



Abschlusspräsentation

B. Böhm A. Nüßlein M. Popov T. Rakowski J. Rekittke A. Sulfrian
Freie Universität Berlin

Softwareprojekt Compilerbau, SoSe 2009

- ▶ erfinden der Quellsprache
 - ▶ Paradigma: imperativ, prozedural
 - ▶ ~~Typsystem: static, weak, infered~~
- ▶ erstellen der EBNF

Gruppe: Quellsprache (Beispiel)

```
fun fib[a]
  if a < 2
    @1
  end
  @(call fib[a-1] + call fib[a-2])
end
```

```
fun main[]
  sum = 0
  i = 1
  while i <= 10
    sum = sum + call fib[i]
    i = i + 1
  end
  @sum
end
```

- ▶ Lexer \Rightarrow Parser:
 - ▶ **Tokenstream**
- ▶ Parser \Rightarrow Zwischencode:
 - ▶ **AST**
- ▶ Zwischencode \Rightarrow Codegenerator:
 - ▶ **3AC**
- ▶ Codegenerator \Rightarrow Assemblercode:
 - ▶ **vereinfachte MIPS ISA**

- ▶ Tokens:
 - ▶ NUMBER, BOOL, IDENT
 - ▶ ASSIGNMENT
 - ▶ OPERATOR
 - ▶ FUN, RETURN
 - ▶ WHILE, IF, ELSE
 - ▶ LBRAK, RBRAK, LPAREN, RPAREN, NEWLINE, COMMA, END
- ▶ Übersetzungsregeln mit Regulären Ausdrücken

- ▶ Typinferenz bei Initialisierung von Variablen
- ▶ Statische Typzuweisung in der Symboltabelle
- ▶ ggf. automatische Typkonvertierung

- ▶ Top-Down-Parsing
- ▶ Ableitungsbaum:
 - ▶ Program(Node)
 - ▶ Function(Node)
 - ▶ Statement(Node)
 - ▶ Sequence(Statement)
 - ▶ IfStatement(Statement)
 - ▶ WhileStatement(Statement)
 - ▶ ReturnStatement(Statement)
 - ▶ AssignStatement(Statement)
 - ▶ FunctionCall(Statement,Expression)
 - ▶ Operation(Node)
 - ▶ Expression(Operation)
 - ▶ UnaryExpression(Expression)
 - ▶ BinaryExpression(Expression)
 - ▶ Variable(Expression)
 - ▶ Constant(Expression)

- ▶ Aufbau der Symboltabelle
- ▶ Gültigkeitsbereiche der Bezeichner überprüfen
- ▶ eindeutige Zuweisungen von Bezeichnern

▶ Drei-Adress-Code (SSA Form):

- ▶ Binäre Operationen: ADD, SUB, MUL, DIV, MOD, AND, OR
- ▶ Unäre Operationen: NOT, MINUS
- ▶ Zuweisungen: ASSIGN
- ▶ Vergleiche: EQ, NE, LE, LT, GE, GT
- ▶ Bedingte Sprünge: BEZ
- ▶ Nichtbedingte Sprünge: JMP
- ▶ Funktionsaufrufe: PARAM, CALL, RETURN

- ▶ ~~Eliminierung gemeinsamer Teilausdrücke~~
- ▶ ~~Kopiepropagation~~
- ▶ ~~Entfernen von totem Code~~
- ▶ ~~Konstantenfaltung~~

- ▶ Zuweisung von Adressen/Registervergabe
- ▶ Zerlegen von Drei-Adress-Code in Grundblöcke
- ▶ Befehlsauswahl
- ▶ Aufrufkonventionen
- ▶ Ausgabe als Assemblercode

- ▶ Erfundene RISC ISA:
 - ▶ Arithmetik: ADD, SUB, MUL, DIV, MOD
 - ▶ Logik: AND, OR
 - ▶ Load & Store: MOV, LW, SW, PUSH, POP
 - ▶ Vergleiche: CMP, EQ, NE, LE, LT, GE, GT
 - ▶ Sprünge: BEZ, JMP, CALL, RET
 - ▶ Andere: SYS
- ▶ keine Memory-Management-Unit
- ▶ I/O durch Nachbildung von System-Calls