



Crearea nodurilor de calcul MPI utilizând Docker

Cemîrtan Cristian

Introducere

- În această prezentare, vom demonstra că, utilizând Docker, putem instanția nodurile virtuale de calcul ce poate rula programele bazate pe biblioteca MPI.
- Aceasta va aduce facilități studenților ce își fac actual studiile în cadrul disciplinei „Programarea paralelă și distribuită”, și că studenții să-și poată compila și să ruleze programele MPI pe calculatoarele personale, fără a necesita o conexiune constantă la Internet.
- MPI, descifrat ca Message Passing Interface, este o bibliotecă pentru limbajele de programare C/C++ și Fortran, ce permite programatorului să realizeze algoritmi paraleli pentru sistemele de calcul paralel cu memorie distribuită.
- În cadrul MPI, comunicarea între sistemele de calcul are loc prin transmiterea mesajelor de tip „punct-la-punct” (MPI_Send/MPI_Recv), sau prin intermediul funcțiilor pentru comunicarea colectivă (MPI_Bcast/MPI_Scatter/MPI_Gather), într-un mediu virtual de comunicare denumit comunicator (de ex. MPI_COMM_WORLD).

Introducere

- Găzduirea unui cluster MPI necesită configurarea exhaustivă a serverelor sau a mașinilor virtuale bazate pe Linux. O parte dificilă din setarea configurărilor este instalarea sistemului de fișiere NFS (Network File System), ce dă posibilitatea nodurilor din clusterul MPI să aibă accesul comun la un singur program partajat.
- Acest pas dificil poate fi înlocuit cu o soluție ușoară – în loc să instalăm Network File System, vom beneficia de volumele partajate Docker fără a modifica imaginea de bază unui container MPI.
- În următoarele slide-uri, vom ilustra pașii necesari pentru a găzdui propriul cluster virtual MPI, utilizând Docker.
- În cazul Docker, configurarea clusterelor MPI devine automatizată și facilă de realizat prin crearea imaginilor.

Construirea imaginii

- Creăm un director nou (mpi-node), și acolo creăm un fișier dockerfile cu următorul conținut, unde:

Imaginea va fi bazată pe sistemul de operare Ubuntu cu versiunea recentă
FROM ubuntu

Setăm parola contului root ca să fie disponibil pentru conexiunile SSH
Parola este deasemenea - root
RUN echo 'root:root' | chpasswd

Instalăm următoarele pachete:
mpich - instrumentele necesare pentru compilarea și rularea programelor MPI
iputils-ping - programul ping pentru a testa conexiunea la o gazdă
curl - client HTTP (opțional)
nano - editor textual
openssh-server - găzduirea serverului SSH
RUN apt-get update && apt-get install -y mpich iputils-ping curl nano openssh-server

Se permite autentificarea ca root într-o conexiune SSH
RUN sed -ri 's/^#PermitRootLogin.+/PermitRootLogin yes/' /etc/ssh/sshd_config

Ne permite să facem conexiuni SSH la fiecare gazdă fără a le înregistra explicit în fișierul known_hosts
Util pentru rularea programelor MPI pe noduri recent adăugate, fără administrare suplimentară
RUN echo StrictHostKeyChecking no >> /etc/ssh/ssh_config

Se generează o cheie SSH pentru realiza conexiuni la gazde în mod implicit fără a introduce parola
Ne permite să distribuim rularea unei aplicații MPI la celelalte gazde
RUN ssh-keygen -b 2048 -t rsa -f /root/.ssh/id_rsa -q -N ""
RUN cp /root/.ssh/id_rsa.pub /root/.ssh/authorized_keys

Portul expus va fi 22, ce va primi conexiuni SSH
EXPOSE 22

Se creează directorul necesar serverului SSH
RUN mkdir /var/run/sshd

Se rulează serverul SSH, cu catalogarea activităților
CMD /usr/sbin/sshd -D

Construirea imaginii

- Construim imaginea, unde eticheta imaginii este cemti/mpi_node, cu versiunea implicită latest:
- `E:\USM\An III\sem I\Cloud\Individual\mpi-node>docker build -t cemti/mpi-node .`

```
E:\USM\An III\sem I\Cloud\Individual\mpi-node>docker build -t cemti/mpi-node .
```

