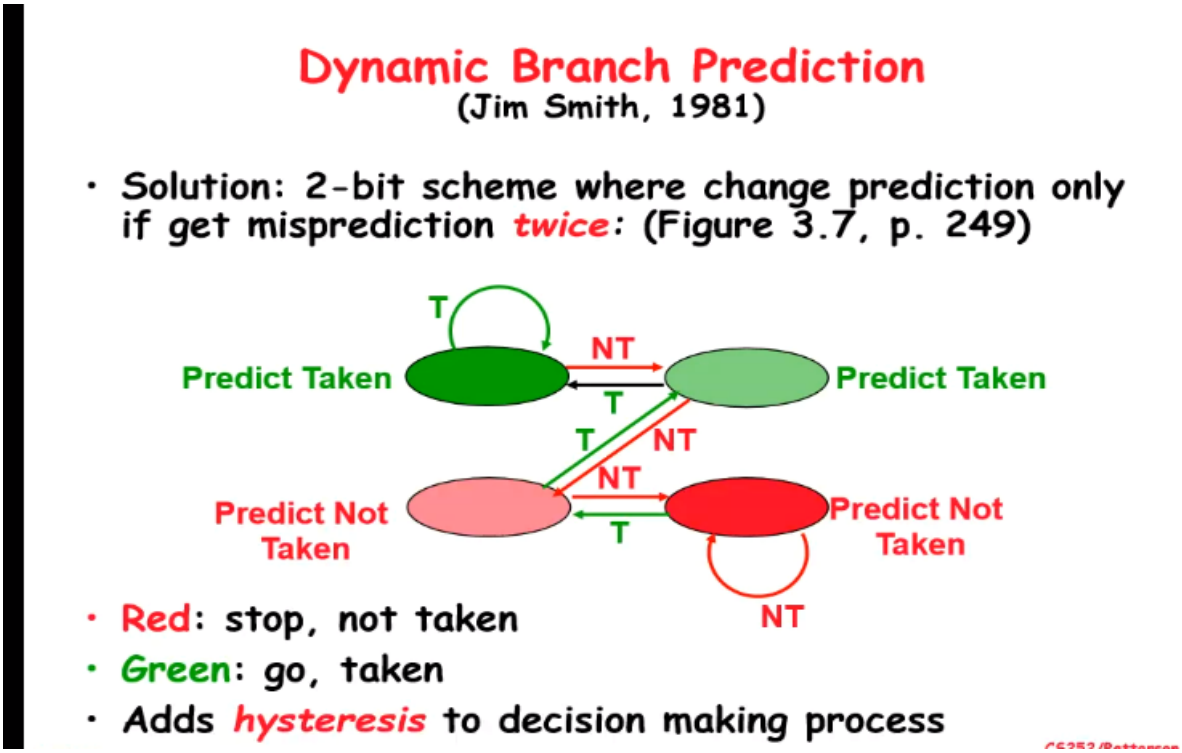
11.16 Notes

1. Reviewing Tomasulo algorithm

1. Reservation Station: rename to larger set of registers
    - prevents registers as bottleneck(critical)
    - avoid WAR, WAW of scoreboard
    - allow loop unrolling
2. Beyond Branches
3. Help cache miss(we don't stop upon cache miss)
4. Key idea
    - Dynamic scheduling
    - Register renaming (ROB id or physical register id)
    - Load/store disambiguation

2. Dynamic Branch Prediction



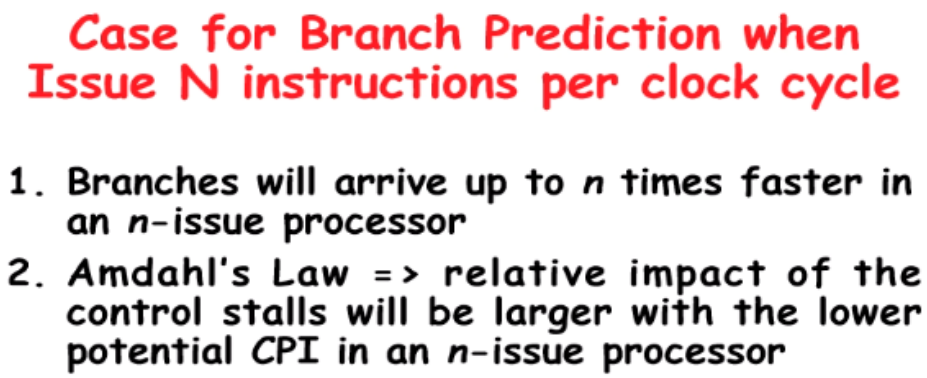
In summary, dynamic branch prediction helps us to add hysteresis and robustness to branch prediction. (Think about a special case to show this?)

3. Tomasulo algorithm and branch prediction

Pipeline prediction: it predict branches without speculation, leading to the case where pipeline stops until the branch was totally resolved.

