Задание 2

Логическая организация кэш-памяти; тестовое покрытие

Для кэш-памяти данных с заданными параметрами:

1. Описать структуру адреса;
2. Построить конечный автомат Мили, описывающий набор кэш-памяти (set); выходной алфавит должен позволять различать адреса (теги) запросов;
3. Описать входную последовательность минимальной длины, приводящую к вытеснению модифицированных данных (исходное состояние: кэш пуст);
4. Рассчитать достигнутое тестовое покрытие в метрике состояний и в метрике переходов построенного конечного автомата.

Возможные события:

- Операции чтения/записи со стороны вычислительного ядра;
- Снуп-инвалидатор со стороны подсистемы памяти: при его получении кэш вычёркивает немодифицированный блок с заданным адресом либо вытесняет модифицированный блок в память.

Структура кэш-памяти (вариант "d"):

- Разрядность адреса (W_a) – 48;
- Размер кэш-памяти (S) – 2 Мбайта;
- Размер блока (B) – 64 байта;
- Ассоциативность (A) – 32;
- Тип записи – отложенная (write-back);
- Политика заведения – первый свободный блок набора;
- Промах по чтению и записи;
- Политика заведения – LRU.

1. Описать структура адреса

Для смещение внутри блока необходимо иметь

$$\log_2(B) = \log_2(64) = 6 \text{ бит.}$$

Для определения поля под "номер набора" нужно иметь

$$2^n = \frac{S}{A * B} = \frac{2 \text{ Мбайт}}{32 * 64 \text{ байт}} = \frac{2^21}{2^5 * 2^6} \Rightarrow n = 10 \text{ бит}$$

Оставшееся место для тега блока: $48 - 6 - 10 = 32$ бита. Структура адреса будет:

Тег	Индекс	Смещение
[48 : 16]	[15 : 6]	[5 : 0]

2. Построить конечный автомат Мили

Ограничимся набором тегов $A + 1 = 33$.

Конечный автомат Мили:

$$Automaton = (S, X, Y, \delta : S \times X \to S, s_0 \in S, \lambda)$$
$$\lambda : S \times X \to Y$$

- Набор состояний: $S_t = (fulltag_i)_{i=0}^{31}$,

$$fulltag_i = \begin{cases} (v = 0, \varnothing) \\ (v = 1, \text{tag} \in \{1, \dots, A+1\}, \text{age} \in \{1, \dots, A\}, \text{modified} \in \{0, 1\}) \end{cases}$$

v – валидность, tag – один из 33 тегов, age – давность использования, $modified$ – флаг изменения.

- Начальное состояние – $s_0 = [\varnothing, \varnothing, ..., \varnothing, \varnothing]$
- Входной алфавит – $X = ld, st, inv \times tag_i, i = \overline{0, 32}$
- Выходной алфавит – $Y = ld, st, st + ld, \varnothing \times tag_i, i = \overline{0, 32}$
- Функции выходов – $\delta : (s, x) \to s'$
- Функции переходов – $\lambda : (s, x) \to y$

Запись

$$\delta : ((st, tag), s) \to s'$$

Попадание: $s : \exists k : v_k = 1 \wedge tag_k = tag$

$$s' : fulltag'_i = \begin{cases} fulltag_i & v_i = 0 \vee (v_i = 1 \wedge age_k < age_i) \\ (1, tag_i, age_i + 1, modified_i) & v_i = 1 \wedge i \neq k \wedge age_i < age_k \\ (1, tag_i, 1, 1) & i = k \end{cases}$$

$$y = \varnothing$$

Промах без вытеснения: $s : \nexists k : v_k = 1 \wedge tag_k = tag \wedge \exists j : v_j = 0$ (j – минимальное из свободных)

$$s' : fulltag'_i = \begin{cases} fulltag_i & v_i = 0 \\ (1, tag_i, age_i + 1, modified_i) & v_i = 1 \wedge i \neq j \\ (1, tag_i, 1, 1) & i = j \end{cases}$$

$$y = st$$

Промах с вытеснения: $s : \nexists k : v_k = 1 \wedge tag_k = tag \wedge \nexists j : v_j = 0$

$$s' : fulltag'_i = \begin{cases} (1, tag_i, age_i + 1, modified_i) & i : age_i \neq 32 \\ (1, tag_i, 1, 1) & i : age_i = 32 \end{cases}$$

$$y = \begin{cases} st, & \exists i : age_i = 32 \wedge modified_i = 1 \\ \varnothing & \exists i : age_i = 32 \wedge modified_i = 0 \end{cases}$$

Чтение

$$\delta : ((ld, tag), s) \to s'$$

Попадание: $s : \exists k : v_k = 1 \wedge tag_k = tag$

$$s' : fulltag'_i = \begin{cases} fulltag_i & v_i = 0 \vee (v_i = 1 \wedge age_k < age_i) \\ (1, tag_i, age_i + 1, modified_i) & v_i = 1 \wedge i \neq k \wedge age_i < age_k \\ (1, tag_i, 1, modified_i) & i = k \end{cases}$$

$$y = \varnothing$$

Промах без вытеснения: $s : \nexists k : v_k = 1 \wedge tag_k = tag \wedge \exists j : v_j = 0$ (j – минимальное из свободных)

$$s' : fulltag'_i = \begin{cases} fulltag_i & v_i = 0 \\ (1, tag_i, age_i + 1, modified_i) & v_i = 1 \wedge i \neq j \\ (1, tag_i, 1, 0) & i = j \end{cases}$$

$$y = ld$$

Промах с вытеснения: $s : \nexists k : v_k = 1 \wedge tag_k = tag \wedge \nexists j : v_j = 0$

$$s' : fulltag'_i = \begin{cases} (1, tag_i, age_i + 1, modified_i) & i : age_i \neq 32 \\ (1, tag_i, 1, 0) & i : age_i = 32 \end{cases}$$

$$y = \begin{cases} ld + st, & \exists i : age_i = 32 \wedge modified_i = 1 \\ ld & \exists i : age_i = 32 \wedge modified_i = 0 \end{cases}$$

