Distributed computing in ROS2

Prof. Ivano Notarnicola

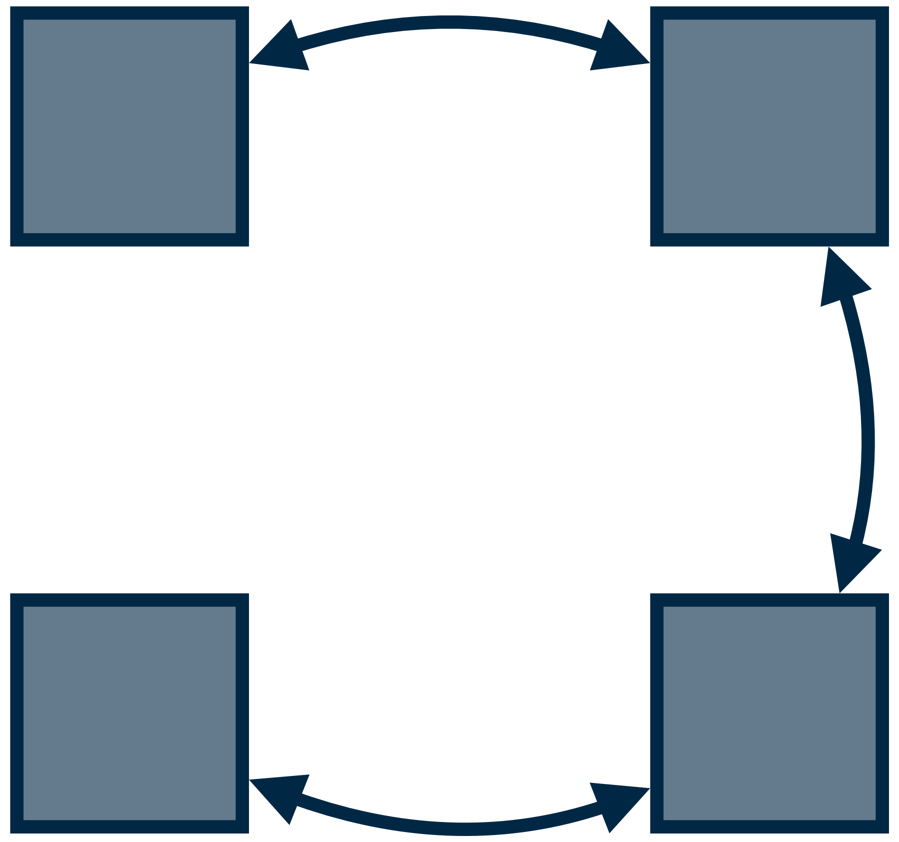
Department of Electrical, Electronic, and Information Engineering
Alma Mater Studiorum Università di Bologna

