Parallel programming laboratory

Imre Szeberényi

BME IIT

<szebi@iit.bme.hu>

MŰEGYETEM 1782

Burglar – cops

- Cops are chasing a burglar on the street (in a labyrinth).

- The cops are communicating with each other or with the police centre on handy (mobile phone)

- Every moving actor (cop, robber) is a thread or a process that communicates with each other.

- Chasing is over when the robbers can not move.

Labyrint server

- The server runs on the local machine, and it is responsible for displaying the labyrinth and the moving actors.

- It is listening to TCP port 15623 and executes simple commands.

- The clients are running on the server para.

- The connection is built up through the TCP tunnel.

Architecture

Burglar

Labyrinth server

Cop #1

Cop #2

Cop #3

Cop #4

Cop #5

Process

TCP/IP communication

MPI processes
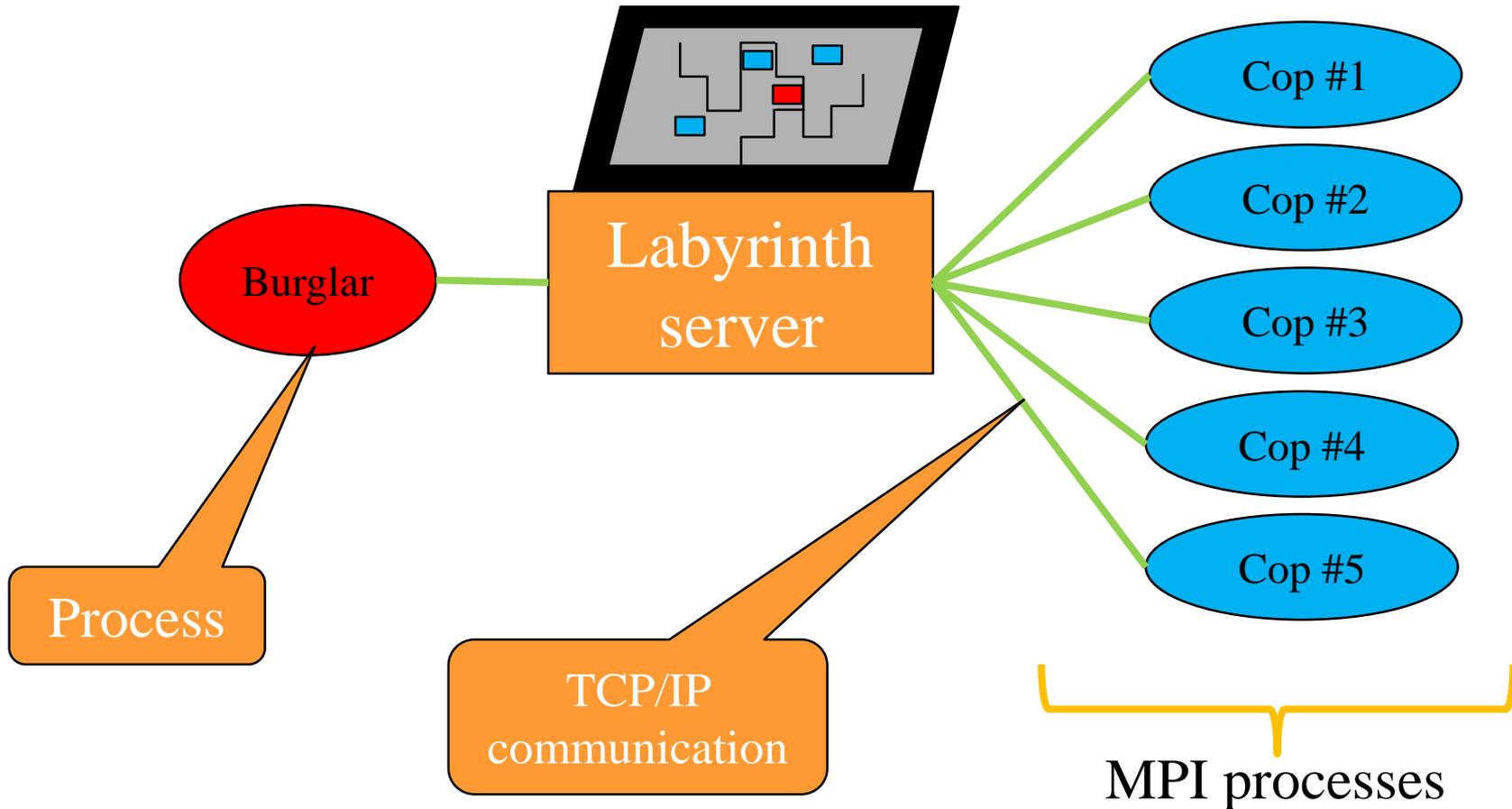
Communication

- Simple text messages in both directions ending with endline ('\n').

