Praktikum Compilerbau
Sitzung 8 – Optimierungen: Common Subexpression Elimination

Lehrstuhl für Programmierparadigmen
Universität Karlsruhe (TH)

10. Juni 2009

1. Letzte Woche
2. Common Subexpression Elimination
3. Sonstiges

Letzte Woche

- Was waren die Probleme?
- Hat soweit alles geklappt?

1 Letzte Woche

2 Common Subexpression Elimination

3 Sonstiges

Common Subexpression Elimination

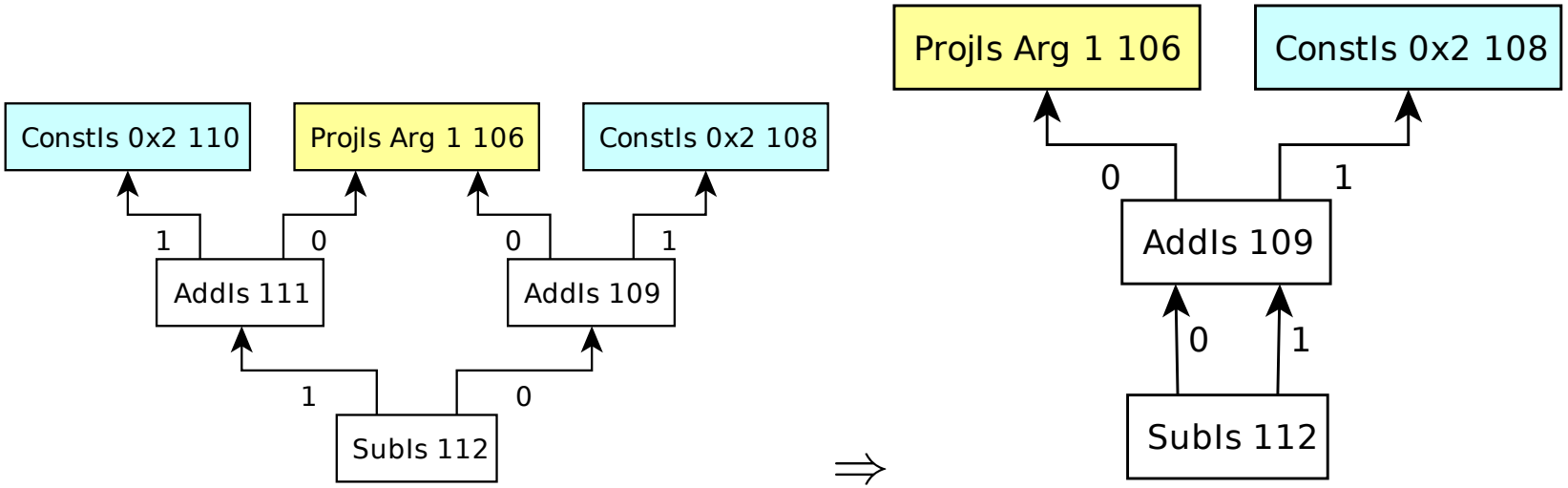
Ziel:

- Vermeidung wiederholter Berechnungen

Vorgehensweise:

- Identifiziere Knoten die den gleichen Wert berechnen.
- Ersetze alle Verwender durch einen der Knoten.

Beispiel


ConstIs 0x2 110
ProjIs Arg 1 106
ConstIs 0x2 108
1
0
0
1
AddIs 111
AddIs 109
1
0
SubIs 112
⇒
ProjIs Arg 1 106
ConstIs 0x2 108
0
1
AddIs 109
0
1
SubIs 112

Werte

Frage: Wann berechnen 2 Knoten den gleichen Wert?

Werte

Frage: Wann berechnen 2 Knoten den gleichen Wert?

- Wenn Sie die gleiche Operation mit dem gleichen Mode durchführen.

Werte

Frage: Wann berechnen 2 Knoten den gleichen Wert?

- Wenn Sie die gleiche Operation mit dem gleichen Mode durchführen.
- Wenn Sie die gleichen Operanden besitzen.

Werte

Frage: Wann berechnen 2 Knoten den gleichen Wert?

- Wenn Sie die gleiche Operation mit dem gleichen Mode durchführen.
- Wenn Sie die gleichen Operanden besitzen.
- Wenn Sie sich im gleichen Grundblock befinden (globale Common Subexpression Elimination wird hier nicht betrachtet).

Werte

Frage: Wann berechnen 2 Knoten den gleichen Wert?

- Wenn Sie die gleiche Operation mit dem gleichen Mode durchführen.
- Wenn Sie die gleichen Operanden besitzen.
- Wenn Sie sich im gleichen Grundblock befinden (globale Common Subexpression Elimination wird hier nicht betrachtet).
- Wenn ihre Attribute gleich sind (wichtig z.B. bei `Proj`, `SymConst`, `Const`, `Sel`)