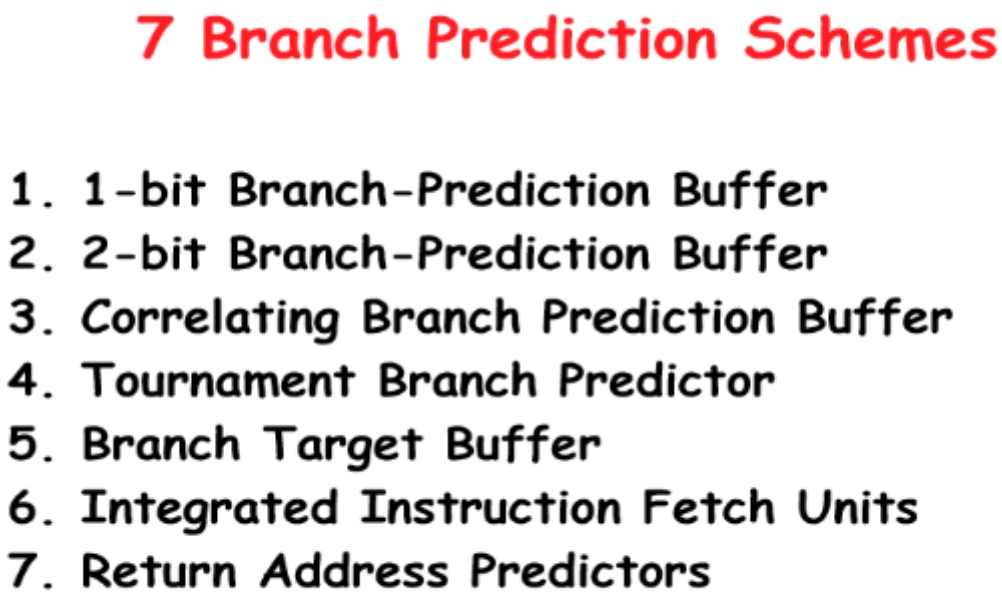
Speculation with ROB: we can easily discard wrong prediction results without flushing the registers, because we commit in an order.

4. Significance of branch prediction



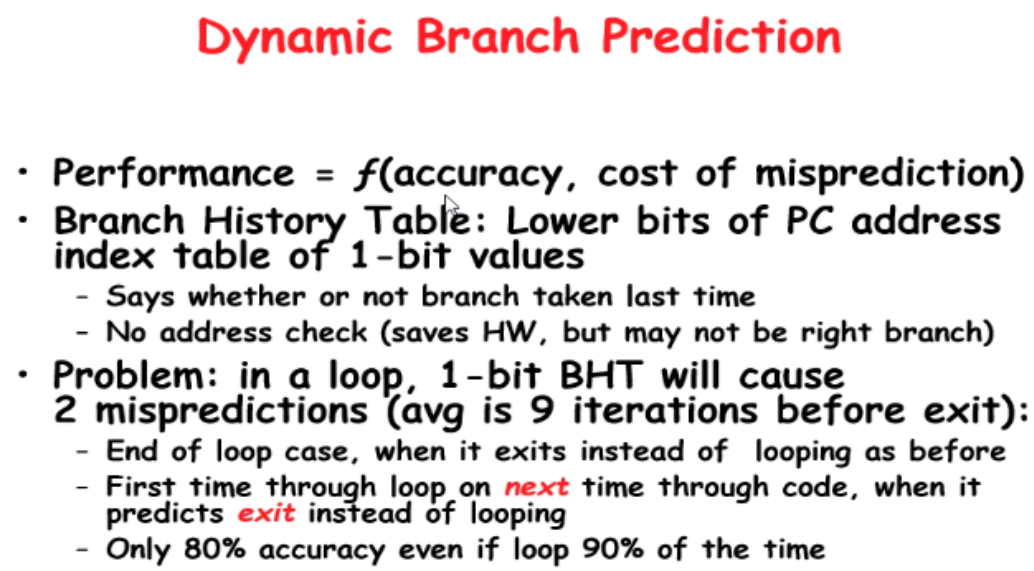
5. Branch prediction implementation and performance analysis

5.1 Methods



5.2 Dynamic Branch Prediction

we mainly focus on accuracy and the cost of rollbacks. The core idea is the Branch History Table(BHT).

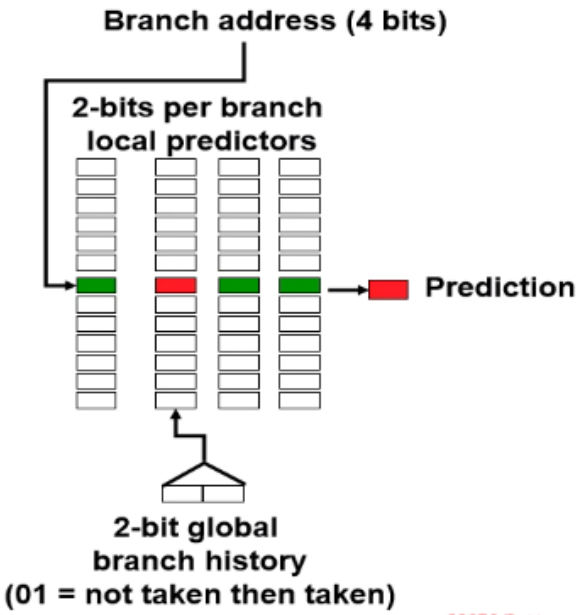


Correlating Branches

Idea: taken/not taken of recently executed branches is related to behavior of next branch (as well as the history of that branch behavior)