[+] Building 5.2s (13/13) FINISHED	docker:default
=> [internal] load .dockerignore	0.1s
=> => transferring context: 2B	0.0s
=> [internal] load build definition from dockerfile	0.2s
=> => transferring dockerfile: 493B	0.1s
=> [internal] load metadata for docker.io/library/ubuntu:latest	1.3s
=> [auth] library/ubuntu:pull token for registry-1.docker.io	0.0s
=> [1/8] FROM docker.io/library/ubuntu@sha256:2b7412e6465c3c7fc5bb21d3e6f1917c167358449fecac8176c6e496e5c1f05f	0.0s
=> CACHED [2/8] RUN echo 'root:root' chpasswd	0.0s
=> CACHED [3/8] RUN apt-get update && apt-get install -y mpich iputils-ping curl nano openssh-server	0.0s
=> CACHED [4/8] RUN sed -ri 's/^#PermitRootLogin.+/PermitRootLogin yes/' /etc/ssh/sshd_config	0.0s
=> [5/8] RUN echo StrictHostKeyChecking no >> /etc/ssh/ssh_config	0.7s
=> [6/8] RUN ssh-keygen -b 2048 -t rsa -f /root/.ssh/id_rsa -q -N ""	0.8s
=> [7/8] RUN cp /root/.ssh/id_rsa.pub /root/.ssh/authorized_keys	0.6s
=> [8/8] RUN mkdir /var/run/sshd	0.7s
=> exporting to image	0.3s
=> => exporting layers	0.2s
=> => writing image sha256:6140d29a0df35186b396183e8ae4150793b05eac8c1378db79d7e308420a3ec0	0.0s
=> => naming to docker.io/cemti/mpi-node	0.0s

What's Next?

View a summary of image vulnerabilities and recommendations → `docker scout quickview`

Crearea rețelei și a volumului

- Crearea explicită unei rețele permite containerele să comunice direct între ele utilizând denumiri de gazdă (hostname) în loc de adrese IP care posibil se va schimba pe parcurs. Crearea unui volum Docker ne scutește de instalarea Network File System pe toate containere.
- Crearea rețelei

```
E:\USM\An III\sem I\Cloud\Individual\mpi-node>docker network  
create mpi-net
```

```
b10a1666adcc00a73860501b492d7ede86b91903bb6e81097aa5e0d981ae662  
9
```

- Crearea volumului partajat

```
E:\USM\An III\sem I\Cloud\Individual\mpi-node>docker volume  
create mpi-vol
```

```
mpi-vol
```


Verificare

- Verificăm dacă s-au creat rețeaua și volumul:

```
E:\USM\An III\sem I\Cloud\Individual\mpi-node>docker network list
```

NETWORK ID	NAME	DRIVER	SCOPE
9ae1255702c7	bridge	bridge	local
f288bb87f89a	host	host	local
b10a1666adcc	mpi-net	bridge	local
497d1a80e476	none	null	local
edbfaf0732194	php-net	bridge	local

```
E:\USM\An III\sem I\Cloud\Individual\mpi-node>docker volume list
```

DRIVER	VOLUME NAME
local	mpi-vol
local	php-vol

Crearea containerelor

- La moment, vom crea 4 containere. Primul container va fi accesibil din exterior. Containerele vor fi bazate pe imaginea noastră cemti/mpi-node.

```
C:\Users\Cristian>docker run -d --name mpi1 --network mpi-net --  
volume=mpi-vol:/root/mpi -p 800:22 cemti/mpi-node  
3b6815b993e2c1db30dd8bd0c2dc08a41f270f9e50677d975bf31bcf54158e0d
```

```
C:\Users\Cristian>docker run -d --name mpi2 --network mpi-net --  
volume=mpi-vol:/root/mpi cemti/mpi-node  
E00bee6b20357d8633c4f4274cdea8e7f838f4bd1d74e03edde72135d325a0df
```

```
C:\Users\Cristian>docker run -d --name angry-birds --network mpi-net --  
volume=mpi-vol:/root/mpi cemti/mpi-node  
08e6c1ba44ab70976779bfb7309e92588dd1f1ba624b421fab656faf18018cc6
```

```
C:\Users\Cristian>docker run -d --name b0r15 --network mpi-net --  
volume=mpi-vol:/root/mpi cemti/mpi-node  
0a8418420621395f2e35d3cc90d61ef894498c88e56ae718ac56d940233f0fd8
```

Testarea conexiunii între containere

- Realizăm o conexiune SSH către mpi1. Utilizând comanda ping, verificăm dacă gazda mpi2 poate fi translatată în adresa IP.

```
C:\Users\Cristian>ssh -p 800 root@localhost
root@localhost's password:
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 6.4.16-linuxkit x86_64)
```

```
* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:        https://ubuntu.com/advantage
```