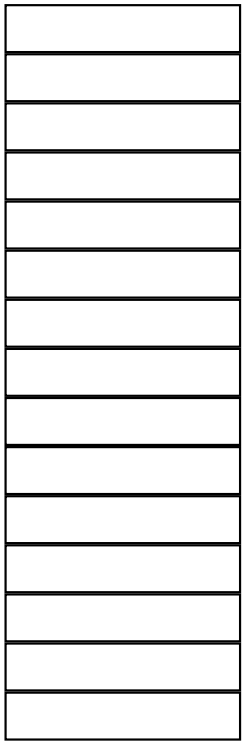
Wertnummerierung

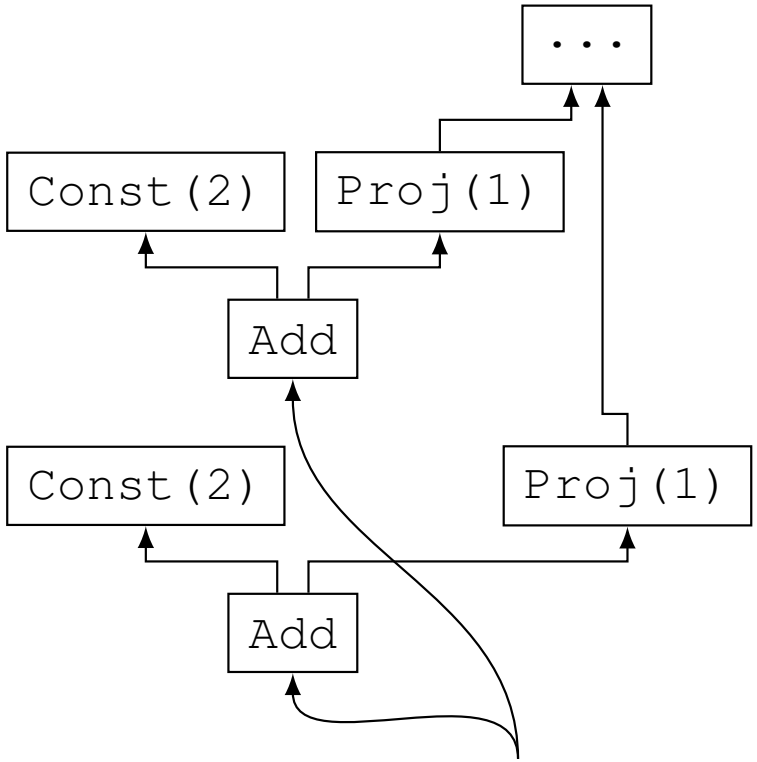
Für einen gegebenen Knoten x, wie finde ich einen Knoten der den gleichen Wert berechnet?

- Erzeuge Hashmap für alle Knoten.
- Berechne Hashwert aus Kriterien von der letzten Folie. Frage: Alle Kriterien nutzen oder nur eine Teilmenge?
- Für jeden Knoten:
  - Betrachte alle Knoten in der Hashmap mit gleichem Hashwert.
  - Falls Kriterium für Gleichheit bei einem Knoten erfüllt, ersetze Knoten durch den Knoten in der Hashmap.
  - Falls kein gleicher Knoten gefunden wurde: Einfügen in die Hashmap.

