Net Neutrality and Quality of Service (QoS) + Python Data Structures

CSCI 77800 – Ethics and Computer Science

Session 3 – 9/12/2024

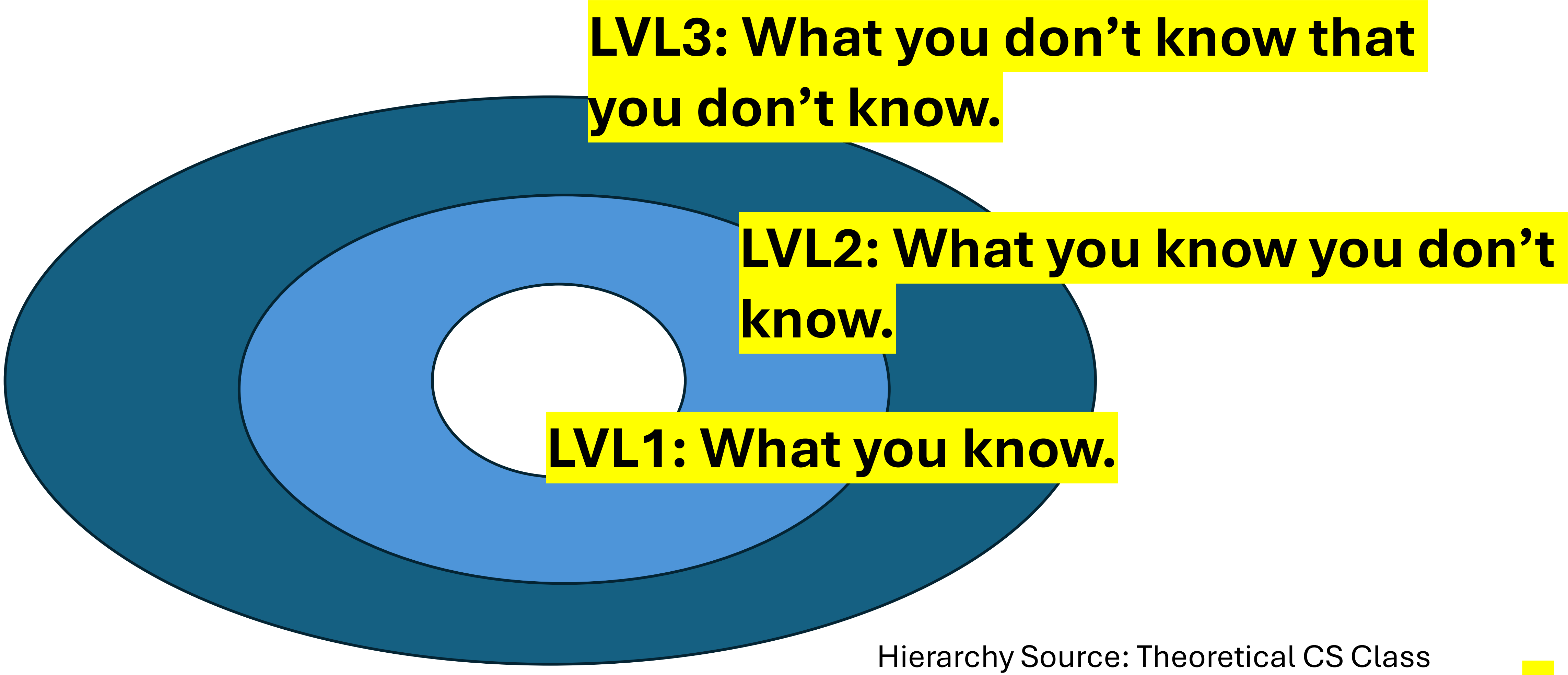
Edgar E. Troudt, Ph.D. <et3076@hunter.cuny.edu>

Finish Up from Last Week / DNS

- Examine second code block, including library aliasing.
- Note the **#!/usr/bin/env python3** for direct execution.
- **! git clone** may not work
- **%load** should.