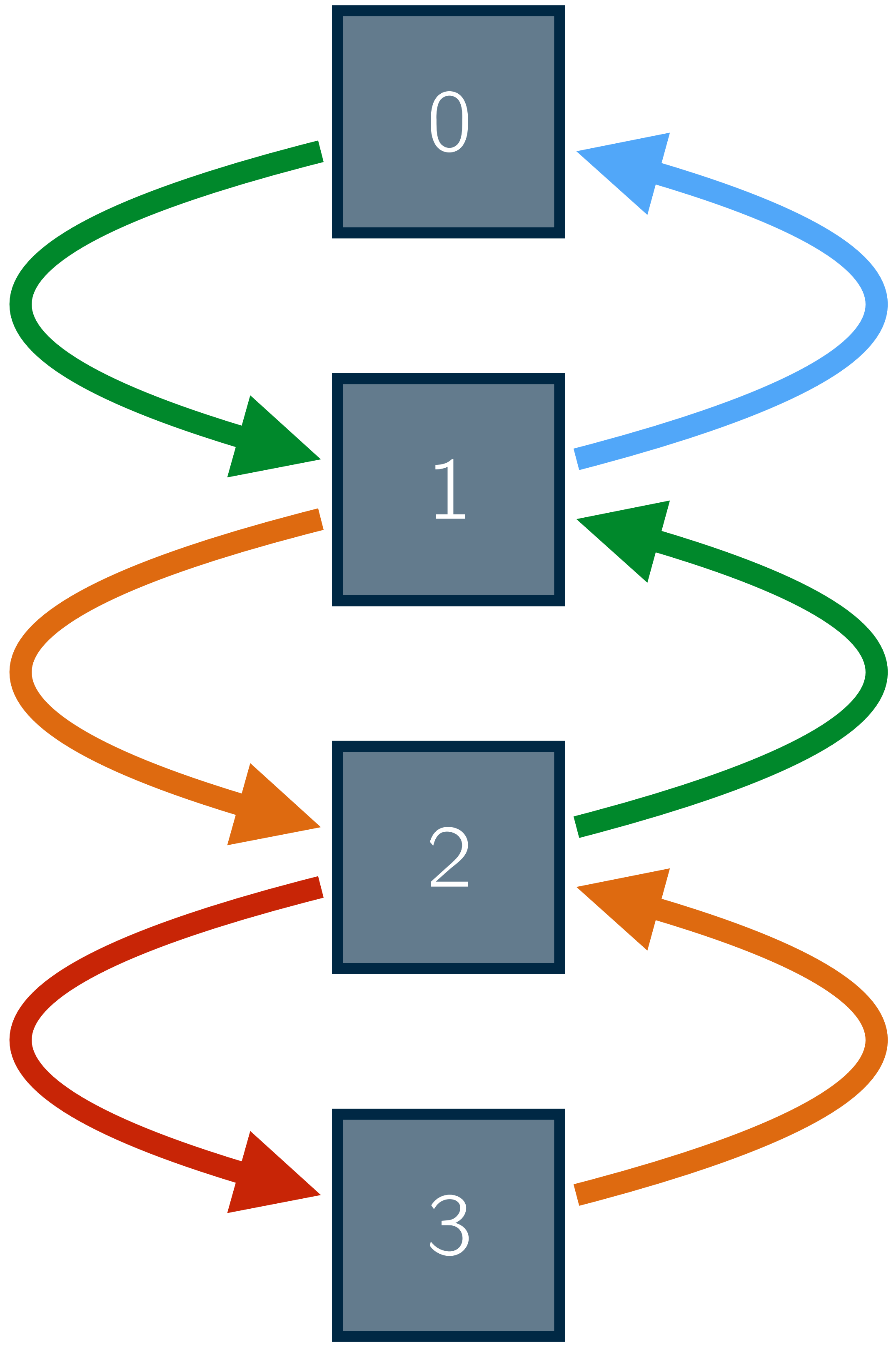
ivano.notarnicola@unibo.it

Distributed Autonomous Systems M
A.A. 2024-2025