Инвалидация

$$\delta : ((inv, tag), s) \to s'$$

Попадание: $s : \exists k : v_k = 1 \wedge tag_k = tag$

$$s' : fulltag'_i = \begin{cases} fulltag_i & v_i = 0 \vee (v_i = 1 \wedge age_k < age_i) \\ (1, tag_i, age_i + 1, modified_i) & v_i = 1 \wedge i \neq k \wedge age_i < age_k \\ (0, \varnothing) & i = k \end{cases}$$

$$y = \begin{cases} st, & modified_k = 1 \\ \varnothing & modified_k = 0 \end{cases}$$

Промах: $s : \nexists k : v_k = 1 \wedge tag_k = tag$

$$s' = s$$

$$y = \varnothing$$

3. Пример – вытеснение модифицированных данных

Исходное состояние – кэш пуст:

$$s_0 = [\varnothing, \varnothing, ..., \varnothing, \varnothing]$$

Зная, что политика вытеснения LRU, что значит вытеснению будет подлежать самый "старый" блок. Этот же самый блок должен быть (а) модифицированным и (б) быть первым

вытесненным для соответствия условиям примера. Но так как изменение ведёт к обнулению возраста, среди прочих блоков, то этот блок должен быть изменён до появления других блоков, приводящих к его вытеснению. А значит этот блок, который мы сразу же модифицируем, будет первым занесённым в кэш:

$$
\begin{array}{lccccccc}
 & [& \varnothing & \varnothing & \varnothing & ... & \varnothing & \varnothing &] \\
\delta(st, tag_0) & [& (1, tag_0, 1, 0) & \varnothing & \varnothing & ... & \varnothing & \varnothing &] \\
\delta(ld, tag_0) & [& (1, tag_0, 1, 1) & \varnothing & \varnothing & ... & \varnothing & \varnothing &] \\
\delta(st, tag_1) & [& (1, tag_0, 1, 1) & (1, tag_1, 1, 1) & \varnothing & ... & \varnothing & \varnothing &] \\
\delta(st, tag_2) & [& (1, tag_0, 1, 1) & (1, tag_1, 1, 2) & \varnothing & ... & \varnothing & \varnothing &] \\
\vdots & & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \\
\delta(st, tag_{30}) & [& (1, tag_0, 1, 1) & (1, tag_1, 1, 2) & (1, tag_3, 1, 1) & ... & (1, tag_{30}, 1, 1) & \varnothing &] \\
\delta(st, tag_{31}) & [& (1, tag_0, 1, 1) & (1, tag_1, 1, 2) & (1, tag_3, 1, 1) & ... & (1, tag_{30}, 1, 1) & (1, tag_{31}, 1, 1) &] \\
\delta(st, tag_{32}) & [& (1, tag_{32}, 1, 1) & (1, tag_1, 1, 2) & (1, tag_3, 1, 1) & ... & (1, tag_{30}, 1, 1) & (1, tag_{31}, 1, 1) &]
\end{array}
$$

4. Тестовое покрытие

- Метрика состояний

  Начнём считать по порядку переменных в $fulltag$.

  Может быть $k \in \{0, 2, \ldots, 32\}$ валидных блоков, расположение которых являются различными состояниями. Всего таких перестановок:

$$C_B^k = \frac{A!}{k!(A-k)!}$$

  Каждый из k валидных блоков может быть заполнен $tag \in \{1, \cdots, 33\}$, но без повторений. То есть для первого из валидных блоков (если такой есть) будет 33 варианта тега, для второго (если есть) будет 32 и так далее. Итогово, если k валидных, то комбинаций с тегами будет:

$$\frac{(A+1)!}{(A+1-k)!}$$

  Каждый из k валидных блоков имеет возраст $age \in \{1, \cdots, k\}$, с учётом этого, вариантов состояний будет:

$$k!$$

  И к тому же, любой валидный блок может быть как модифицированный, так и немодифицированный:

$$2^k$$

  Итого состояний:

$$|S| = \sum_{k=0}^{32} \frac{A!}{k!(A-k)!} \frac{(A+1)!}{(A+1-k)!} k! 2^k = \sum_{k=0}^{32} \frac{A!}{(A-k)!} \frac{(A+1)!}{(A+1-k)!} 2^k$$

$$|S| \approx 6.1 * 10^{107}$$

точное значение: 6106587494906339209056285369085736099574205379657692333119731334920194396843541984819365535103870474297211 21

В примере было пройдено 34 состояний, тестовое покрытие:

$$\frac{34}{|S|} \approx 5.8 * 10^{-107}$$

- Метрика переходов

  На каждый $fulltag$ в любом состоянии может произойти 8 различных операций с 33 различными тегами, тогда всего переходов будет: $8 * 33 * 6.1 * 10^{107} \approx 1.6 * 10^{110}$. С учётом того что переходов было 33, то тестовое покрытие будет:

$$\frac{33}{8 * 33 * |S|} \approx 2 * 10^{-109}$$