- Vocabulary:
  - DNS,
  - IP address,
  - Python virtual environment.
- Conceptual: how does ad blocking DNS work?

Hierarchy of Knowledge



LVL3: What you don’t know that you don’t know.

LVL2: What you know you don’t know.

LVL1: What you know.

Hierarchy Source: Theoretical CS Class
at the CUNY Graduate Center

Have you thought about the cellular market?

- **Tower owners –**
  - e.g., American Tower, Crown Castle.
- **Network operators which own/lease network towers** –
  - e.g., Verizon, T-Mobile, AT&T, Boost Mobile, Uscellular
  - Source: https://en.wikipedia.org/wiki/List_of_mobile_network_operators_in_the_United_States
- **Mobile Virtual Network Operators (MVNOs) –**
  - e.g., Mint Mobile, Google Fi, Metro by T-Mobile.
  - https://en.wikipedia.org/wiki/List_of_mobile_virtual_network_operators_in_the_United_States

- What are MVNOs actually selling to you? Why do the major carriers have MVNOs?
- Is unlimited data unlimited?
- **Optional reading:**
  - https://www.verizon.com/about/news/verizon-vertical-bridge-expansion-cell-towers
  - https://www.steelintheair.com/cell-tower-companies/

Are all data sent at the same speed?

- Often not. You started to learn about Net Neutrality.

- **QoS** = quality of service.
- **QCI** = quality of service class indicator.
- **Brief pair-wise discussion:**
  - Skim this: https://www.reddit.com/r/NoContract/comments/1cyfjpp/data_prioritization_policies_of_the_carriers_and/
  - Remember: the lower the number the faster.
  - Find an interesting fact to share with the class.
  - Overtime: look at the MVNO list linked from the previous slide.

- **Optional reading:** https://en.wikipedia.org/wiki/Quality_of_service

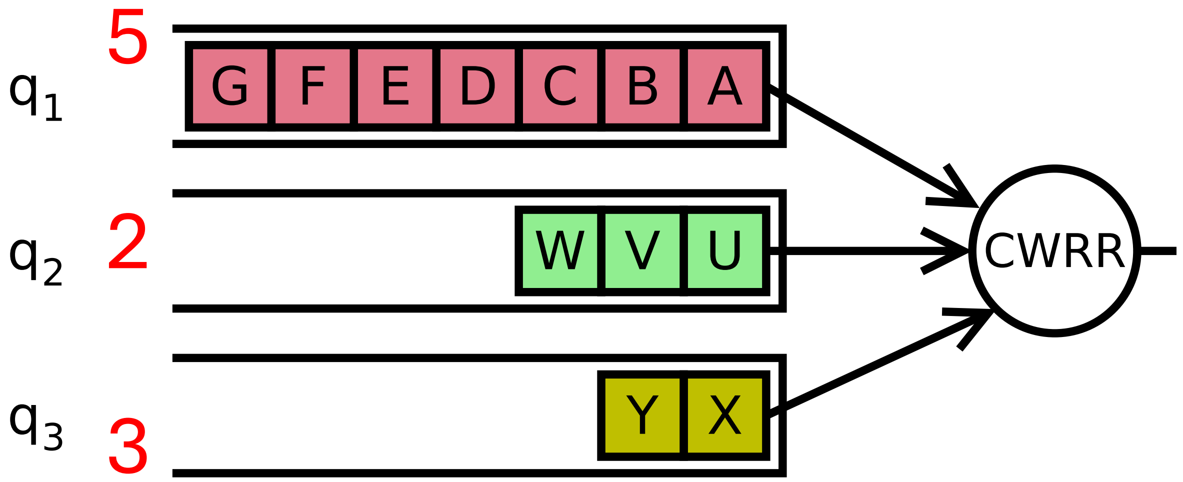
What is a packet?

- A box for your internet message. May contain:

- Source/Destination Addresses.
- Error detector (e.g., checksum).
- Hop limit.
- Length.
- Protocol.
- Priority.
- Data.