Initialization: Player → Server

Cmd	Answer	Meaning
M		Burglar step in
	X,Y	Burglar initial position
N		Cop step in
	X,Y	Cop initial position

Communication/2

Other commands: Player → Server

Cmd	Meaning
U,D,L,R	Step (up, down, left, right)

Answer	Meaning
F	Stepping is not possible
C	Game over
view	Successful step. *view* is the view what is viewable path from the new position.

Communication/3

view:= U#{#} D#{#} L#{#} R#{#}X

→ R
← U2220R2223D3L223X

Legend:
- Cop (0) — green
- Burglar (1) — red
- Street (2) — yellow
- House (3) — blue
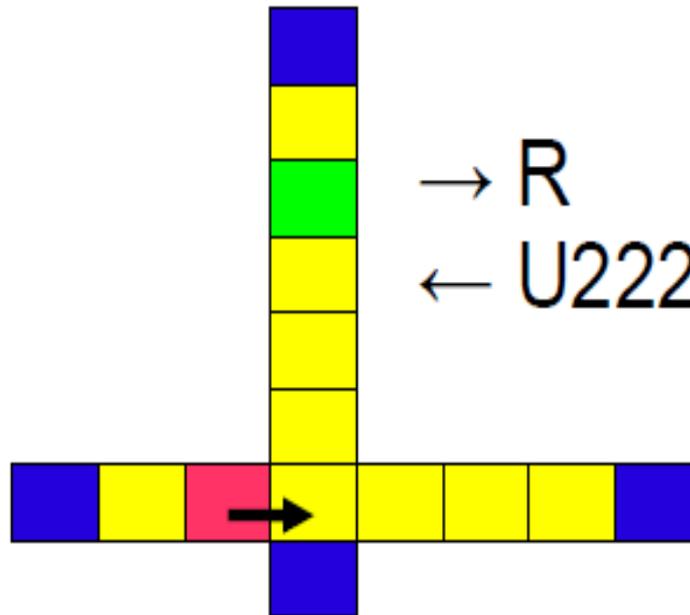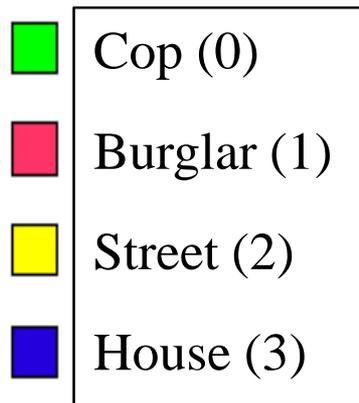
Helper class: player

```cpp
class Player {
 public:
  typedef enum {up, down, left, right}
  direction;
  typedef enum {burglar, cop} player_t;
  Player(char* host, int port, player_t cop);
  char *move(direction d);
  int getx(){ return x; }
  int gety(){ return y; }
  player_t gettyp() { return me; }
  void setx(int x) { this->x = x; }
  void sety(int y) { this->y = y; }
 private:
  ...
```

