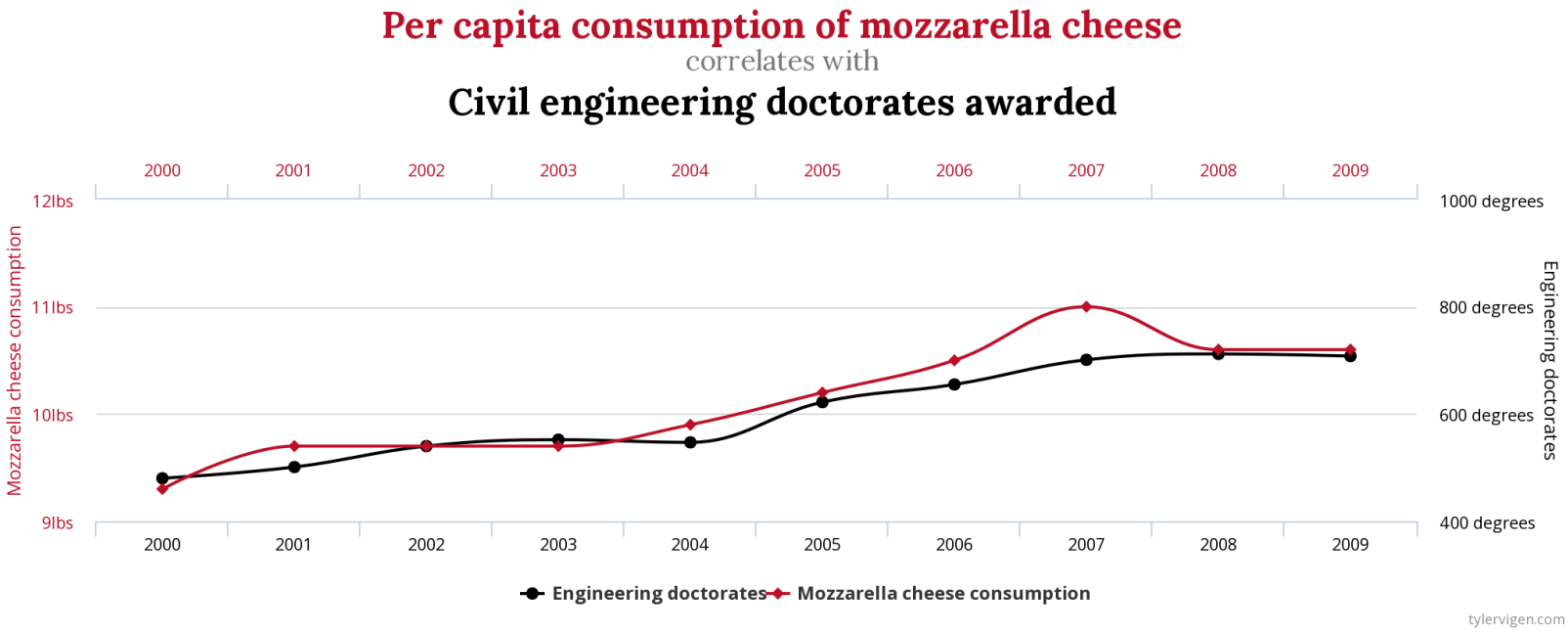
Wireshark and ns-3

Introduzione

SAPIENZA
UNIVERSITÀ DI ROMA

Andrea Lacava

Risultati questionario



Source: https://www.tylervigen.com/spurious-correlations

VM/ Docker Telecomunicazioni Homework

- Per usare questi tools è necessario un ambiente Linux-based
- FAQ:
  - È obbligatorio usare una VM o Docker?
    - No, ma nessun supporto tecnico se non funziona
  - Come sarà l'homework?
    - Esercizi e analisi dati su ns-3 e TCP
  - Ho un mac con M1, come faccio?
    - Installa barebone
    - Prova a fare una vm su vmWare Fusion Tech Preview
    - https://communities.vmware.com/t5/Fusion-22H2-Tech-Preview/How-to-install-Ubuntu-on-Fusion-Tech-preview-for-M1-June-2022/td-p/2913103
    - sudo apt install g++ python3 cmake git qtbase5-dev qtchooser qt5-qmake qtbase5-dev-tools wireshark git build-essential
    - Install vscode or your favorite editor
    - git clone https://github.com/nsnam/ns-3-dev-git

