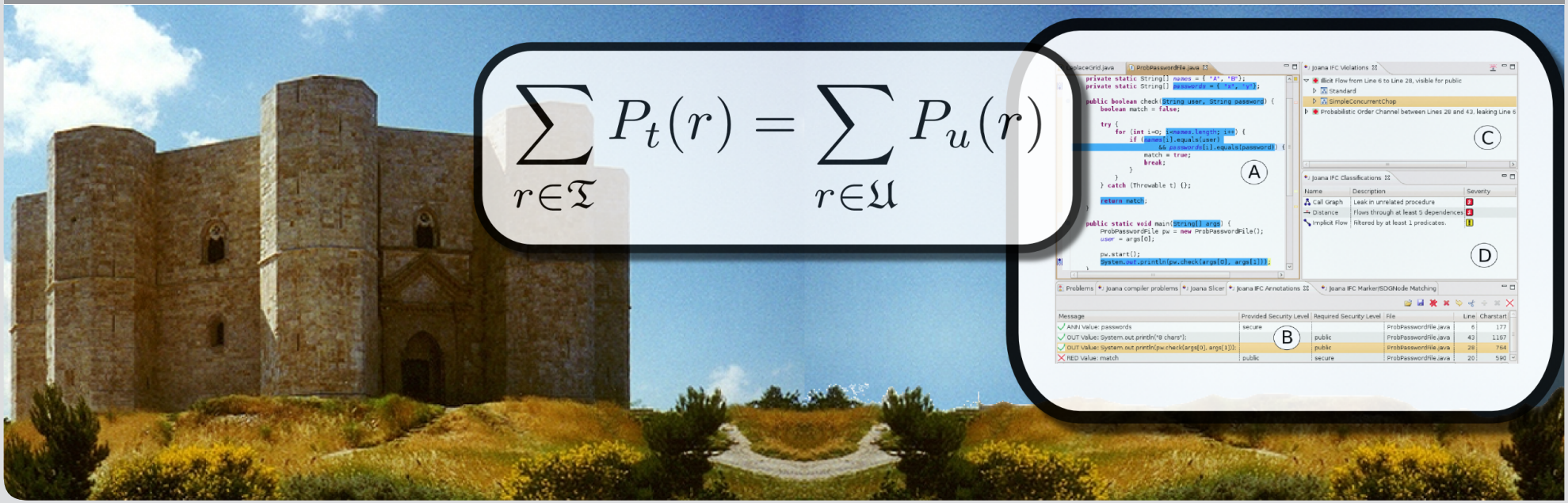
Karlsruher Institut für Technologie

PSE: Programmabhängigkeitsgraphen

M. Radermacher, S. Bischof, S. Buchwald, M.Hecker

IPD Snelting, ITI Wagner



Ein einfaches Programm

```
public int m(int y, int z) {
    int calc = z*z;
    int result = calc-1;
    if (y < 0) {
        result = 0;
    }
    return result;
}
```

- Hängt das Ergebnis von **z** ab?
- Und hängt es von **y** ab?

