Universidad de Costa Rica - Escuela de Ingeniería eléctrica

IE0521 – Estructuras de Computadoras Digitales II

Tarea01 Branch Prediction

Estudiantes:

Edwin Camacho Mora B21304
Sixto Loría Villagra C04417

Profesor:
Erick Carvajal Barboza, PhD

Primer ciclo de 2024

3 de mayo de 2024

Resultados de los experimentos

Resultados de P-share

Tamaño historia global	Bits del PC para indexar				
	4	8	12	16	20
2	66.836 %	74.220 %	84.939 %	86.597 %	86.692 %
6	68.543 %	76.427 %	88.027 %	89.488 %	89.581 %
8	69.710 %	77.388 %	89.109 %	90.518 %	90.589 %
16	84.616 %	85.820 %	92.497 %	93.507 %	93.559 %
20	86.464 %	87.913 %	92.391 %	93.401 %	93.447 %

Cuadro 1: P-share

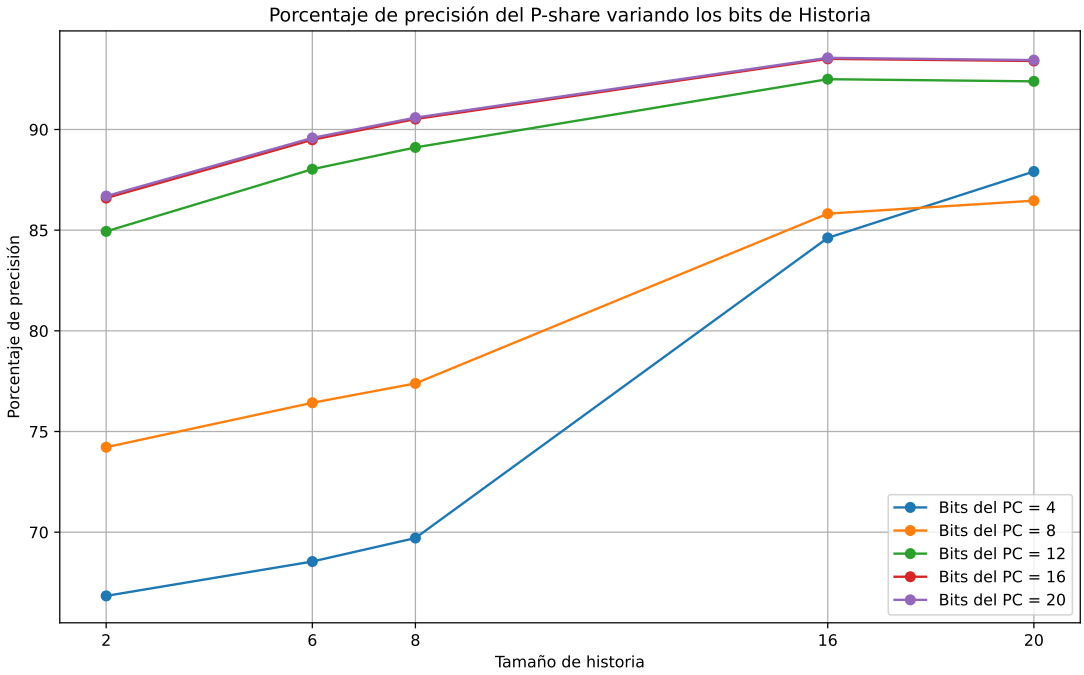

Figura 1: Gráfica de los porcentajes de predicción del P-Share variando los bits de historia