Distributed communication: naive approach

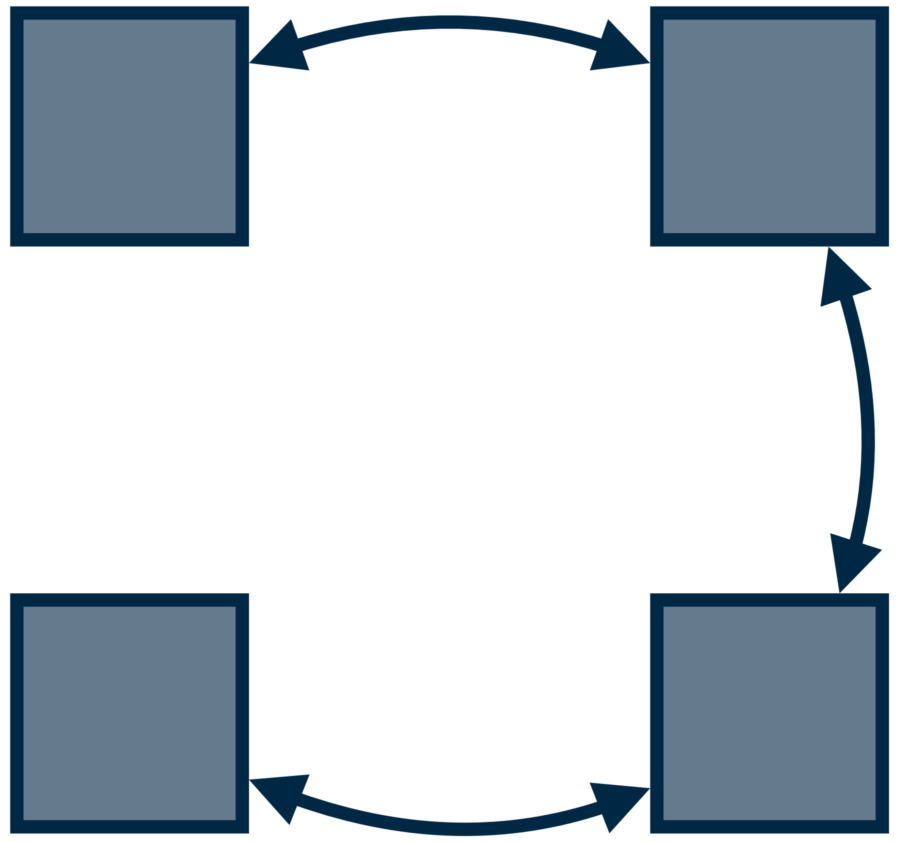
Consider an undirected ***path graph*** with $N = 4$ nodes