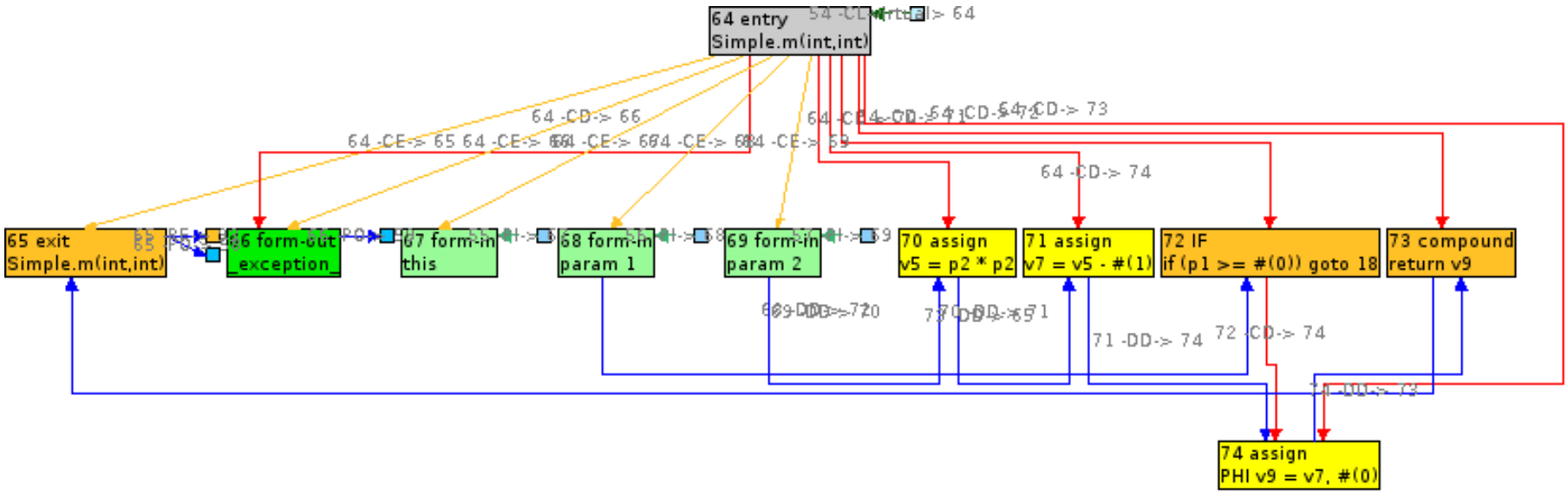
Informationsflusskontrolle

- es gibt geheime und öffentliche Werte
- Fragestellung: Kann der Wert einer geheimen Eingabe eine öffentliche Ausgabe beeinflussen?
- In diesem Fall kann ein Angreifer etwas über geheime Werte lernen

Informationsflusskontrolle

- es gibt geheime und öffentliche Werte
- Fragestellung: Kann der Wert einer geheimen Eingabe eine öffentliche Ausgabe beeinflussen?
- In diesem Fall kann ein Angreifer etwas über geheime Werte lernen
- Allein mit Programmcode für Menschen schwierig zu erkennen
- Graphen helfen bei der Visualisierung
- Hier am Lehrstuhl entwickelt: JOANA

Der Programmabhängigkeitsgraph (PDG)



Wie entsteht ein solcher PDG?

- Knoten sind Statements und Ausdrücke
- 2 wichtige Arten von Kanten

Wie entsteht ein solcher PDG?

- Knoten sind Statements und Ausdrücke
- 2 wichtige Arten von Kanten
  - Datenabhängigkeitskanten (blau), falls der Wert eines Knotens für die Berechnung eines anderen Knotens benötigt wird
  - Kontrollabhängigkeitskanten (rot), falls ein Knoten entscheidet, ob ein anderer ausgeführt wird
  - In strukturierten Sprachen wie Java: Eine (if, while, ...)-Bedingung kontrolliert die Statements in dem entsprechenden Rumpf

Wie entsteht ein solcher PDG?