Network Simulator 3

- Simulatore di reti discreto ad eventi.
  - Simulatore:
    - Uno strumento software che predice il comportamento di una rete, senza che sia presente una rete reale, una rete di telecomunicazioni reale senza i costi reali
  - Evento discreto:
    - Ogni evento si verifica in un particolare istante nel tempo
    - Vengono processati solo gli eventi effettivi della rete e non le pause tra una comunicazione e la prossima.
    - Se il numero di eventi è basso, la simulazione è molto veloce
- Open Source

Tesi di implementazione su ns-3 disponibili

Main entities ns-3

- Modelli:
  - Rappresentano le stack simulate.
- Topology Helpers:
  - Classi di management per la creazione e il collegamento tra i vari modelli.
  - Automatizzano l'installazione e il setup della stack nei Nodes
- Examples:
  - Script di esempio che usano i modelli
- Scratch folder:
  - Cartella all'interno di ns-3 per creare nuovi esempi usando i modelli già preesistenti
- Nodes:
  - Rappresentano una singola entità generica all'interno di ns-3
- Application:
  - Strato che genera attività di rete, e.g., HTTP, FTP, etc.
- Channel:
  - Il canale che collega i nodi, e.g., PointToPointChannel (wired point-to-point link), WifiChannel (wireless link), CsmaChannel (ethernet-like link), etc.
- Net Device:
  - L'adattatore di rete che viene installato sul nodo per usare il Channel, e.g., PoinToPointNetDevice, WifiNetDevice, CsmaNetDevice, etc.

Workflow ns-3

- Definire Topologia di rete
  - Quanti nodi compongono la rete?
  - Come sono connessi i nodi fra di loro? Che tipo di link stanno usando?
- Creare i layer di strato superiore:
  - Cosa montani i nodi?
  - Quale è la loro stack?
  - Che protocollo usano?
- Configurazione dei links
  - Quale è l'indirizzo di ogni link?
  - Quale è la MTU?
  - Quali sono i parametri di default del modello di traffico per la simulazione?
  - Etc. Etc.

Workflow ns-3

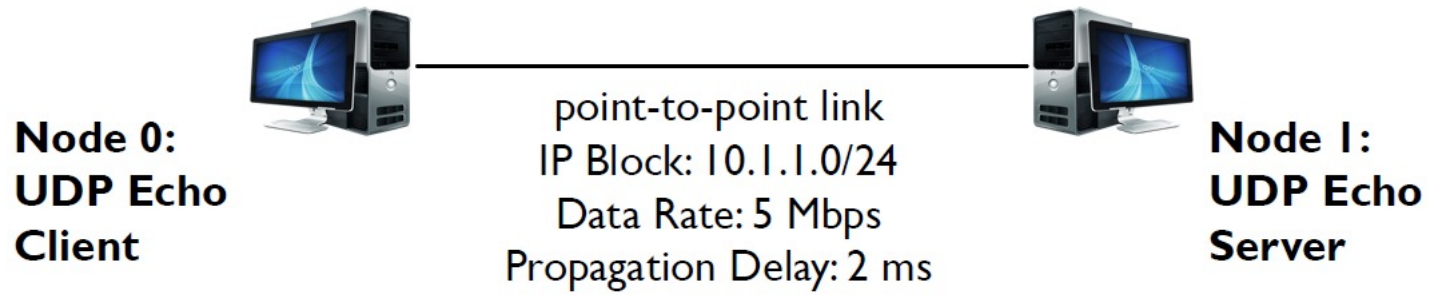
- Esecuzione
  - I nodi generano eventi a seconda di come sono programmati
  - Vengono quindi scambiati i pacchetti secondo il modello di traffico
  - I pacchetti possono essere tracciati e salvati
- Performance analysis
  - End-to-end delay
  - End-to-end packet delivery probability
  - Energy consumption
  - The sky is the limit