Multiuser environment
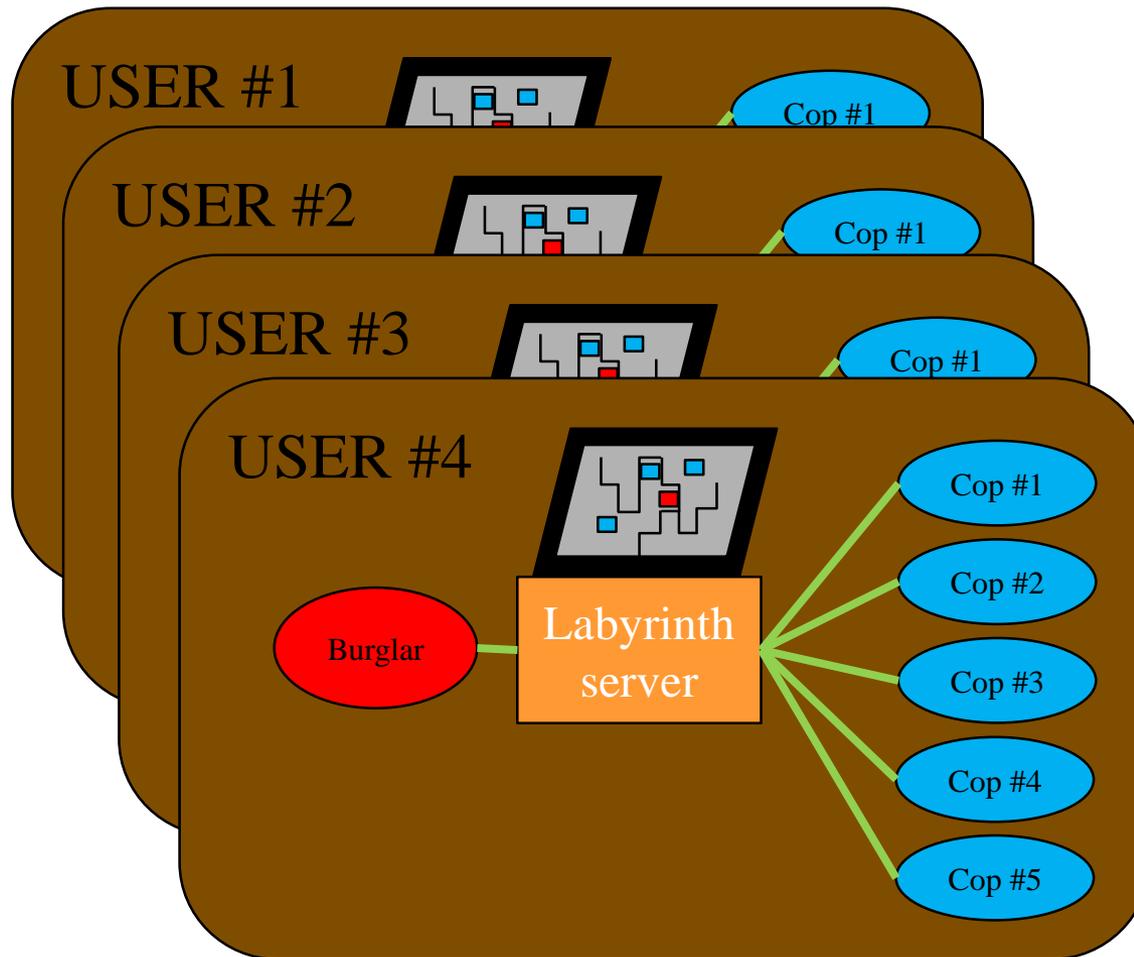
Multiuser environment #2

- The labyrinth servers are running on the local machines and listen at port number 15623.
- The clients are running on para cluster.
- A tunnel should be used between the para and the local machine.
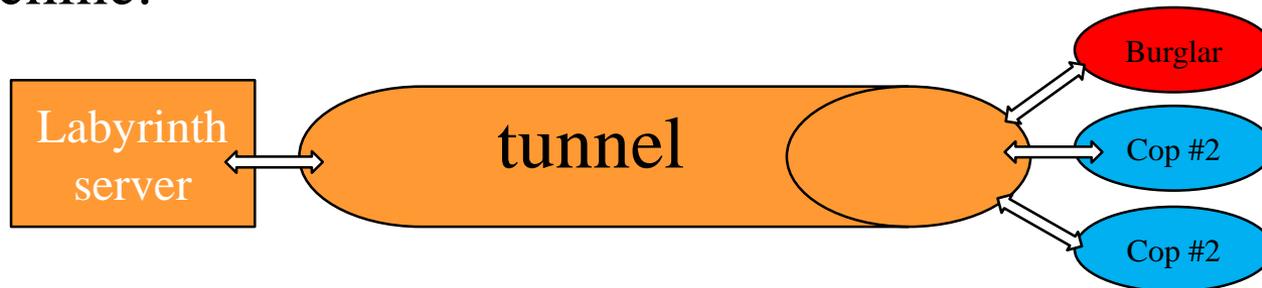
- The tunnels must end at unique port number on para:
- Suggested unique port number is 15000+UID
  - where UID is the user id on para cluster.
  - id -l command prints the UID.