Beispiel

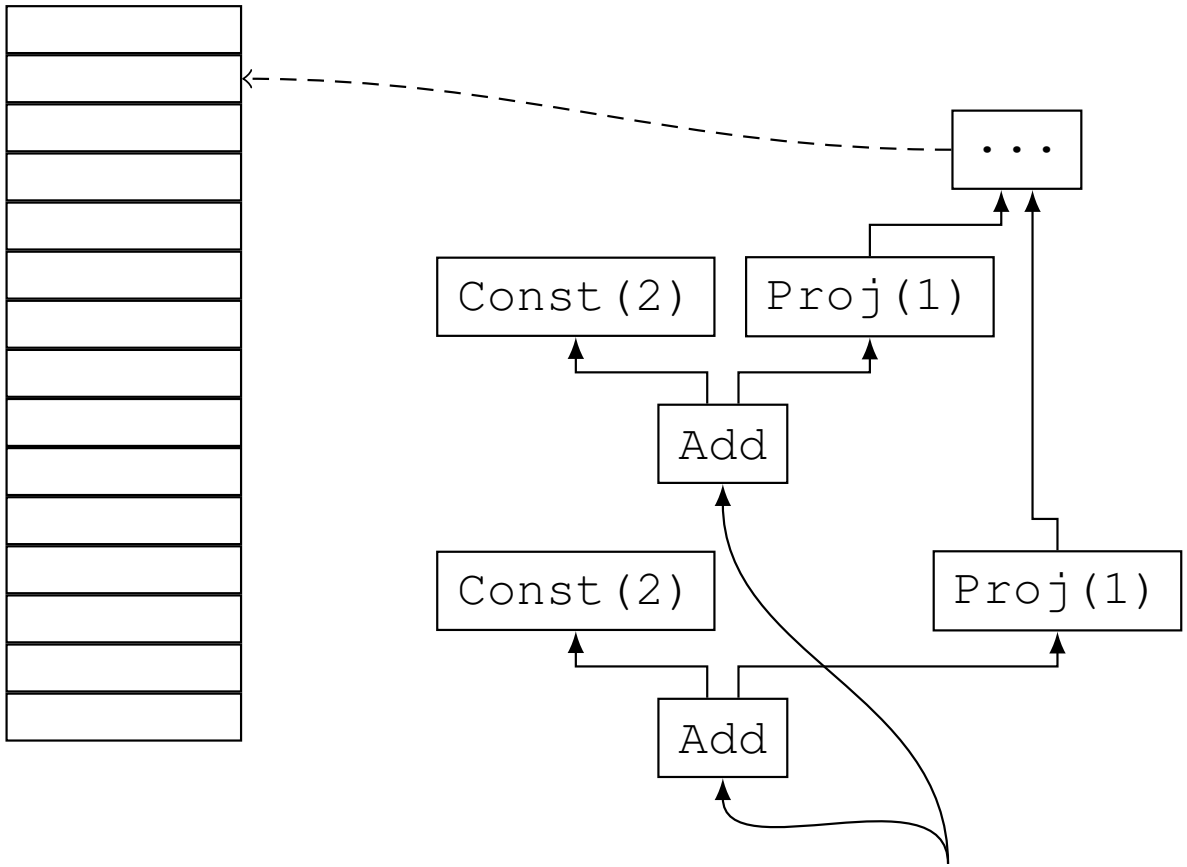
Hashtabelle





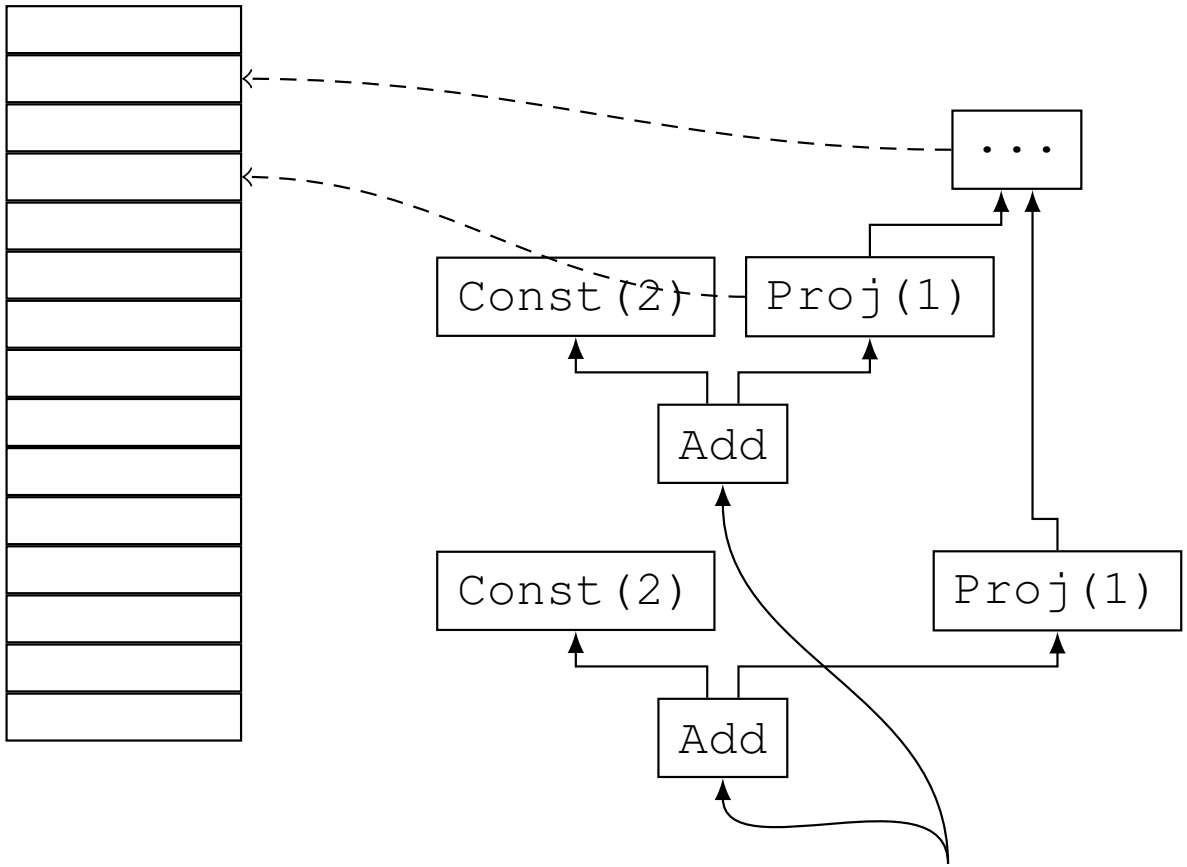
Beispiel

Hashtabelle



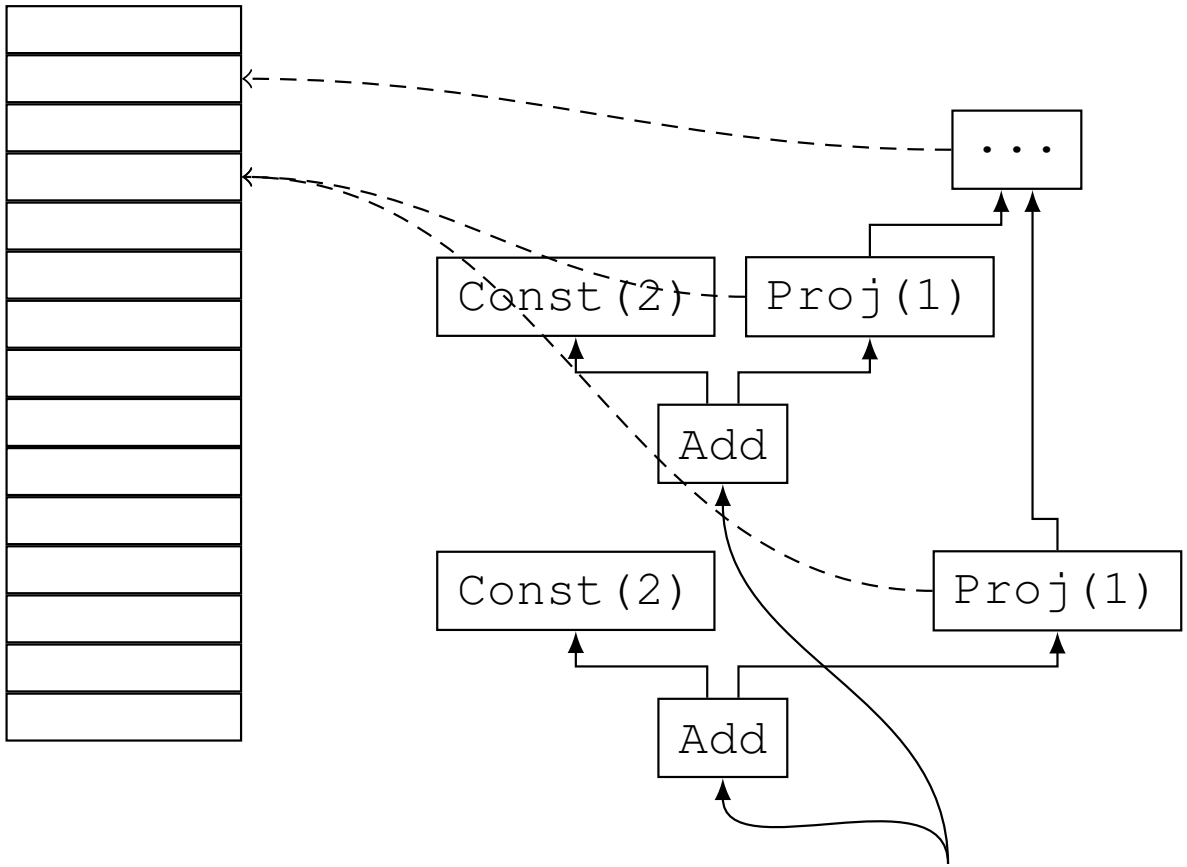
Beispiel

Hashtabelle



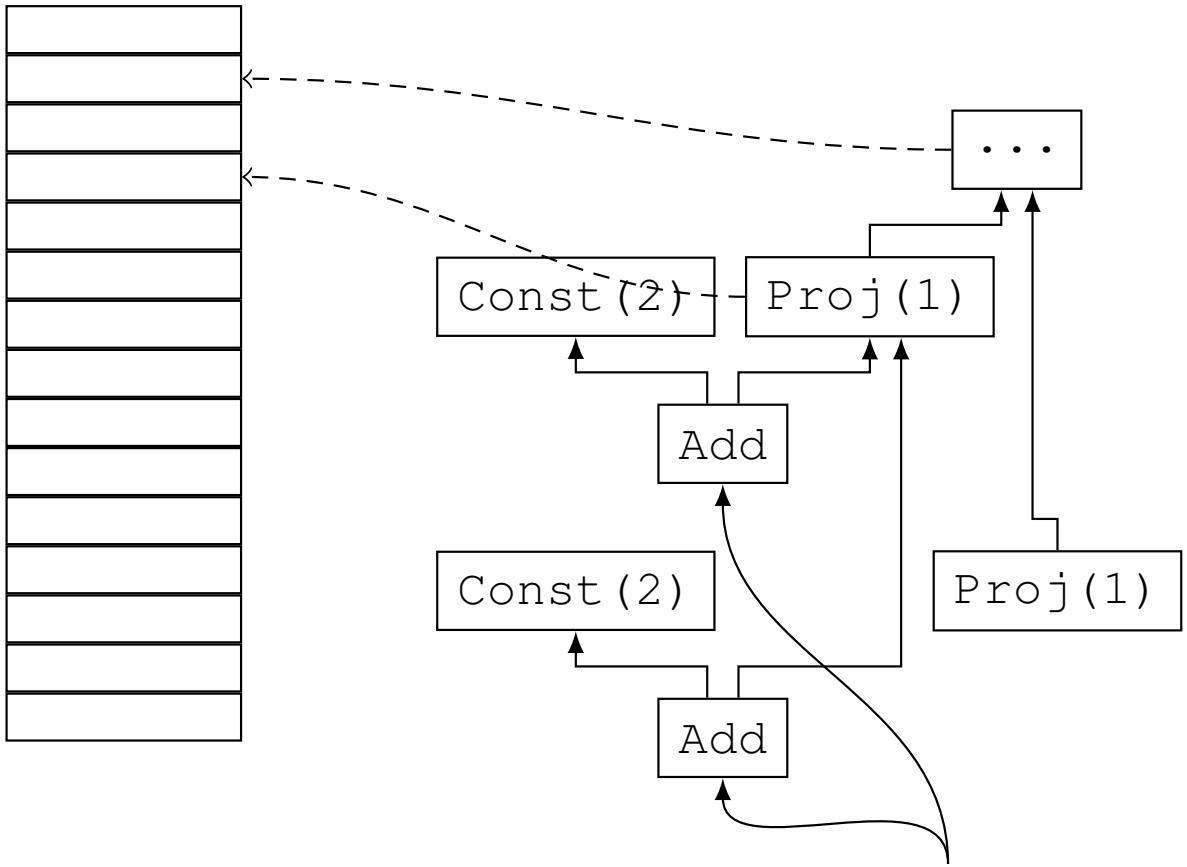
Beispiel

Hashtabelle