This system has been minimized by removing packages and content that are not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

```
Last login: Mon Nov 20 01:22:52 2023 from 192.168.65.1
```

```
root@3b6815b993e2:~# ping mpi2
```

```
PING mpi2 (172.19.0.3) 56(84) bytes of data.
```

```
64 bytes from mpi2.mpi-net (172.19.0.3): icmp_seq=1 ttl=64 time=0.049 ms
```

```
64 bytes from mpi2.mpi-net (172.19.0.3): icmp_seq=2 ttl=64 time=0.050 ms
```

```
^C
```

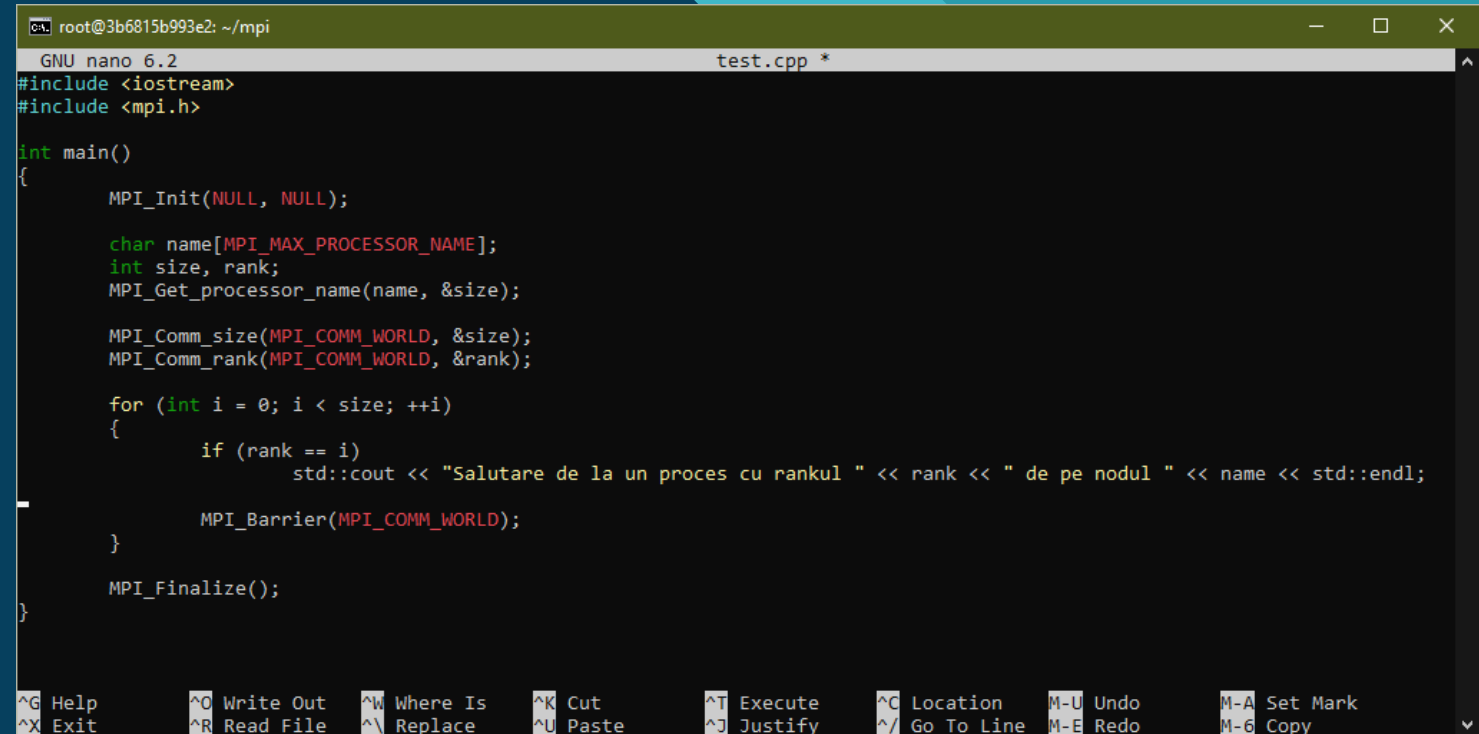
```
--- mpi2 ping statistics ---
```

```
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
```

```
rtt min/avg/max/mdev = 0.049/0.049/0.050/0.000 ms
```

Crearea unui program MPI

```
root@3b6815b993e2:~# cd mpi
root@3b6815b993e2:~/mpi# pwd
/root/mpi
root@3b6815b993e2:~/mpi# nano text.cpp
root@3b6815b993e2:~/mpi# mpicxx test.cpp
```

