Problem of the DAY

Create a regular context-free grammar that generates L= {w $\in$ {a,b}* : the number of a's in w is not divisible by 3}

Hint: start by designing a DFA first then get the grammar from the DFA.

Recall: A regular context-free grammar can have at most one non-terminal on the RHS of each rule, if there is a non-terminal, it is the last symbol on the RHS of the rule.

Announcements:

Assignment 3 has been posted.

If you do the 10 point bonus question, make sure you submit your code on connex.

Midterm: Wednesday June 27, in class.

Old midterms are available from the course web page.

Assignment #1 and #2 solutions are available in the connex resources.

Alan Turing's 100th Birthday Celebration

A DAY OF FREE EVENTS in ECS 660

Friday, June 22nd, 2012

No class Friday June 22.
Please attend Turing Celebration Events instead

9:00-9:30	The Life of Alan Turing - **Daniel German**
9:30-11:00	Turing's Seminal 1936 Paper - **John Ellis**
11:00-12:00	The history of the Entscheidungsproblem, from Llull to Turing and beyond - **Bruce Kapron**
12:00 – 1:30	Pizza Lunch, Turing Art, Music & Wumpus World
1:30-2:30	The Turing Test and Searle's "Chinese Room" - **Maarten van Emden**
2:30-3:30	The Enigma machine and Turing's Cryptoanalysis - **Venkatesh Srinivasan**
3:30-4:00	Birthday cake, coffee and tea with special guest **Olive Bailey** - worked with Turing at Bletchley Park
4:00-6:00	**"Breaking the Code"** - Acclaimed film about Turing's life – ECS 104

www.csc.uvic.ca/Events/turing100.htm

Sponsors:

PIMS

University of Victoria
Computer Science

Create a NDFA which accepts the language generated by this context-free grammar. Start symbol: S

$S \rightarrow aa\ S$

$S \rightarrow \varepsilon$

$S \rightarrow M$

$M \rightarrow bbb$

$M \rightarrow ab\ M$

$M \rightarrow b\ S$

$M \rightarrow E$

$E \rightarrow aa$

Side note: I am using a more general definition of a NDFA here which permits transitions on strings. That is,

Δ is a relation which is a finite subset of

$K \times \Sigma^* \times K$

current state — string — next state

Theorem: If L has a regular context-free grammar, then L is a regular language.

Proof: A NDFA can be constructed that accepts the language that the regular context-free grammar generates.

Note: this is in the text as Exercise 3.1.10.

Given the regular context-free grammar $G=(V, \Sigma, R, S)$ construct a NDFA

$M = (K, \Sigma, \Delta, s, F)$ where

$K= (V - \Sigma) \cup \{ f \}$, $s= S$, $F = \{ f \}$

For each rule $T \rightarrow u\,R$ with $u \in \Sigma^*$, $R \in V-\Sigma$,
add a transition (T, u, R) to Δ.

For each rule $T \rightarrow u$ with u in Σ^*,
add a transition (T, u, f) to Δ.

$L = \{ a^n b^p : \quad n \leq p \leq 3n, \quad n,p \geq 0 \}$

Start symbol S.

$S \rightarrow a\ S\ b$ $\qquad\qquad$ $S \rightarrow \varepsilon$

$S \rightarrow a\ S\ bb$

$S \rightarrow a\ S\ bbb$

This works because any integer p can be expressed as:

$p = r + 2(n - r)$ when $n \leq p \leq 2n$, and

$p = 2r + 3(n - r)$ when $2n \leq p \leq 3n$.

Parse Trees:

Definition of parse trees.

Parse trees are created when programs are compiled to aid in checking syntax and determining semantics of a program.

Leftmost derivations.

Example of how parse trees are used to prove the pumping theorem for context-free languages.

L= {w in {a, b}* : w has the same number of a's and b's}

Start symbol S

$S \rightarrow e$	$A \rightarrow a$	$B \rightarrow b$
$S \rightarrow a\ B$	$A \rightarrow a\ S$	$B \rightarrow b\ S$
$S \rightarrow b\ A$	$A \rightarrow b\ A\ A$	$B \rightarrow a\ B\ B$

S- generates strings with #a's = #b's

A- means need one "a"

B- means need one "b"

Three derivations for aabbab:

1. $\underline{S} \Rightarrow a\underline{B} \Rightarrow aa\underline{B}B \Rightarrow aab\underline{S}B \Rightarrow aabb\underline{A}B \Rightarrow aabba\underline{B} \Rightarrow aabbab$

2. $\underline{S} \Rightarrow a\underline{B} \Rightarrow aa\underline{B}B \Rightarrow aab\underline{B} \Rightarrow aabb\underline{S} \Rightarrow aabba\underline{B} \Rightarrow aabbab$

3. $\underline{S} \Rightarrow a\underline{B} \Rightarrow aa\underline{B}\underline{B} \Rightarrow aaBb\underline{S} \Rightarrow aaBba\underline{B} \Rightarrow aa\underline{B}bab \Rightarrow aabbab$

Leftmost derivation: leftmost non-terminal replaced at each step.

Which ones are leftmost?

Two of the derivations are essentially the same- they correspond to the same leftmost derivation. The other one is different.

A context-free grammar is ambiguous if there is a string in the language which has two or more distinct parse trees (or equivalently, two or more distinct leftmost derivations).

Parse Trees- $G = (V, \Sigma, R\ S)$

Nodes of parse trees are each labelled with one symbol from $V \cup \{\varepsilon\}$.

Node at top- root, labelled with start symbol S.

Leaves- nodes at the bottom.