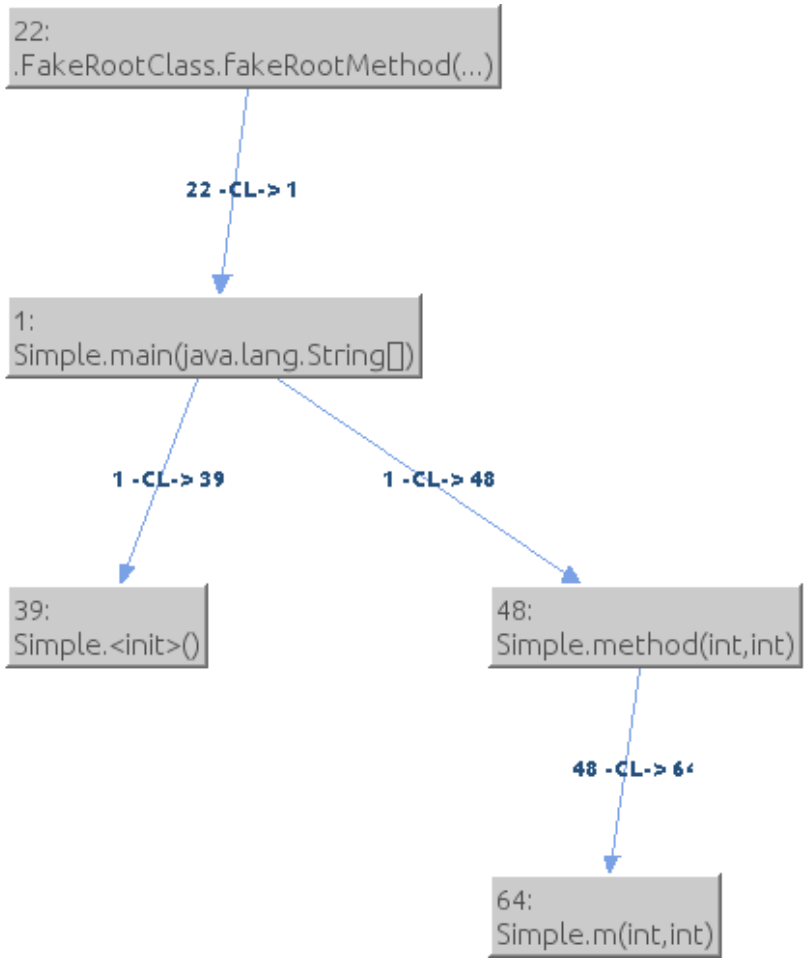
- Knoten sind Statements und Ausdrücke
- 2 wichtige Arten von Kanten
  - Datenabhängigkeitskanten (blau), falls der Wert eines Knotens für die Berechnung eines anderen Knotens benötigt wird
  - Kontrollabhängigkeitskanten (rot), falls ein Knoten entscheidet, ob ein anderer ausgeführt wird
  - In strukturierten Sprachen wie Java: Eine (if, while, ...)-Bedingung kontrolliert die Statements in dem entsprechenden Rumpf
- JOANA kann aus Source- oder Bytecode den PDG generieren

Wie sieht es aus bei mehreren Methoden?

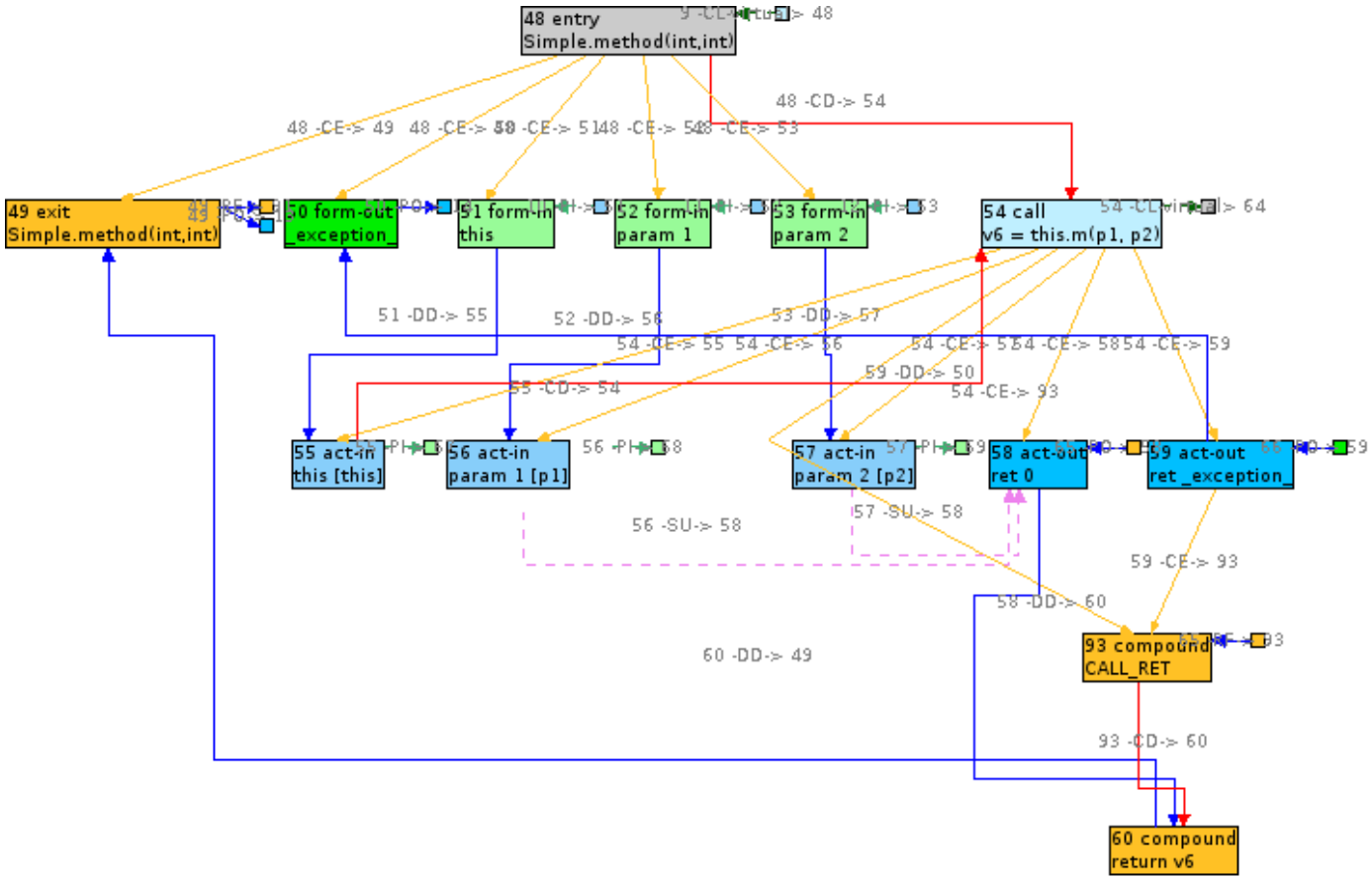
```
public int m(int y, int z) {
    int calc = z*z;
    int result = calc-1;
    if (y < 0) {
        result = 0;
    }
    return result;
}

public int method(int a, int b) {
    return m(a, b);
}
```

