**Sobre la relació entre IPC i el % d’encert dels predictors:**

Quan una instrucció de salt es descodificada s’utilitzen mètodes per a predir la direcció i el resultat del salt. D’aquesta manera, es pot fer *prefetching* de les instruccions situades a partir de la direcció de memòria predita com a destí del salt abans que el salt sigui resolt.

A aquest fenomen se’l anomena **execució especulativa** ja que s’executen instruccions sense saber si son realment les que s’han d’executar.

Tot i així els predictors de salt no tenen una fiabilitat del 100%, pel que de vegades cometen errors i prediuen el resultat o direcció d’un salt de manera incorrecta, fent *prefetching* d’instruccions equivocades. En quant el processador rep en la fase d’execució la instrucció de salt i detecta que s’ha executat especulativament les instruccions de la branca equivocada; es provoca el buidat del *pipeline* i es replena amb les instruccions procedents de la branca correcta. Aquest buidat i replenat del *pipeline* dura molts cicles i causa una baixada de rendiment.

Efectivament, un predictor de salts més efectiu provoca que hi hagi menys buidats i replenats del pipeline, per tant, fa augmentar el nombre d’instruccions per cicle, augmentant el rendiment.

Predictors estàtics:

Com ja sabem aquests predictors no recopilen ni utilitzen la informació de les instruccions de salts en temps d’execució, sinó que sempre es comporten igual front al mateix salt.

Els predictor “taken” i “not taken” sempre prediuen tots els salts com a presos o com a no presos respectivament.

Per estadística podem veure que la majoria dels salts solen ser presos, pel que en la majoria dels programes el predictor “taken” sol tenir més taxa d’encerts que el “not taken”.

El predictor perfect en canvi, té una taxa d’encerts del 100 %, pel que, será el predictor amb el que obtindrem un IPC més gran degut a que no tindrem mai cap penalització de buidat de pipeline degut a un branch miss.

Podem observar que en els benchmarks Applu, twolf i vpr el IPC varia molt segons si utilitzem el predictor perfect o algun dels altres dos. Això ens indica que en aquests benchmarks hi ha un nombre d’execució de salts major que en els altres, pel que en utilitzar un predictor amb una taxa d’encerts baixa, el IPC es veu molt ressentit degut a la major quantitat de branch misses.

En el predictor bimodal la PHT conté els 2-bit saturating counter. Quan la PHT és petita (el nombre d’instruccions de salt en el codi és major que el nombre d’entrades) es produeixen col·lisions degut a que dues instruccions de salts diferents utilitzen el mateix comptador per a dur a terme la seva predicció de salt. Això resulta en una eficiencia baixa ja que estem utilitzant el mateix comptador per a predir el resultat de més d’una instrucció de salt.

Aquests salts obviament poden tenir comportaments diferents, pel que la predicció resulta incongruent.

A mesura que anem augmentant la PHT, disminueixen les colisions, pel que augmenta la taxa d’encert en la predicció i disminueixen els misses. Això provoca una pujada en el IPC.

La situació ideal en aquest predictor es que cada instrucció de salt tingui la seva propia entrada en la PHT. Aconseguit això, la taxa d’encerts entraria en una fase estacionària respecte a la longitud de la PHT.

Podem observar aquest fenomen en el gràfic:

En l’applu la taxa d’encerts és manté gairebé constant respecte al nombre d’entrades de la PHT. Podem deduir doncs, que es tracta d’un benchmark amb poques instruccions de salt.

En la resta de benchmarks podem observar que els resultats de la taxa de predicció són més o menys els mateixos amb una PHT de 1024 o 4096. Podem deduir doncs que amb una taula de 1024 hi ha suficient per a tenir una entrada per a cada instrucció i que no es produeixin col·lisions. Augmentar la mida fins a 4096 resulta en tenir la major part de la taula buida.

Podem seguir veient una correlació directa entre la taxa d’encerts i el IPC.

En el predictor G-Share s’utilitza una porta XOR entre el valor de PC quan apunta a la instrucció de salt a predir i el valor de BHR per a generar l’índex per a accedir a la taula del PHT.

Com que ara també utilitzem els bits del BHR, una sola instrucció pot accedir a diferents entrades de la PHT, això millora l’eficiència de predicció front a patrons repetitius de salt, però també augmenta el nombre de col·lisions inclús amb un nombre d’entrades del PHT major que el nombre d’instruccions de salt. Degut a això, i comparant amb el predictor bimodal podem veure que inclús en passar de 1024 entrades a 4096 la taxa d’encerts segueix augmentant. Per tant el predictor G-Share té una taxa d’encerts molt gran, però que podria augmentar encara més si disposessim de més entrades.

La única diferència entre aquest predictor i el G-Share és que aquí per a generar l’índex a la taula de PHT utilitzem la concatenació del BHR i el PC quan apunta a la instrucció de salt de la qual volem predir el salt. Podem deduir llavors que generem un índex amb més bits dels necessaris per accedir a la PHT. En aquest cas la no utilització de la porta XOR com a funció de hashing fa que el nombre de col·lisions augmenti inevitablement respecte al predictor G-Share. I es per això que la seva taxa d’encerts disminueix respecte a aquest.

Trobem doncs, que un dels factors més limitants per a un predictors de salt per a obtenir bona taxa d’encerts és que el nombre de colisions sigui baix, de tal manera que la predicció de cada salt está més individualitzada i per tant és més acurada.

El predictor Pag funciona de manera molt similar al predictor G-Select, però en aquest cas en lloc de tenir un BHR tenim un BHT, de tal manera que podem guardar les últimes n ocurrencies d’un determinat nombre de salts. Es a dir, utilitzem un BHR de manera local per a un determinat nombre d’instruccions.

Com més gran sigui el BHT menys instruccions hauran de compartir el mateix BHR per tant la precisió serà major ja que tindrem les últimes ocurrencies de menor nombre de salts. La situació ideal, seria que hi hagués un BHR per cada instrucció de salt.

Després, el contingut de l’entrada del BHT corresponent s’opera amb una xor junt amb la direcció de la instruccio de salt actual.

Això genera un index a la PHT per a predir el salt.

Podem observar que entre el cas 128-2048 i 64-4096, hi ha una millor predicció quan la BHT té una capacitat major. Per tant, resulta tan important tenir una PHT suficientment gran per aque no hi hagin colisions, com tenir una BHT per a que hi hagin poques instruccions de salt que comparteixin el mateix registre.