Sono tecniche che consentono di predire il salto da prendere, in modo da migliorare le performance
Vengono implementare attraverso l'utilizzo di hardware aggiuntivo(e costoso).
Due tipologie:

- **statiche:** gestite dal compilatore attraverso una analisi preliminare del codice
- **dinamiche:** implementate dall'hardware e si basano sul comportamento del codice

Static branch prediction

Utile quando affiancata a tecniche come *branch delayed slot* e *rescheduling* per evitare *data hazards*.
Il **compilatore** può decidere di:

- prendere **sempre** il branch
- prendere **in base alla direzione** del branch
  - quelli in avanti(**forward**) sono spesso **untaken**
  - quelli indietro(**backward**) sono spesso **taken**
- prendere il branch in base **alla profilazione** di run precedenti

Dynamic branch prediction

Tecniche che sfruttano hardware dedicato dove viene usato l'indirizzo dei salti per fare la scelta.
Queste tecniche si basano su principi di località:

- **Temporale:** se una certa area di memoria viene referenziata ad un certo punto, è probabile che verrà referenziata di nuovo in futuro
- **Spaziale:** se una certa area di memoria viene referenziata ad un certo punto, è probabile che verrà referenziata anche un'area adiacente
- **Branch:** ci sono poche alternative di salto;i salti sono distribuiti localmente e non creano spesso conflitti tra loro(??)

Funzionamento base

1. La predizione viene fatta nello stadio di **Decode**
2. L'istruzione predetta viene mandata come successiva nello stadio di **Execute**
3. Il **branch** viene completato e nell'**EXE** viene verificata la predizione
   - se la predizione è vera, il flusso continua senza problemi
   - altrimenti la pipeline viene *flushata* e viene caricata l'istruzione corretta

Branch History Table(BHT)

Piccola memoria che può essere indicizzata dalla parte più bassa dell'indirizzo del branch.
Ogni entry contiene 1 o più bit che indicano se quel branch è stato preso l'ultima volta.

Ogni volta che il branch viene **decodificato**, viene fatto un accesso alla **BHT** utilizzando la parte bassa dell'indirizzo.
Viene utilizzata la predizione salvata nella tabella e il PC viene aggiornato di conseguenza.

Questa tecnica non porta vantaggi in un processore MIPS dal momento il controllo della condizione viene fatto nella fase di **Decode**,dove verrebbe fatta la **predizione**

Two-bit Prediction schemas

Si utilizzano due bit per decidere la predizione.
Di solito $00, 01$ significa **not taken**, mentre $10, 11$ significa **taken**
Uno schema simile si può realizzare con n bit.

Correlating(two-level) predictors

Effettuano predizioni in base ai risultati di branch precedenti.

```
if (aa==2)
     aa = 0;
if (bb==2)
     bb = 0;
if (aa !== bb)
{    }
```