Putty configuration

Session->SSH->Tunnels
√ Local ports accept connection from other hosts
Source port: unique PORT nr.
Destination: localhost:15623
√ Remote
√ Ipv4
Add

PuTTY Configuration

Options controlling SSH port forwarding

Port forwarding

☑ Local ports accept connections from other hosts
☐ Remote ports do the same (SSH-2 only)

Forwarded ports: Remove

Add new forwarded port:

Source port: 15502   Add

Destination: localhost:15623

○ Local   ● Remote   ○ Dynamic
○ Auto   ● IPv4   ○ IPv6

- Rlogin
- SSH
  - Kex
  - Auth
  - TTY
  - X11
  - Tunnels

About   Help   Open   Cancel
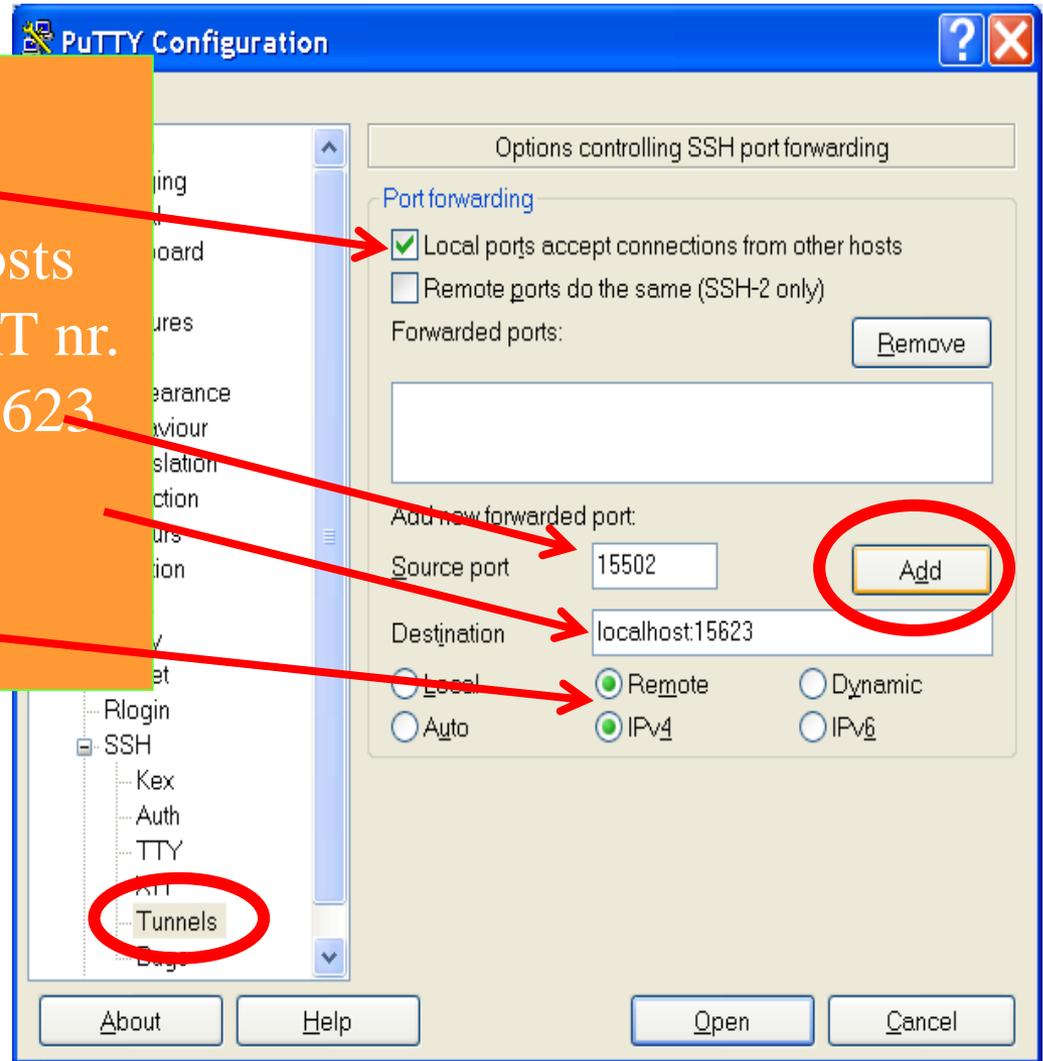
Possible configurations

- Option 1:
  - Your local machine runs the labyrinth server
  - para cluster runs the clients

- Option 2:
  - Virtual machine running on cloud runs the labyrinth server.
  - para cluster runs the client.

