

Simuladores aula 2

Bruno Pereira

Universidade Federal de Minas Gerais

bruno.ps@dcc.ufmg.com

30 de agosto de 2016

Agenda

1 Continuando...

- Aula passada

2 Outros simuladores

- The Network Simulator - NS2
- OMNeT++
- Castalia
- SUMO
- The ONE
- TOSSIM
- Cooja

3 Dúvidas sobre o projeto

- Árvore de Roteamento

Exercício PingPong

Conceitos

- ① Comportamento do nó
- ② Modelos
- ③ Criar um arquivo de configuração

Exercício PingPong

Conceitos

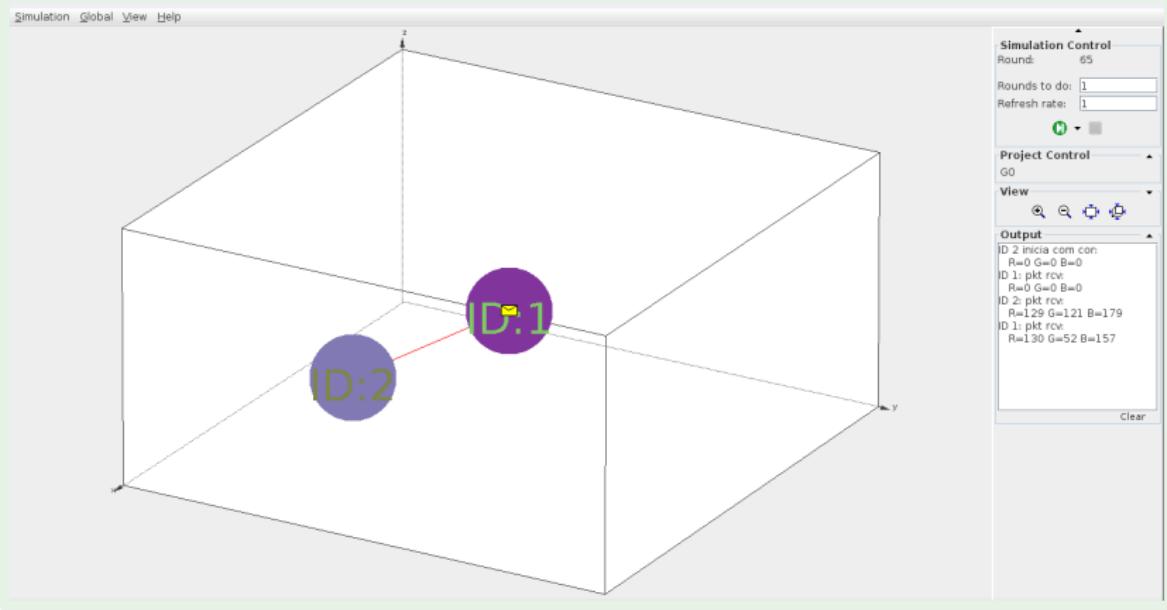
- ① Comportamento do nó
- ② Modelos
- ③ Criar um arquivo de configuração

Comportamento PingPong

- Nesta simulação, dois nós vão trocar mensagens entre si.
- Os nós geram cores em RGB de modo aleatório.
- Cada cor gerada deve ser anexada em uma mensagem, que será enviada para o vizinho.
- Ao receber uma mensagem:
 - O nó deve alterar sua cor conforme os valores RGB recebidos.
 - Gerar uma nova cor RGB e enviar por broadcast.

Exercícios da aula passada

Ping Pong



Exercícios da aula passada

Tarefa 1

- ① Execute os 6 exemplos do Sinalgo.
- ② Descreva a finalidade do exemplo.
- ③ Quais conceitos visto em sala de aula que são demonstrados em cada exemplo.
- ④ Descreva as limitações de cada exemplo.
- ⑤ Quais os pontos fortes e fracos do Sinalgo?