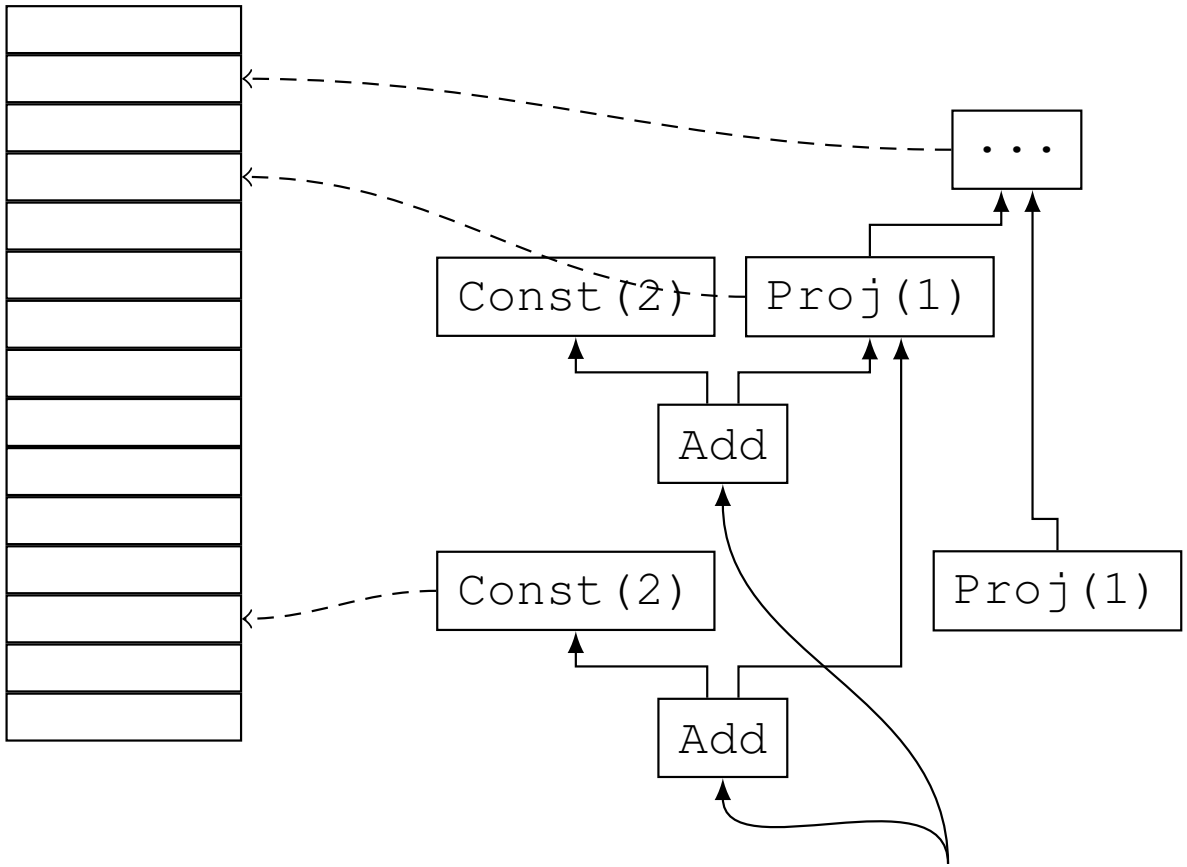
Beispiel

Hashtabelle



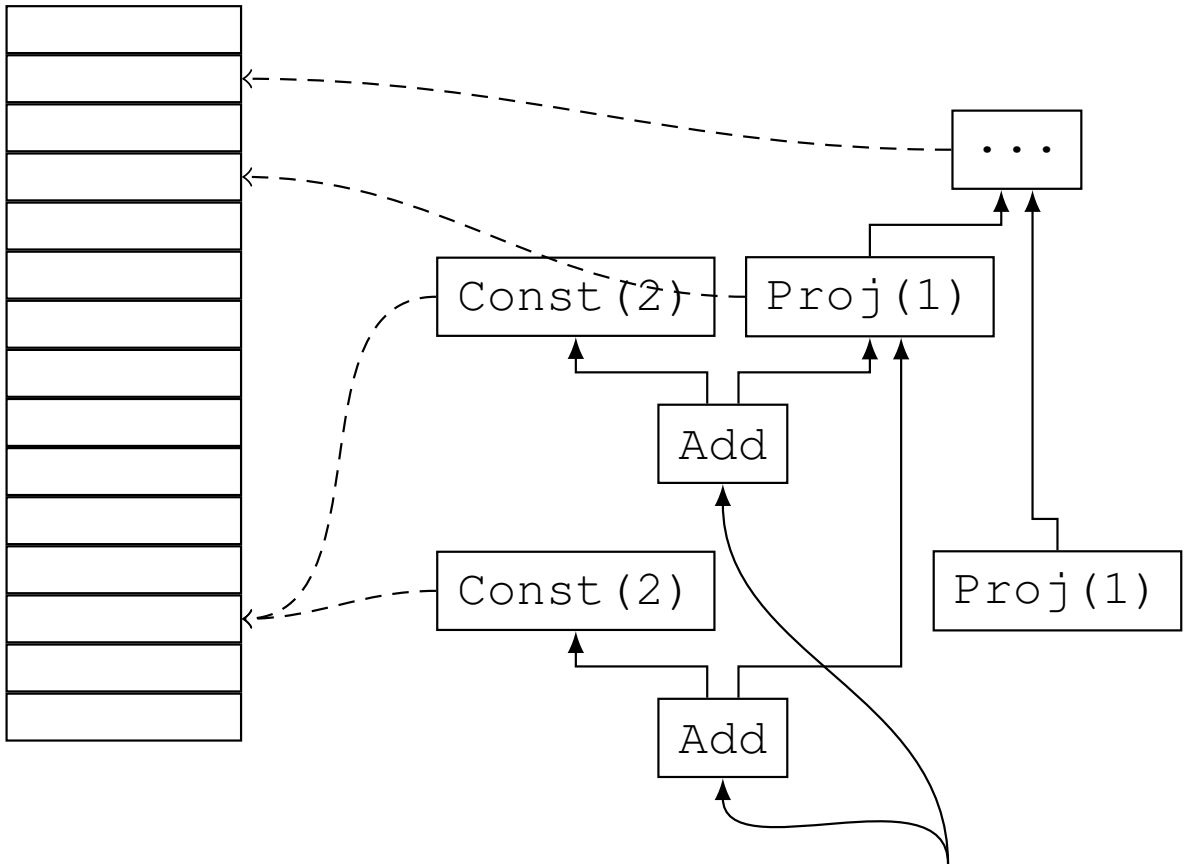
Beispiel

Hashtabelle



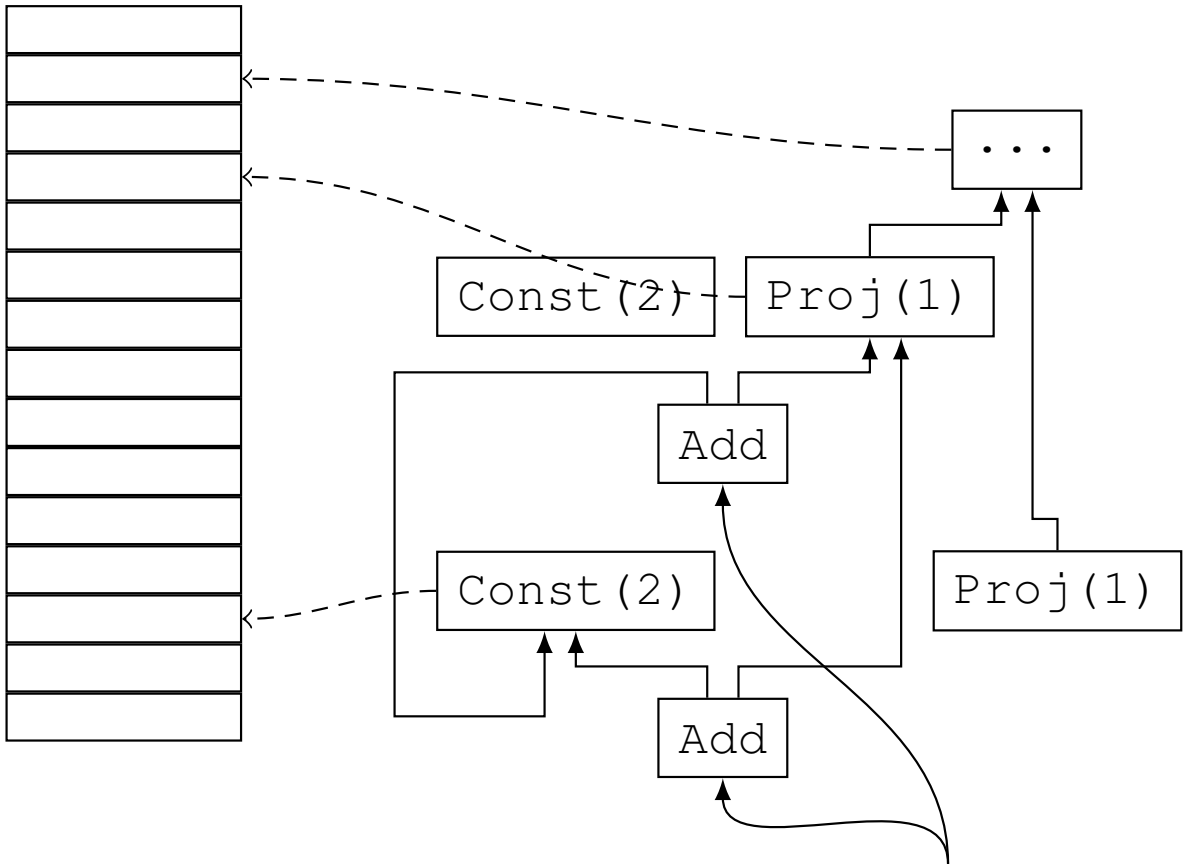
Beispiel

Hashtabelle



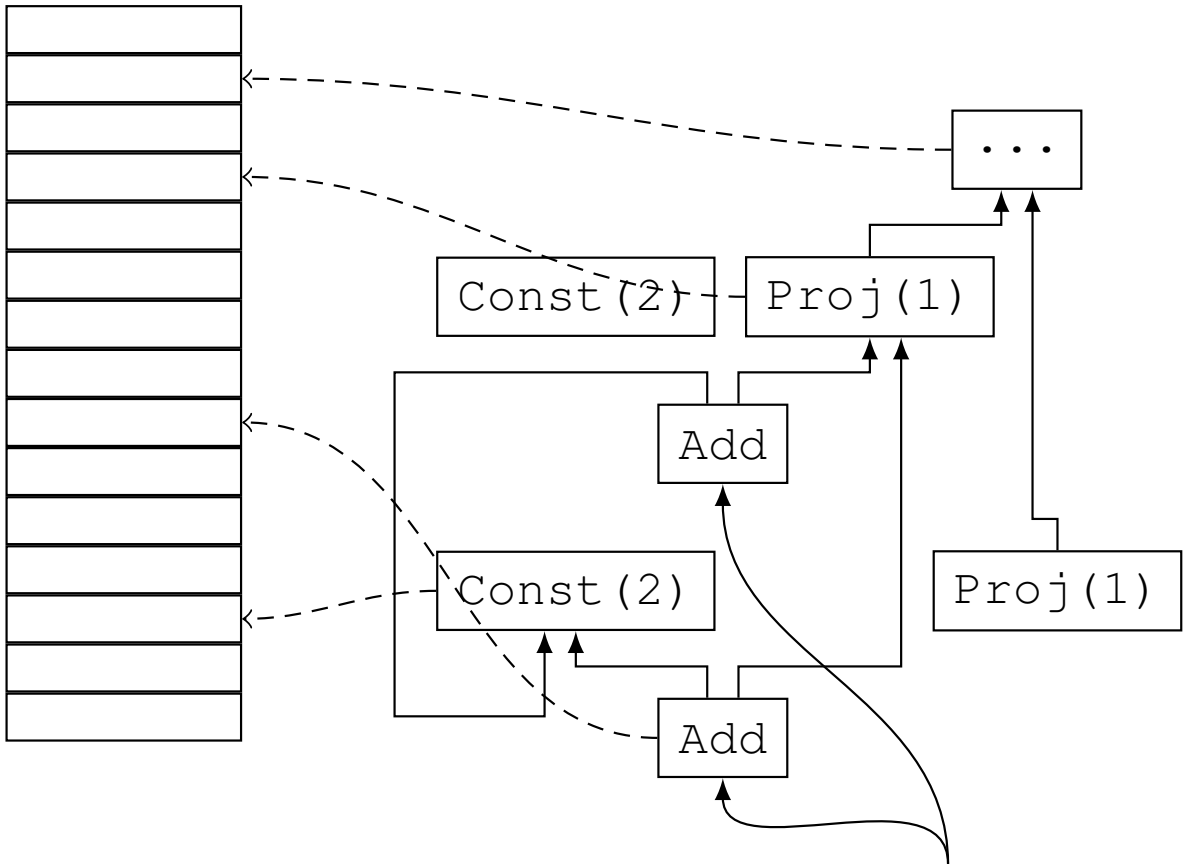
Beispiel

Hashtabelle



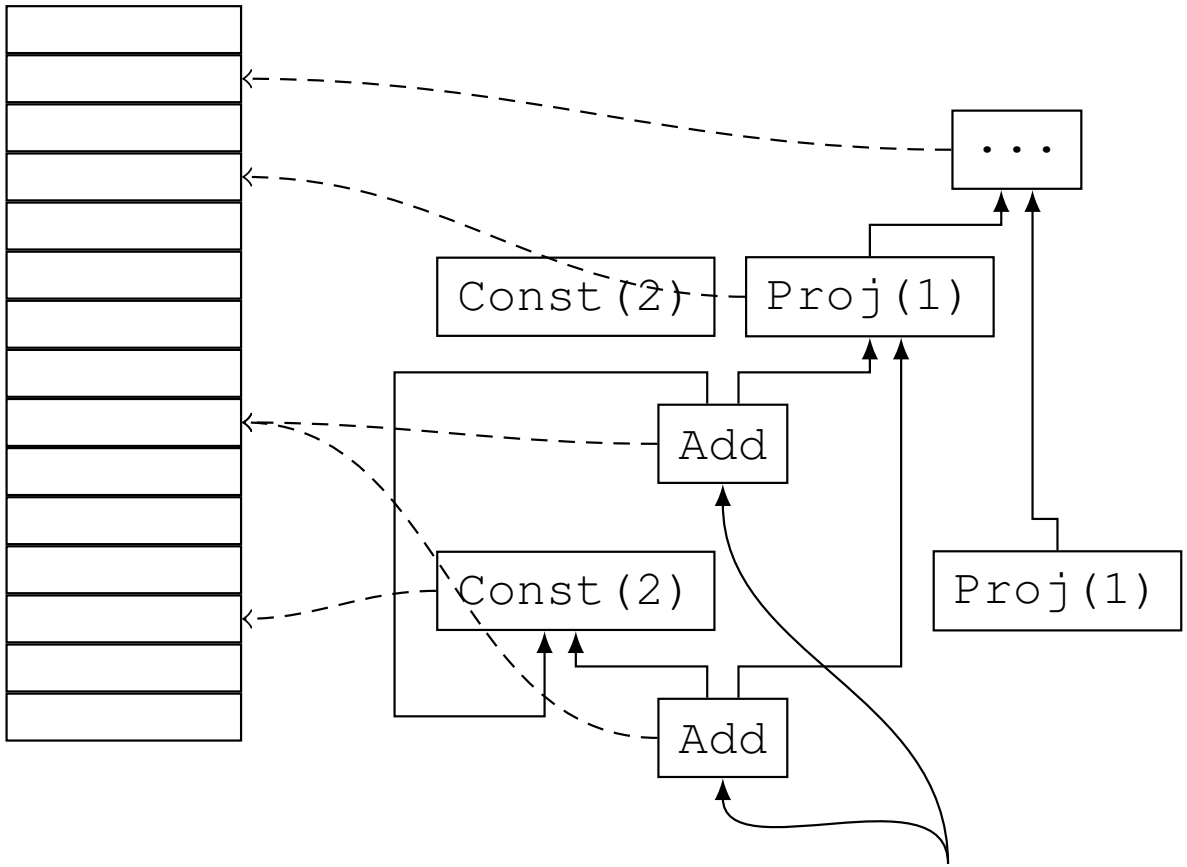