- Then behavior of recent branches selects between, say, 4 predictions of next branch, updating just that prediction

- (2,2) predictor: 2-bit global, 2-bit local

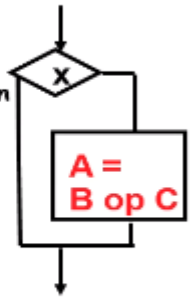


3/01

CS252/Patterson
Lec 17.8

Predicated Execution

- Avoid branch prediction by turning branches into conditionally executed instructions:
  if (x) then A = B op C else NOP
  - If false, then neither store result nor cause exception
  - Expanded ISA of Alpha, MIPS, PowerPC, SPARC have conditional move; PA-RISC can annul any following instr.
  - IA-64: 64 1-bit condition fields selected so conditional execution of any instruction
  - This transformation is called "if-conversion"
- Drawbacks to conditional instructions
  - Still takes a clock even if "annulled"
  - Stall if condition evaluated late
  - Complex conditions reduce effectiveness; condition becomes known late in pipeline



5.3 Tournament predictors

motivation: combine global saturated predictor with local saturated predictor.

This is conducive to the overall performance as we can use global information to speculate the important branches and use the local history to predict the locally frequently used branches.

We select right predictor by a selector. Hope it works.

Here's an example:

Tournament Predictor in Alpha 21264

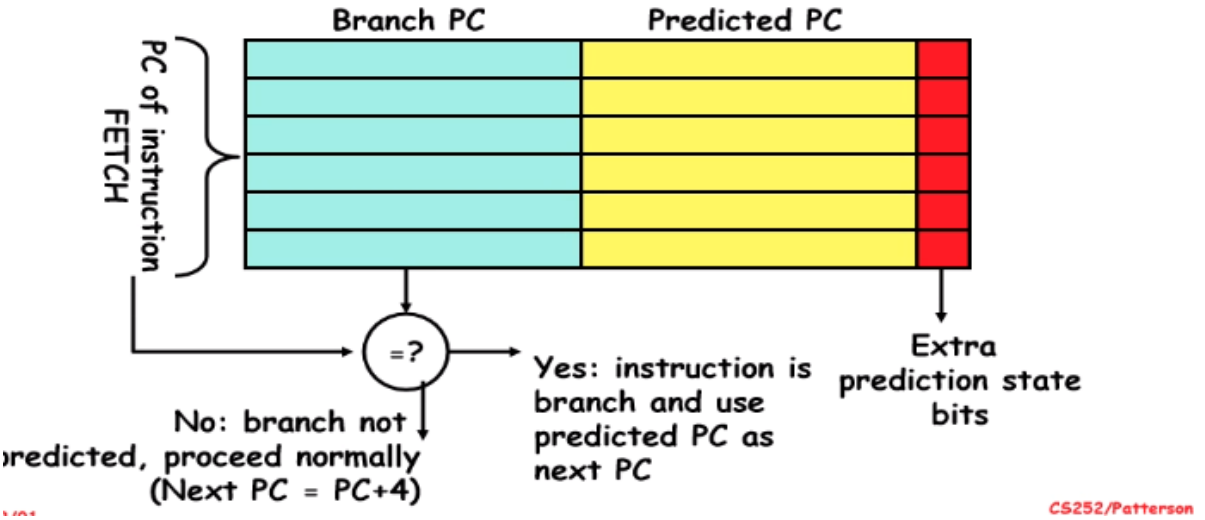
- 4K 2-bit counters to choose from among a global predictor and a local predictor
- Global predictor also has 4K entries and is indexed by the history of the last 12 branches; each entry in the global predictor is a standard 2-bit predictor
  - 12-bit pattern: ith bit 0 => ith prior branch not taken; ith bit 1 => ith prior branch taken;
- Local predictor consists of a 2-level predictor:
  - Top level a local history table consisting of 1024 10-bit entries; each 10-bit entry corresponds to the most recent 10 branch outcomes for the entry. 10-bit history allows patterns 10 branches to be discovered and predicted.
  - Next level Selected entry from the local history table is used to index a table of 1K entries consisting a 3-bit saturating counters, which provide the local prediction
- Total size: 4K*2 + 4K*2 + 1K*10 + 1K*3 = 29K bits! (~180,000 transistors)

5.4 Another optimization

Need Address at Same Time as Prediction

Branch Target Buffer (BTB): Address of branch index to get prediction AND branch address (if taken)

- Note: must check for branch match now, since can't use wrong branch address (Figure 3.19, p. 262)



6. More reference

1. Typical Ways of Implementing Branch Predictor