Projekt
Simulation Branch Prediction

Prof. Dr. Rafael Mayoral Malmström
Fakultät Informatik
Hochschule Kempten

18. Mai 2022

Zusammenfassung

Das Ziel dieses Projektes ist es die Vorhersagegenauigkeit von verschiedenen Methoden der Sprungvorhersage anhand von Simulationen für bestimmte *trace files* zu bestimmen. Damit soll ein Vergleich der unterschiedlichen Methoden möglich sein.

Implementieren und Simulieren Sie folgende Algorithmen zur Sprungvorhersage:

- Lokaler 2-Bit-Prädiktor
- Two-level global predictor. Das GHT ist 4 Bit lang und die *states* sind 2 Bit lang
- gshare. Das GHT ist 4 Bit lang und die *states* sind 2 Bit lang
- tournament. Das GHT ist 4 Bit lang und die globalen *states* sind 2 Bit lang. Für den lokalen Prädiktor sind die jeweiligen Historien 4 Bit lang und die *states* 2 Bit lang. Der *Prädiktor-Prädiktor*-Bit kommt noch dazu

Für die Lokalen Prädiktoren berücksichtigen Sie folgende zwei Varianten:

1. Alle Bits der Adresse einer Verzweigung werden als Index zu den Datenstrukturen verwendet, d.h. jede Verzweigung hat ihre eigenen Eintrag
2. Es werden nur die 10 Bits mit den niedrigsten Stellenwert der Adresse einer Verzweigung als Index zu den Datenstrukturen verwendet

Da wir nur an die Vorhersagegenauigkeit interessiert sind, ist der Branch Target Buffer (BTB) nicht erforderlich.

Für die Simulation verwenden Sie die auf moodle zur Verfügung gestellten *trace*-Dateien. Diese Dateien bestehen aus Zeilen, die jeweils eine Verzweigungsadresse zusammen mit dem Tatsächlichen Ergebnis der Verzeigung (taken/not taken) beinhalten. Die Kodierung ist wie folgt:

- 0: not taken: kein Sprung, die Ausführung bleibt sequentiell

- 1: taken: Sprung, der Programmfluss ändert sich

Mit den Ergebnissen der Simulationen, können Sie folgende Fragen beantworten:

- Welche Vorhersagegenauigkeit wird mit den verschiedenen Methoden für die *traces* erreicht?
- Welcher Einfluss hat die Verwendung von nur partiellen Addresen?
- Welche andere Beobachtungen haben Sie gemacht?

Weitere Anmerkungen zum Projekt

- Sie dürfen eine beliebige Programmiersprache verwenden. Standard-Programmiersprachen (C, C++, Java, python, C#, usw.) gestalten das Verstehen des Codes einfacher
- Stellen Sie genug Dokumentation zur Verfügung, um die Programme ausführen zu können, z.B. erläutern Sie eventuelle Argumente
- Sie können Ihre Ergebnisse über moodle hochladen (eine Zip-Datei) oder einen Link zu einem Github-Repo zur Verfügung stellen
- Um die Methoden noch generischer zu gestalten, könnten Sie Argumente vorsehen, die z.B. die Länge des GHT oder der States bestimmen. Dann können Sie verschiedene Varianten der gleichen Methode ebenfalls vergleichen