Outros simuladores

NS2

- Discrete event simulator
- Support for simulation of TCP
- Support routing protocols
- Multicast protocols over wired and wireless (local and satellite) networks
- **Aqua-SIM**
- **NS3.**

Continuando...
○○○

The Network Simulator - NS2

Outros simuladores
○●○○○○○○○○

Dúvidas sobre o projeto
○○○○

Vídeo

OMNeT++

- Component-based C++
- Support for sensor networks
- Wireless ad-hoc networks
- Internet protocols

Vídeo

OMNeT++

- Component-based C++
- Support for sensor networks
- Wireless ad-hoc networks
- Internet protocols

Castalia

Castalia is a simulator based on the OMNeT++ for WSN.

- Advanced channel model based on empirically measured data
- Advanced radio model based on real radios for low-power communication
- Extended sensing modelling provisions
- MAC and routing protocols available
- *Body Area Networks (BAN)*

Sumo – Simulation of Urban MObility

- Microscopic simulation - vehicles, pedestrians and public transport are modeled explicitly
- Online interaction – control the simulation with TraCI
- Simulation of multimodal traffic, e.g., vehicles, public transport and pedestrians
- Time schedules of traffic lights can be imported or generated automatically by SUMO
- No artificial limitations in network size and number of simulated vehicles
- Supported import formats: OpenStreetMap, VISUM, VISSIM, NavTeq
- SUMO is implemented in C++ and uses only portable libraries

Continuando...

○○○

SUMO

Outros simuladores

○○○○●○○○○

Dúvidas sobre o projeto

○○○○

Vídeo

The Opportunistic Network Environment simulator

- Generating node movement using different movement models
- Routing messages between nodes with various DTN routing algorithms and sender and receiver types
- Visualizing both mobility and message passing in real time in its GUI
- ONE can import mobility data from real-world traces or other mobility generators.

The Opportunistic Network Environment simulator

- Generating node movement using different movement models
- Routing messages between nodes with various DTN routing algorithms and sender and receiver types
- Visualizing both mobility and message passing in real time in its GUI
- ONE can import mobility data from real-world traces or other mobility generators.

Vídeo

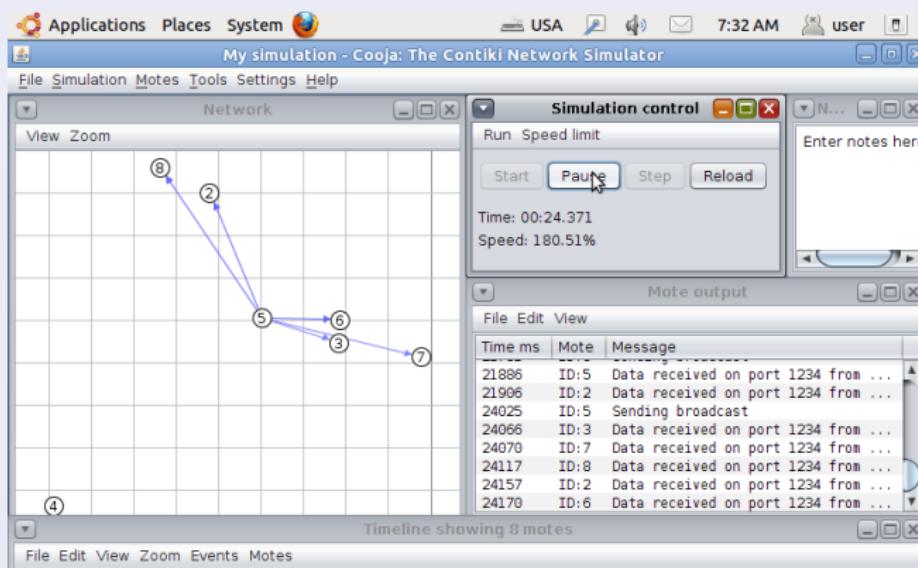
TOSSIM – TinyOS

- TOSSIM is a TinyOS library
- TinyOS code is the same for TOSSIM
- TOSSIM supports two programming interfaces: Python and C++
 - Python allows you to interact with a running simulation dynamically