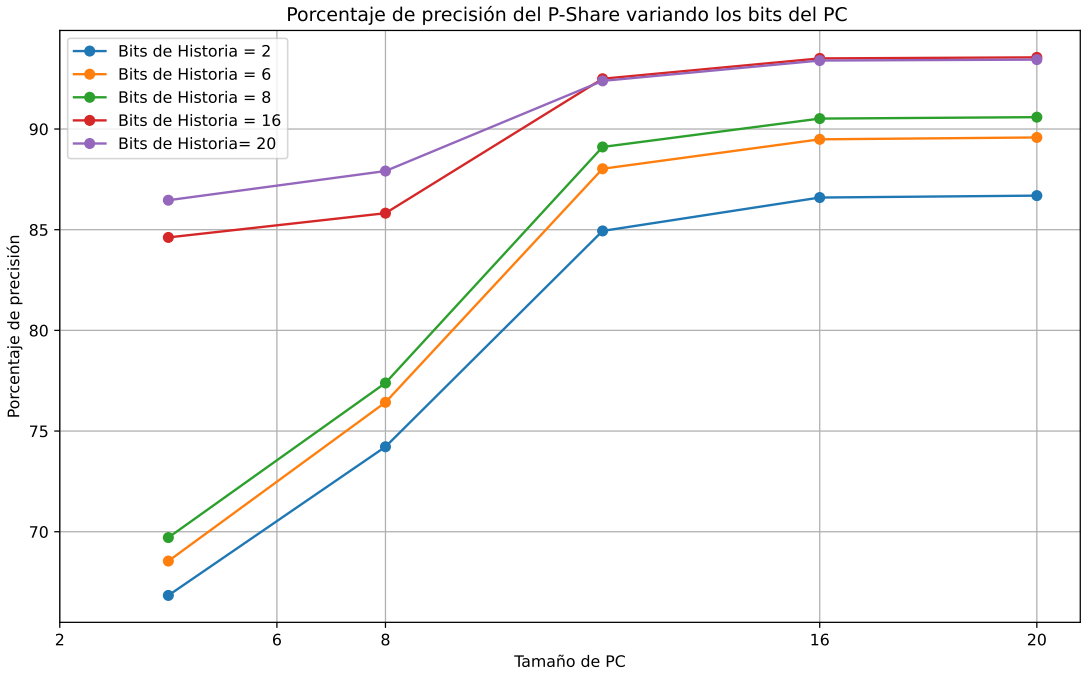

Figura 2: Gráfica de los porcentajes de predicción del P-Share variando los bist del PC

Resultados de Perceptron

Tamaño historia global	Bits del PC para indexar				
	4	8	12	16	20
2	71.496 %	84.303 %	90.859 %	91.466 %	91.480 %
6	74.266 %	90.029 %	93.665 %	93.847 %	93.860 %
8	74.783 %	91.282 %	94.306 %	94.445 %	94.455 %
16	76.459 %	93.697 %	95.942 %	96.062 %	96.070 %
20	77.019 %	94.154 %	96.242 %	96.369 %	96.377 %

Cuadro 2: Perceptron

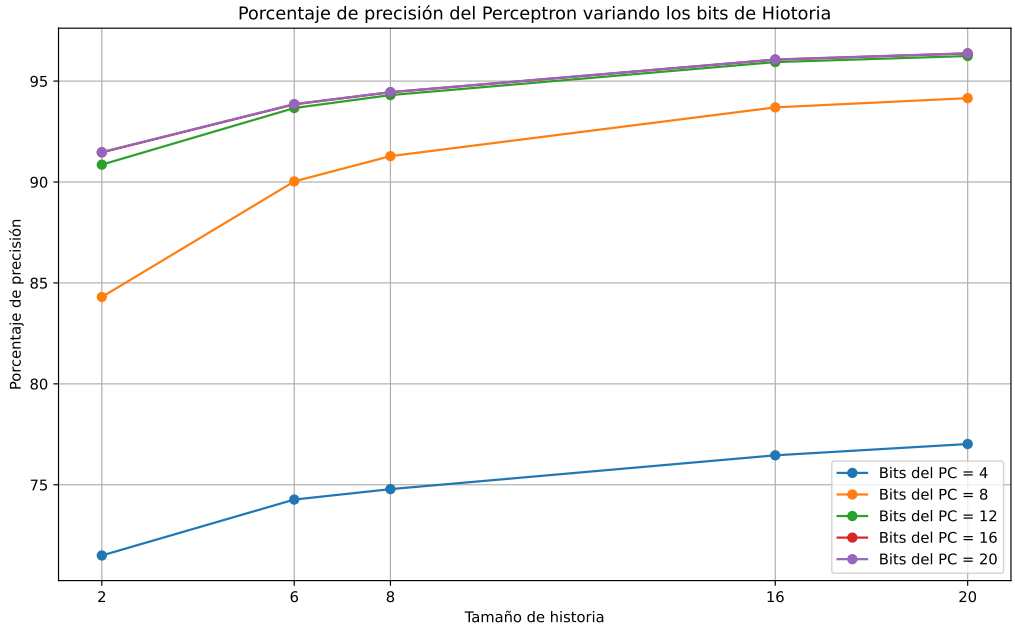

Figura 3: Gráfica de los porcentajes de predicción del Perceptron variando los bits de historia

