A Presentation of our Work

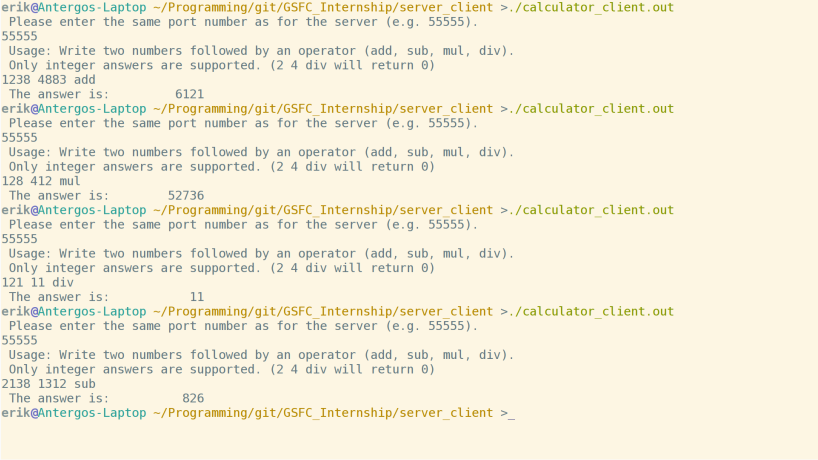
The Swedish Interns

2016-08-18

Created for NVI Inc. at the Goddard Space Flight Centre

Weeks 1-2: Learning Fortran

Calculator



```
erik@Antergos-Laptop ~/Programming/git/GSFC_Internship/server_client >./calculator_client.out
 Please enter the same port number as for the server (e.g. 55555).
55555
 Usage: Write two numbers followed by an operator (add, sub, mul, div).
 Only integer answers are supported. (2 4 div will return 0)
1238 4883 add
 The answer is:         6121
erik@Antergos-Laptop ~/Programming/git/GSFC_Internship/server_client >./calculator_client.out
 Please enter the same port number as for the server (e.g. 55555).
55555
 Usage: Write two numbers followed by an operator (add, sub, mul, div).
 Only integer answers are supported. (2 4 div will return 0)
128 412 mul
 The answer is:        52736
erik@Antergos-Laptop ~/Programming/git/GSFC_Internship/server_client >./calculator_client.out
 Please enter the same port number as for the server (e.g. 55555).
55555
 Usage: Write two numbers followed by an operator (add, sub, mul, div).
 Only integer answers are supported. (2 4 div will return 0)
121 11 div
 The answer is:           11
erik@Antergos-Laptop ~/Programming/git/GSFC_Internship/server_client >./calculator_client.out
 Please enter the same port number as for the server (e.g. 55555).
55555
 Usage: Write two numbers followed by an operator (add, sub, mul, div).
 Only integer answers are supported. (2 4 div will return 0)
2138 1312 sub
 The answer is:          826
erik@Antergos-Laptop ~/Programming/git/GSFC_Internship/server_client >_
```

Figure 1: Example usage of the TCP calculator.

Weeks 3-7: Our First Project

Rewrite how globl/solve handles its passing of data to and from usrpartials and usrprogs.

By minimizing disc I/O we want to increase the speed at which data is sent.

Week 3-4: Testing I/O performance

I/O Performance

- A couple of contenders:

I/O Performance

- A couple of contenders:
    - Read/Write with files

I/O Performance

- A couple of contenders:
  - Read/Write with files
  - Read/Write with pipes

I/O Performance

- A couple of contenders:
  - Read/Write with files
  - Read/Write with pipes
  - Sending/Receiving with TCP Sockets

I/O Performance

- A couple of contenders:
    - Read/Write with files
    - Read/Write with pipes
    - Sending/Receiving with TCP Sockets
    - Sending/Receiving with OpenMPI

I/O Performance

- A couple of contenders:
    - Read/Write with files
    - Read/Write with pipes
    - Sending/Receiving with TCP Sockets
    - Sending/Receiving with OpenMPI
    - Sending/Receiving with ZeroMQ (MQ)

Performance Test

1. The producer generates a list of length n and fills it with integers.

Performance Test

1. The producer generates a list of length n and fills it with integers.
2. The producer writes the list to file (or sends it over the designated transfer protocol).