Wie sieht es aus bei mehreren Methoden?



Wie sieht es aus bei mehreren Methoden?



Wie sieht es aus bei mehreren Methoden?

Karlsruher Institut für Technologie

Zusätzliche Knoten:

- Einfügen eines Call-Knotens
- Knoten für berechnete Parameter
- Knoten für Verwendung des Rückgabewerts

Wie sieht es aus bei mehreren Methoden?

Karlsruher Institut für Technologie

Zusätzliche Knoten:

- Einfügen eines Call-Knotens
- Knoten für berechnete Parameter
- Knoten für Verwendung des Rückgabewerts

Zusätzliche Kanten:

- Call-Knoten zum Eintrittsknoten der aufgerufenen Funktion (gelb)
- Call-Knoten zu den anderen eingefügten Knoten (gelb)
- Berechnete Parameter zu den Parameterknoten in der aufgerufenen Funktion
- Rückgabewert in der aufgerufenen Funktion zu dessen Verwendung
- Summary-Kanten (lila gestrichelt)

Summary-Kanten

- Abhängigkeiten können durch Methodenaufrufe gehen
- Problem: Direkte Modellierung algorithmisch schwierig

Summary-Kanten

Karlsruher Institut für Technologie

- Abhängigkeiten können durch Methodenaufrufe gehen
- Problem: Direkte Modellierung algorithmisch schwierig
- Lösung: Summary-Kanten in aufrufender Funktion fassen solche Abhängigkeiten zusammen
- Verbinden Parameter mit der Verwendung des Rückgabewerts

Spezielle Anforderungen

- Pfade zwischen Knoten zeigen, falls solche existiert
- Einfache Navigation zwischen aufrufender und aufgerufener Funktion
- Für Summary-Kanten: Möglichkeit, den zusammengefassten Pfad zu sehen

Spezielle Anforderungen

- Pfade zwischen Knoten zeigen, falls solche existiert
- Einfache Navigation zwischen aufrufender und aufgerufener Funktion
- Für Summary-Kanten: Möglichkeit, den zusammengefassten Pfad zu sehen
- Genaue Ausgestaltung frei
- Eigene Ideen willkommen (Wunschkriterien)

Zum selbst ausprobieren

- JOANA-Webseite: `http://pp.ipd.kit.edu/projects/joana/` mit IFC-Konsole und Graphviewer als Webstart-Anwendungen
- Eigenes Programm in *.jar-Datei übersetzen
- IFC-Konsole starten, Datei auswählen, PDG bauen und speichern
- Dann mit Graphviewer anschauen