EIE

Escuela de
Ingeniería Eléctrica

Universidad de Costa Rica
Facultad de Ingeniería
Escuela de Ingeniería Eléctrica

UNIVERSIDAD DE COSTA RICA

IE0521 - Estructuras de Computadores Digitales ll

TAREA 2: Simulación de Procesador con Ejecución Fuera de Orden (Parte 1)

Oscar Fallas Cordero. B92861

Entrega: 21/Mayo/2023

---

1. Experimentos

	600.perlbench_s	**605.mcf_s**	**619.lbm_s**	**648.exchange2_s**	**654.roms_s**
La cantidad de instrucciones que, en promedio, se ejecutan por ciclo	1.9041	0.359	0.337	1.1343	0.7957
La cantidad de ciclos que, en promedio, tarda en ejecutarse una instrucción	0.525	2.86	2.967	0.882	1.257
La cantidad total de instrucciones ejecutadas	150000004	150000002	150000001	150000000	150000004
El miss rate del caché de instrucciones del primer nivel	5.17e-4	0	0	9.70e-4	3.00e-6
El miss rate del caché de datos del primer nivel	2.10e-4	0.204	0.217	1.15e-4	0.028
El miss rate del caché de segundo nivel	0.665	0.744	0.335	1.85e-3	0.744
El miss rate del último nivel de caché	0.942	0.467	0.556	1	0.744
El porcentaje de branches predichos incorrectamente	0.0249	0.0877	0.0166	0.236	5.5e-4
El MPKI del predictos de saltos	3.4622	23.0144	0.207	33.162	0.0573
La cantidad de instrucciones que, e promedio, habían en el ROB cuando ocurrieron malas predicciones de saltos	147.773	25.958	240.32	11.707	317.027

Cuadro 1: Resultados Simulaciones

1.1. Preguntas

1. ¿Cuál sería la aplicación que se beneficiaría más de aumentar el tamaño de caché L1I? ¿Cuál se beneficiaría menos?¿Por qué?

   Sabemos que entre más tamaño de caché se pueden almacenar más instrucciones cercana al procesador, mejorando su desempeño. De acuerdo con la tabla, la aplicación más beneficiada sería la 648.exchange2_s que posee la tasa de fallos más alta, por lo añadir más caché permitiría almacenar más instrucciones cercanas, por otro lado podemos ver que al 605.mcf_s y al 619.lbm_s no les beneficiería en nada al menos en la porción de aplicación que estamos simulando.

2. ¿Cuál sería la aplicación que se beneficiaría más de aumentar el tamaño de caché L1D? ¿Cuál se beneficiaría menos? ¿Por qué?

   La aplicación más beneficiada es 613.lbm_s que tiene un 21.7 % de miss rate, mientras que la menos beneficiada sería la 648.exchange2_s que tiene un un 0.0115 %, es decir, que al menos en la porción de aplicación simulada el aumentar el tamaño no correspondería a un aumento significante.

3. La tendencia general de las aplicaciones es que el miss rate aumenta en los niveles de caché más bajos (L2, LL). ¿Por qué?

   Esto se debe a varios factor, se sabe que por lo general el nivel de caché más grande es el primero, mientras que sucesivamente van disminuyendo su capacidad de almacenamiento. También es debido al principio de localidad y temporalidad, que dicta que las aplicaciones tienden a reutilizar los datos que han sido utilizados más recientemente, mientras que si un dato no se ha utilizado en un rato se encontrará más alejado.

4. ¿Cuál sería la aplicación que se beneficiaría más de utilizar un mejor predictor de saltos condicionales? ¿Cuál se beneficiaría menos? ¿Por qué?

   La aplicación más beneficiada sería 648.exchange_s ya que tiene un porcentaje de fallos del predictor de saltos bastante elevado. Mientras que para el 654.roms_s el mejorar o aplicar un predictor mejor puede que siquiera baje el porcentaje de fallos.

5. ¿Cuál sería la aplicación que se beneficiaría más de utilizar un mejor predictor de saltos indirectos? ¿Cuál se beneficiaría menos? ¿Por qué?

   Si nos basamos MPKI podemos notar que también la aplicación 648.exchange2_s tiene un porcentaje alto, lo que indica que no se están prediciendo correctamente una gran cantidad de saltos indirectos, lo cuál si se mejorará el predictor beneficiaría el rendimiento. Por otra parte, si vemos el 654.roms_s podemos ver que ya el predictor es capaz de casi obtener el mejor resultado posible, por lo que una mejora sería insignificante.

6. ¿Cuál sería la aplicación que se beneficiaría más de utilizar un pipeline más ancho? ¿Cuál se beneficiaría menos? ¿Por qué?

   Si vemos la cantidad de instrucciones habientes en el ROB del 654.roms_s es demasiado alta, lo cuál produce un gran estancamiento en el procesador y disminuyendo significativamente el rendimiento, con un pipeline más ancho reduciríamos ese estancamiento procesando más instrucciones en paralelo. Por otra parte el 648.exchange2_s su cantidad de instrucciones promedio en ROB al momento de ocurrir malas predicciones es demasiado bajo, por lo que añadir mas ancho no tendría una significancia alta en la cantidad de instrucciones procesadas.