Beispiel

Hashtabelle



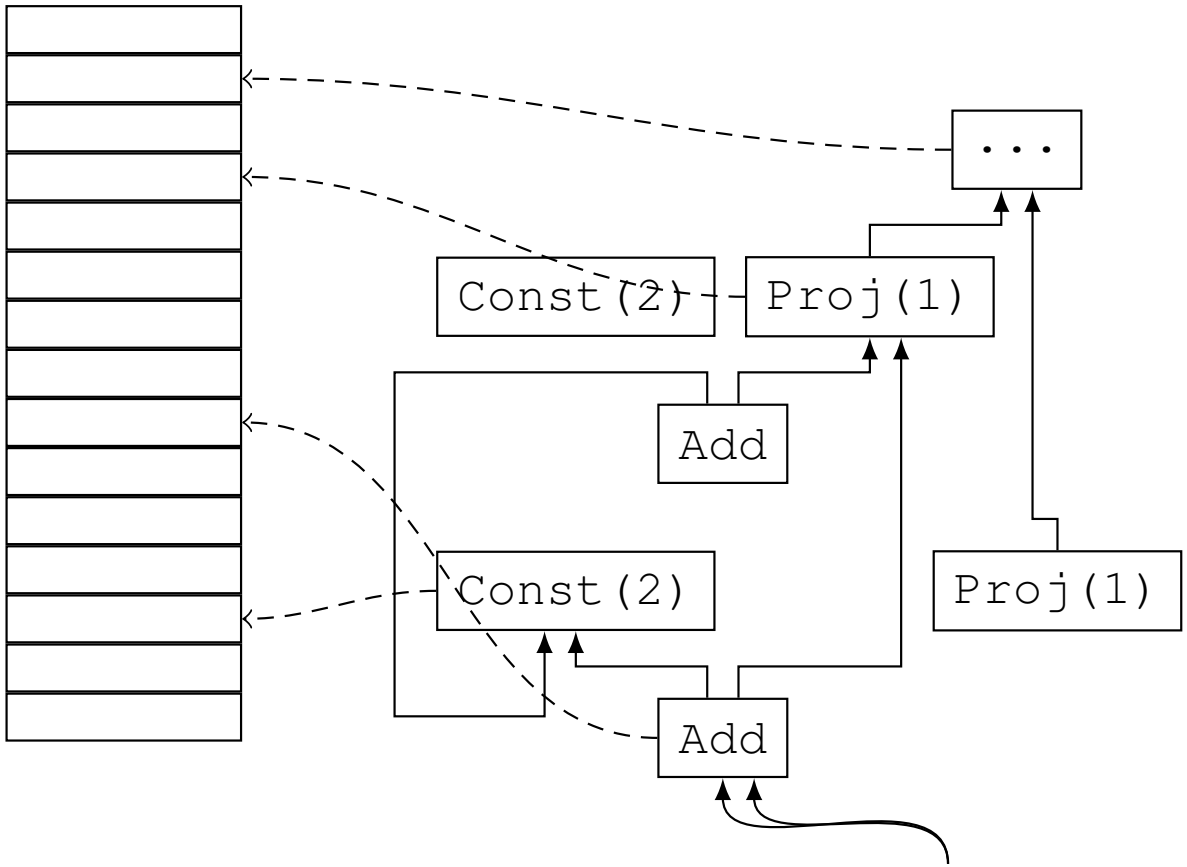
Beispiel

Hashtabelle



Beispiel

Hashtabelle



Achtung

Probleme:

- Sinnvolle Reihenfolge für CSE? Austauschen von Knoten kann zeigen, dass weitere Knoten gleich sind.
- Kombination mit lokalen Optimierungen.

Java Implementierung

Standard-Bibltiothek enthält bereits Implementierung einer Hashtabelle: `java.util.HashMap<K,T>`.
Problem: `Node` Klasse vorgegeben, eigene `equals` und `hashCode` Methoden nicht möglich.

Java Implementierung

Standard-Bibltiothek enthält bereits Implementierung einer Hashtabelle: `java.util.HashMap<K,T>`.
Problem: `Node` Klasse vorgegeben, eigene `equals` und `hashCode` Methoden nicht möglich.

Lösung: Benutze Wrapperklasse für Schlüsselwerte in der Hashtabelle:

```
public class CSENode {
  private Node node;
  public CSENode(Node node) {
    this.node = node;
  }
  @Override
  public int hashCode() { /* ... */ }
  @Override
  public int equals(Object other) { /* ... */ }
}
```

1 Letzte Woche

2 Common Subexpression Elimination

3 Sonstiges

Feedback! Fragen? Probleme?

- Anmerkungen?
- Probleme?
- Fragen?