Cooja – Contiki

- Cooja is the Contiki network simulator.
- Cooja allows large and small networks
- Motes can be emulated at the hardware level

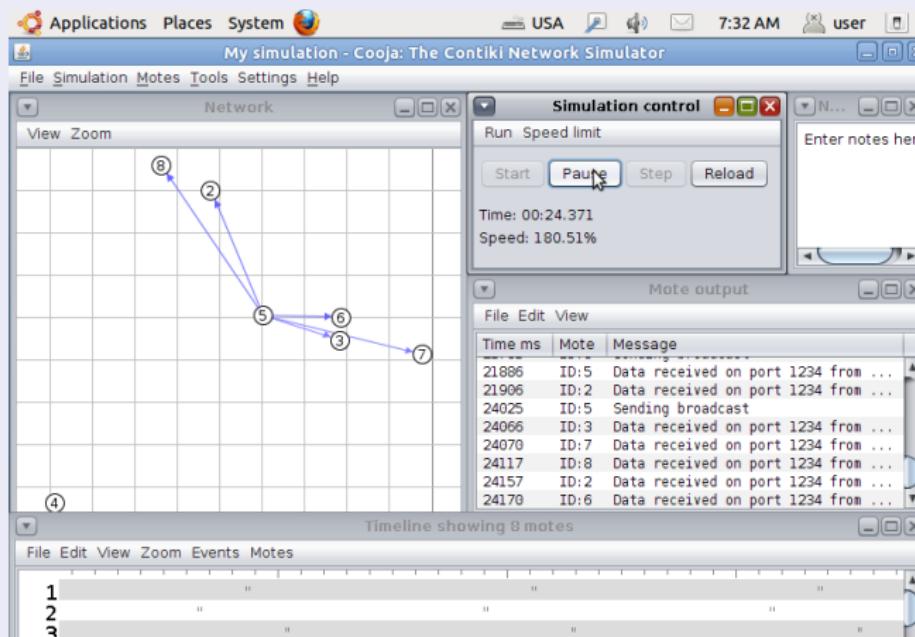
Cooja – Contiki



Cooja – Contiki

- **Minicurso - SBCR 2016**
- **Hands-on**

Cooja – Contiki



Tarefa 2

Árvore para coleta de dados

- ① Faça uma inundação para descobrir o menor caminho (em saltos) de cada nó para uma Estação Base (EB).
- ② A EB deve ser iniciada através de **@NodePopupMethod**
- ③ As mensagens podem ser do tipo rota ou dados
- ④ O nó deve identificar se a mensagem é de construção de rota ou de dados.
 - Se a mensagem é de construção de rota
 - O nó deve atualizar sua rota para a EB.
 - Se a mensagem é de dados
 - O nó deve mandar uma mensagem unicast para o próximo salto até a EB, caso exista uma rota válida

Tarefa 2

Árvore para coleta de dados

- ① A EB ao receber uma mensagem deve imprimir no Output informações sobre a mensagem coletada.
- ② Cada nó após ter um caminho válido para EB, deve ser permitido enviar dados periodicamente (10 rounds) para EB, isto deve ser ativado através de um **@NodePopupMethod**.
- ③ O payload da mensagem de dado pode ser uma amostra de temperatura ou alguma variável de ambiente geralmente analisada por RSSF.
- ④ A mensagem não pode trafegar mais que um tempo de vida estabelecido (Ex: 30 saltos)

Dúvidas sobre o projeto.

Talk is cheap

Show me the

CODE

–Linus



Git é vida!

git clone

<https://github.com/BrunoPereiraSantos/aula-simuladores-redes-sem-fio.git>