A screenshot of a terminal window with a dark background. The window title is 'root@3b6815b993e2: ~/mpi'. Inside the terminal, the GNU nano 6.2 editor is open, editing a file named 'test.cpp'. The code in the file is as follows:

```
#include <iostream>
#include <mpi.h>

int main()
{
    MPI_Init(NULL, NULL);

    char name[MPI_MAX_PROCESSOR_NAME];
    int size, rank;
    MPI_Get_processor_name(name, &size);

    MPI_Comm_size(MPI_COMM_WORLD, &size);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);

    for (int i = 0; i < size; ++i)
    {
        if (rank == i)
            std::cout << "Salutare de la un proces cu rankul " << rank << " de pe nodul " << name << std::endl;

        MPI_Barrier(MPI_COMM_WORLD);
    }

    MPI_Finalize();
}
```

The bottom of the terminal window shows the nano editor's help menu with various shortcuts like ^G for Help, ^O for Write Out, etc.

```
root@3b6815b993e2:~/mpi# mpiexec -n 8 ./a.out
Salutare de la un proces cu rankul 0 de pe nodul 3b6815b993e2
Salutare de la un proces cu rankul 1 de pe nodul 3b6815b993e2
Salutare de la un proces cu rankul 2 de pe nodul 3b6815b993e2
Salutare de la un proces cu rankul 3 de pe nodul 3b6815b993e2
Salutare de la un proces cu rankul 4 de pe nodul 3b6815b993e2
Salutare de la un proces cu rankul 5 de pe nodul 3b6815b993e2
Salutare de la un proces cu rankul 6 de pe nodul 3b6815b993e2
Salutare de la un proces cu rankul 7 de pe nodul 3b6815b993e2
```

```
root@3b6815b993e2:~/mpi# mpiexec -host mpi1:2,mpi2:4 ./a.out
Salutare de la un proces cu rankul 0 de pe nodul 3b6815b993e2
Salutare de la un proces cu rankul 1 de pe nodul 3b6815b993e2
Salutare de la un proces cu rankul 2 de pe nodul e00bee6b2035
Salutare de la un proces cu rankul 3 de pe nodul e00bee6b2035
Salutare de la un proces cu rankul 4 de pe nodul e00bee6b2035
Salutare de la un proces cu rankul 5 de pe nodul e00bee6b2035
```

Studiu de caz

- Vom încerca să rulăm un program ce realizează următoare cerință pentru o lucrare de laborator din partea disciplinei „Programarea paralelă și distribuită”:

Fie dat un joc bimatriceal $\Gamma = \langle I, J, A, B \rangle$ unde I – mulțimea de indici ai liniilor matricelor, J – mulțimea de indici ai coloanelor matricelor, iar $A = \|a_{ij}\|_{\substack{i \in I \\ j \in J}}$, $B = \|b_{ij}\|_{\substack{i \in I \\ j \in J}}$ reprezintă matricele de câștig ale jucătorilor.

Elementul $i \in I$ respectiv $j \in J$, se numește strategie pură a jucătorului 1, respectiv a jucătorului 2, perechea de indici (i, j) reprezintă o situație în strategii pure. Jocul se realizează astfel: fiecare jucător independent și „concomitent” (adică alegerea de strategii nu depinde de timp) alege strategie sa, după care se obține o situație în baza căreia jucătorii calculează câștigurile care reprezintă elementul a_{ij} pentru jucătorul 1, respectiv b_{ij} pentru jucătorul 2 și cu aceasta jocul este sfârșit.

Situația de echilibru este perechea de indici (i^*, j^*) , pentru care se verifică sistemul de inegalități:

$$(i^*, j^*) \Leftrightarrow \begin{cases} a_{i^*j^*} \geq a_{ij^*} \quad \forall i \in I \\ b_{i^*j^*} \geq b_{i^*j} \quad \forall j \in J \end{cases}$$

Vom spune că *linia i strict domină linia k* în matricea A dacă și numai dacă $a_{ij} > a_{kj}$ pentru orice $j \in J$. Dacă există j pentru care inegalitatea nu este strictă, atunci vom spune că *linia i domină linia k* . Similar, vom spune: *coloana j strict domină coloana l* în matricea B dacă și numai dacă $b_{ij} > b_{il}$ pentru orice $i \in I$. Dacă există i pentru care inegalitatea nu este strictă, atunci vom spune: *coloana j domină coloana l* .