- **List adapted from:** https://en.wikipedia.org/wiki/Network_packet

Classic Weighted Round Robin



INSTRUCTIONS

1. Go to each queue.
2. If elements present, send a number of elements up to the weight of the queue.
3. Repeat 1 and 2.

Algorithm source: https://en.wikipedia.org/wiki/Weighted_round_robin

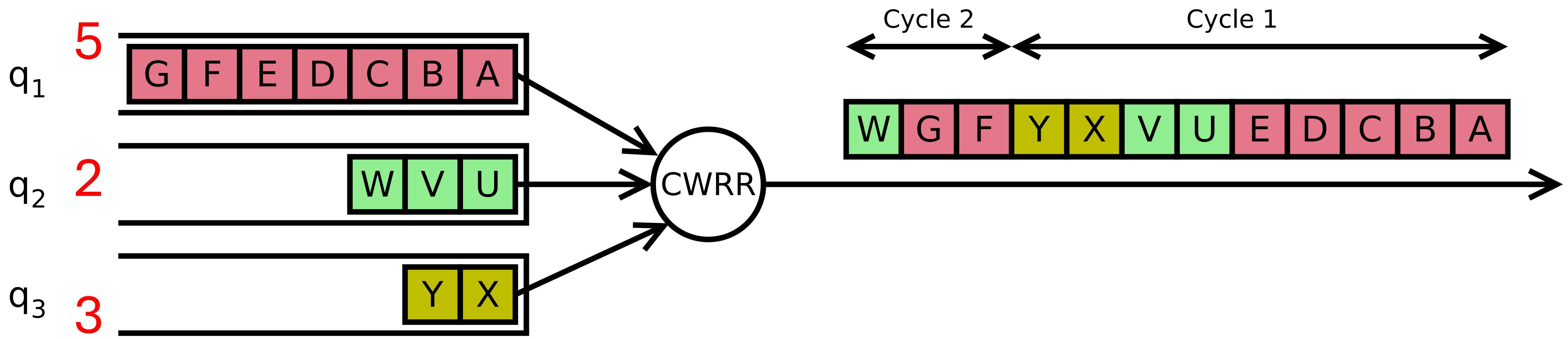
Source: https://commons.wikimedia.org/wiki/File:WRR-Examples.svg#/media/File:WRR-Examples.svg
Author: MarcBoyerONERA - Own work
License: https://creativecommons.org/licenses/by-sa/4.0/
Modification: Image with two algorithms separated over two slides. Also added weight numbers. Right side result hidden for this slide.

Pair-wise exercise 1: Show the output of CWRR on the previous slide.

HINT: Don’t look at the answer on the next slide until you’ve finished.

OVERTIME QUESTION: Do you like this algorithm? Does this seem fair?

Classic Weighted Round Robin



Algorithm source: https://en.wikipedia.org/wiki/Weighted_round_robin

Source: https://commons.wikimedia.org/wiki/File:WRR-Examples.svg#/media/File:WRR-Examples.svg

Author: MarcBoyerONERA - Own work

Modification: Image with two algorithms separated over two slides. Also added weight numbers.