Modelli su ns-3 (subset)

- Physical Layer
  - Wired links: point-to-point, shared bus, etc.
  - Wireless links: line-of-sight, non-line-of-sight, indoor and outdoor propagation models, etc
- Data Link Layer:
  - CSMA, CSMA-CA, CSMA-CD
- Network Layer:
  - Protocolli di Routing: AODV, DSDV, DSR, OLSR
  - IPv4, IPv6
- Transport Layer:
  - TCP, UDP
- Full standard implementations:
  - IEEE 802.11 and variants, LoRaWAN
  - LTE, 5G mmWave, O-RAN

Hands on ns-3!

- /examples/tutorial/first.cc
- UDP Echo è un applicazione classica in cui il client invia informazioni al server che risponde con lo stesso pacchetto
- In questo esempio, mandiamo un solo pacchetto a tempo t=2 sec



/examples/tutorial/first.cc

- Entrypoint del codice:
  - int main(int argc, char * argv[])

- Topology Helper:
  - **NodeContainer,** definisce ed istanzia i nodi nella rete
    - NodeContainer nodes;
    - nodes.Create(2);
  - **PoinToPointHelper,** collega il PointToPointNetDevice sul PointToPointChannel
    - PointToPointHelper pointToPoint;
    - pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
    - pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));
  - **NetDeviceContainer,** installa i PointToPoint devices all'interno dei nodi definiti
    - PointToPointHelper pointToPoint;
    - pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
    - pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));
  - **InternetStackHelper**: installa la stack di internet sui nodi
    - InternetStackHelper stack;
    - stack.Install (nodes);
  - **Ipv4AddressHelper**: sets the IP addresses to the nodes
    - Ipv4AddressHelper address;
    - address.SetBase ("10.1.1.0", "255.255.255.0");
    - Ipv4InterfaceContainer interfaces = address.Assign (devices);

/examples/tutorial/first.cc

- Applicazioni:
  - **UdpEchoServerHelper,** Crea un echo server sul nodo 1
    - UdpEchoServerHelper echoServer (9);
    - ApplicationContainer serverApps = echoServer.Install (nodes.Get (1));
    - serverApps.Start (Seconds (1.0));
    - serverApps.Stop (Seconds (10.0));
  - **UdpEchoClientHelper,** Crea un echo client sul nodo 0
    - UdpEchoClientHelper echoClient (interfaces.GetAddress (1), 9);
    - echoClient.SetAttribute ("MaxPackets", UintegerValue (1));
    - echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
    - echoClient.SetAttribute ("PacketSize", UintegerValue (1024));
    - ApplicationContainer clientApps = echoClient.Install (nodes.Get (0));
    - clientApps.Start (Seconds (2.0));
    - clientApps.Stop (Seconds (10.0));