Studiu de caz

- Copiem programul și fișierele cu datele de intrare pentru 2 matrici 6x6 și 2 matrici 500x500, în volumul mpi-vol.

```
E:\USM\An III\sem I\Calcul paralel\Lab\12>docker cp lab5.cpp
```

```
mpi1:/root/mpi
```

```
Successfully copied 10.8kB to mpi1:/root/mpi
```

```
E:\USM\An III\sem I\Calcul paralel\Lab\12>docker cp ..\6\lab1\mtx1.txt
```

```
mpi1:/root/mpi
```

```
Successfully copied 2.05kB to mpi1:/root/mpi
```

```
E:\USM\An III\sem I\Calcul paralel\Lab\12>docker cp ..\6\lab1\mtx5.txt
```

```
mpi1:/root/mpi
```

```
Successfully copied 1.45MB to mpi1:/root/mpi
```

- Pe containerul mpi1:

```
root@3b6815b993e2:~/mpi# ls
```

```
a.out  lab5.cpp  mtx1.txt  mtx5.txt  test.cpp
```

```
root@3b6815b993e2:~/mpi# mpicxx -std=c++20 lab5.cpp -o lab5.out
```

```
root@3b6815b993e2:~/mpi# mpiexec -host mpi1:1,mpi2:1,angry-birds:2 ./lab5.out mtx1.txt 2 2 3 2
```

Situatiile Nash de echilibru:

(2, 2), (3, 3), (4, 4),

Total: 3

```
root@3b6815b993e2:~/mpi# mpiexec -host mpi1:2,mpi2:4,angry-birds:3,b0r15:6 ./lab5.out mtx5.txt 0 0 6 8 time
```

Situatiile Nash de echilibru:

(35, 489), (37, 132), (40, 229), (61, 284), (62, 41), (67, 424), (69, 321), (77, 243), (97, 42), (106, 313), (107, 44),
(116, 425), (120, 429), (127, 297), (128, 388), (133, 362), (147, 217), (153, 381), (155, 166), (156, 361), (161, 169),
(163, 64), (165, 110), (200, 318), (225, 119), (232, 322), (246, 160), (247, 97), (261, 206), (263, 326), (266, 297),
(276, 364), (293, 365), (301, 245), (317, 383), (318, 185), (321, 307), (340, 81), (352, 141), (354, 407), (358, 210),
(362, 10), (362, 103), (397, 154), (407, 390), (416, 27), (418, 319), (423, 95), (428, 487), (434, 378), (440, 190),
(444, 31), (457, 260), (493, 211), (498, 28),

Total: 55

Crearea fisierului: 0.0432585

Media: 0.199623, Max: 0.266628, Min: 0.0692021

Concluzii

- Procesul de creare unui cluster virtual MPI, pe baza tehnologiei Docker, este concluzionat în felul următor:
 1. Ne asigurăm că este instalat Docker pe calculatorul personal;
 2. Creăm o imagine pe baza SO Ubuntu unde se instalează pachetele necesare pentru a compila și a rula aplicațiile MPI, de asemenea se instalează și se configurează serverul SSH pentru a face posibilă distribuirea execuției programului la celelalte containere.
 3. Pentru ca denumirile nodurilor de calcul (containerelor) să fie translatate în adrese IP, vom crea o nouă rețea Docker;
 4. Pentru a oferi un spațiu de lucru partajat pentru toate nodurile de calcul, vom crea un nou volum Docker;
 5. Rularea containerelor noi pe baza imaginii create.

Docker Hub

<https://hub.docker.com/r/cemti/mpi-node/>



Webografie

- 1. Docker Docs. Install Docker Desktop on Windows. <https://docs.docker.com/desktop/install/windows-install/>.
- 2. MPICH. <https://www.mpich.org/>.
- 3. OpenSSH. <https://www.openssh.com/>.
- 4. Boris Hîncu, Elena Calmîs. Modele de programare paralelă pe clustere. Partea I. Programare MPI. <https://www.scribd.com/document/440661312/Modele-de-programare-paralela-pe-clustere-Programare-MPI-docx>.
- 5. Benedikt Steinbusch. Introduction to parallel programming with MPI and OpenMP. https://www.fz-juelich.de/en/ias/jsc/education/training-courses/training-materials/course-material-introduction-to-mpi-and-openmp-feb-2022/mpi-omp-article/@_@download/file.
- 6. Ubuntu documentation. Package management with APT. <https://help.ubuntu.com/community/AptGet/Howto>.
- 7. IBM Cloud Docs. Generating and using an SSH key. <https://cloud.ibm.com/docs/power-iaas?topic=power-iaas-creating-ssh-key>.
- 8. Docker Docs. Dockerfile reference. <https://docs.docker.com/engine/reference/builder/>.
- 9. Docker Docs. Networking Overview. <https://docs.docker.com/network/>.
- 10. Docker Docs. Volumes. <https://docs.docker.com/storage/volumes/>.



Vă mulțumim
pentru atenție