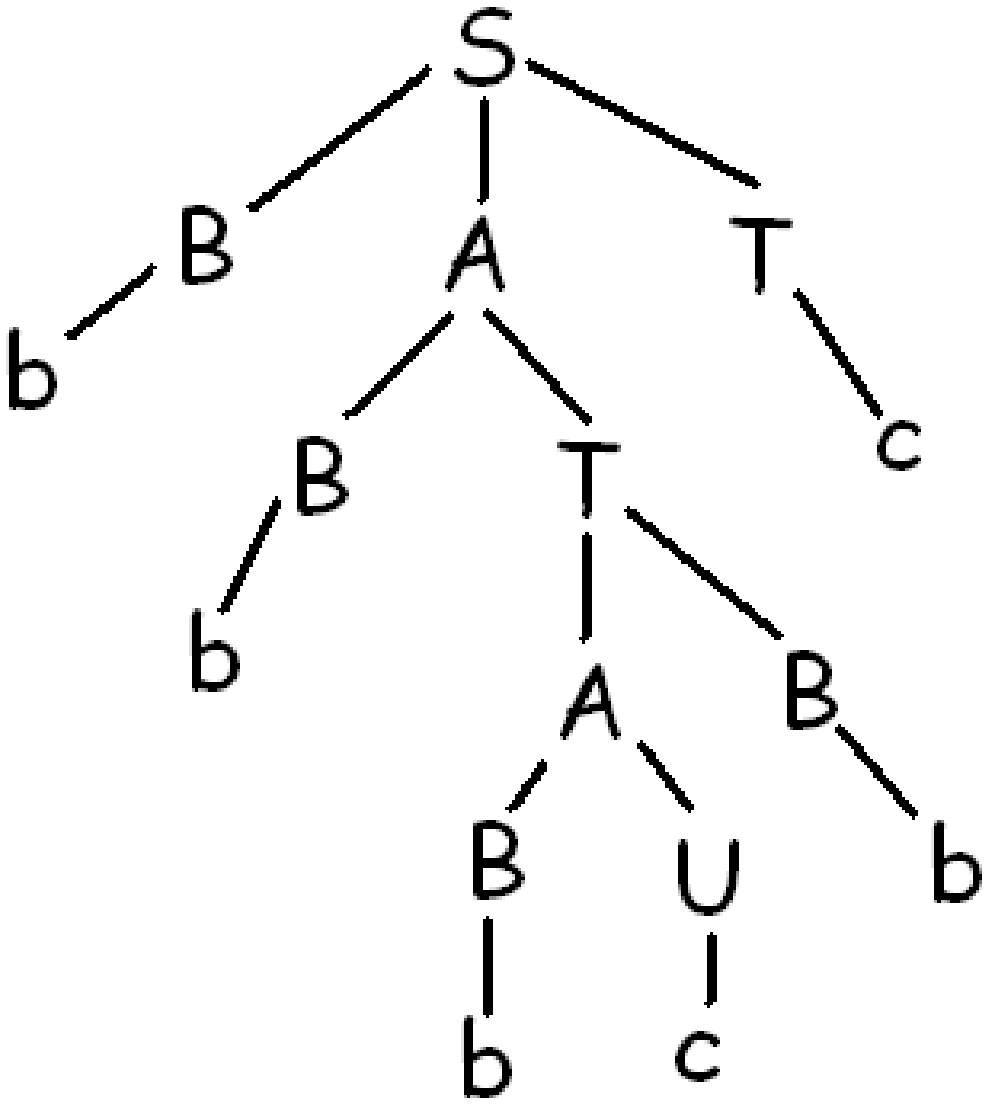
Yield- concatenate together the labels on the leaves going from left to right.

Leftmost derivation: always replace the leftmost non-terminal at each step.

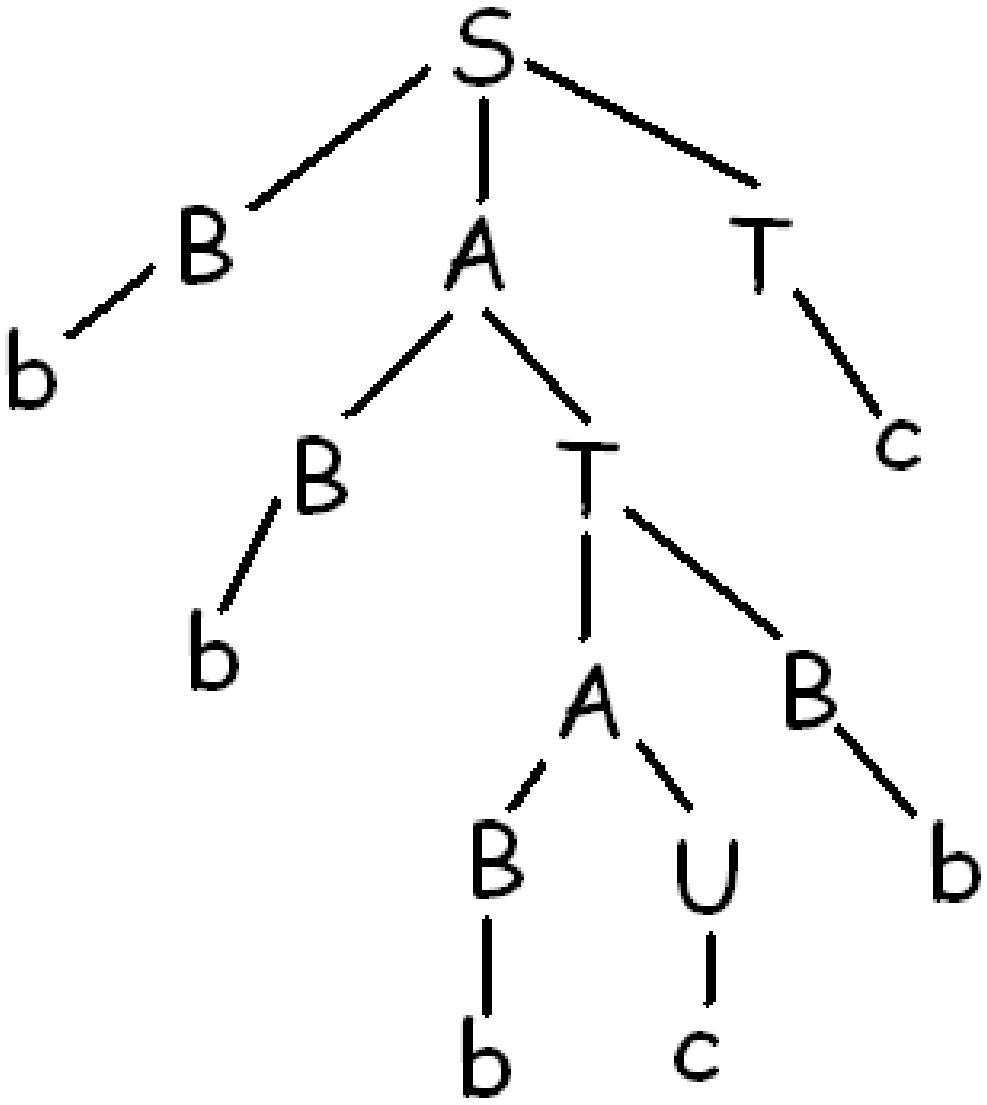
Theorem: Every derivation in a grammar G corresponds to a unique leftmost derivation.

Proof:

Construct the parse tree and determine the leftmost derivation from it.