/examples/tutorial/first.cc

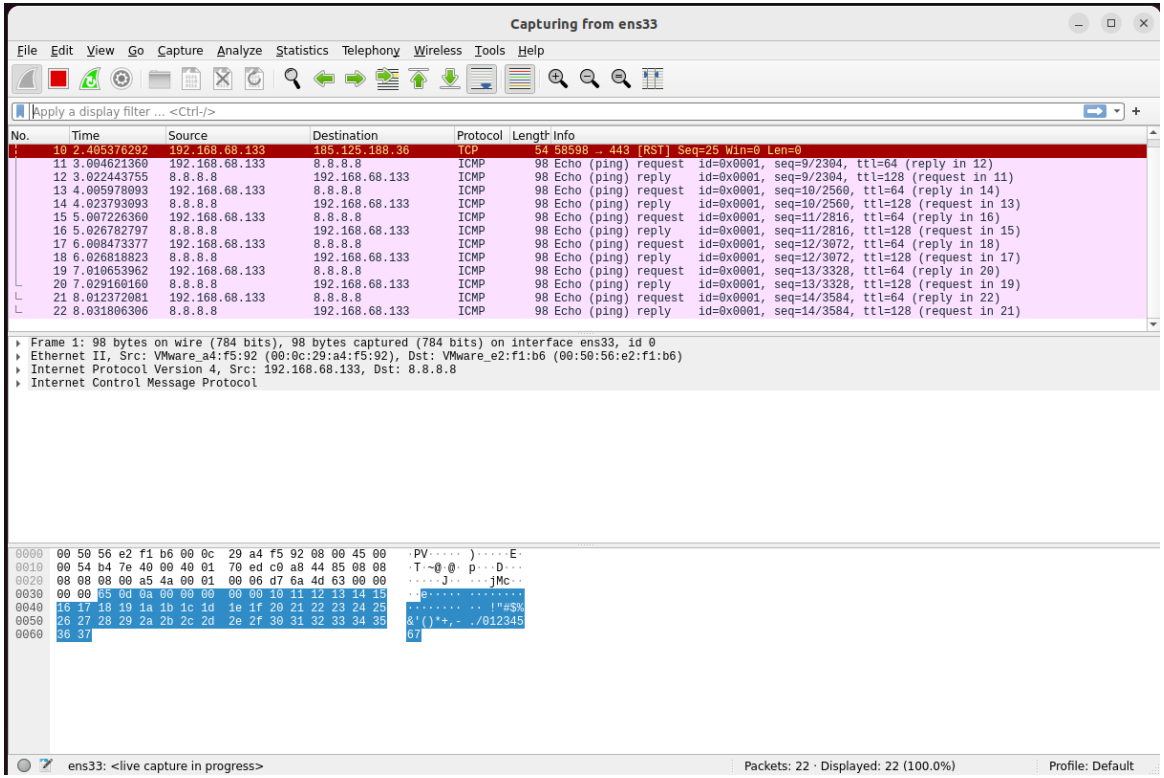
- Simulatore:
  - **Lancio effettivo della simulazione:**
    - Simulator::Run ();
    - Simulator::Destroy ();
    - return 0;
  - **Lista comandi:**
    - ./ns3 configure --enable-tests --enable-examples # only the first time
    - ./ns3 build # only the first time
    - ./ns3 run first # questo è il comando più importante
  - **Output (expected):**
    - At time 2s client sent 1024 bytes to 10.1.1.2 port 9
    - At time 2.00369s server received 1024 bytes from 10.1.1.1 port 49153
    - At time 2.00369s server sent 1024 bytes to 10.1.1.1 port 49153
    - At time 2.00737s client received 1024 bytes from 10.1.1.2 port 9

/scratch/myfirst.cc

- Possiamo salvare i pacchetti generati da ns-3:
  - **Copiare first.cc nella cartella scratch rinominarlo in myfirst**
  - **Salvataggio pcap files:**
    - pointToPoint.EnablePcapAll ("myfirst");
    - Simulator::Run ();
  - **Come si visualizzano?**
    - **tcpdump**
    - **aircrack-ng**

WIRESHARK

- Open source packet analyzer and dissector
  - Analisi dei pacchetti salvati:
    - Qualsiasi traccia catturata può essere analizzata con la giusta estensione
  - Cattura dati live:
    - Servono privilegi di root



- Pcap di catture vere sono pieni di informazioni
  - Usare i filtri
  - Risoluzione dei nomi
    - Visualizza -> Risoluzione nomi
  - Modalità promiscua
  - Stream TCP (Statistiche)
    - Segui flusso TCP (stream TCP plots)
    - Flow graph o Grafico del flusso
  - Etc. (esplorate e divertitevi)

Link utili

- Tutorial ns-3:
  - https://www.nsnam.org/docs/release/3.36/tutorial/ns-3-tutorial.pdf
- Guida wireshark:
  - https://www.wireshark.org/docs/
- Link per download VM (no Mac M1):
  - https://drive.google.com/drive/u/1/folders/1ca0y-7m1b8ZRH4QXlblB0wVcZD_oc0-M