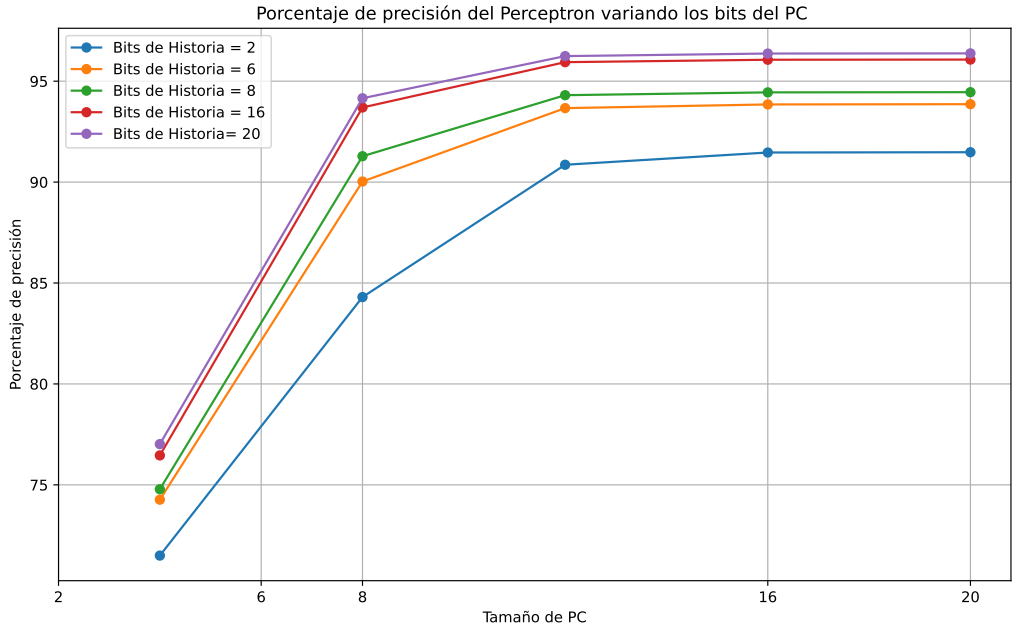

Figura 4: Gráfica de los porcentajes de predicción del Perceptron variando los bits del PC

Análisis de tendencia de datos en los resultados

Análisis de P-Share

- **Si la cantidad de bits del PC se mantiene constante, ¿cómo cambia la precisión ante cambios en el tamaño del registro de historia?**

  Si mantenemos constante los bits del PC para indexar y lo que variamos es la historia vemos se obtiene un aumento considerable en el porcentaje de predicciones correctas, ya que en un predictor de branch P-share, el tamaño del registro de historia es crucial para determinar la cantidad de información sobre los patrones de ramificación pasados que se utiliza para tomar decisiones de predicción. Al aumentar el tamaño del registro de historia, se esta permitiendo que el predictor recuerde patrones de ramificación más largos.

  Al dejar constante los bits de indexado del PC, se esta manteniendo la forma en que se calcula la dirección en la tabla de historia, pero se está cambiando la cantidad de historia que se utiliza para indexar esa tabla. Al aumentar el tamaño del registro de historia, es más probable que el predictor recuerde patrones de ramificación más largos, lo que puede mejorar la precisión de la predicción al permitir que el predictor detecte y responda a patrones más complejos en el comportamiento de la ramificación. Tal y como se puede apreciar en la figura (1)

- **Si la cantidad de bits del registro de historia global se mantiene constante, ¿cómo cambia la precisión ante cambios en la cantidad de bits del PC utilizados para indexar?**

  Si la cantidad de bits del registro de historia global se mantiene constante y solo se cambia la cantidad de bits del PC utilizados para indexar la presicion del predictor de branch P-share puede verse afectada ya sea si se aumentan o si se reducen.