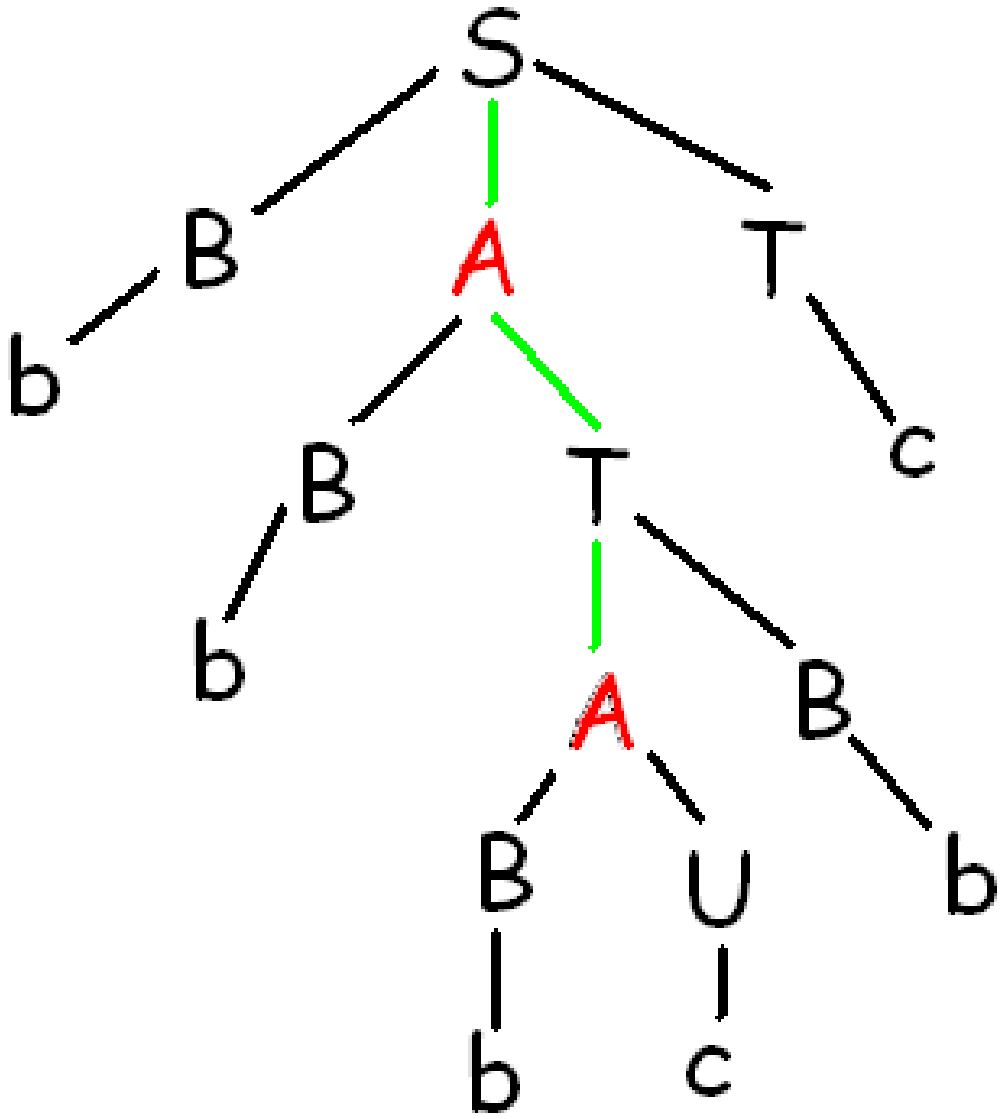


Find the leftmost derivation for this parse tree.



To pump:

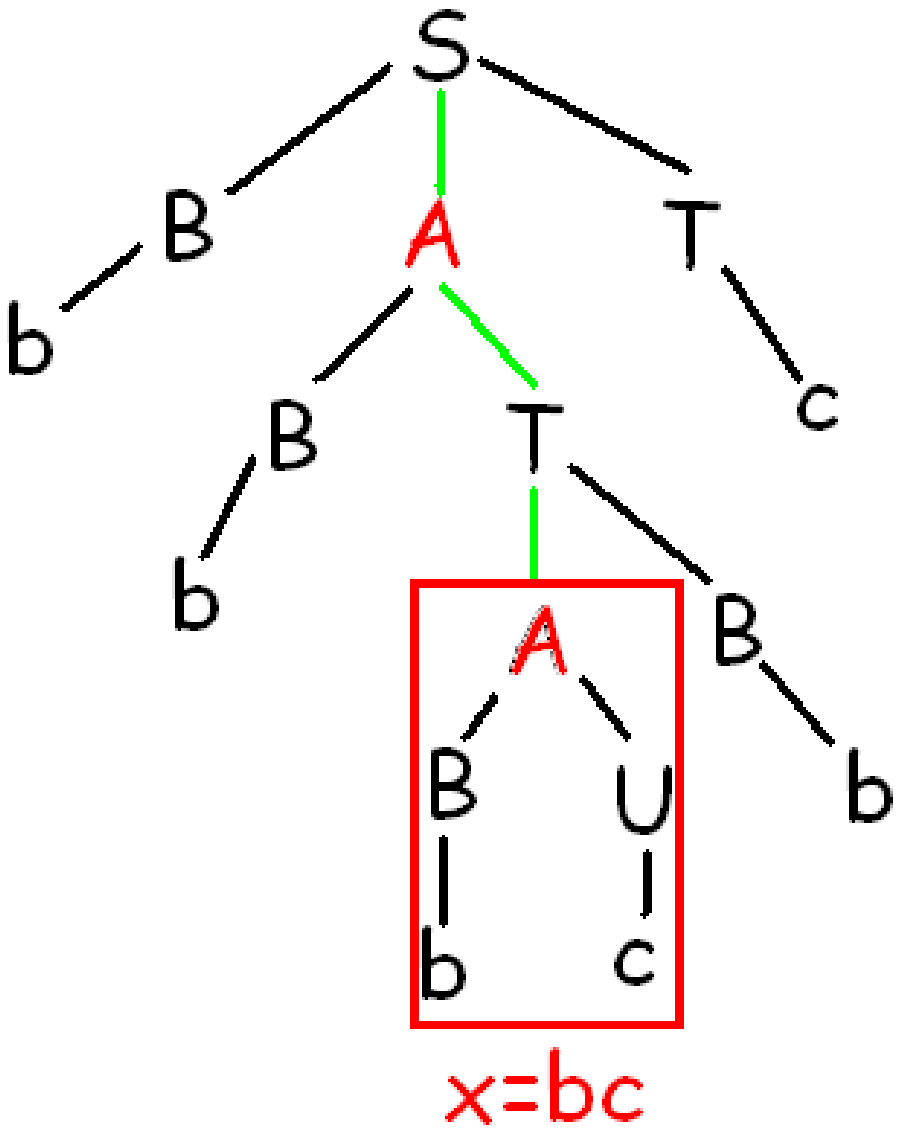
Look for a path from root with a repeated non-terminal.