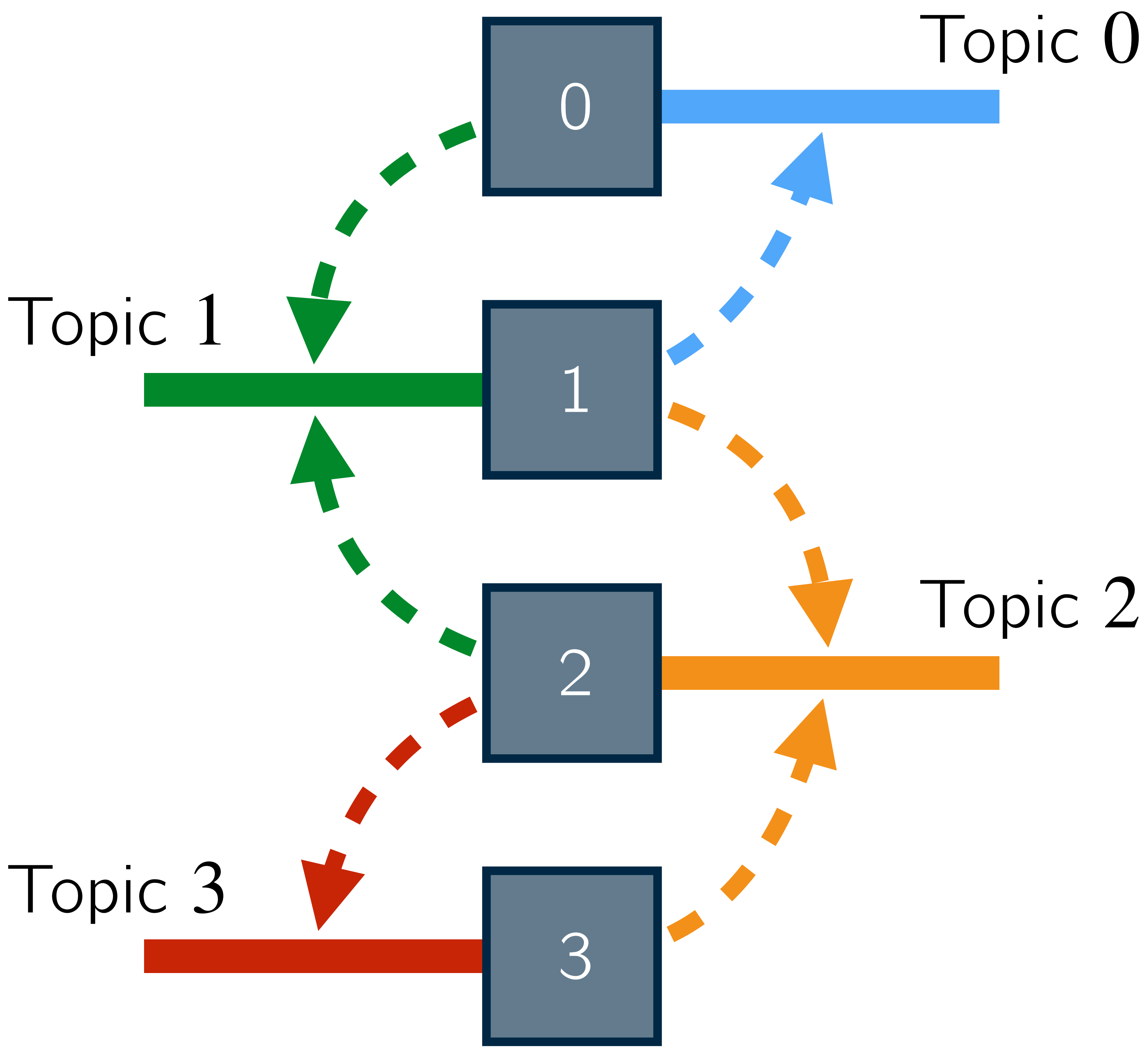




Distributed communication: efficient implementation

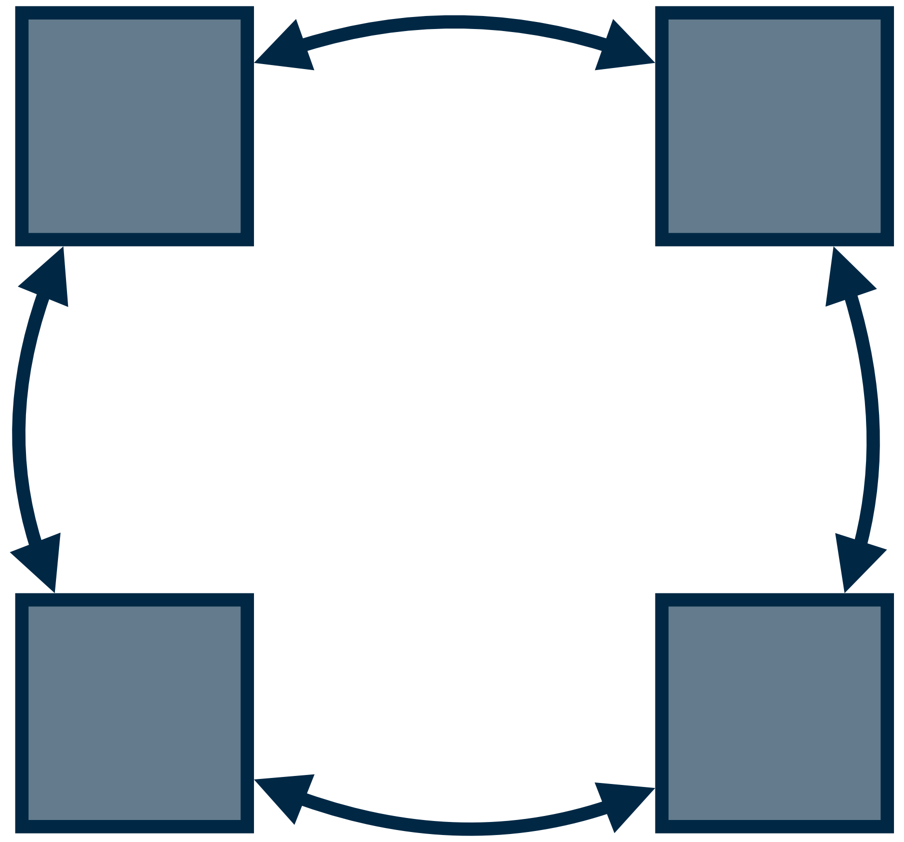
Consider an undirected ***path graph*** with $N = 4$ nodes

