**CS249 - Homework 5b – Maximal Consistent Cut**

The program runs a sample execution plan for three processors P0, P1, and P2. The corresponding vector clocks are calculated and then maximal consistent cut is computed for given inconsistent cut.

**Instructions to run:**

$ git clone https://github.com/Mayuri-Wad-012447851/CS-249-Distributed-Computing

Above command will download entire repo with all the previous homework.

**Homework5b folder: HW5b-Maximal-consistent-cut**

Instructions to run the program from command prompt:

1. On command prompt, change directory to location of file (cloning location).

2. To compile and run, issue following commands:

Example command 1:

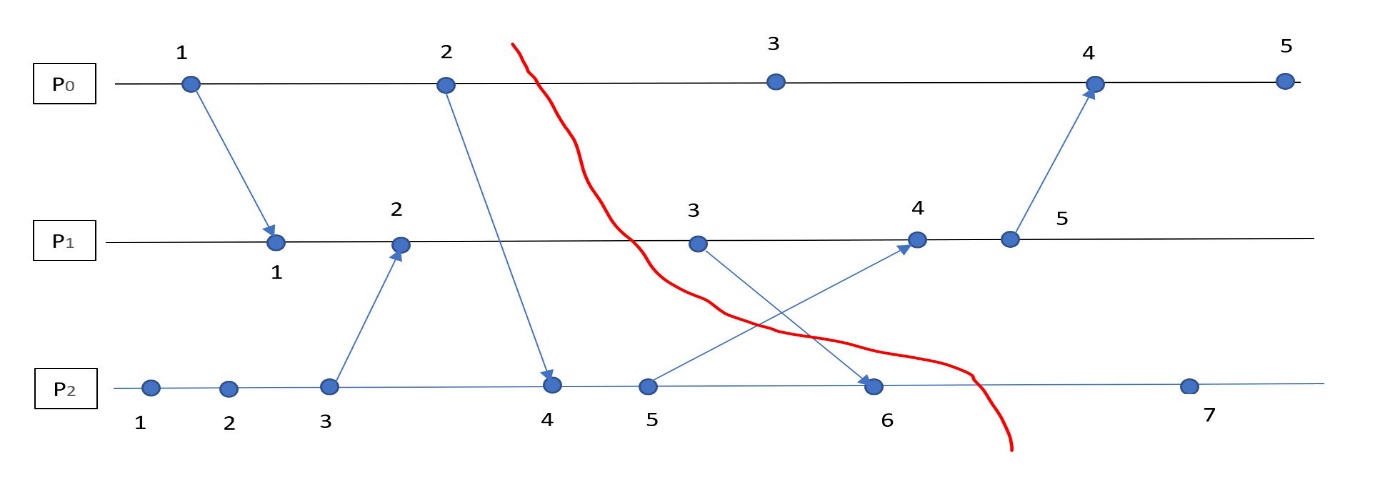
$..\CS-249-Distributed-Computing\HW5b-Maximal-consistent-cut\src>"C:\Program Files\Java\jdk1.8.0\_121\bin\javac" \*.java

Example command 2:

$..\CS-249-Distributed-Computing\HW5b-Maximal-consistent-cut\src >"C:\Program Files\Java\jdk1.8.0\_121\bin\java" Algorithm

**Input:**

1. List of events of a processor along with their vector clock values
2. Inconsistent Cut {2,2,6}



**Output of Program:**

VectorClock before compute event at P2

0 0 0

COMPUTING at P2

VectorClock after compute event at P2

0 0 1

P0 SEND to P1

VectorClock before SEND event at P0

0 0 0

VectorClock after SEND event at P0

1 0 0

VectorClock before compute event at P2

0 0 1

COMPUTING at P2

VectorClock after compute event at P2

0 0 2

VectorClock before RECEIVE event at P1

0 0 0

VectorClock after RECEIVE event at P1

1 1 0

P2 SEND to P1

VectorClock before SEND event at P2

0 0 2

VectorClock after SEND event at P2

0 0 3

VectorClock before RECEIVE event at P1

1 1 0

VectorClock after RECEIVE event at P1

1 2 3

P0 SEND to P2

VectorClock before SEND event at P0

1 0 0

VectorClock after SEND event at P0

2 0 0

VectorClock before RECEIVE event at P2

0 0 3

VectorClock after RECEIVE event at P2

2 0 4

P2 SEND to P1

VectorClock before SEND event at P2

2 0 4

VectorClock after SEND event at P2

2 0 5

P1 SEND to P2

VectorClock before SEND event at P1

1 2 3

VectorClock after SEND event at P1

1 3 3

VectorClock before compute event at P0

2 0 0

COMPUTING at P0

VectorClock after compute event at P0

3 0 0

VectorClock before RECEIVE event at P2

2 0 5

VectorClock after RECEIVE event at P2

2 3 6

VectorClock before RECEIVE event at P1

1 3 3

VectorClock after RECEIVE event at P1

2 4 6

P1 SEND to P0

VectorClock before SEND event at P1

2 4 6

VectorClock after SEND event at P1

2 5 6

VectorClock before RECEIVE event at P0

3 0 0

VectorClock after RECEIVE event at P0

4 5 6

VectorClock before compute event at P2

2 3 6

COMPUTING at P2

VectorClock after compute event at P2

2 3 7

VectorClock before compute event at P0

4 5 6

COMPUTING at P0

VectorClock after compute event at P0

5 5 6

Event count at p0: 5

Vector Clock at Processor P0: [5 5 6 ]

Event count at p1: 5

Vector Clock at Processor P1: [2 5 6 ]

Event count at p2: 7

Vector Clock at Processor P2: [2 3 7 ]

Computing maximal consistent cut at {2,2,6}..

Events and VCs at P0:

VectorClock [VC=[1, 0, 0]]SEND

VectorClock [VC=[2, 0, 0]]SEND

VectorClock [VC=[3, 0, 0]]COMPUTE

VectorClock [VC=[4, 5, 6]]RECEIVE

VectorClock [VC=[5, 5, 6]]COMPUTE

Events and VCs at P1:

VectorClock [VC=[1, 1, 0]]RECEIVE

VectorClock [VC=[1, 2, 3]]RECEIVE

VectorClock [VC=[1, 3, 3]]SEND

VectorClock [VC=[2, 4, 6]]RECEIVE

VectorClock [VC=[2, 5, 6]]SEND

Events and VCs at P2:

VectorClock [VC=[0, 0, 1]]COMPUTE

VectorClock [VC=[0, 0, 2]]COMPUTE

VectorClock [VC=[0, 0, 3]]SEND

VectorClock [VC=[2, 0, 4]]RECEIVE

VectorClock [VC=[2, 0, 5]]SEND

VectorClock [VC=[2, 3, 6]]RECEIVE

VectorClock [VC=[2, 3, 7]]COMPUTE

**Maximum consistent cut:**

**2 2 5**

Green line represents Maximal Consistent Cut.