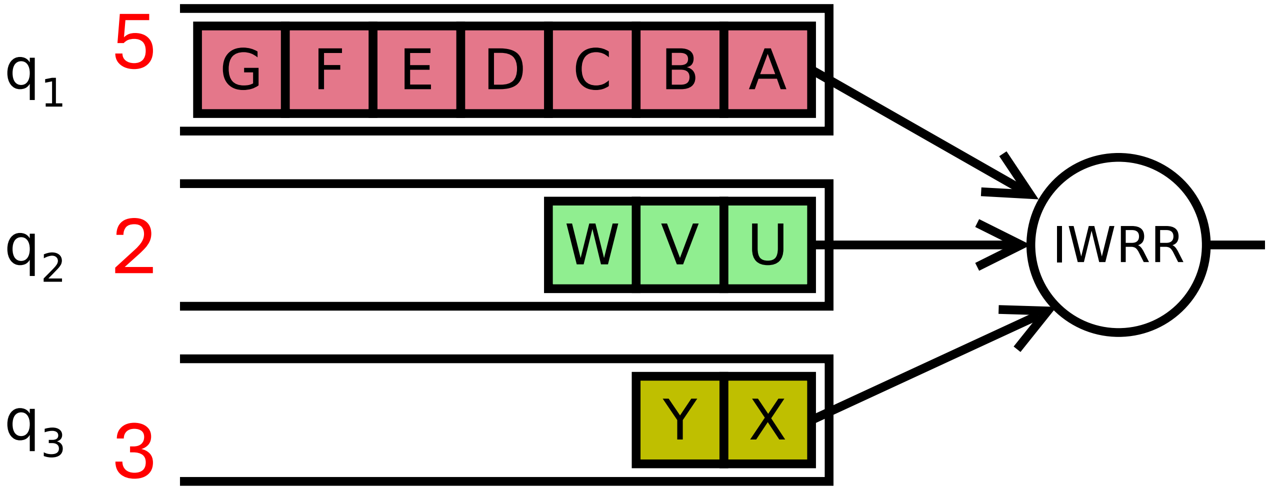
Classic Weighted Round Robin

- **View original pseudocode:** https://en.wikipedia.org/wiki/Weighted_round_robin

- **Edgar’s demos:**

1. Packet.py
2. PacketTest.py
3. cwrr.py

Interleaved Weighted Round Robin



INSTRUCTIONS

1. Find the maximum weight.
2. Rotate across each of the queues. Keep track of what “round” this is.
3. At the queue, if the current round is less than the queue’s weight, remove one element.
4. Move to next queue.
5. Repeat 2, 3, and 4.

Algorithm source: https://en.wikipedia.org/wiki/Weighted_round_robin

Source: https://commons.wikimedia.org/wiki/File:WRR-Examples.svg#/media/File:WRR-Examples.svg

Author: MarcBoyerONERA - Own work

License: https://creativecommons.org/licenses/by-sa/4.0/

Modification: Image with two algorithms separated over two slides. Also added weight numbers. Result on the right side hidden for this slide.

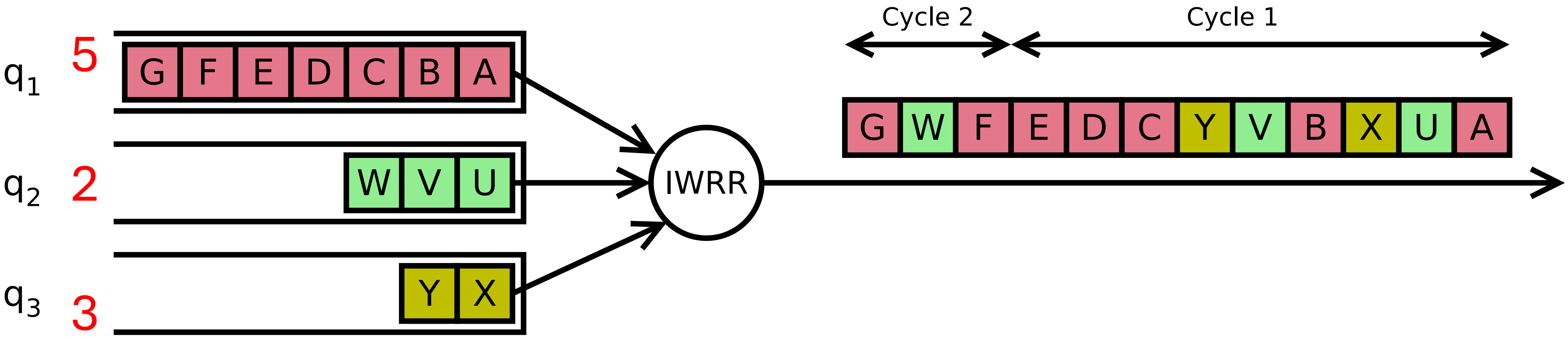
Pair-wise exercise 2: Show the output of IWRR on previous slide.