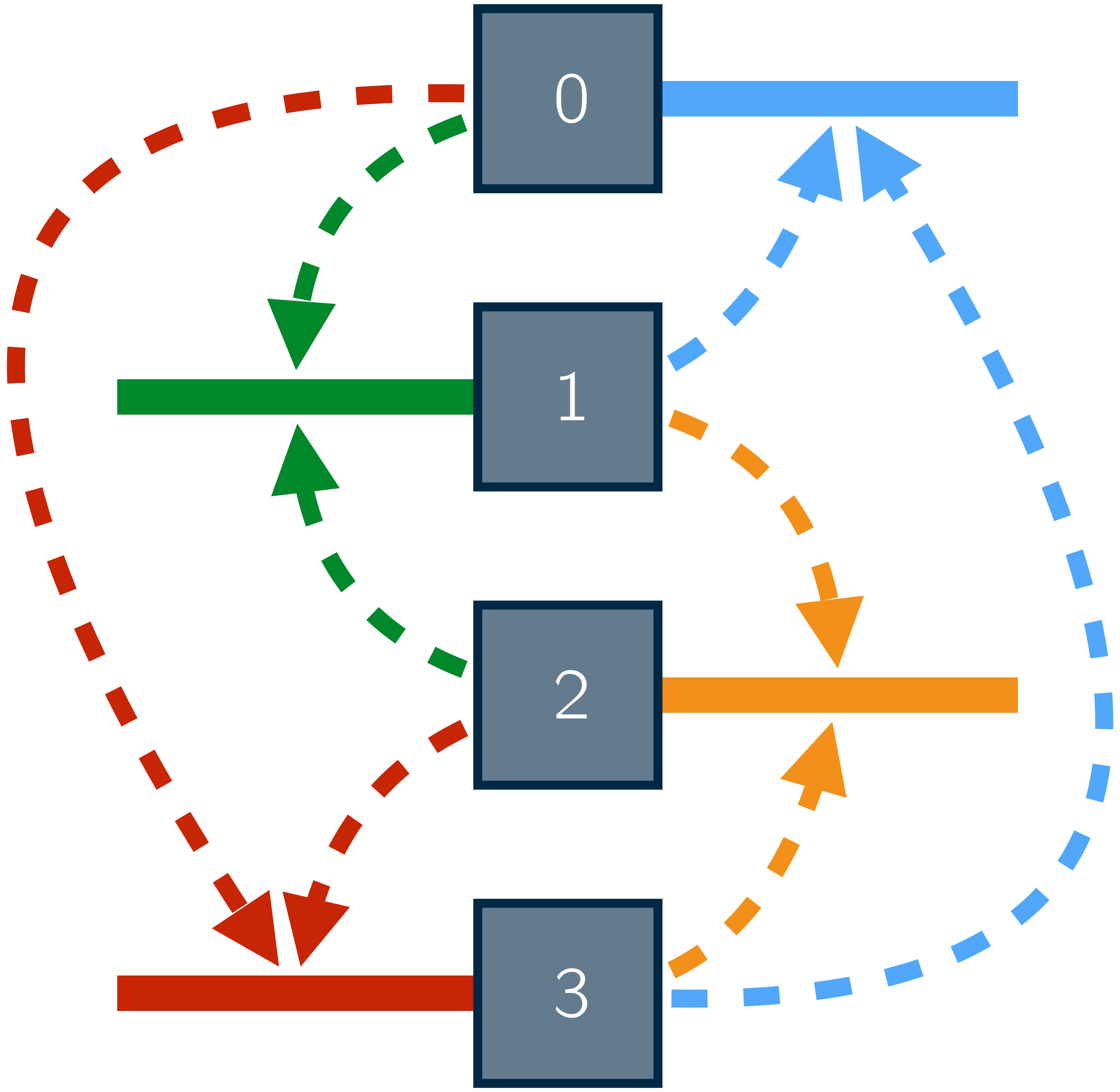




Distributed communication: efficient implementation

Consider an undirected ***cycle graph*** with $N = 4$ nodes



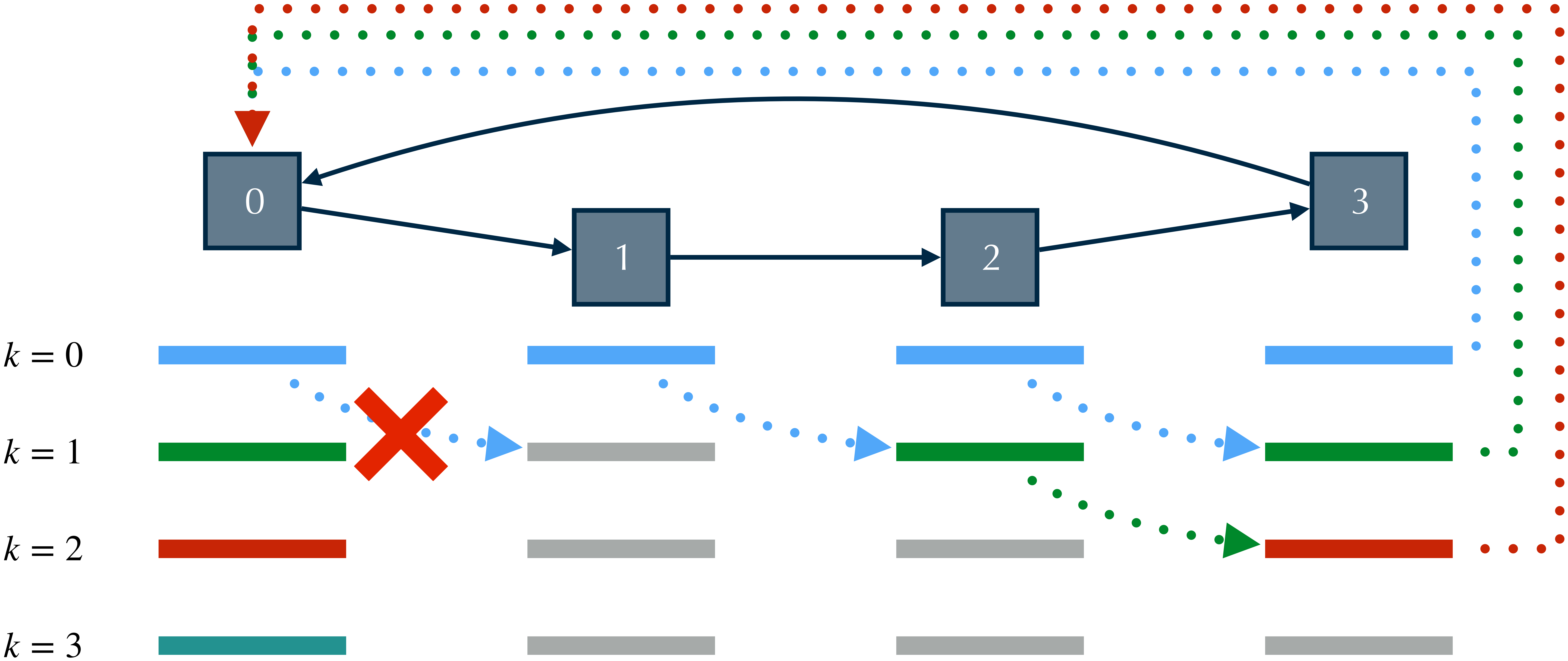


Handling multiple received messages

In a real network, ***synchronization*** issues may arise

Each node i may receive several messages (the exact number depends on the graph structure) from the same neighbor before having received “synchronous” messages from all the remaining neighbors

Synchronization issues in a distributed protocol



Synchronization issues in a distributed protocol

Each node i must simultaneously act as

- a ***publisher***, and
- a ***subscriber*** (to the topic of each neighbor)

The messages to be exchanged should contain the information about the sender and the iteration, i.e.,

$$\text{msg} = [i, k, x_i]$$

Received messages are collected in neighbor-specific buffer (a FIFO queue, namely a **list**)

The new message at time k is computed (and sent) only when the states x_j^{k-1} ***for all*** $j \in N_i$ have been received

Workspace preparation

Activate ROS2

. /opt/ros/humble/setup.bash

Create a new directory that will contain the ROS2 workspace

mkdir -p distributed_ros2_ws/src
cd distributed_ros2_ws/src

Create a package called **distributed_algs** from the **src** directory using

ros2 pkg create --build-type ament_python distributed_algs

Package configuration

Add dependencies in **package.xml**

<exec_depend>rclpy</exec_depend>

<exec_depend>std_msgs</exec_depend>
<exec_depend>ros2launch</exec_depend>

Edit the **setup.py** to specify the launch file **max_launch.py** as

(i) include the header "**from** glob **import** glob" and to the **data_files** list:
("share/" + package_name, glob("launch_folder/max_launch.py"))

(ii) specify the entry points, i.e., the name of the ROS2 node associated to the source file **the_agent.py**
"generic_agent = distributed_algs.the_agent:main"

Package build and run

Include the (single) source file **the_agent.py** of the ROS2 node, which is to be located at
distributed_ros2_ws/src/distributed_algs/distributed_algs

From the ROS2 workspace root **distributed_ros2_ws** build the package
colcon build --symlink-install --packages-select distributed_algs

Then

- activate ROS2 (if needed)
  . /opt/ros/humble/setup.bash
- run
  . install/setup.bash
- execute the launch file
  ros2 launch distributed_algs max_launch.py