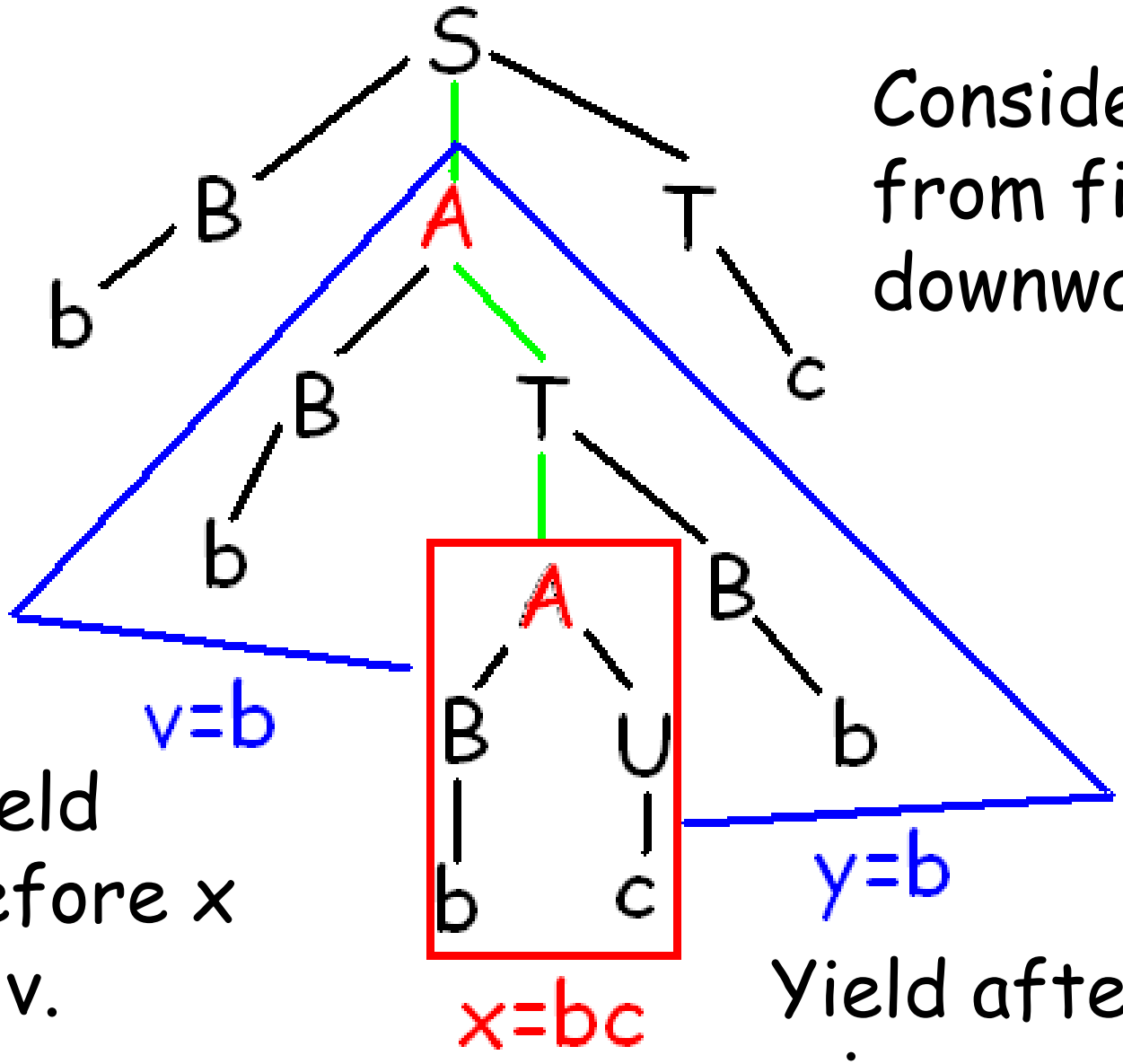
Path:

S - A - T - A

Non-terminal A is repeated.



Yield from second copy downwards gives x.



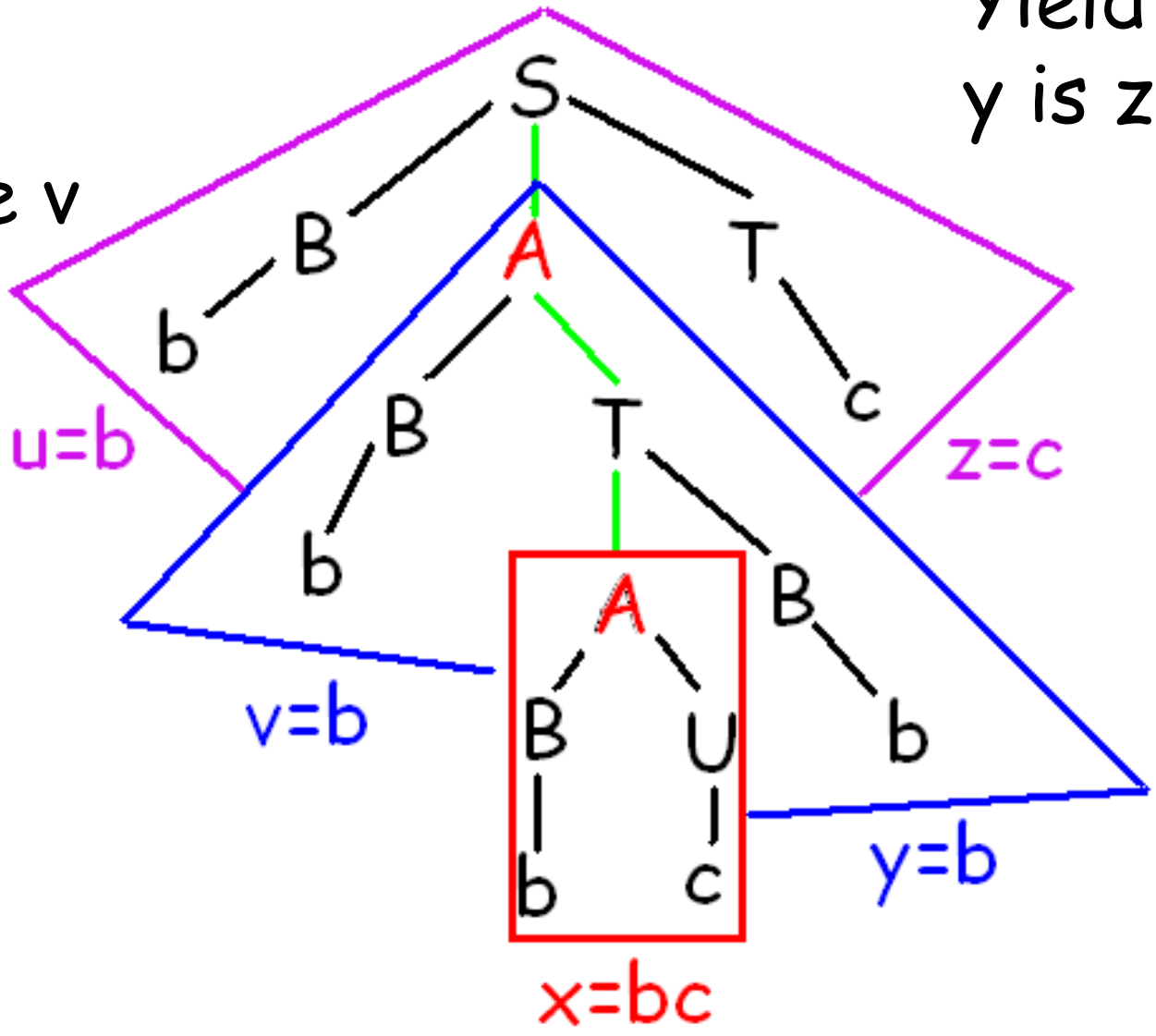
Consider tree from first copy downwards.

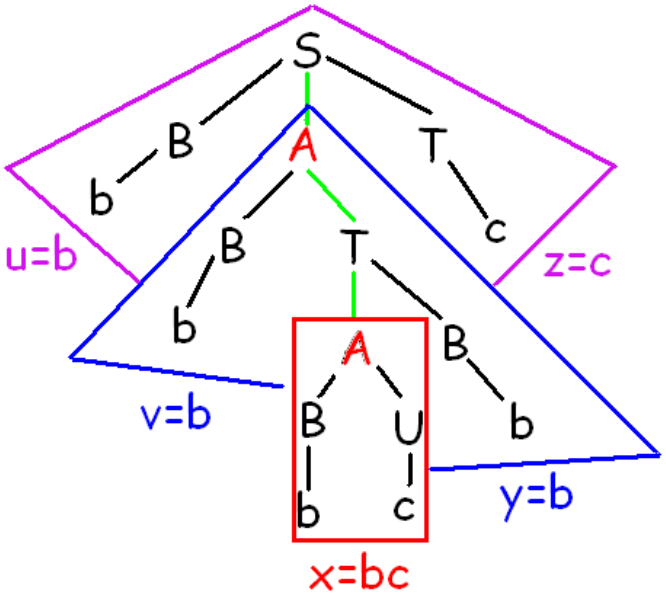
Yield before x is v.

