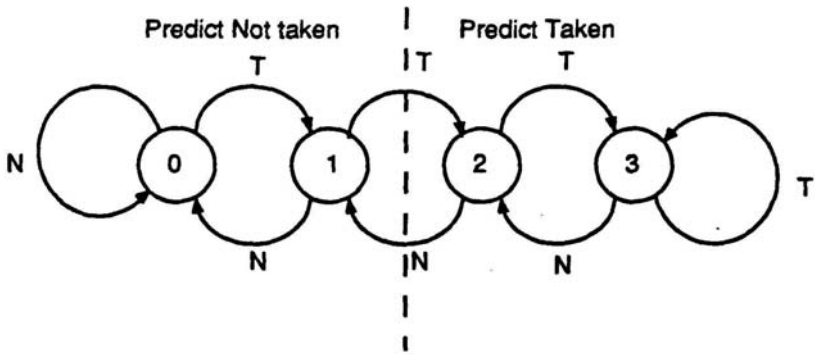
2-bit branch prediction counter



Question:

The above counter requires 2 bits per branch in a branch history table.
Suppose you may keep as much state (bits) as you want, devise a general scheme to improve branch prediction accuracy. Use the code sequence and branch history trace below to devlop your answer. Show specifically how your scheme would be an improvement over the 2-bit counter prediction accuracy for branch b3.

```
b1:   if (aa == 2)
          aa = 0;

b2:   if (bb == 2)
          bb = 0;

b3:   if (aa != bb)
      {
          .....
      }
```

Time →

b1:	T	N	N	N	N	T	T	N	T	T	T	T
b2:	N	N	T	T	N	T	T	T	T	T	N	N
b3:	T	T	N	T	T	N	N	T	N	N	N	N

Answer:

There are many possible answers. The best approach is to realize that the behavior of branch b3 is correlated with the behaviors of branches b1 and b2. And that in general branch behavior may be corelated with previous branches. Thus, by keeping track of whether the last n branches were taken or not taken and by associating a 2 -bit counter with each combination of branch outcomes we may significantly increase branch prediction accuracy. Really good answers showed how the branch history trace for b3 can be broken into four cases using the branch histories of b1 and b2. The best answers showed how you would implement this scheme in a branch history table.