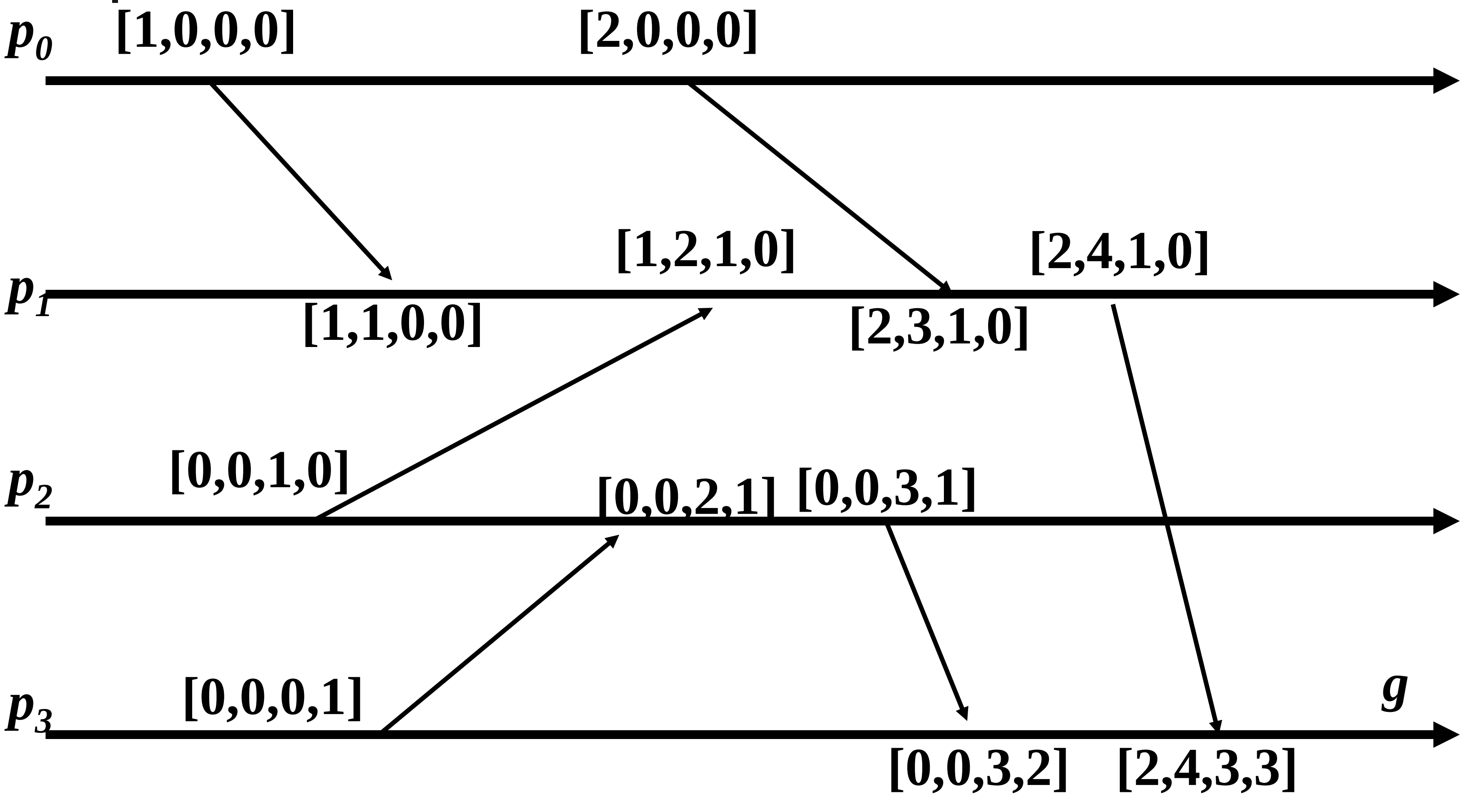
Vector timestamps
(auxiliary material from optional text [Bir05])

- Extend logical timestamps into a list of counters, one per process in the system
- Again, each process keeps its own copy
- Event e occurs at process p*:*
  p increments VT(p)[p]
  (p’th entry in its own vector clock)
- q receives a message from p:
  q sets VT(q)=max(VT(q),VT(p))
  (element-by-element)

Illustration of vector timestamps [DAVE CHANGED]



Vector timestamps accurately represent the happens-before relationship!

- Define VT(e)<VT(e’) if,
  - for all i, $VT(e)[i] \leq VT(e')[i]$, and
  - for some j, $VT(e)[j] < VT(e')[j]$
- Example: if VT(e)=[2,1,1,0] and VT(e’)=[2,3,1,0] then VT(e)<VT(e’)
- Notice that not all VT’s are “comparable” under this rule: consider [4,0,0,0] and [0,0,0,4]

Vector timestamps accurately represent the happens-before relationship!

- Now can show that VT(e)<VT(e') if and only if $e \rightarrow e'$:
  - If $e \rightarrow e'$, there exists a chain $e_0 \rightarrow e_1 \ldots \rightarrow e_n$ on which vector timestamps increase "hop by hop"
  - If VT(e)<VT(e') suffices to look at VT(e')[proc(e)], where proc(e) is the place that e occured. By definition, we know that VT(e')[proc(e)] is at least as large as VT(e)[proc(e)], and by construction, this implies a chain of events from e to e'

Examples of VT's and happens-before

- Example: suppose that VT(e)=[2,1,0,1] and VT(e')=[2,3,0,1], so VT(e)<VT(e')
- How did e' "learn" about the 3 and the 1?
  - Either these events occured at the same place as e', or
  - Some chain of send/receive events carried the values!
- If VT's are not comparable, the corresponding events are concurrent!

Notice that vector timestamps require a static notion of system membership

- For vector to make sense, must agree on the number of entries
- Later will see that vector timestamps are useful within groups of processes
- Will also find ways to compress them and to deal with dynamic group membership changes