Preparing Slides Using LaTeX, Pstricks, and Beamer

Uday Khedker

(www.cse.iitb.ac.in/~uday)

Department of Computer Science and Engineering,

Indian Institute of Technology, Bombay

August 2010

Outline

- Using LaTeX for document preparation
- Using Pstricks for drawing pictures
- Using Beamer for making presentations

Part 2

Using LaTeX for Document Preparation

Document Preparation

- Typesetting = Text (To Be Typeset) + Typesetting Commands
- Document Structure : Position, size, shape of entities etc.
  - Visual Structure : Governed by visual aesthetics
  - Logical Structure : Governed by the meaning
    (List, Table, Chapter, Section, etc.)

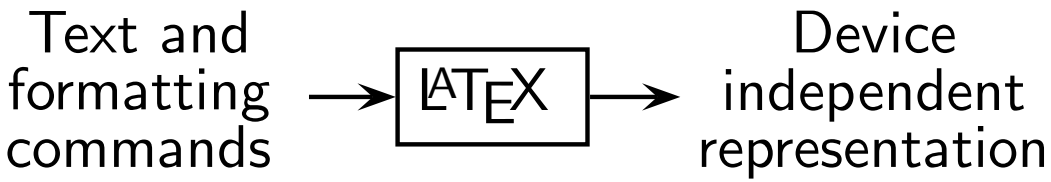
WYSIWYG Preparation

- What You See Is What You Get (E.g. MS Word.)
- Interactive : Interleaved typing and typesetting.
  - As you type the text, the resulting formatting is shown immediately and automatically.
  - Visual structure is more prominent.

Non-WYSIWYG Preparation

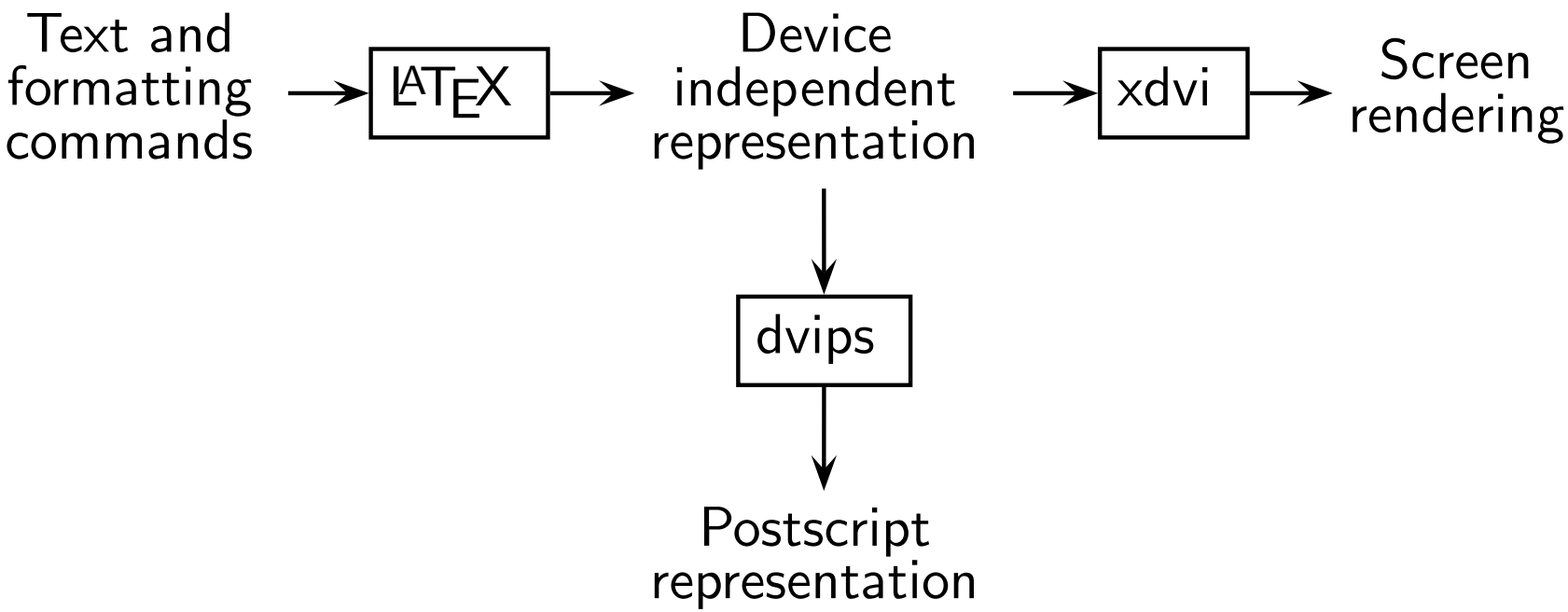
- Execution of formatting commands separate from keying in the text.
  E.g. LaTeX.
- Multi-step batch mode process
  - Type the text
  - Execute the formatting commands
  - View the resulting document
- Visual structure de-emphasized :
  Can't see immediately and automatically.