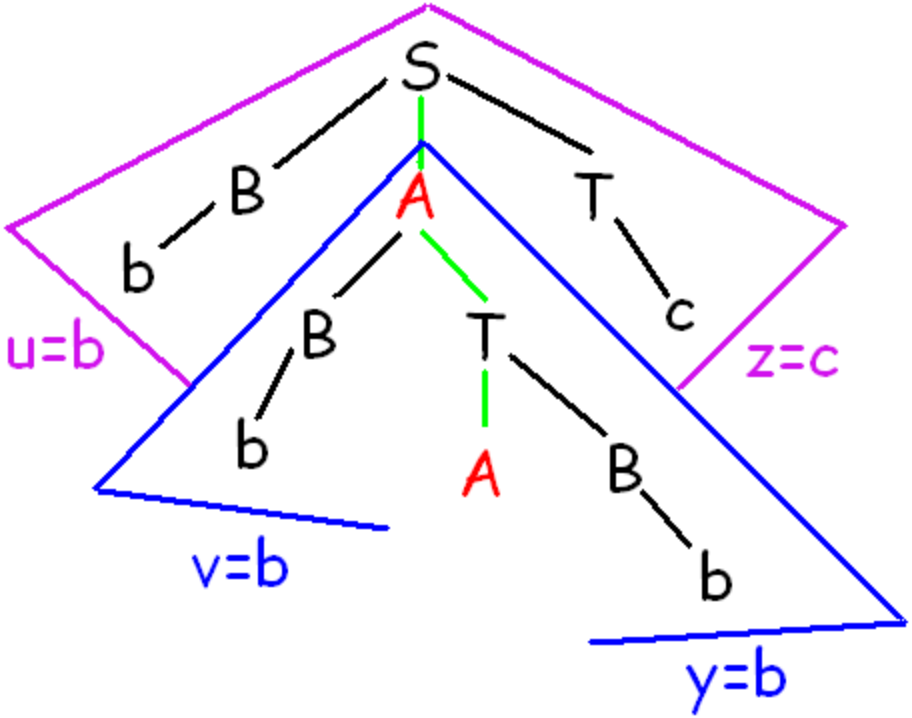
Yield after x is y.

Yield of whole tree before v is u.

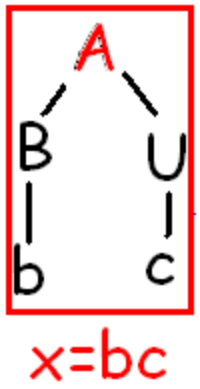
Yield after y is z.

