

Small step Semantik

Beispiel zur λ -Abstraktion mit einem Argument vom Produkttyp: *sum_of_squares*

```
let val sum_of_squares =  $\lambda z:\text{int} * \text{int}. \#1 z * \#1 z + \#2 z * \#2 z$   
in sum_of_squares (3, 4)  
end
```

steht für

```
let val sum_of_squares =  $\lambda z:\text{int} * \text{int}. + (* (\#1 z, \#1 z), * (\#2 z, \#2 z))$   
in sum_of_squares (3, 4)  
end
```

→ $(\lambda z:\text{int} * \text{int}. + (* (\#1 z, \#1 z), * (\#2 z, \#2 z))) (3, 4)$
mit Regel (LET-EXEC)

→ $+ (* (\#1 (3, 4), \#1 (3, 4)), * (\#2 (3, 4), \#2 (3, 4)))$
mit Regel (BETA-V)

→ $+ (* (3, \#1 (3, 4)), * (\#2 (3, 4), \#2 (3, 4)))$
mit Regel (PROJ)

→ $+ (* (3, 3), * (\#2 (3, 4), \#2 (3, 4)))$
mit Regel (PROJ)

→ $+ (9, * (\#2 (3, 4), \#2 (3, 4)))$
mit Regel (OP)

→ $+ (9, * (4, \#2 (3, 4)))$
mit Regel (PROJ)

→ $+ (9, * (4, 4))$
mit Regel (PROJ)

→ $+ (9, 16)$
mit Regel (OP)

→ 25
mit Regel (OP)