Performance Test

1. The producer generates a list of length n and fills it with integers.
2. The producer writes the list to file (or sends it over the designated transfer protocol).
3. The consumer reads (or receives) the list.

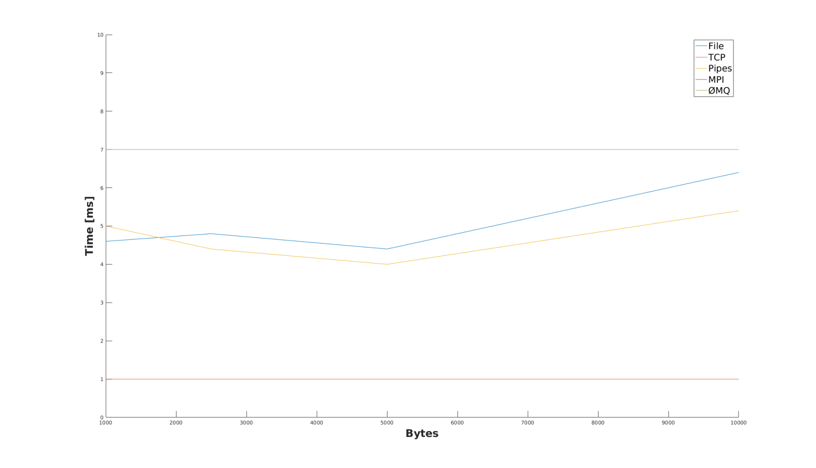
Performance Test

1. The producer generates a list of length n and fills it with integers.
2. The producer writes the list to file (or sends it over the designated transfer protocol).
3. The consumer reads (or receives) the list.
4. The consumer squares each int in the list and sends it back to the producer.

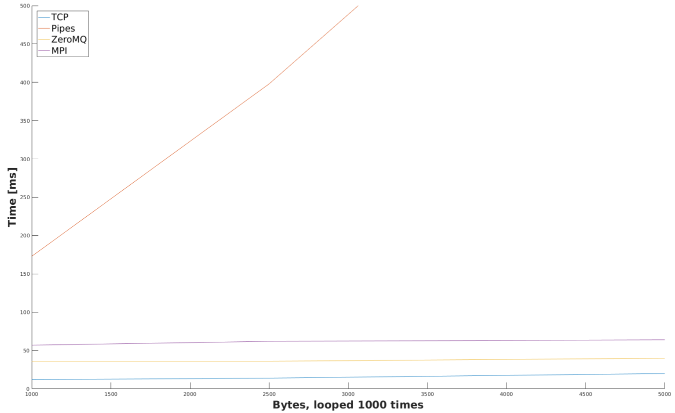
Performance Test

1. The producer generates a list of length n and fills it with integers.
2. The producer writes the list to file (or sends it over the designated transfer protocol).
3. The consumer reads (or receives) the list.
4. The consumer squares each int in the list and sends it back to the producer.
5. The producer reads (or receives) the modified list.

Result for I/O Performance



Result for I/O Performance



Result for I/O Performance

- TCP was the fastest, but the most difficult to implement.

Result for I/O Performance

- TCP was the fastest, but the most difficult to implement.
- Since we assumed a lot of data would be passed we opted for MQ due to its presumptive ease of use and performance.

Weeks 5-7: Implementation

Implementation

- Installation of Software on **bootes**

Implementation

- Installation of Software on **bootes**
- Porting our code to ifort

Implementation

- Installation of Software on **bootes**
- Porting our code to ifort
- A lot of coding.

Results for Project One

Results for Project One

Results for Project One

Results for Project One

Why?