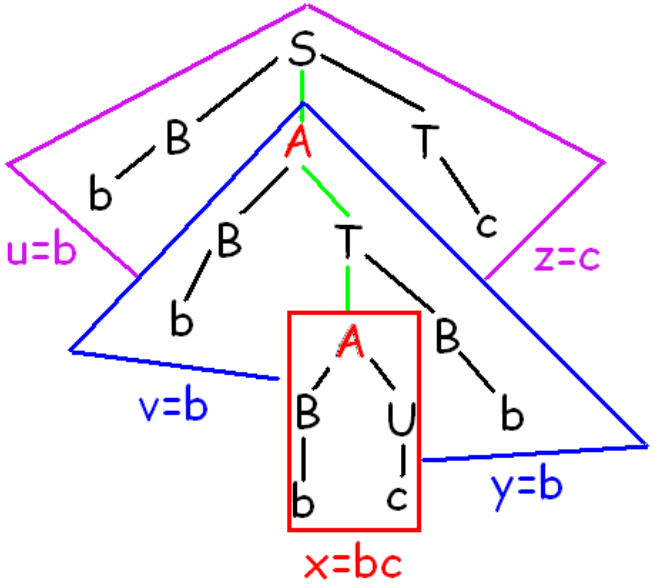




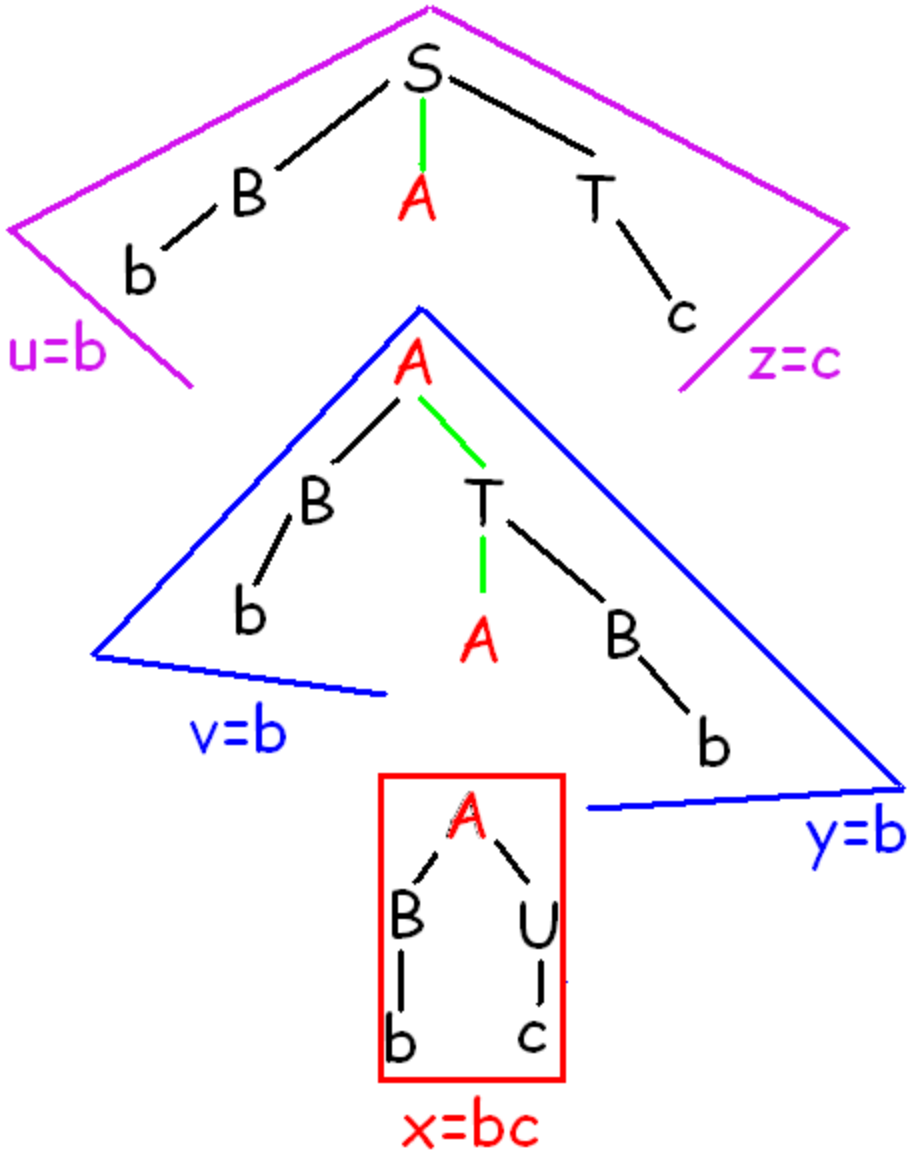


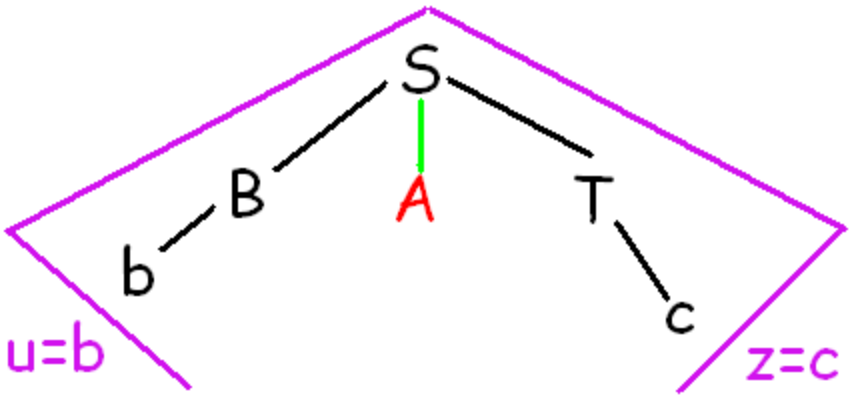
Cut off part of tree from second copy downwards.



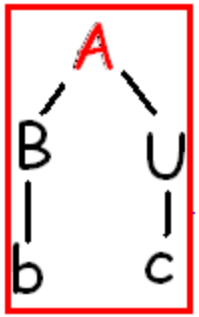


Cut off part from first copy downwards.





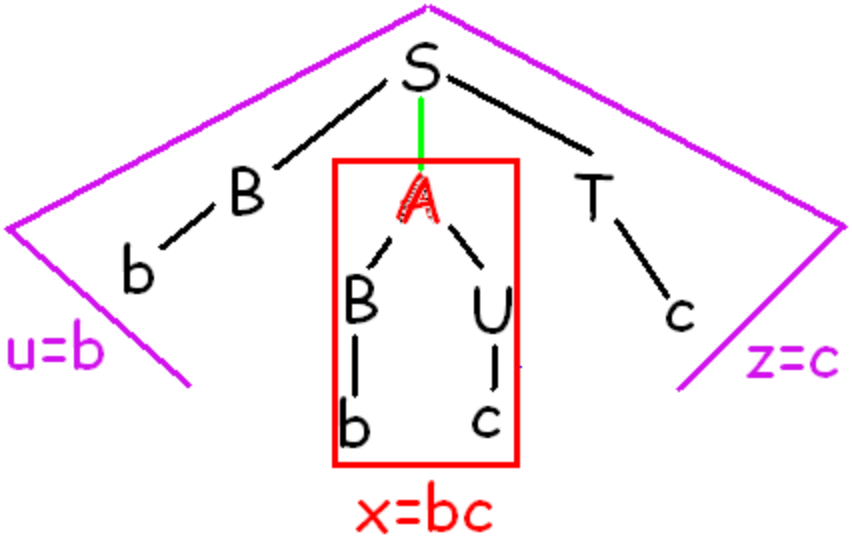
To pump zero times:

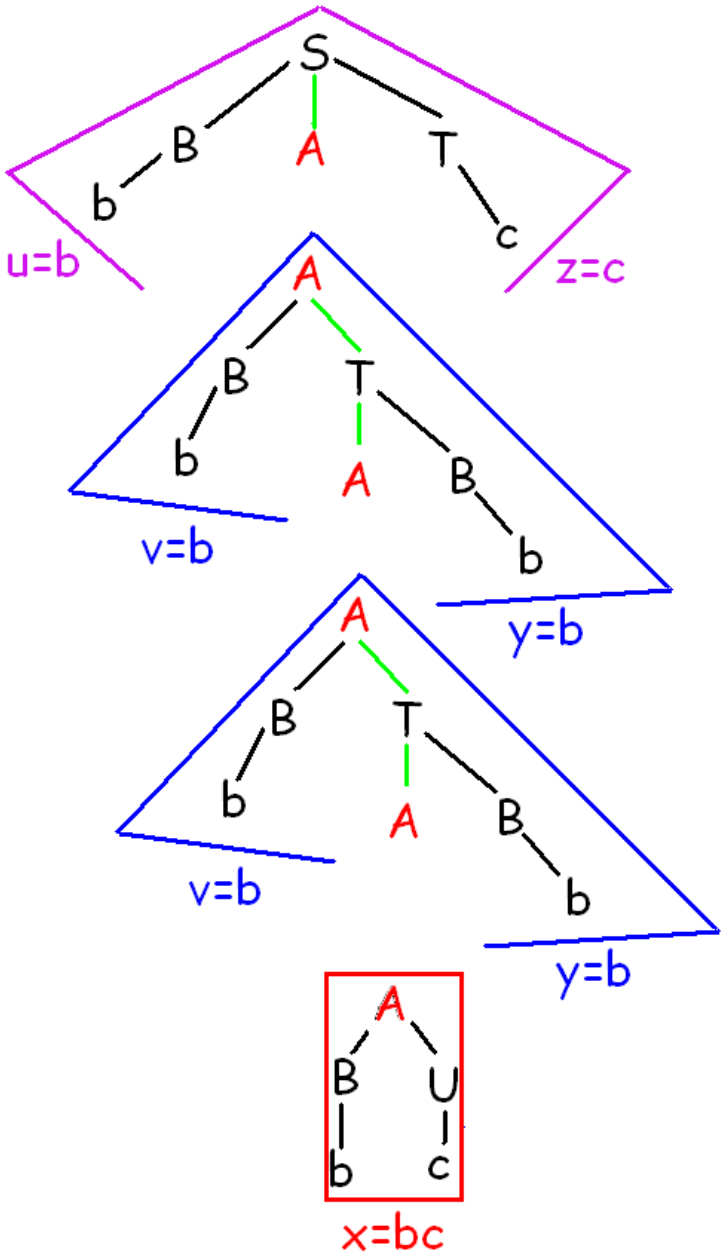


New yield:

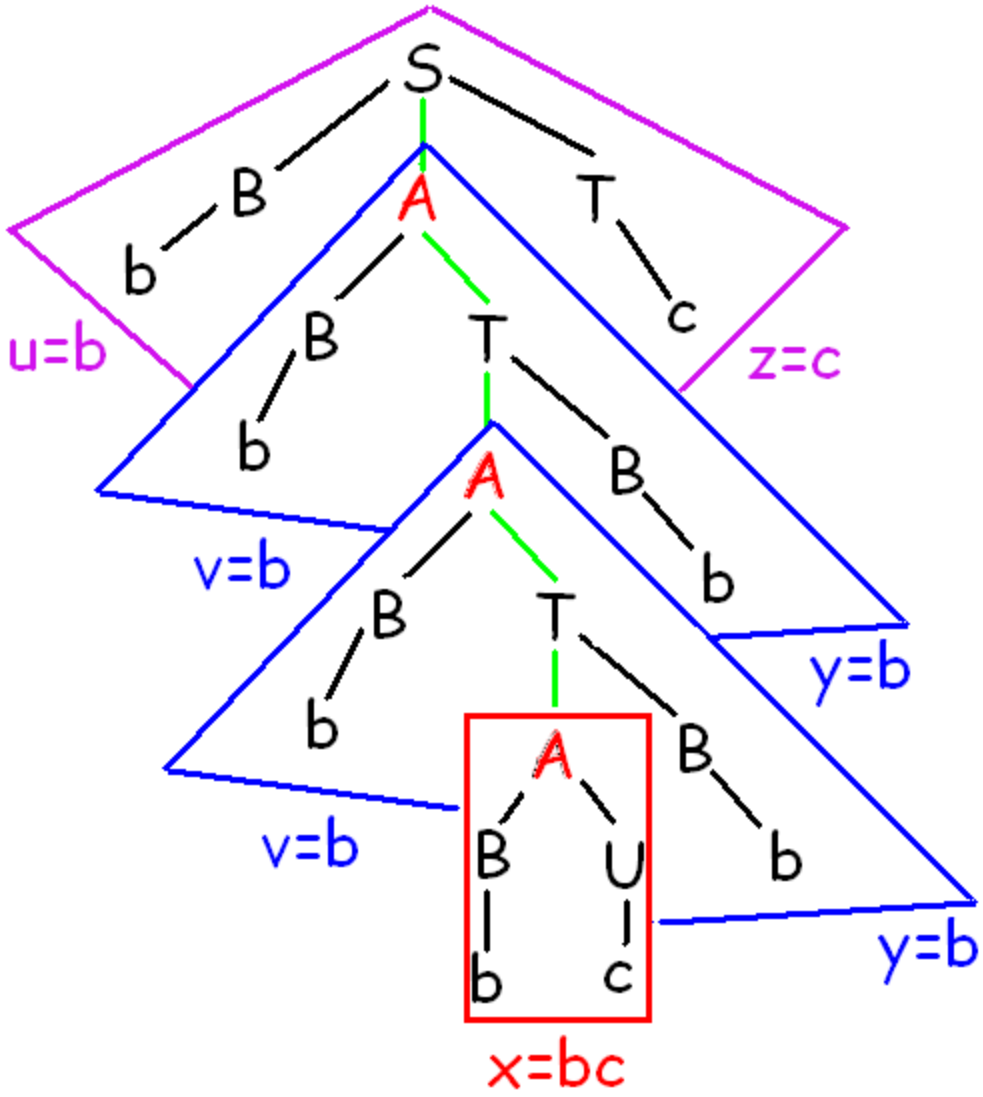
u x z = b bc c

= u v^0 x y^0 z

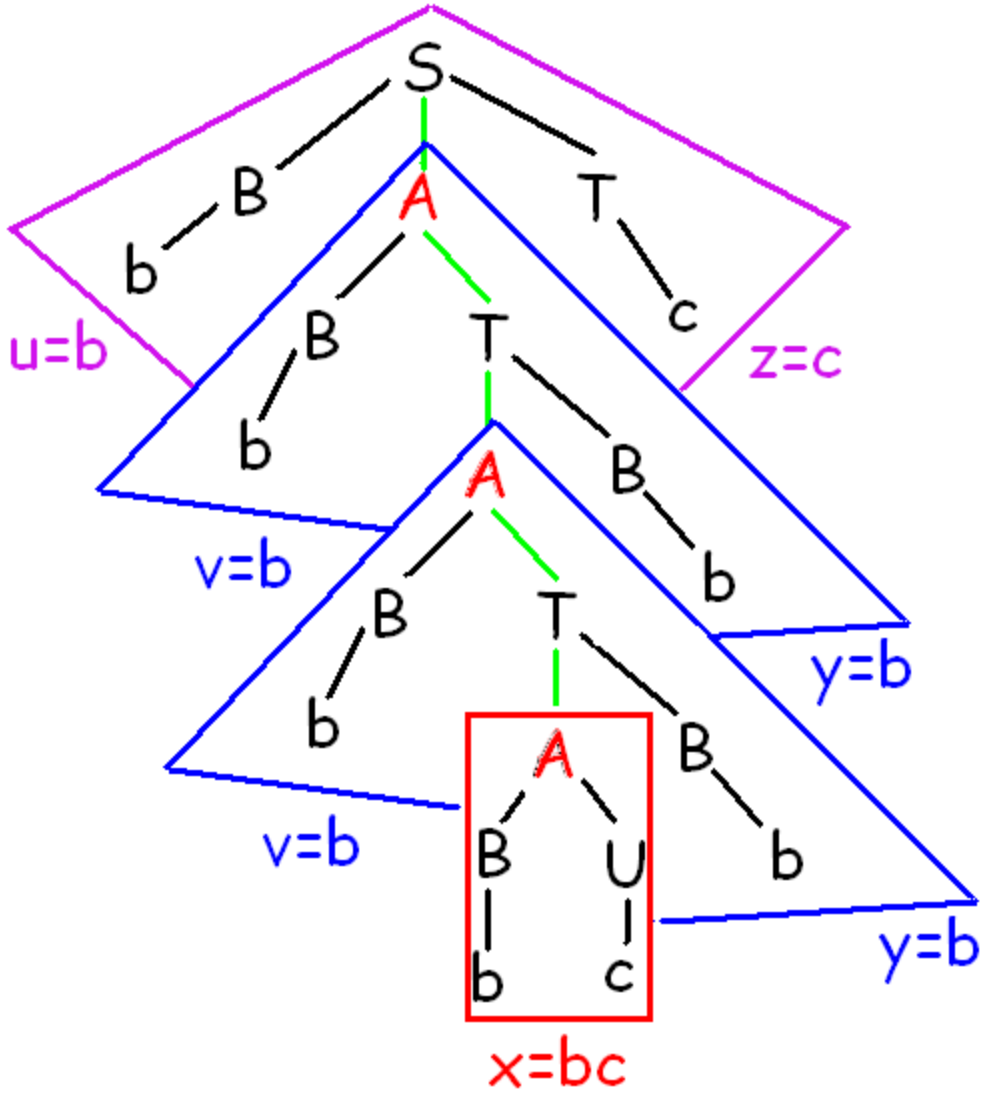




To pump twice:



Pumping twice:



New yield is:

u v v x y y z

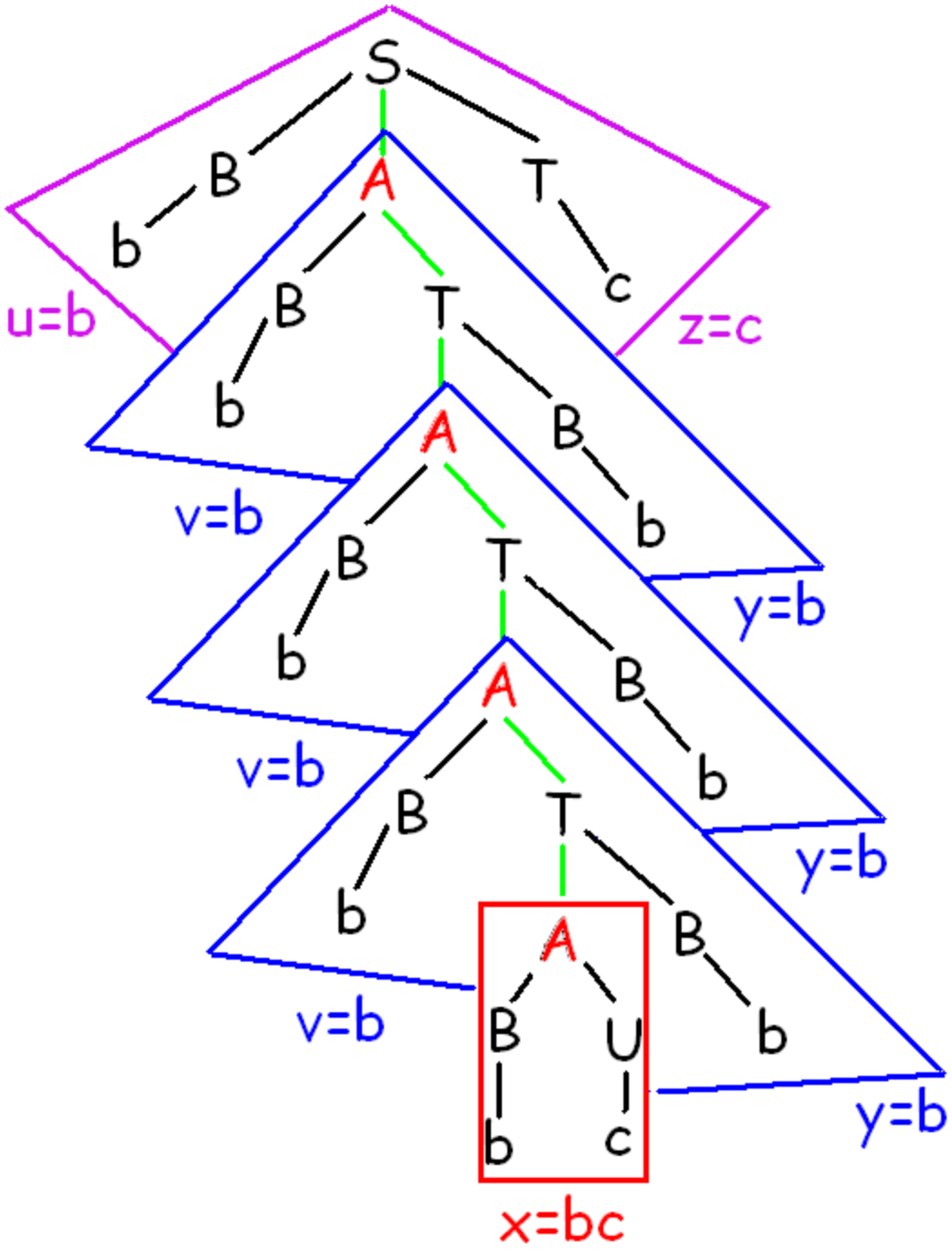
= u v^2 x y^2 z

To pump three times:

New yield is:

u vvv x yyy z

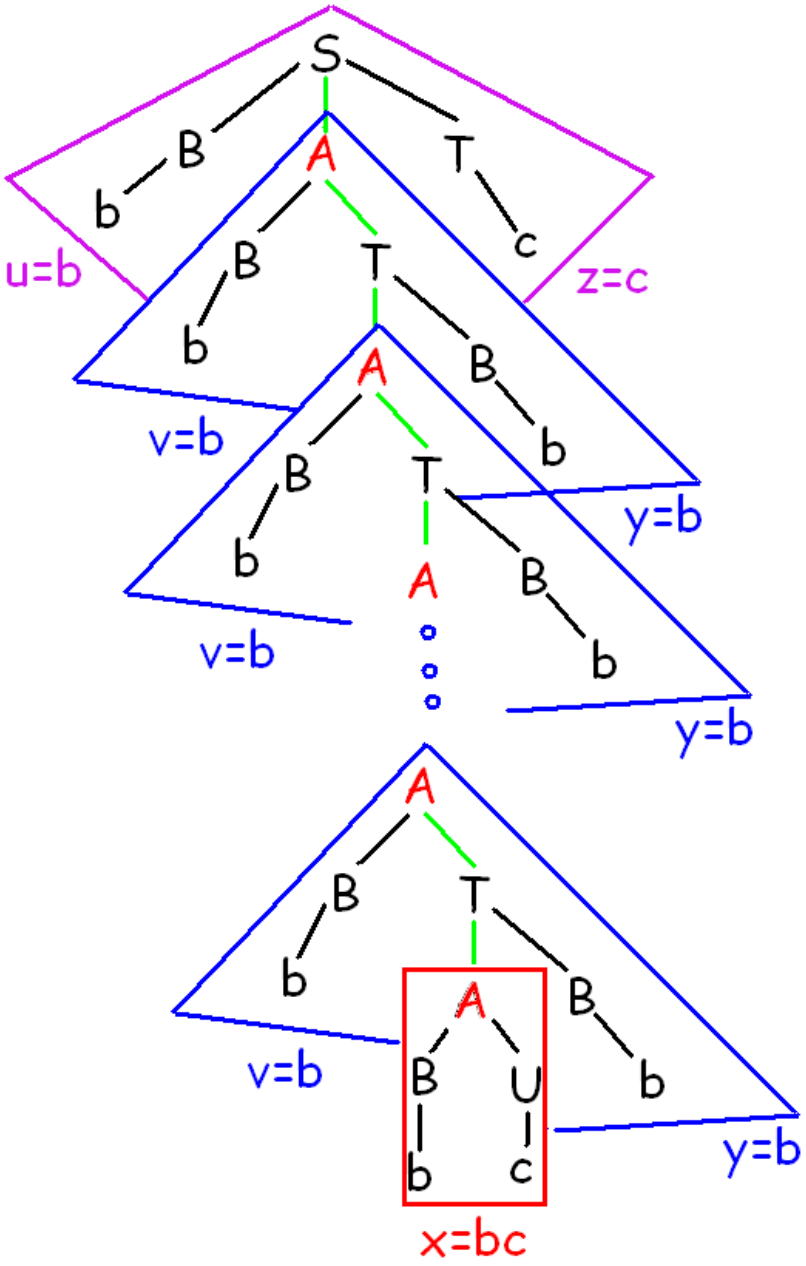
= u v^3 x y^3 z



To pump n times:

New yield is:

$= u\, v^n\, x\, y^n\, z$



The Pumping Theorem for Context-Free Languages:

Let G be a context-free grammar.

Then there exists some constant k which depends on G such that for any string w which is generated by G with $|w| \geq k$,

there exists u, v, x, y, z, such that

1. $w = u\ v\ x\ y\ z$,
2. $|v| + |y| \geq 1$, and
3. $u\ v^n\ x\ y^n\ z$ is in L for all $n \geq 0$.

Prove the following languages are context-free by designing context-free grammars which generate them:

$L_1 = \{a^n b^n c^p : n, p \geq 0\}$

$L_2 = \{a^n b^p c^n : n, p \geq 0\}$

$L_3 = \{a^n b^m : n \neq m, \ n, m \geq 0\}$

Hint: $L_3 = \{a^n b^m : n < m\} \cup \{a^n b^m : n > m\}$

$L_4 = \{ c\ u\ c\ v\ c : |u| = |v|, \ u, v \in \{a, b\}^* \}$

What is $L_1 \cap L_2$?