HINT: Don’t look at the answer on the next slide until you’ve finished.

OVERTIME QUESTION: Is this algorithm more fair than the previous? Could anything about a packet break this algorithm?

OVERTIME #2: Take the CWRR code (cwrr.py) from our repo. Try to convert it to iwrr. The solution to IWRR and the pseudo code for IWRR exist on the next two slides.

Interleaved Weighted Round Robin



Algorithm source: https://en.wikipedia.org/wiki/Weighted_round_robin

Source: https://commons.wikimedia.org/wiki/File:WRR-Examples.svg#/media/File:WRR-Examples.svg
Author: MarcBoyerONERA - Own work
License: https://creativecommons.org/licenses/by-sa/4.0/
Modification: Image with two algorithms separated over two slides. Also added weight numbers.

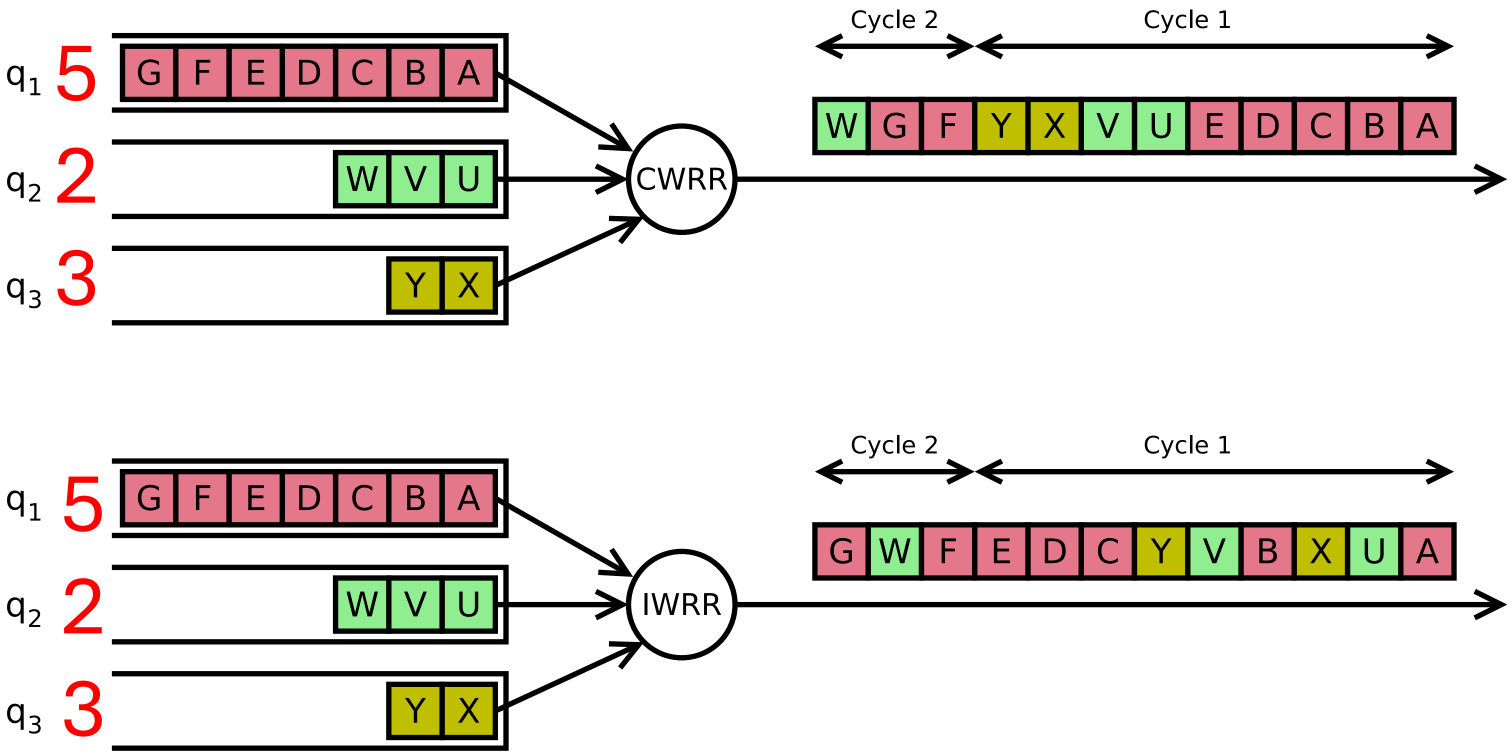
Interleaved Weighted Round Robin

- **See original pseudocode:** https://en.wikipedia.org/wiki/Weighted_round_robin