  Si se aumentan la cantidad de bits del PC utilizados para indexar, se reduce la posibilidad de colisiones en la tabla de historia, lo que puede mejorar la precisión de la predicción. Esto se debe a que más bits de PC permiten una mayor granularidad en la dirección de la tabla de historia, lo que significa que cada patrón de ramificación tiene una dirección única en la tabla de historia. Este aumento al principio es significativo según se puede ver en la tabla de datos y el gráfico de la figura(2), pero conforme nos acercamos a un valor considerable de bits el aumento de precisión va disminuyendo, esto debido a que ya posee bastante granularidad con la que cada praton ya casi que es único por lo que por mas que aumente no sale rentable el beneficio.

Análisis de Perceptron

- **Si la cantidad de bits del PC se mantiene constante, ¿cómo cambia la precisión ante cambios en el tamaño del registro de historia?**

  Si mantenemos constante los bits del PC para indexar y lo que variamos es la historia vemos que la precisión aumenta ligeramente, esto debido a que el predictor de perceptrón se basa en el concepto de redes neuronales y utiliza un registro de historia global para tomar decisiones de predicción. Este registro de historia global es esencialmente una cadena de bits que registra la historia de las ramificaciones tomadas y no tomadas cuando se aumenta tamaño del registro de historia (es decir, la cantidad de bits en el registro), se esta permitiendo que el predictor tenga en cuenta una historia más larga de ramificaciones. Esto puede mejorar la precisión de la predicción ya que se considera más contexto histórico. sin embargo hay tener cuidado pues, también puede llevar a un mayor riesgo de sobreajuste, donde el predictor puede empezar a adaptarse demasiado a las peculiaridades específicas de la historia de ramificaciones y perder su capacidad para generalizar bien a nuevas ramificaciones [1].

- **Si la cantidad de bits del registro de historia global se mantiene constante, ¿cómo cambia la precisión ante cambios en la cantidad de bits del PC utilizados para indexar?**

  Si la cantidad de bits del registro de historia global se mantiene constante y solo se cambia la cantidad de bits del PC utilizados para indexar la precisión se ve mejorada en un inicio significativamente, esto debido a que cuando se aumentan la cantidad de bis del PC utilizados para indexar, se esta permitiendo al predictor que tenga en cuenta mas información sobre la ubicación de la instruccion.

  Sin embargo, hay un límite en cuánto puede mejorar la precisión al aumentar la cantidad de bits del PC utilizados para indexar se refiere. Vemos en la figura (4) que inicialmente se puede ver una mejora significativa en la precisión. Pero después de cierto punto, las mejoras en la precisión pueden comenzar a estabilizarse. Esto se debe a que llega un punto en el que la información adicional que se va agregando con mas bits deja de ser significativa por lo que no contribuyen casi nada a la mejora, es por eso que vemos que los valores de mejora de precisión tienden al estabilizarse al alcanzar una cantidad de bits de PC considerables.

Predictor UCR-IE0521 (P-share vs Preceptron)

Explicación del algoritmo

Para el predictor propuesto elegimos, realizar un predictor de torneo como el que se vio en clases solo que en lugar de ser entre el P-share y el G-share lo realizaremos entre el P-share y el predictor basado en perceptrones [1], implementado como parte de esta tarea. Además se implementara un contador de 2

bits que se satura, este tiene la función de que según que predictor que este teniendo mejores resultados en la predicción usaremos ese para el resultado final y según el que vaya teniendo mejor comportamiento ese se estará usando.

Por ejemplo se tiene un salto, por lo que se instancian ambos predictores similar a como se hace en el archivo de branch-predictor.py, ambos predictores predicen el resultado y se evalua cual tuvo un acierto, si el predictor de P-share tuvo un acierto se resta 1 al sumado y si el predictor de perceptrones tuvo un acierto se le suma 1 al contador, dependiendo del estado del contador (Valores entre 00 y 11) se decidirá que predictor seguir usando para tomar los resultados, cada que un predictor falle perderá tendencia y se pasara a usar el siguiente y así repetidamente.

Obteniendo de esta manera que predictor utilizar según sea el caso, como duda teníamos de si era valido usar el predictor P-share como parte de nuestro algoritmo, dado que fue visto en clases. Pero se consulto con el profesor y si nos dio permiso de implementarlo, dado que el conjunto de todo es un algoritmo distinto.

