



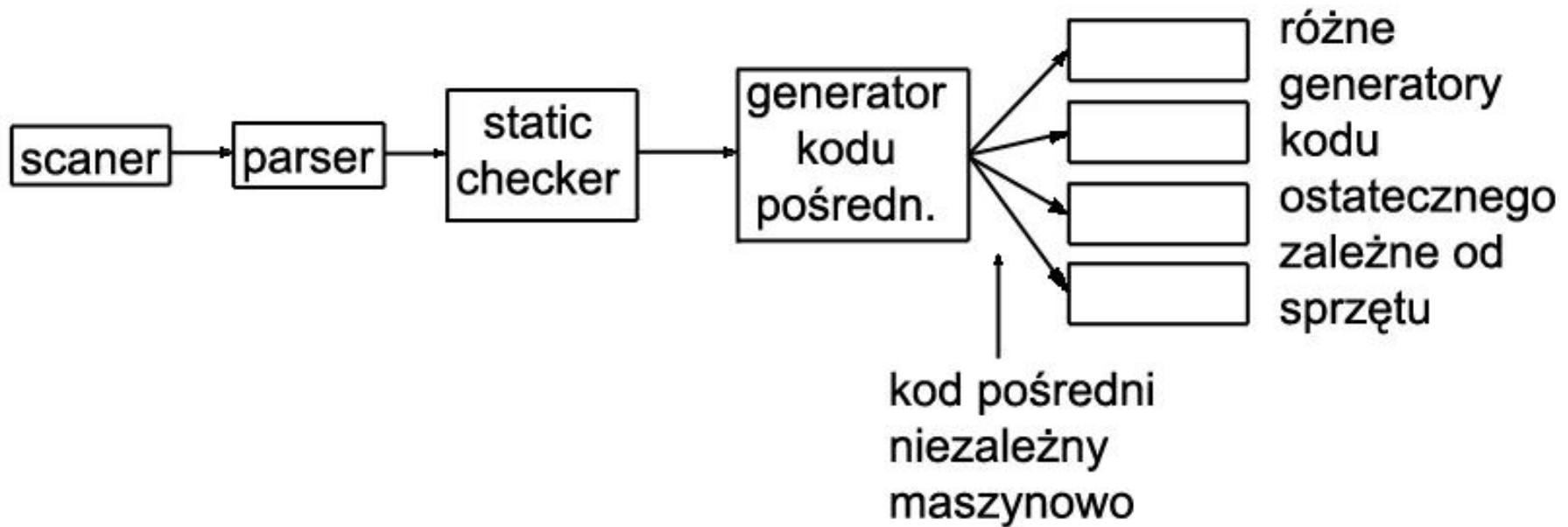
**AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE**

Generacja kodu pośredniego

**Dr inż. Janusz Majewski
Języki formalne i automaty**

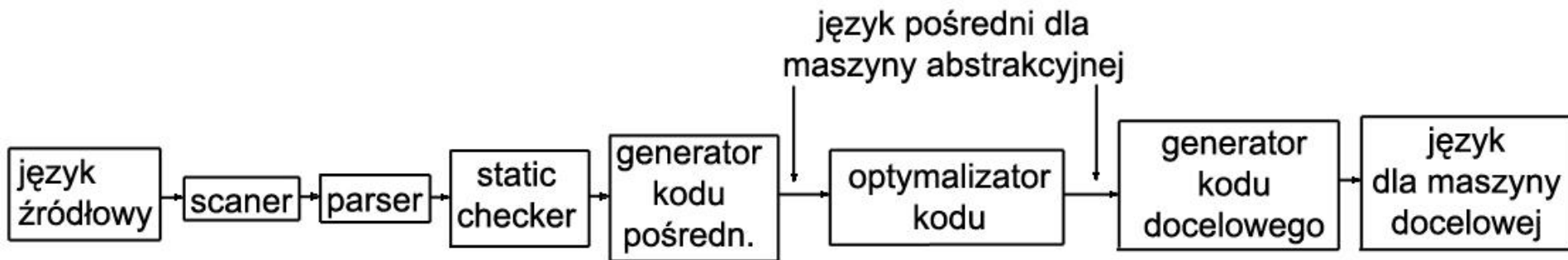
Przyczyny dwustopniowego tłumaczenia

- Łatwość generowania kompilatorów tego samego języka dla różnych platform systemowo-sprzętowych



Przyczyny dwustopniowego tłumaczenia

- *Łatwość przeprowadzania optymalizacji na bazie kodu pośredniego niezależnie od sprzętu*



Języki kodu pośredniego

Języki kodu pośredniego są językami dla pewnej maszyny abstrakcyjnej :

- Odwrotna notacja polska (notacja postfiksowa) → maszyna stosowa
- Drzewa syntaktyczne lub grafy skierowane acykliczne
- Kod trójadresowy

Przykład – translacja wyrażeń do odwrotnej notacji polskiej (ONP)

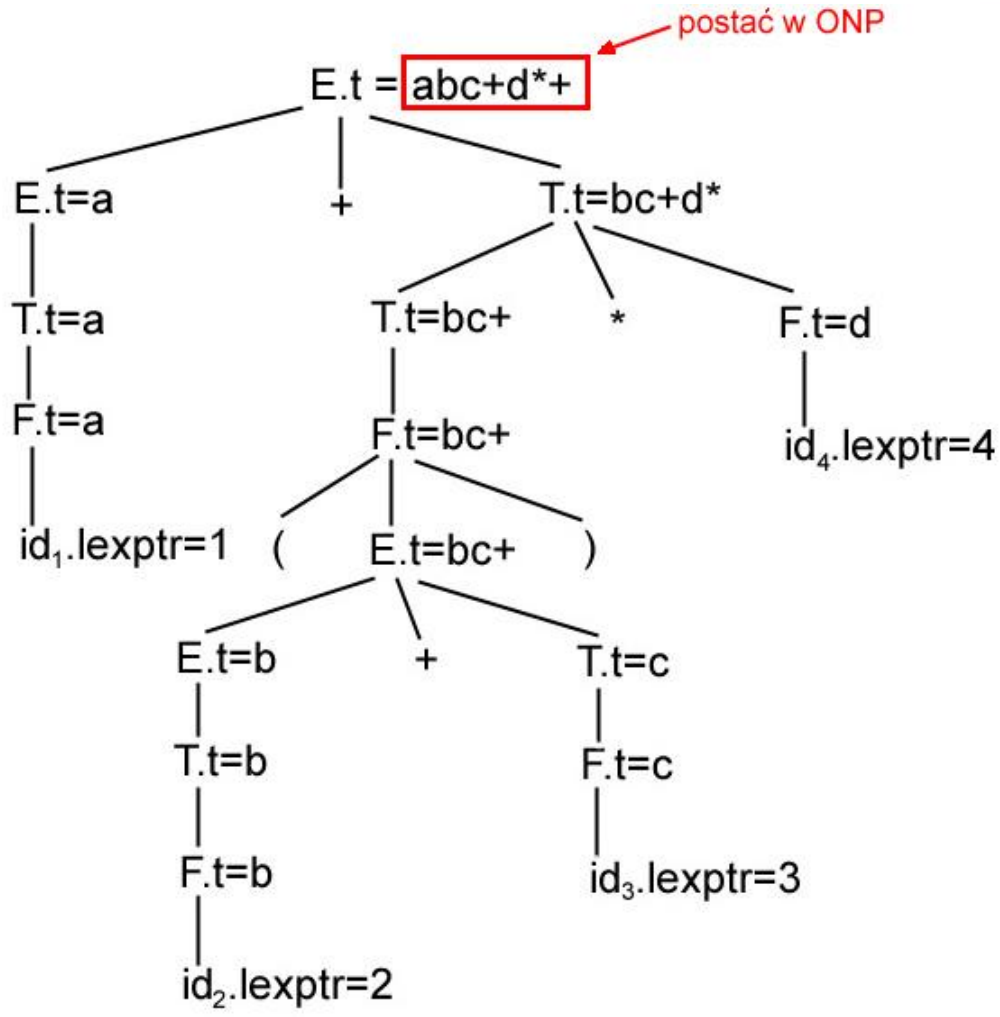
$E \rightarrow E_1 + T$	$E.t \leftarrow E_1.t \parallel T.t \parallel '+'$
$E \rightarrow T$	$E.t \leftarrow T.t$
$T \rightarrow T_1 * F$	$T.t \leftarrow T_1.t \parallel F.t \parallel '**'$
$T \rightarrow F$	$T.t \leftarrow F.t$
$F \rightarrow (E)$	$F.t \leftarrow E.t$
$F \rightarrow \underline{id}$	$F.t \leftarrow \text{name}(\underline{id.lexptr})$

gdzie: \parallel - operator konkatencji tekstów

Rozważane słowo źródłowe: **a+(b+c)*d**

Po analizie leksykalnej: **id₁+(id₂+id₃)*id₄**

<u>id.lexptr</u>	name(<u>id.lexptr</u>)
1	a
2	b
3	c
4	d





Maszyna wirtualna działająca w oparciu o ONP

Przykład:

- źródło:
- ONP:

maszyna wirtualna = maszyna stosowa

day := (1461 * y) div 4 + (153 * m + 2) div 5 + d

day 1461 y * 4 div 153 m * 2 + 5 div + d + :=

Instrukcje maszyny stosowej:

- **push v** - złożenie stałej na stos
- **rvalue l** - złożenie zawartości l na stos
- **lvalue l** - złożenie adresu l na stos
- **(operacja, np: +)** - wykonanie operacji na dwóch argumentach na wierzchołku stosu i bezpośrednio pod wierzchołkiem, zdjęcie obu argumentów ze stosu złożenie tam wyniku.
- **:=** - r-wartość z wierzchołka stosu przesyłana jest do pamięci pod adres (l-wartość) znajdujący się bezpośrednio pod wierzchołkiem. Obie wartości są zdejmowane ze stosu

Program dla maszyny stosowej

- źródło: $\text{day} := (1461 * y) \underline{\text{div}} 4 + (153 * m + 2) \underline{\text{div}} 5 + d$
- ONP: $\text{day } 1461 \ y * 4 \underline{\text{div}} 153 \ m * 2 + 5 \underline{\text{div}} + d + :=$

- Tłumaczenie dla maszyny stosowej:

lvalue	day
push	1461
rvalue	y
*	
push	4
<u>div</u>	
push	153
rvalue	m
*	
push	2
+	
push	5
<u>div</u>	
+	
rvalue	d
+	
:=	

Przykład – translacja instrukcji przypisania do kodu trójadresowego

$S \rightarrow \underline{id} := E$	$S.zm \leftarrow name(\underline{id}.lexptr)$ $gen(S.zm \parallel " := " \parallel E.zm)$
$E \rightarrow E_1 + E_2$	$E.zm \leftarrow new_temp()$ $gen(E.zm \parallel " := " \parallel E_1.zm \parallel " + " \parallel E_2.zm)$
$E \rightarrow E_1 * E_2$	$E.zm \leftarrow new_temp()$ $gen(E.zm \parallel " := " \parallel E_1.zm \parallel " * " \parallel E_2.zm)$
$E \rightarrow (E_1)$	$E.zm \leftarrow E_1.zm$
$E \rightarrow \underline{id}$	$E.zm \leftarrow name(\underline{id}.lexptr)$

słowo źródłowe: **d := a + (b + c) * d**

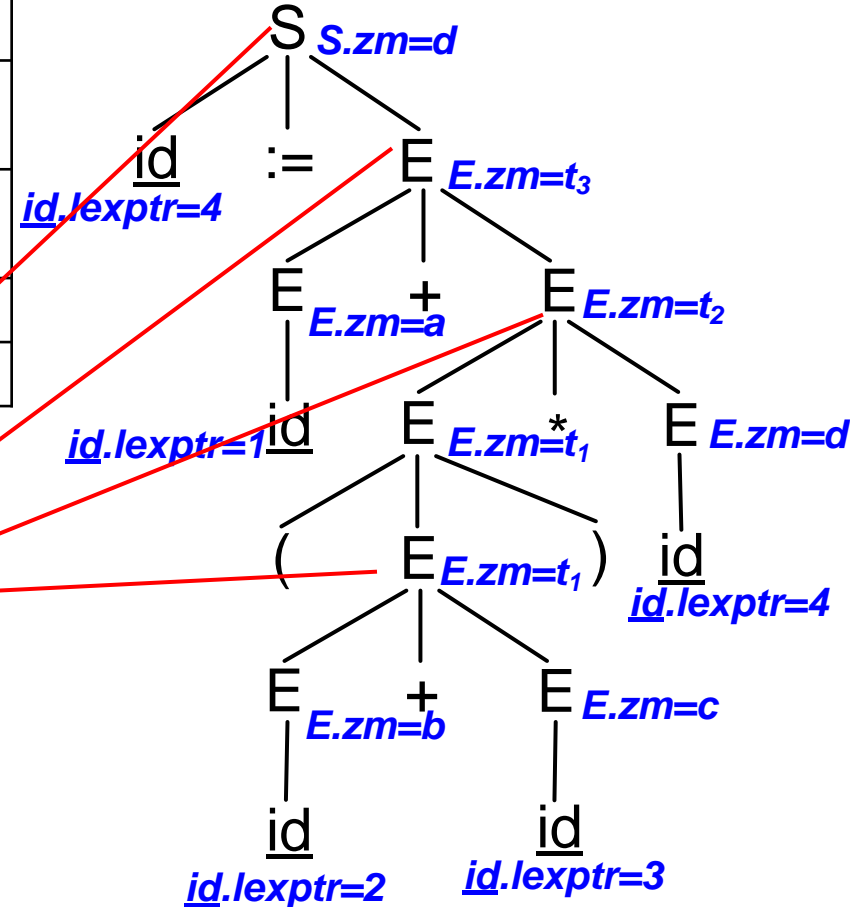
Tłumaczenie:

$t_1 := b + c$

$t_2 := t_1 * d$

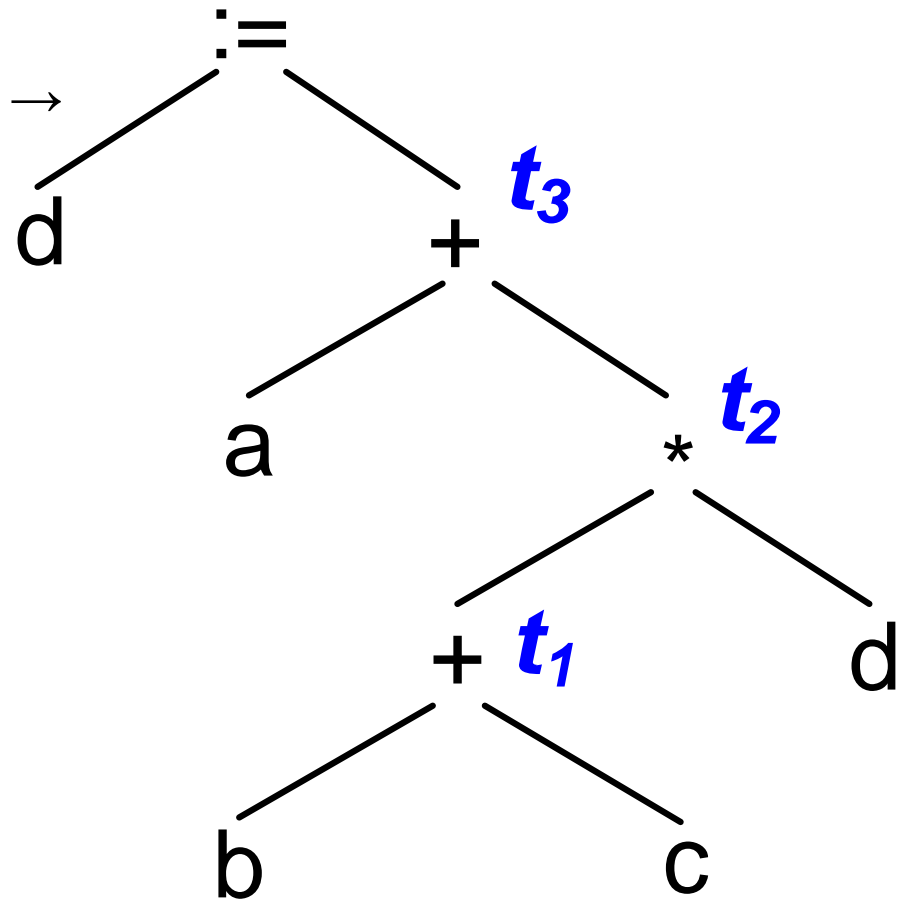
$t_3 := a + t_2$

$d := t_3$



Przykład – translacja instrukcji przypisania do kodu trójadresowego

Drzewo syntaktyczne →



Słowo źródłowe: **d := a + (b + c) * d**

Tłumaczenie:

$t_1 := b + c$

$t_2 := t_1 * d$

$t_3 := a + t_2$

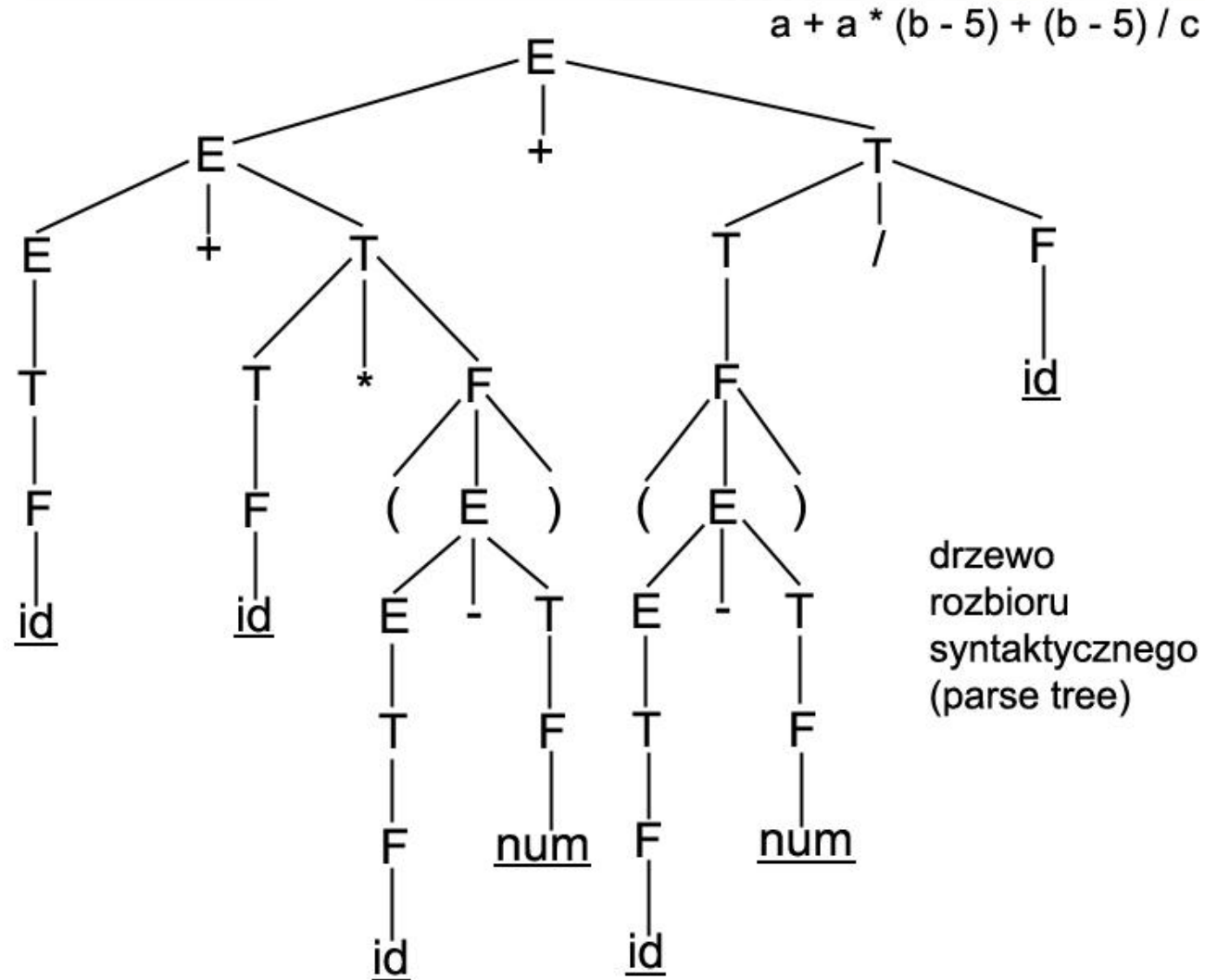
$d := t_3$

Zestaw instrukcji trójadresowych

- (a) **$x := y \text{ op } z$** dla operacji dwuargumentowych
- (b) **$x := \text{op } y$** dla operacji jednoargumentowych
- (c) **$x := y$** kopiowanie
- (d) **goto L** skok bezwarunkowy
- (e) **if x relop y goto L** skok warunkowy
- (f) **param x**
call p, n } do obsługi procedur
return y
- (g) **$x := y[i]$** } do obsługi tablic
 $x[i] := y$ }
*i – odległość elementu od początku tablicy
liczona w jednostkach pamięci, np. w bajtach*
- (h) **$x := \&y$** }
 $x := *y$ } do obsługi wskaźników
 $*x := y$ }

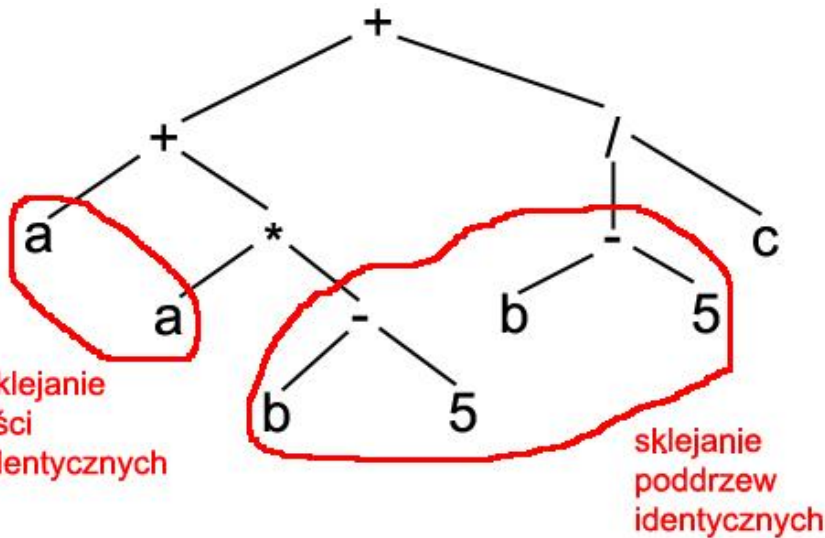
Przypomnienie: drzewa rozbioru

$E \rightarrow E + T \mid E - T \mid T$
 $T \rightarrow T * F \mid T / F \mid F$
 $F \rightarrow (E) \mid \underline{id} \mid \underline{num}$

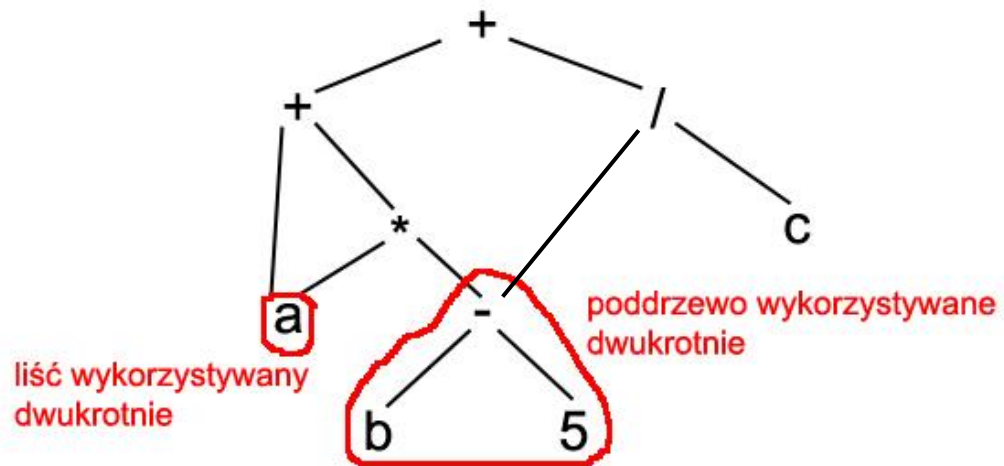


Drzewa syntaktyczne i dagi

$$a + a * (b - 5) + (b - 5) / c$$



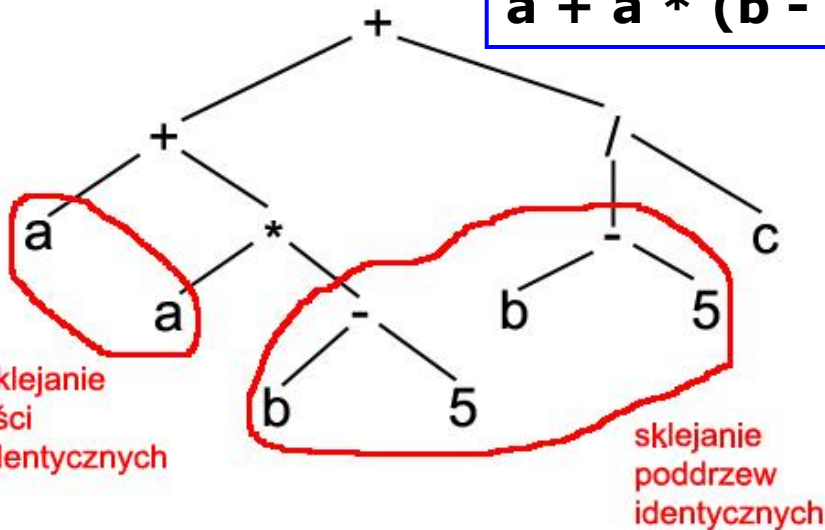
drzewo syntaktyczne
(syntax tree)



skierowany graf acykliczny
(directed acyclic graph - dag)

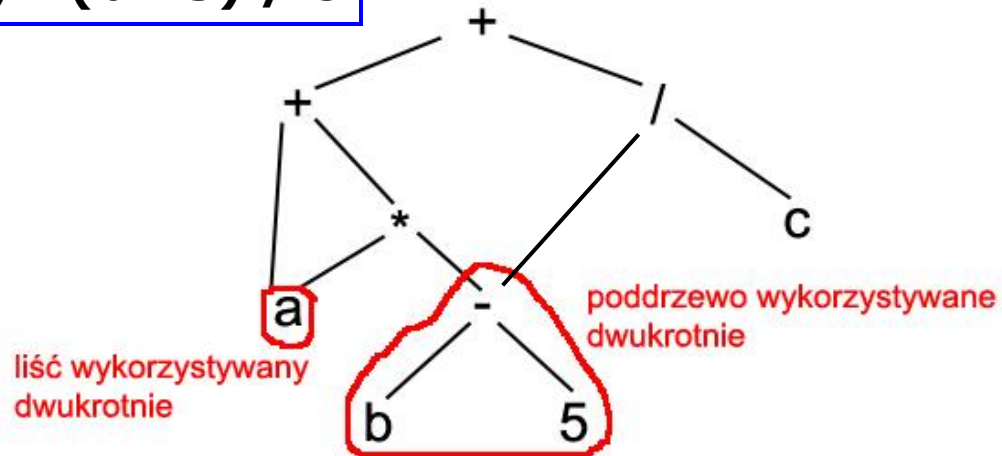
Drzewa syntaktyczne i dagi a kod trójadresowy

$a + a * (b - 5) + (b - 5) / c$



drzewo syntaktyczne
(syntax tree)

$t_1 := b - 5$
 $t_2 := a * t_1$
 $t_3 := a + t_2$
 $t_4 := b - 5$
 $t_5 := t_4 / c$
 $t_6 := t_3 + t_5$

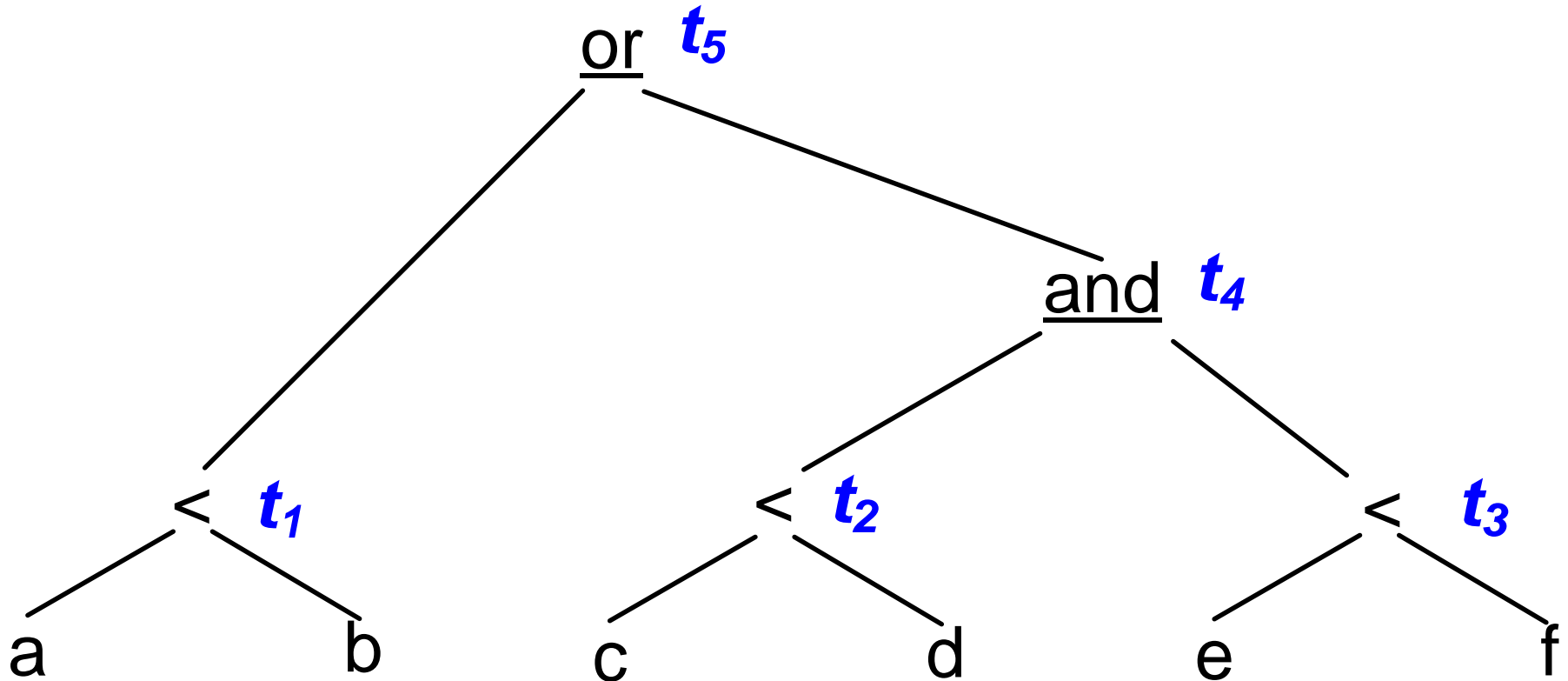


skierowany graf acykliczny
(directed acyclic graph - dag)

$t_1 := b - 5$
 $t_2 := a * t_1$
 $t_3 := a + t_2$
 $t_4 := t_1 / c$
 $t_5 := t_3 + t_4$

Translacja wyrażeń logicznych

Przykład: $a < b$ or $c < d$ and $e < f$



Przykład: **a < b or c < d and e < f**

100 : if a < b goto 103

101 : t₁ := 0

102 : goto 104

103 : t₁ := 1

104 : if c < d goto 107

105 : t₂ := 0

106 : goto 108

107 : t₂ := 1

108 : if e < f goto 111

109 : t₃ := 0

110 : goto 112

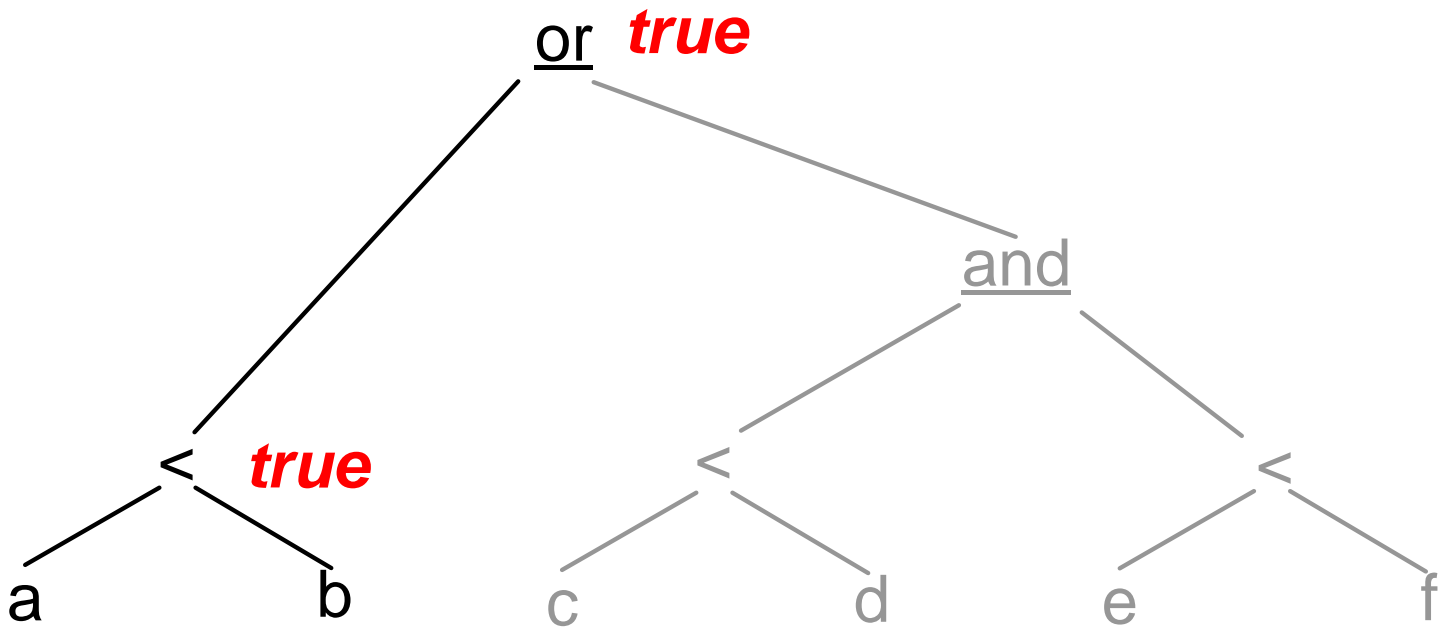
111 : t₃ := 1

112 : t₄ := t₂ and t₃

113 : t₅ := t₁ or t₄

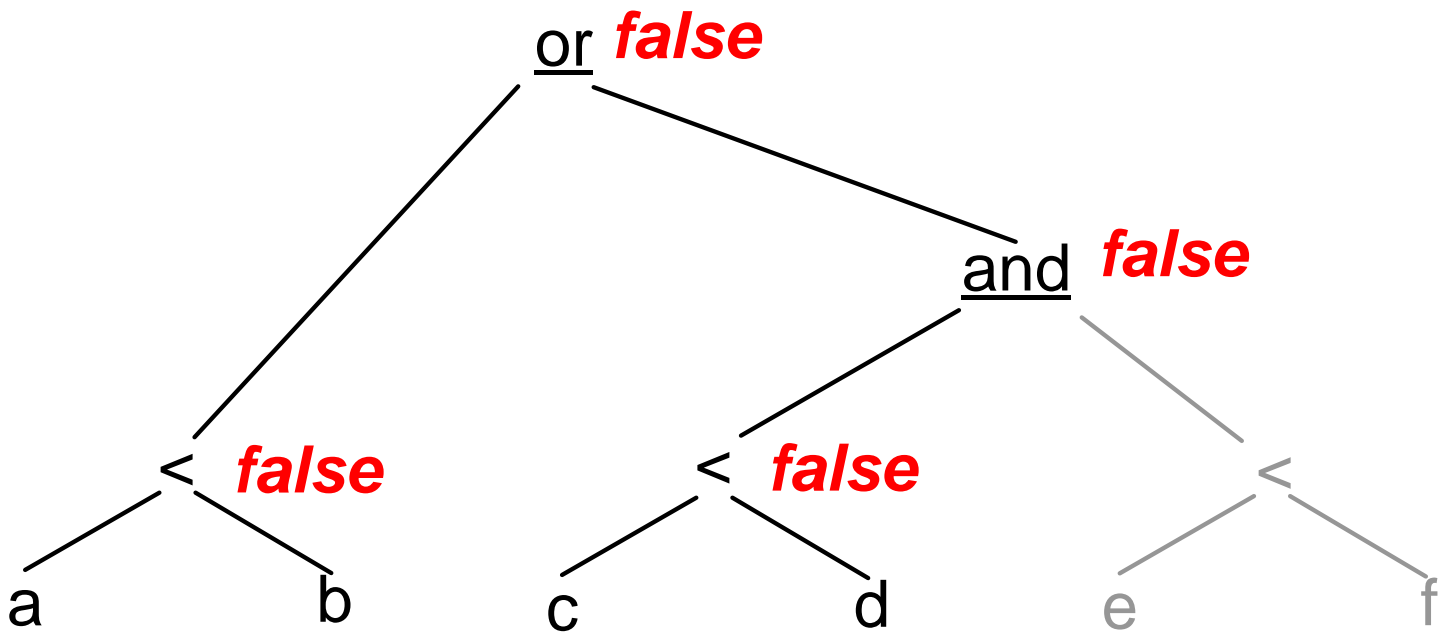
Translacja wyrażeń logicznych

Przykład: **a < b** or **c < d** and **e < f**



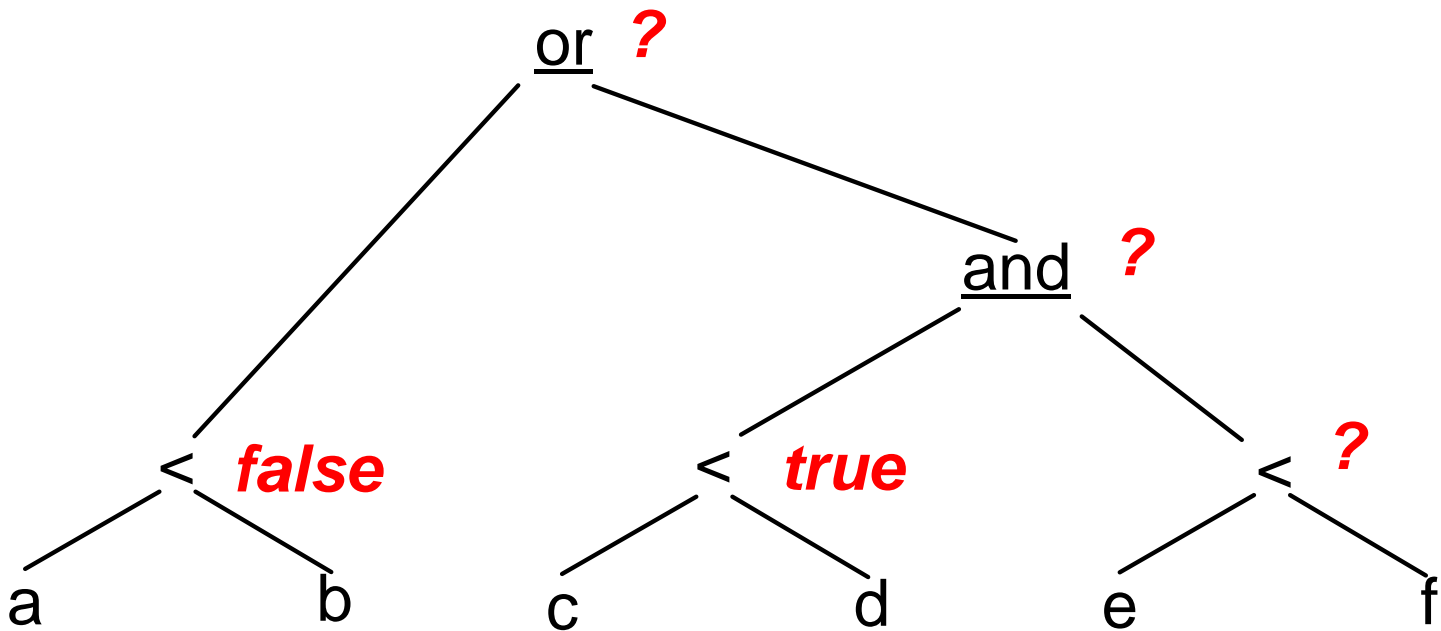
Translacja wyrażeń logicznych

Przykład: $a < b$ or $c < d$ and $e < f$

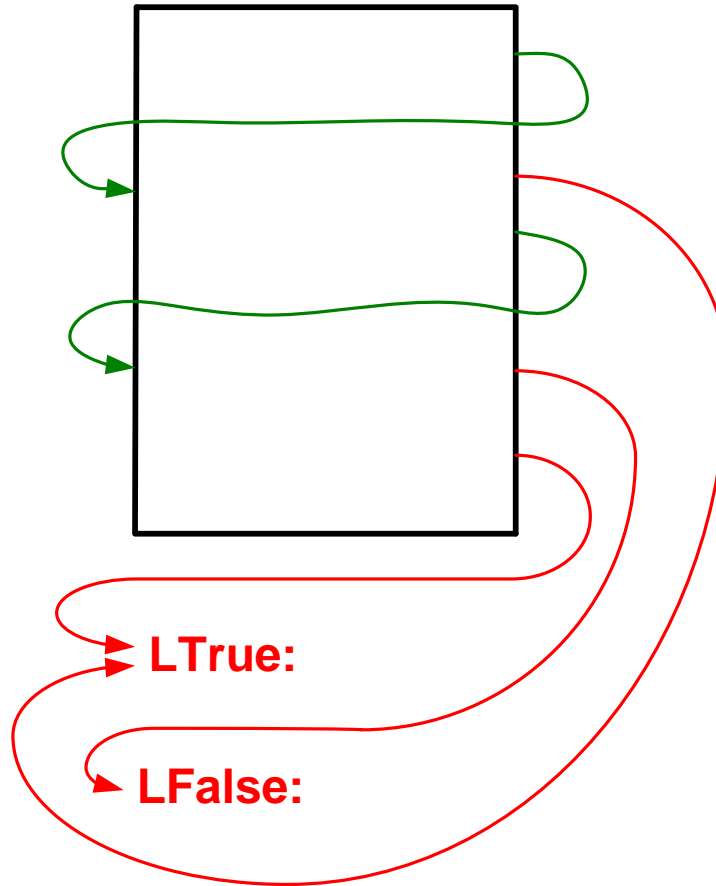


Translacja wyrażeń logicznych

Przykład: $a < b$ or $c < d$ and $e < f$



Translacja wyrażeń logicznych



Przykład: $a < b$ or $c < d$ and $e < f$

if $a < b$ goto L.true

goto L1

L1: if $c < d$ goto L2

goto: L.false

L2: if $e < f$ goto L.true

goto L.false

Po optymalizacji...

```
if  $a < b$  goto L.true  
if  $c \geq d$  goto L.false  
if  $e < f$  goto L.true  
goto L.false
```