Il terzo branch è dipendente dai branch precedenti

(m,n) predictors

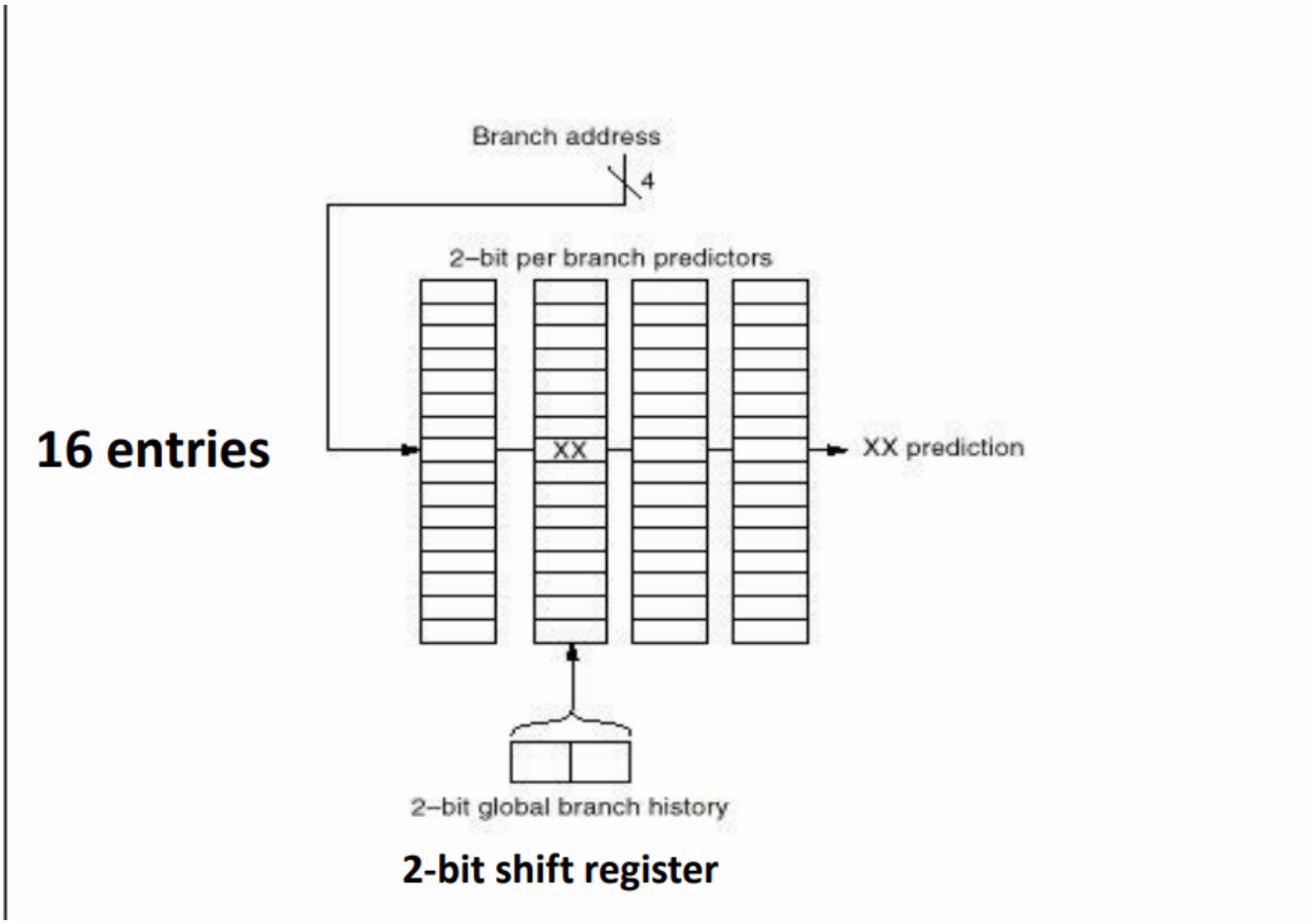
Analizza il comportamento degli ultimi m branch per scegliere tra 2^m predittori, ognuno dei quali è un predittore(simile ai predittori analizzati sopra) a n bit.

(1,1) predictor

Ogni branch è associato a 2^1 predittori da 1 bit:

- uno che indica la predizione nel caso il branch precedente non sia stato preso
- uno che indica la predizione nel caso il branch precedente sia stato preso

