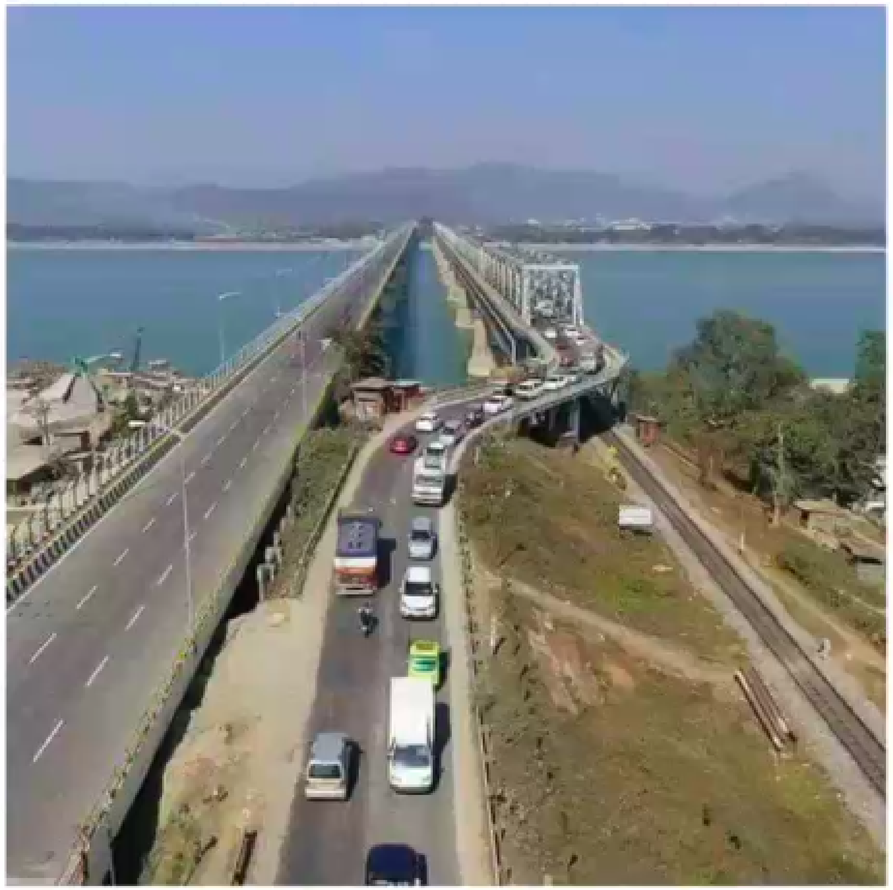
Parallel case …

- Thus memory system is a point in which hardware determines the order of access
- There is not fixed sequence of access among the processes
- There is arbitrary interleaving
- Fairness -> as each process will eventually access memory
- Our understanding of “last” or “subsequent” access will be defined in this hypothetical serial order
- As the serial order must be consistent, the processes see the writes to a location in the same order

Parallel case ...

- Thus memory system is a point in which hardware determines the order of access
- There is not fixed sequence of access among the processes
- There is arbitrary interleaving
- Fairness -> as each process will eventually access memory
- Our understanding of “last” or “subsequent” access will be defined in this hypothetical serial order
- As the serial order must be consistent, the processes see the writes to a location in the same order

Merging of cars
=
Merging of mem accesses



What happens in practice?

- In practice we do not want to construct this serial order. In the presence of caches, ordering is varied
- We just need to make sure that the program behaves as if some serial order was enforced
- *FORMALLY*: Multiprocessor system is coherent if the results of any execution of a program are such that, for each location, it is possible to construct a hypothetical serial order of all operations to that location, that is consistent with the results of execution and in which
  - (1) operations issued by a process occur in its program order, AND
  - (2) value returned by read operation is the value written by the last write to that location in the serial order

Defining Correctness Metrics

Definition: Coherence

- Informally, we could say that a memory system is coherent if any read of a data item returns the most recently written value of that data item
- Easy in uniprocessor. But too vague and simplistic => involves 2 aspects of memory system behaviour
- Coherence and Consistency

COHERENCE	CONSISTENCY
(1) Defines what values can be returned by a read	(1) Determines when a written value will be returned by a read
(2) Defines behaviour of same location	(2) Defines behaviour to other locations

Defining Coherent Memory System

3 conditions:

(1) Preserve Program Order

(2) Coherent View of memory

(3) Write Serialisation