- Option 3:
  - Virtual machine running on cloud runs the labyrinth server and clients. (no tunnel needed)

Using option 1

Unpacking the sources:

    server dir: → local machine

    client: → para cluster

Setting up the tunnel

    see putty config or

    ssh PORT:localhost:15623 U@H

Staring the server

   cd server

    server.bat or ./server.sh

On para cluster:

    cd server

    make

    ./test.sh,

 or

    ./testmpi.sh
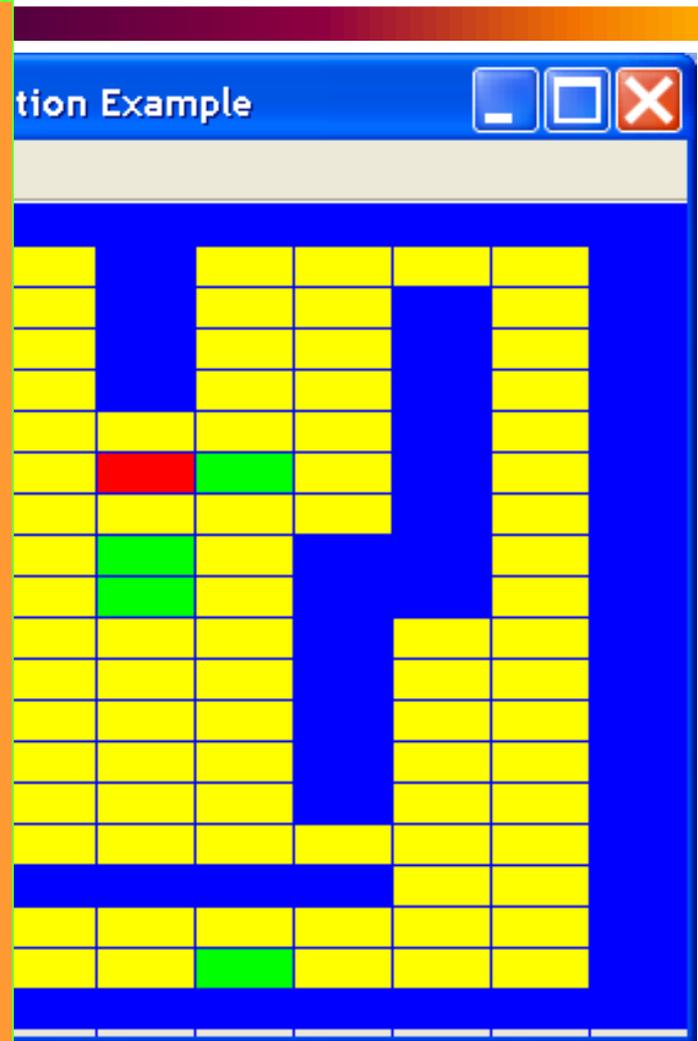
Using option 2

- login to the smalville cluster
- start a VM from template: parab and login to
  - git clone https://git.ik.bme.hu/Parlab/burglar.git
  - cd burglar/server
  - ./server.sh
  - open a new terminal window
  - ssh USER@FULL_HOSTNAME_OF_PARA
  - git clone https://git.ik.bme.hu/Parlab/burglar.git
  - cd client
  - ./test.sh → **record your unique port number**
  - exit
  - ssh PORT:localhost:1623U@H
  - cd client
  - make
  - ./test.sh

Assignment

- Create an MPI program, models the behaviour of cops.

- Simple solution:
  - If the cop saw the burglar, broadcasts a message with the exact coordinates.
  - The cops try to move to that direction.
    - Problem: walls could block the move
  - The example cop forgets to tell the position.

Assignment cont. #1

- Sophisticated solution #1:
  - While the cops are moving, they build a map from the streets (labyrinth)
  - If the cop saw the burglar, broadcasts a message with the exact coordinates.
  - The cops use the map to calculate the way to last place of the burglar.

Assignment cont. #2

- Sophisticated solution #2:
  - While the cops are moving, they send information to the centre.
  - The centre bulid up the city map.
  - The centre sends movement commands to the cops.

Assignment cont. #3

Helper materials:

https://git.ik.bme.hu/Parlab/burglar.git

or on para: ~szebi/para/burglar

- server – labyrinth server (jar)
- client – example client programs
- tests

- Task to be submitted :
  - short description: pdf, or txt
  - sources,
  - makefile