Adicionalmente se implemento en el código de branch_prediction.py un pequeño algoritmo que permite ejecutar la predicion y la ejecución a la vez en una sola función.

Como correr el predictor

Para correr el predictor, ubíquese en el directorio donde se encuentra el archivo **branch_prediction.py** y ejecute el siguiente comando desde la terminal:

- **python3 branch_predictor.py -n 9 –bp 4 -g 7**

Donde:

- n = 9: Cantidad de bits del PC para indexado.
- g = 7: Tamaño del registro de la historia global.
- –bp 4: Selector de que branch usar (E0521_bp.py en este caso)

Estos parámetros son los valores máximos que se pueden poner para obtener los mejores valores de porcentaje de precisión cumpliendo con el presupuesto debido de 32Kb.

Calculo del presupuesto

Para calcular el presupuesto se deben considerar las tablas que hay en el P-Share y las del Perceptron, en conjunto con los bits del sumador que se utiliza para saturar y elegir un método sobre el otro.

1) **Bits utilizados por el P-Share.**

   Para el P-Share se debe considerar dos tablas, una que es la tabla de la historia global y otra es la tabla de patrones.

   - **Tabla de Patrones:** El total de bits que necesita esta dada por la siguiente ecuación:

   $$Bits_{patrones} = 2^{PC} \cdot g \tag{1}$$

     Donde **PC** representa la cantidad de bits a indexar y **g** el tamaño de la historia global.

   - **Tabla de historia global:** El total de bits que necesita se puede obtener por la siguiente ecuación:

   $$Bits_{historia} = 2^{g} \cdot 2 \tag{2}$$

     Se multiplica por 2 porque se utilizan dos bits para cada registro de la historia, el cual muestra la historia de las predicciones.

2) **Bits utilizados por el Perceptron.**

   Para este se debe considerar la tabla de perceptores, la cual es una matriz de **n** filas por **j** columnas la cual contiene los pesos de los perceptrones. El índice **n** depende del **PC** y se obtiene al elevar al PC con una base 2 (2^{PC}) y **j** depende del tamaño de la historia global y un bit más que representa el peso de W_0.

   Se debe considerar que los pesos son números enteros, y la cantidad de bits que se deben considerar dependen del umbral, que esta dado por $1{,}93 \cdot h + 14$, donde h es el tamaño de la historia global. Los pesos pueden ser mayores al umbral, por lo tanto se se consideran los bits necesarios para representar números mayores al umbral. Teniendo esto en cuenta, la ecuación que representa la cantidad de bits para la tabla del perceptron esta dada por:

   $$Bits_{Perceptron} = 2^{PC} \cdot (g + 1) \cdot x \tag{3}$$

   Donde: x: bits que se ocupan para representar los pesos.

3) Para el contador se utilizan 4 bits, dos para seleccionar un método y los otros dos para seleccionar el otro método.

Por lo tanto, la ecuación con la que se obtiene la cantidad de bits que usa el predictor se muestra a continuación:

$$Bits = 2^{PC} \cdot g + 2^{g} \cdot 2 + 2^{PC} \cdot (g + 1) \cdot x + 4 \tag{4}$$

Considerando la ecuación, la mejor combinación posible respetando el presupuesto, es utilizando $n = 9$ y $g = 7$. Como g es igual a 7, el umbral es $1{,}93 \cdot 7 + 14 = 27{,}51$, para ese umbral se necesita como mínimo 5 bits, como los pesos pueden ser mayores al umbral y teniendo en cuenta un bit extra para el signo,;se consideran 7 bits para los pesos.

Sustituyendo los valores se obtiene que los bits totales son:

$$Bits = 2^9 \cdot 7 + 2^7 \cdot 2 + 2^9 \cdot (7 + 1) \cdot 7 + 4 = 32516 \; bits \tag{5}$$

Que son ligeramente menor al limite de 32768 bits, propuestos por el enunciado.

Referencias

[1] D. Jimenez and C. Lin, "Dynamic branch prediction with perceptrons," in Proceedings HPCA Seventh International Symposium on High-Performance Computer Architecture, 2001, pp. 197–206.