!
Στην συνέχεια, επιλέγουμε την 3 και δίνουμε έναν αριθμό:

```

./bin/rpcvec_server
* Iteration: 4:
⇒ input_vec_val[4] = 5.000000

⇒ Sum = 15.000000
* Iteration: 5:
⇒ input_vec_val[5] = 6.000000

⇒ Sum = 21.000000
⇒ Average is 3.500000
⇐ Return results...
⇒ Call for function minmax_1_svc with rq_proc 2
Current is 2.000000
Checking vec_val[1] = 2.000000
Current is 3.000000
Checking vec_val[2] = 3.000000
Current is 4.000000
Checking vec_val[3] = 4.000000
Current is 5.000000
Checking vec_val[4] = 5.000000
Current is 6.000000
Checking vec_val[5] = 6.000000
Max is 6.000000
Min is 1.000000
⇐ Return results...
⇒ Call for function product_1_svc with rq_proc 3
Calculated input_args→product→vec_val[0]=5.000000
Calculated input_args→product→vec_val[1]=10.000000
Calculated input_args→product→vec_val[2]=15.000000
Calculated input_args→product→vec_val[3]=20.000000
Calculated input_args→product→vec_val[4]=25.000000
Calculated input_args→product→vec_val[5]=30.000000
Calculated input_args→product→vec_val[6]=0.000000
Product vector calculated!

```

```

./bin/rpcvec_middleware 3020
* Initialized vector with 6 elements!
⇒ Received value of vector from client side, value is 1.000000
⇒ Received value of vector from client side, value is 2.000000
⇒ Received value of vector from client side, value is 3.000000
⇒ Received value of vector from client side, value is 4.000000
⇒ Received value of vector from client side, value is 5.000000
⇒ Received value of vector from client side, value is 6.000000
⇒ Received rcv_choice number from client side, rcv_choice is 1
⇒ Average of vector is: 3.500000
⇒ Received rcv_choice number from client side, rcv_choice is 2
⇒ The minimum of the vector is: 1.000000
⇒ The maximum of the vector is: 6.000000
⇒ Received rcv_choice number from client side, rcv_choice is 3
⇒ Received number from client side, number is 5.000000
⇐ Sending info to server...
⇒ The product vector is:
product[0] = 5.000000
product[1] = 10.000000
product[2] = 15.000000
product[3] = 20.000000
product[4] = 25.000000

```

```
./bin/rpcvec_user_client localhost 3020

0. Exit program
1. Average of vector
2. Minimum and Maximum element of vector
3. Product of vector with a real number
Choice: 3

You chose to calculate the product of the vector with a number
Please input the number you wish to multiply the vector with: 5

⇒ The product vector is:
product[0] = 5.000000
product[1] = 10.000000
product[2] = 15.000000
product[3] = 20.000000
product[4] = 25.000000
product[5] = 30.000000
Please choose calculation to make or exit:
0. Exit program
1. Average of vector
2. Minimum and Maximum element of vector
3. Product of vector with a real number
Choice: █
```

Το γινόμενο του διανύσματος με έναν αριθμό βρέθηκε και εμφανίστηκε σωστά!
Στην συνέχεια, επιλέγουμε την 0:

```
.../Katanemimena_Systhmata/2022/concurrent-vector-calculator <2>

3. Product of vector with a real number
Choice: 3

You chose to calculate the product of the vector with a number
Please input the number you wish to multiply the vector with: 5

⇒ The product vector is:
product[0] = 5.000000
product[1] = 10.000000
product[2] = 15.000000
product[3] = 20.000000
product[4] = 25.000000
product[5] = 30.000000
Please choose calculation to make or exit:
0. Exit program
1. Average of vector
2. Minimum and Maximum element of vector
3. Product of vector with a real number
Choice: 0
→ Exiting... █

On █ main ~/uni/classes/Eksamino_6/Katanemimena_Systhmata/2022/concurrent-vector-calculator
$ [0]
```

Το πρόγραμμά μας τερμάτισε.

Όπως αναφέρθηκε και προηγουμένως, έχουν γίνει ορισμένες προσπάθειες ώστε να προληφθούν κάποια λάθη του χρήστη αλλά και να ανταποκρίνεται στα σφάλματα ο κώδικας όσο το δυνατό καλύτερα. Μία από τις πολλές περιπτώσεις:

