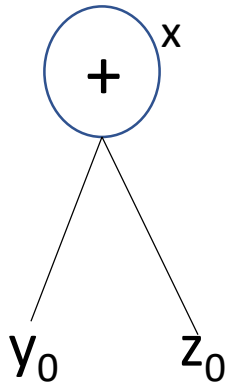
DAG Representation of Basic Blocks

Introduction

- DAG = Directed Acyclic Graph
- By using DAG, we identify the common subexpressions
- Leaves are used for representing variable names or constants
- Initial values are subscripted with zero (0)
- The interior nodes of the graph are labelled with an operator symbol.
- Internal nodes also represents result of the expressions

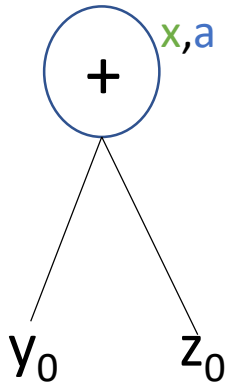
x = y + z



x=y+z

a=y+z

- DAG is constructed statement by statement. First it will take a=b+c
- Whenever we are creating a new node, we will have to check if that node is already available or not. This is how the common subexpressions are removed
- We will simply attach the identifier as a list of identifier



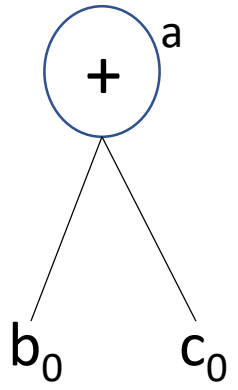
Example

a=b+c

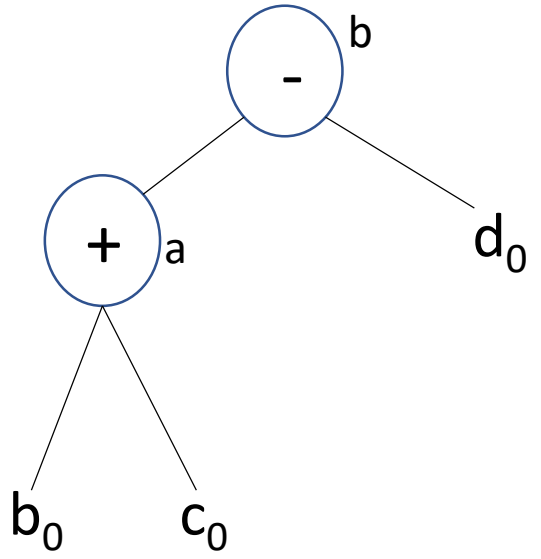
b=a-d

c=b+c

d=a-d

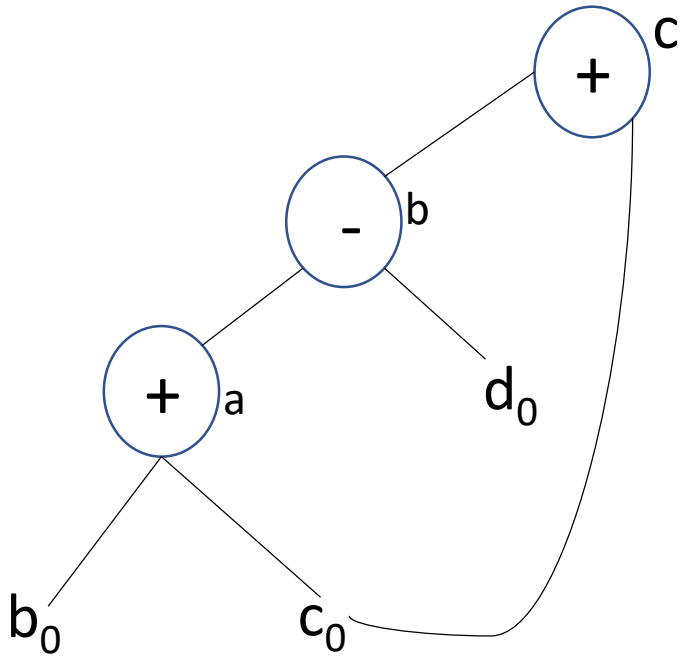


- As we will go statement by statement, we will first construct ‘+’



- Next we will construct b=a-d. So, first we will check if node ‘-’ is already available or not.
- Here ‘-’ is not available to it will be created with left child as ‘a’ and right child as ‘d’
- The left child ‘a’ is previously constructed

- Next, we will have to construct c=b+c
- We have already constructed ‘b’ previously. And from ‘b’ there is no node constructed



The value of ‘c’ is changed from c_0 to this. So, for future constructions, we will use this for ‘x’

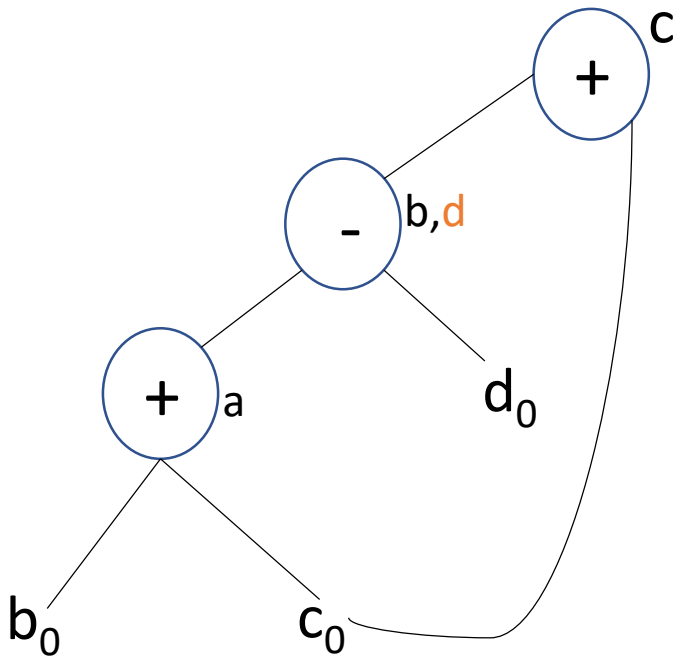
a=b+c

b=a-d

c=b+c

d=a-d

- Here, the final value is ‘a-d’.
- We check if ‘a-d’ is already available or not.
- Though it is available, it is having value ‘b’
- So, we just simply attach ‘d’ to this node as the 2nd and 4th expressions are common

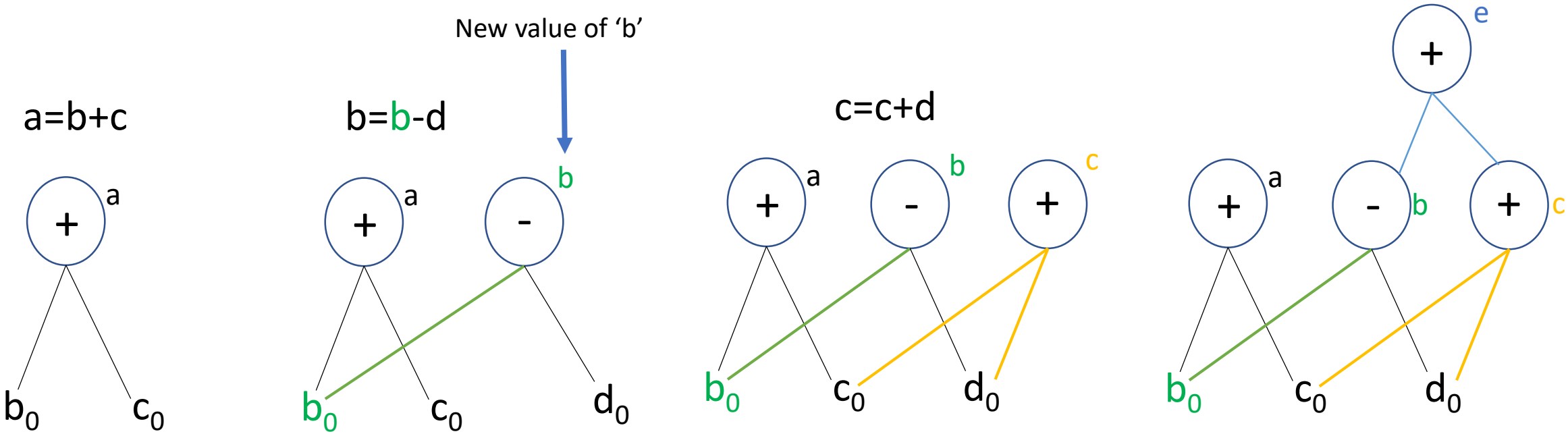


a=b+c

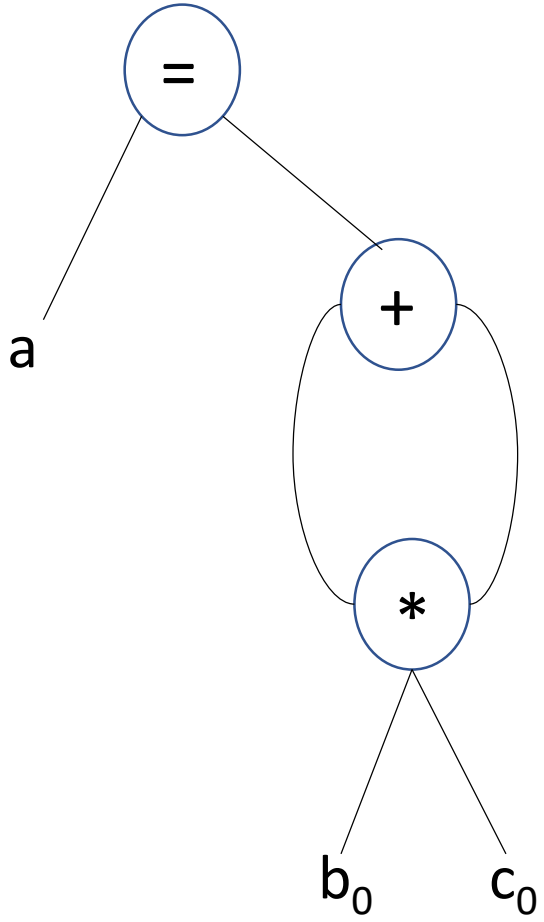
b=b-d

c=c+d

e=b+c

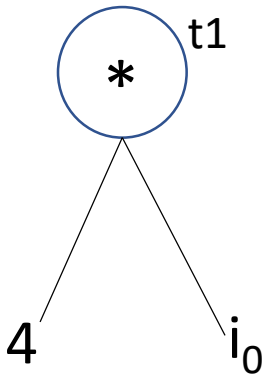


a= b * -c + b * -c

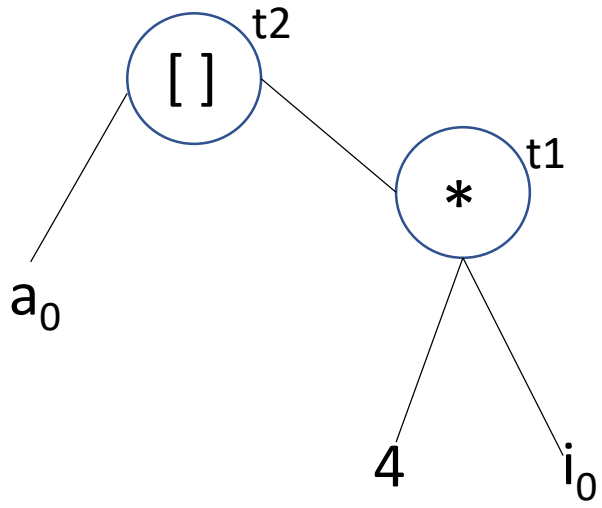


(1) t1 = 4*i
(2) t2 = a[t1]
(3) t3 = 4*i
(4) t4 = b[t3]
(5) t5 = t2 *t4
(6) t6 = p + t5
(7) p = t6
(8) t7 = i+1
(9) i = t7
(10) if i <=20 goto (1)

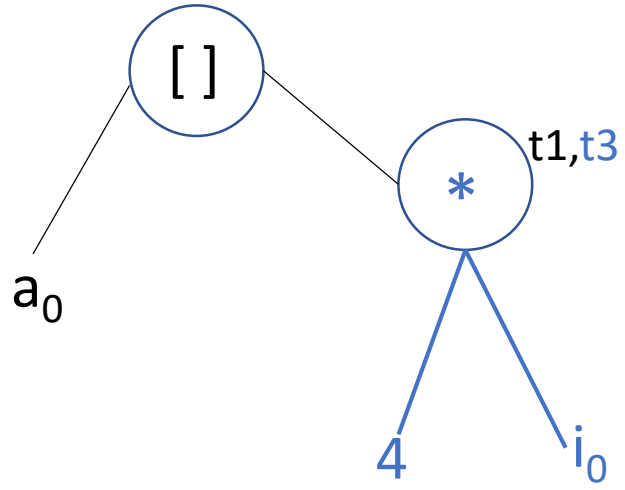
t1 = 4*i



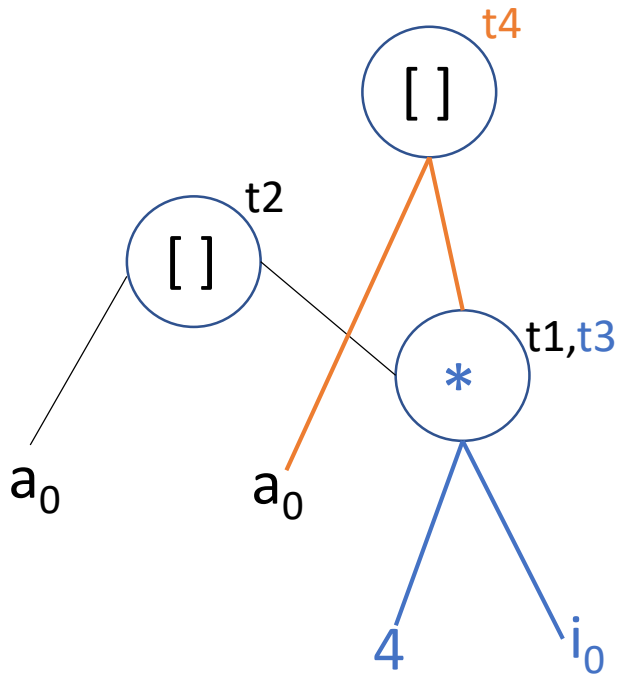
t2 = a[t1]



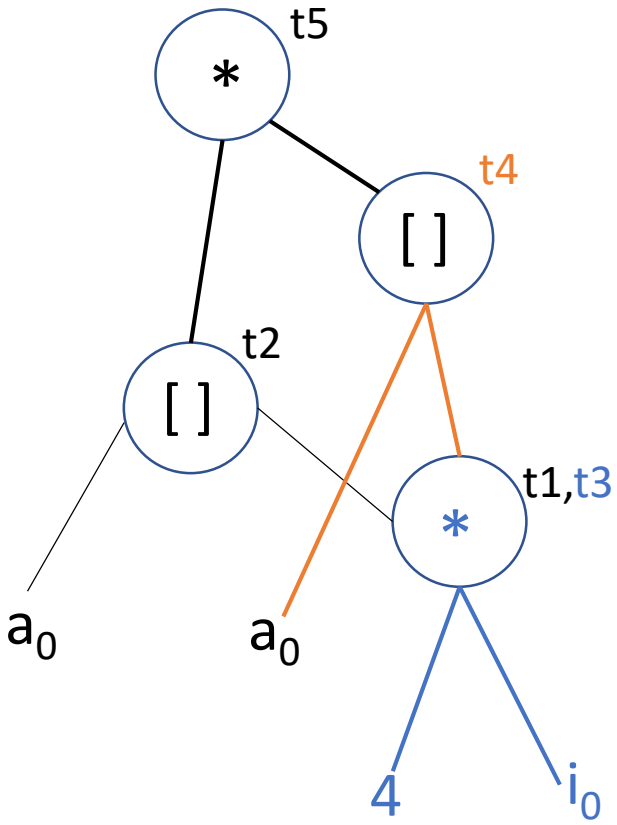
t3 = 4*i



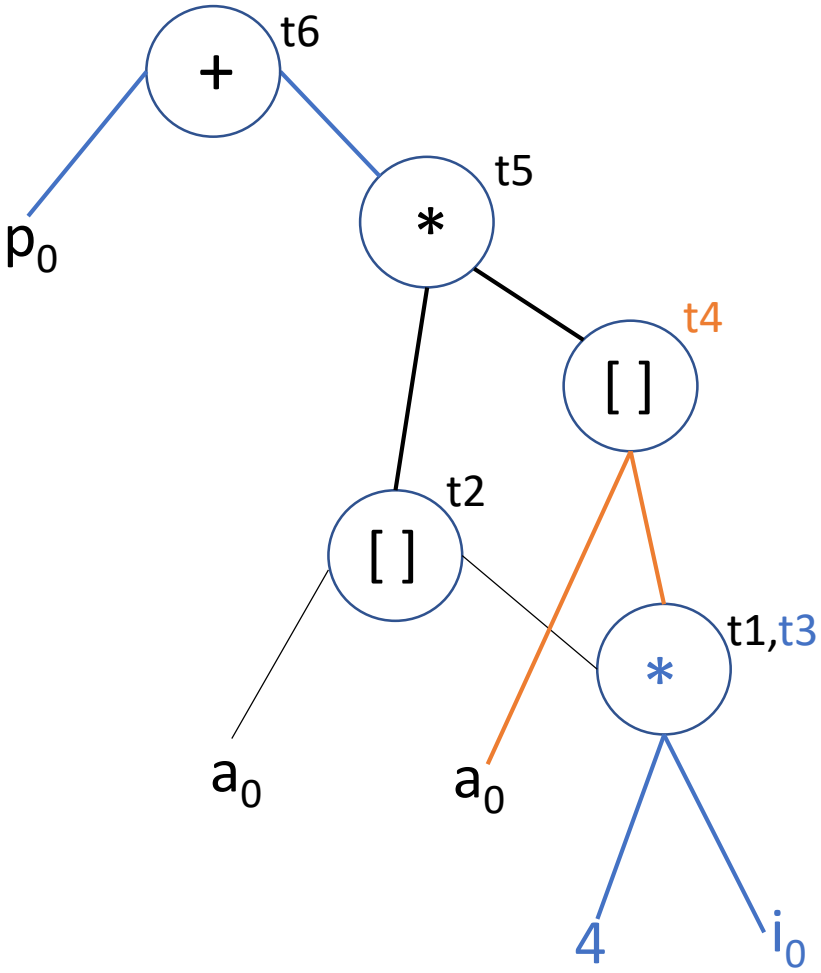
t4 = b[t3]



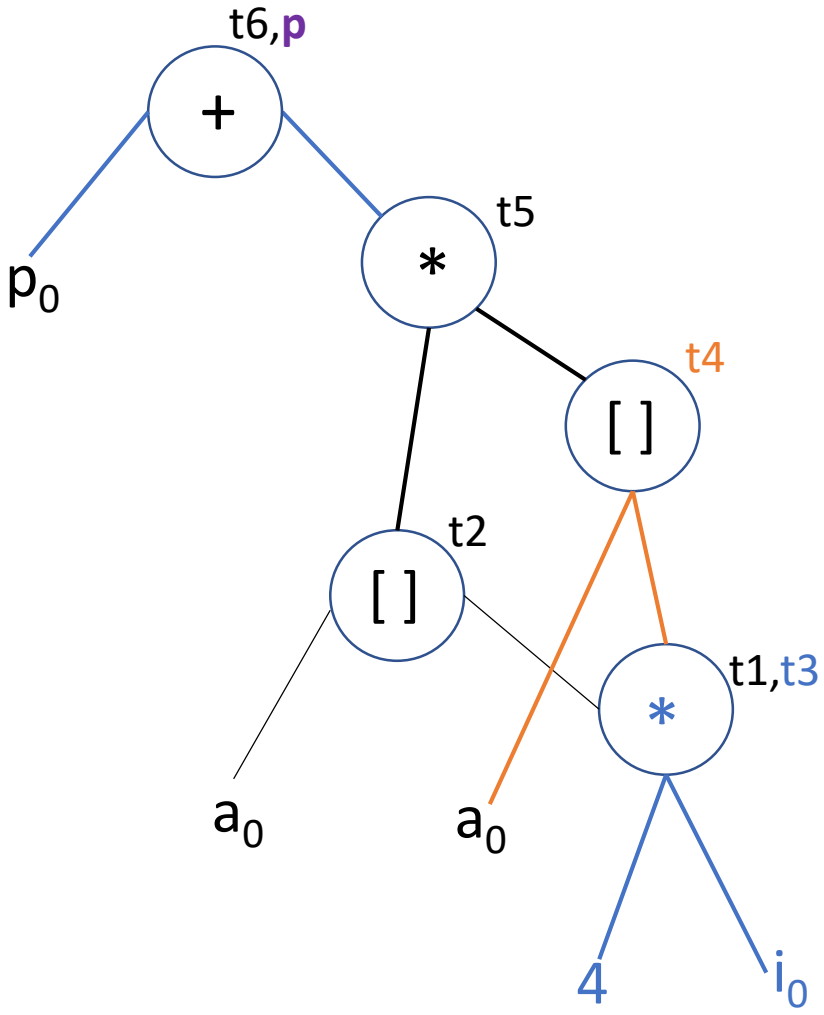
t5 = t2 *t4



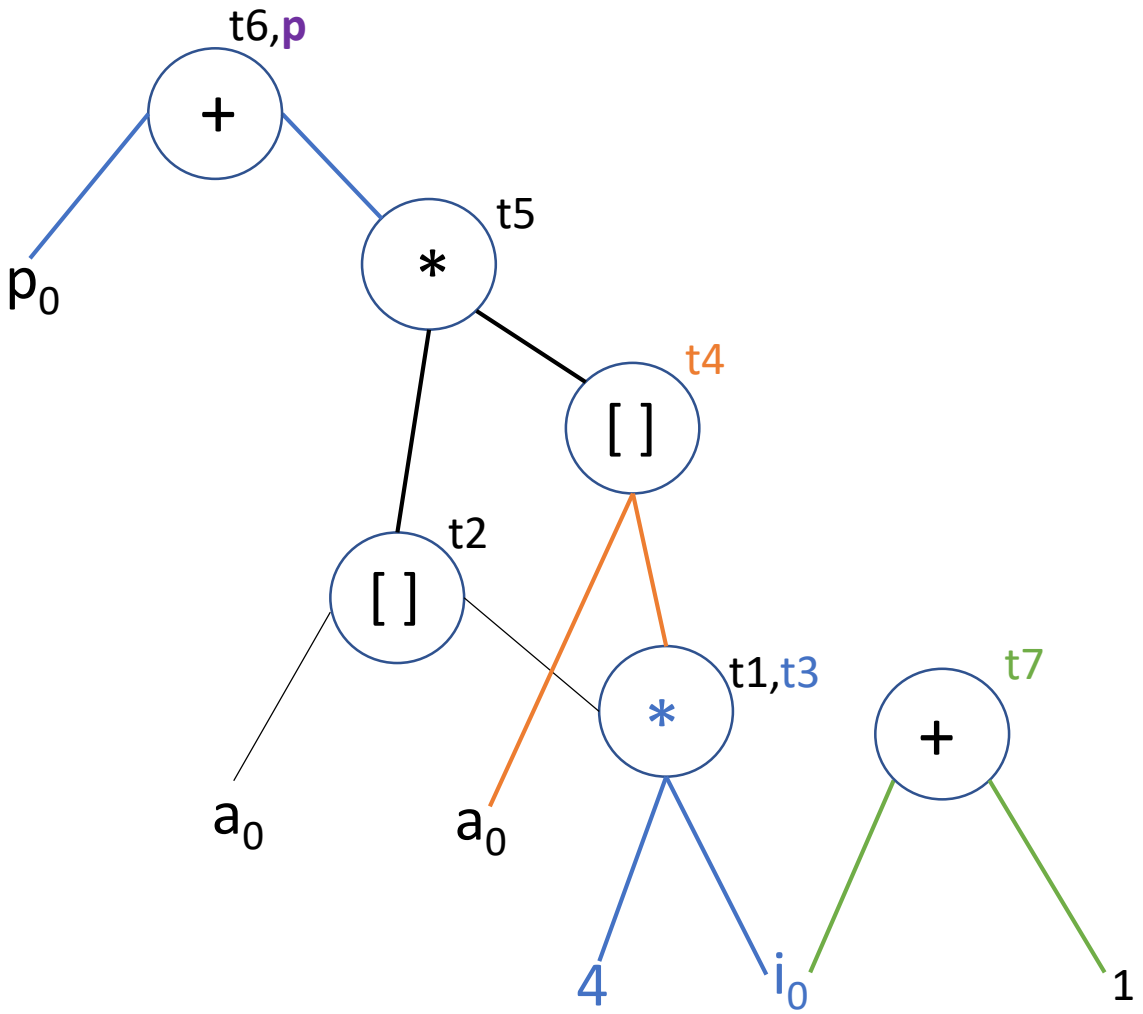