(2,2) predictor



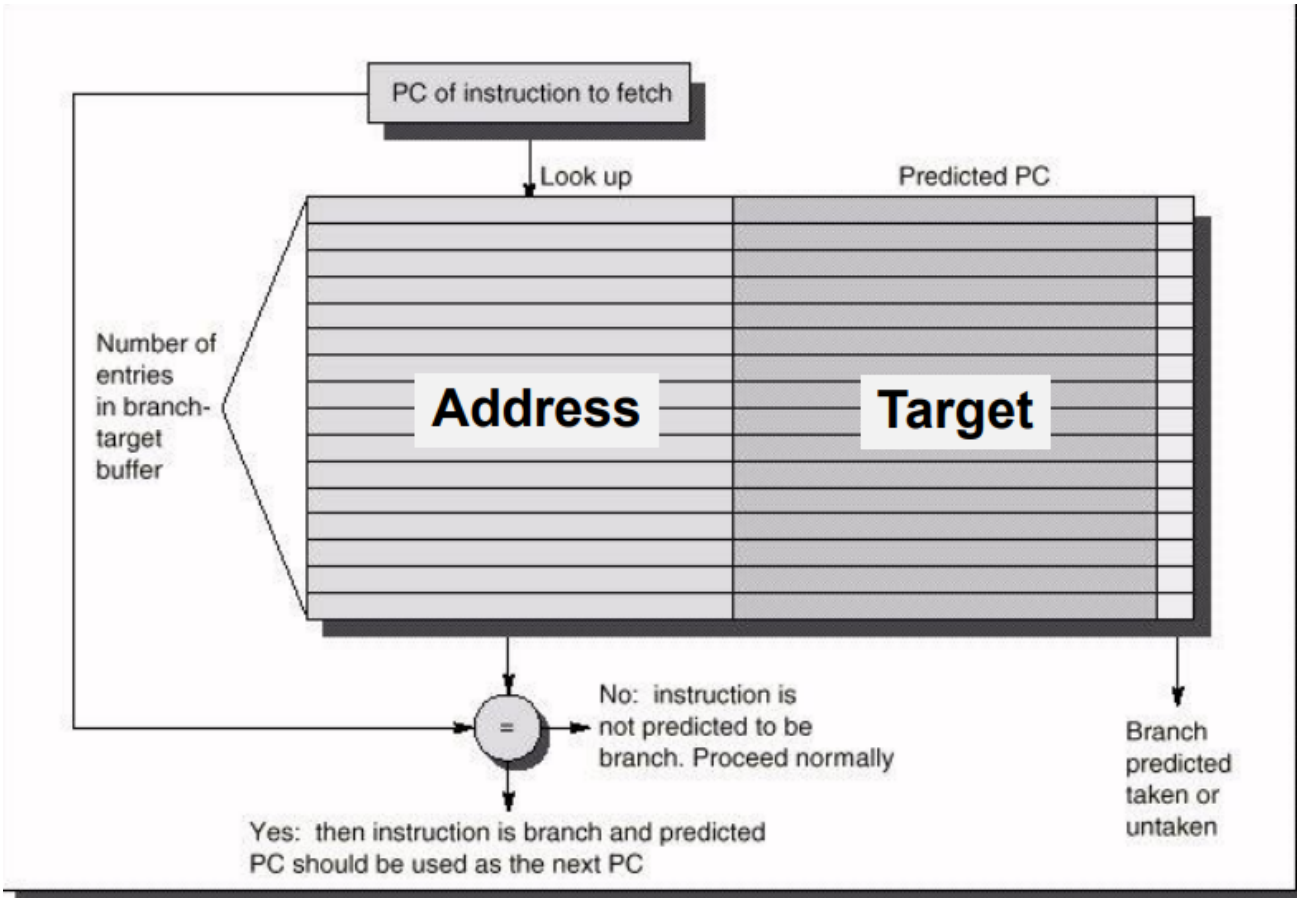
Viene utilizzato uno shift-register a 2 bit per selezionare tra le 2^2 storie.
Quando un branch viene preso(**taken**), viene shiftato da destra nello shift-register un 1, altrimento uno 0.
Il valore della cella poi viene analizzato secondo quanto visto prima per decidere se la predizione è vera oppure no.