- **Edgar’s demos:**

1. iwrr.py

Both Round Robin Algorithms



Algorithm source: https://en.wikipedia.org/wiki/Weighted_round_robin

Source: https://commons.wikimedia.org/wiki/File:WRR-Examples.svg#/media/File:WRR-Examples.svg

Author: MarcBoyerONERA - Own work

Modification: . Added weight numbers.

Are all packets the same size?

What breaks the fairness of IWRR?

<table>
<tr><th></th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td>Q1:W=5</td><td></td><td></td><td></td><td colspan="3">E</td><td>D</td><td>C</td><td>B</td><td>A</td></tr>
<tr><td>Q2:W=2</td><td></td><td></td><td></td><td>W</td><td colspan="3">V</td><td colspan="3">U</td></tr>
<tr><td>Q3:W=3</td><td></td><td></td><td></td><td></td><td>Z</td><td colspan="3">Y</td><td colspan="2">X</td></tr>
</table>

Edgar’s demos:

1. iwrr2.py

Algorithm Source: https://en.wikipedia.org/wiki/Deficit_round_robin

Deficit Round Robin

	6	5	4	3	2	1	0
1:=5	E			D	C	B	A
2:=2	W	V			U		
3:=3		Z	Y			X	

Recognize that packets may have different sizes. Use bytes not packets.

1. **Allowed_Bytes:** Keep a separate counter of how many bytes are allowed this round. It starts at zero.
2. Add the weight of each queue to the **allowed_bytes**.
3. Visit each queue.
4. If queue not empty and the size of the first packet is less than the remaining allowed bytes, send packet.
5. Decrease the remaining allowed_bytes.
6. Repeat 2-6.

Algorithm Source: https://en.wikipedia.org/wiki/Deficit_round_robin

Deficit Round Robin

- See original pseudocode: https://en.wikipedia.org/wiki/Deficit_round_robin

- Edgar's demo:
  - **dwrr.py**

Python Data Structures

- Set: { .. , .. , .. }
  - [unordered, no duplicates, mutable/changeable]
- Dictionaries { key:item, key:item …}
  - [unordered, no duplicates, mutable/changeable]
- List: [.., .., ..] – use this instead of arrays, addressable with [subscript].
  - [ordered, mutable/changeable]
- Tuples: (.., .., ..)
  - [ordered, immutable]
  - Show **tupleTest.py**

Reference and optional reading: https://www.educative.io/answers/list-vs-tuple-vs-set-vs-dictionary-in-python

More on Python Data Structures

- Visit: https://docs.python.org/3/library/stdtypes.html#typesseq
- Examine the common operators, including slicing.

- Visit: https://docs.python.org/3/library/stdtypes.html#set-types-set-frozenset
- Examine operations available.

Pair-wise Breakout

Example 1:

Write a program that asks a user for an animal and prints their sound.

pig	→	oink
cat	→	meow
dog	→	woof
sheep	→	baa

- Use a dictionary.

Example 2:

During the runtime of the program, read in an unlimited number of words from the user until “STOP” is indicated. Then print the words out in sorted order.

Use lists.

The Python lists reference to help you: https://docs.python.org/3/tutorial/datastructures.html.

Net Neutrality and Quality of Service (QoS) + Python Data Structures

CSCI 77800 – Ethics and Computer Science

Session 3 – 9/12/2024

Edgar E. Troudt, Ph.D. <et3076@hunter.cuny.edu>