t6 = p + t5



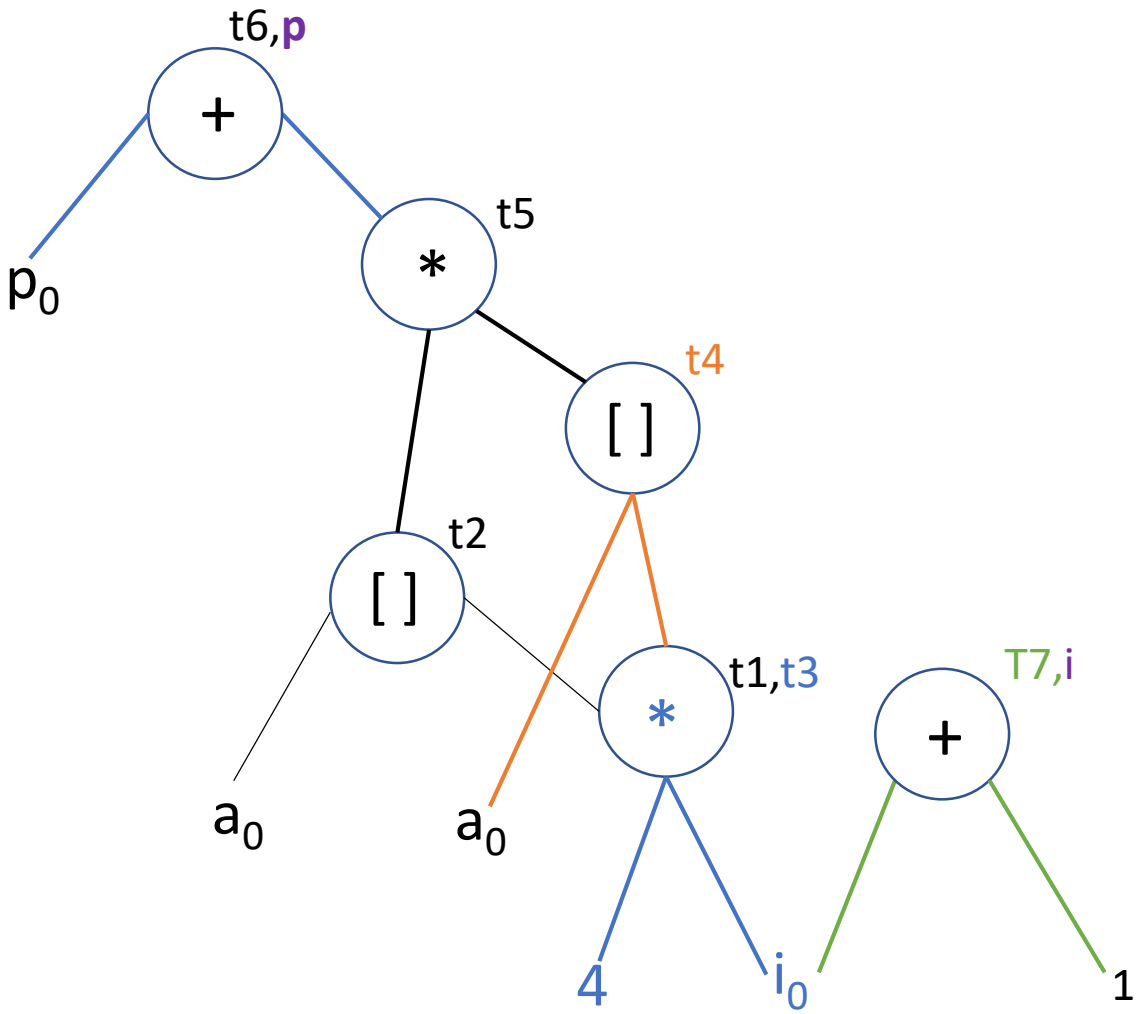
p = t6



t7 = i+1



i = t7



if i <=20 goto (1)