Ein Modell für den Cache des Motorola ColdFire

Christoph Berg

Universität des Saarlandes

`cb@cs.uni-sb.de`

9. Februar 2004

(K)ein Modell für den Cache des Motorola ColdFire

Christoph Berg

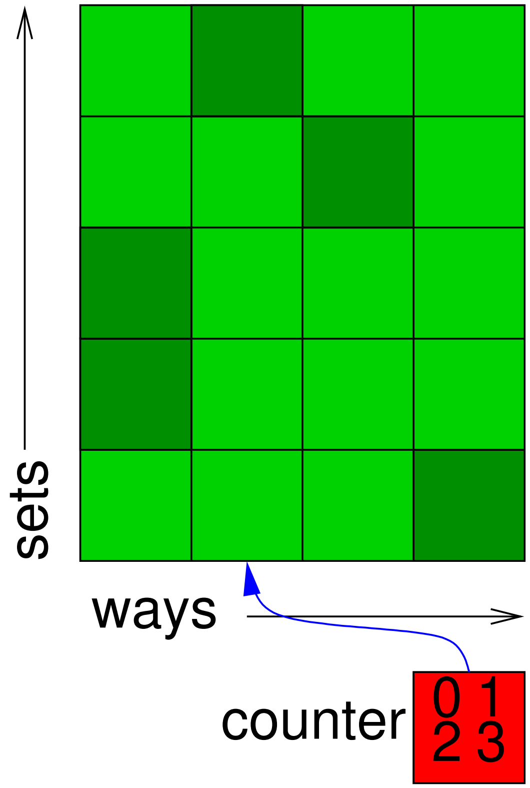
Universität des Saarlandes

`cb@cs.uni-sb.de`

9. Februar 2004

Statische Cache-Analyse

- ColdFire-Cache hat 128 Sets à 4 Ways
- Counter zeigt auf nächsten zu ersetzenden Way
- Problem in der (Must-, May-) Analyse:
  Wo steht der Counter?
- Bisheriges benutztes Modell:
  Direct-Mapped-Cache für Must-Analyse
  - Wirft 3/4 des Cache-Inhalts weg
  - Geht das besser?
  - May-Analyse?



Abstrakte Interpretation

- Konkreter Cache-Zustand $s = (\bar{l}, c) \in \mathbb{S}$:
  - Counter $c \in \{0, \ldots, 3\}$
  - Inhalt $\bar{l} \in (\mathbb{N} \cup \{\perp\})^{4 \times 128}$
- „Bestes“ Modell = keine Abstraktion
- Verband = Potenzmenge der (konkreten) Cache-Zustandsmenge: $L = \mathcal{P}(\mathbb{S})$
- Unbekannter Zustand: $\top = \mathbb{S}$ („Chaos-Cache“)
- Der Einfachheit halber hier:
  - Cache voll belegt ($\perp \notin \bar{l}$)
  - Nur ein Set

Analysebeginn: Chaos-Cache

- Cache-Inhalt unbekannt, Counter unbekannt: Zustandsmenge S
- Zugriff auf Line l_1, neuer Zustand S':

$$S' = \begin{cases} s & l_1 \in s \in S \quad \text{(hit)}, \\ s_{\bar{l}[c^{++}] \mapsto l_1} & l_1 \notin s \in S \quad \text{(miss)}. \end{cases}$$

- l_1 im Cache, aber Beziehung $c \sim l_1$ unbekannt:

$$S' = \{s | l_1 \in s \in S\}$$

Analysebeginn: Chaos-Cache (2)

- Zugriff auf Line $l_2 \neq l_1$ im gleichen Set:

$$S'' = \begin{cases} s' & l_2 \in s' \in S' \quad \text{(hit)}, \\ s'_{\bar{l}[c^{++}] \mapsto l_2} & l_2 \notin s' \in S' \quad \text{(miss)}. \end{cases}$$

$$= \begin{cases} s & l_1, l_2 \in s \in S \quad \text{(hit)}, \\ s_{\bar{l}[c^{++}] \mapsto l_2} & l_1 \in s, l_2 \notin s \in S \quad \text{(miss)}. \end{cases}$$

- Wir wissen nicht was ersetzt wird, es kann l_1 sein:

$$S'' = \{s | l_2 \in s \in S\}.$$

- Nur eine Line sicher im Set
- *Alles bisheriges Wissen verloren!*

Analysebeginn: bekannter Zustand

- $S = \{([l_0, l_1, l_2, l_3], c)\}$
- Solange c bekannt ist können wir das ganze Set verfolgen
- Werte-Analyse kann genaue Adresse nicht ermitteln: $S' = S \cup S_{\bar{l}[c^{++}] \mapsto l_1}$
- Control flow join: $S' = S_1 \cup S_2$
- Sobald ein l_i an jeder Stelle im Cache stehen kann, läßt ein Zugriff auf l_i alle anderen Lines (vielleicht) rausfliegen, c bleibt unbekannt
- nichts wird wirklich rausgeworfen
- ... siehe Chaos-Cache

Analyse

- (maximal) eine Line pro Set sicher im Cache
  - Anfangs mehr, aber Wissen schnell verloren
- Keine wirklichen Cache-Misses
- Analyse bewegt sich in Teilmenge von $\mathcal{P}(\mathbb{S})$ die isomorph zu Direct-Mapped-Cache ist

Weitere Fragen

- Sichere Voraussagen nur begrenzt möglich
- Gibt es Caches ($\neq$ LRU), die analysierbar sind?
- Branch prediction:
  - gshare: PHT mit Hashfunktion
  - Welche Arten von Branch predictors sind analysierbar?
- Welche Pipelines „vergessen“ ihre Vorgeschichte?
- Soft Realtime: wie kombiniert man WCET und Wahrscheinlichkeiten/Statistik?