Branch Target Buffer(o cache)

Ogni entry di questo buffer contiene:

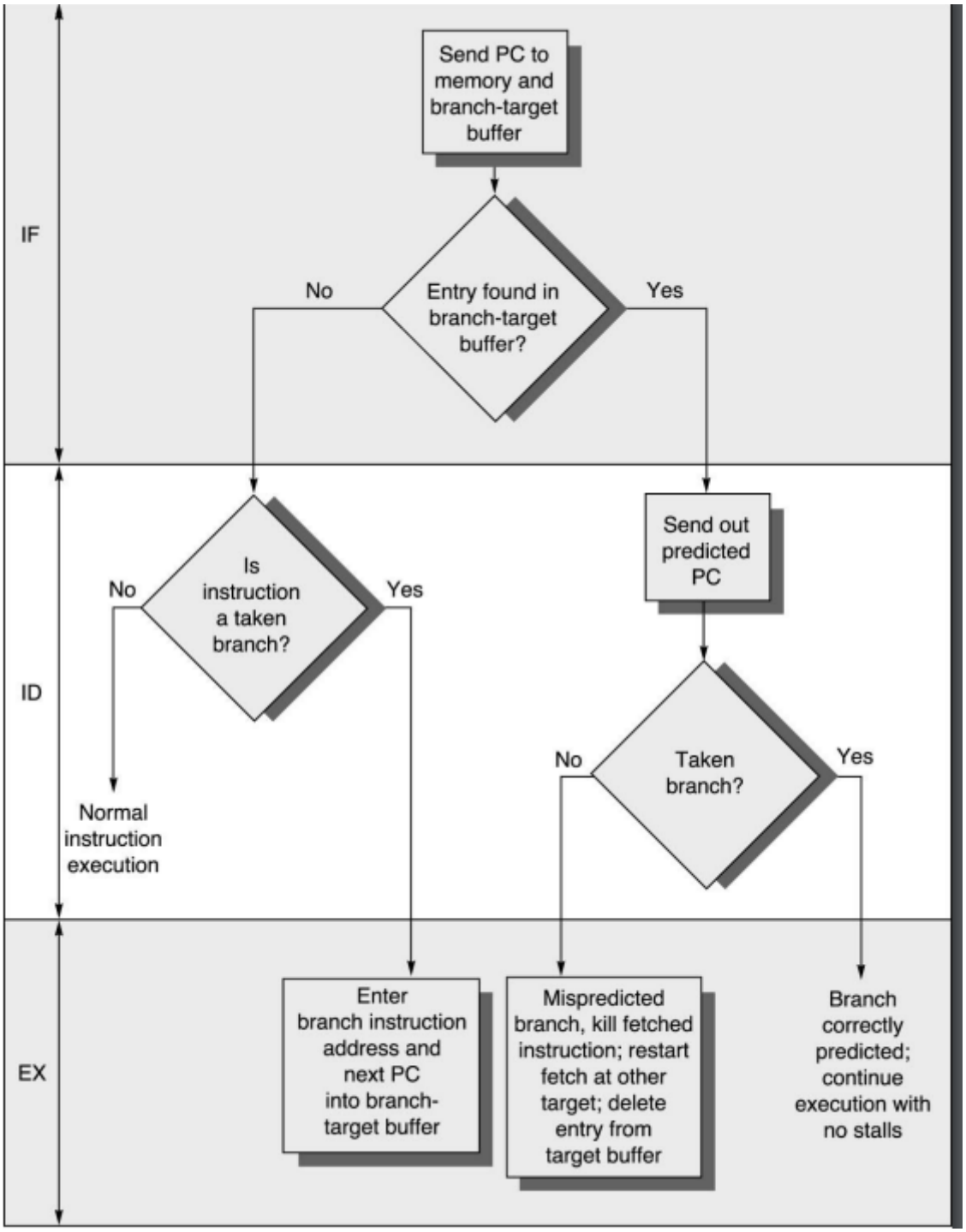
- **indirizzo** del branch che stiamo considerando

- **destinazione** target da caricare nel **PC.**



Prima che l'istruzione venga decodificata, viene controllato che sia presente nel buffer:

- se è presente allora è una istruzione di salto, e il prossimo PC è dato dal target
- se non è presente allora non è una istruzione di salto oppure questo salto non l'ho ancora incontrato



Flowchart che illustra il funzionamento del Branch Target Buffer nei vari stadi della pipeline