Document Preparation with LaTeX

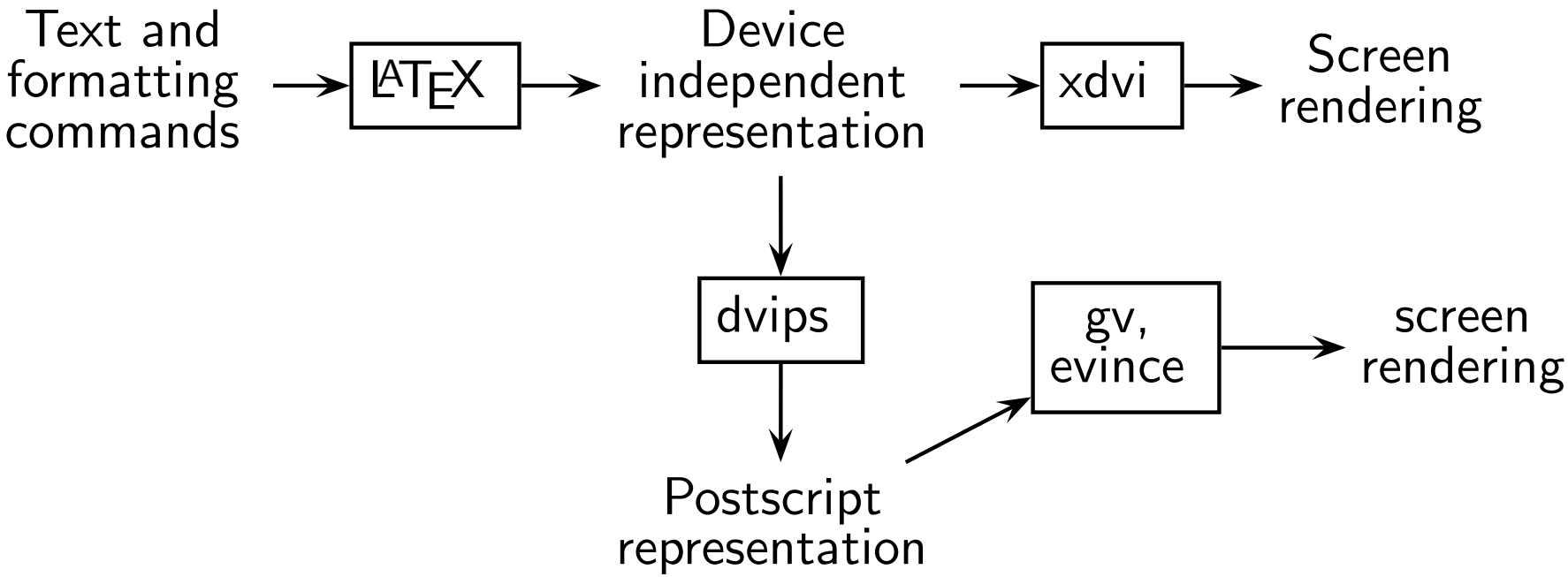


Document Preparation with LaTeX

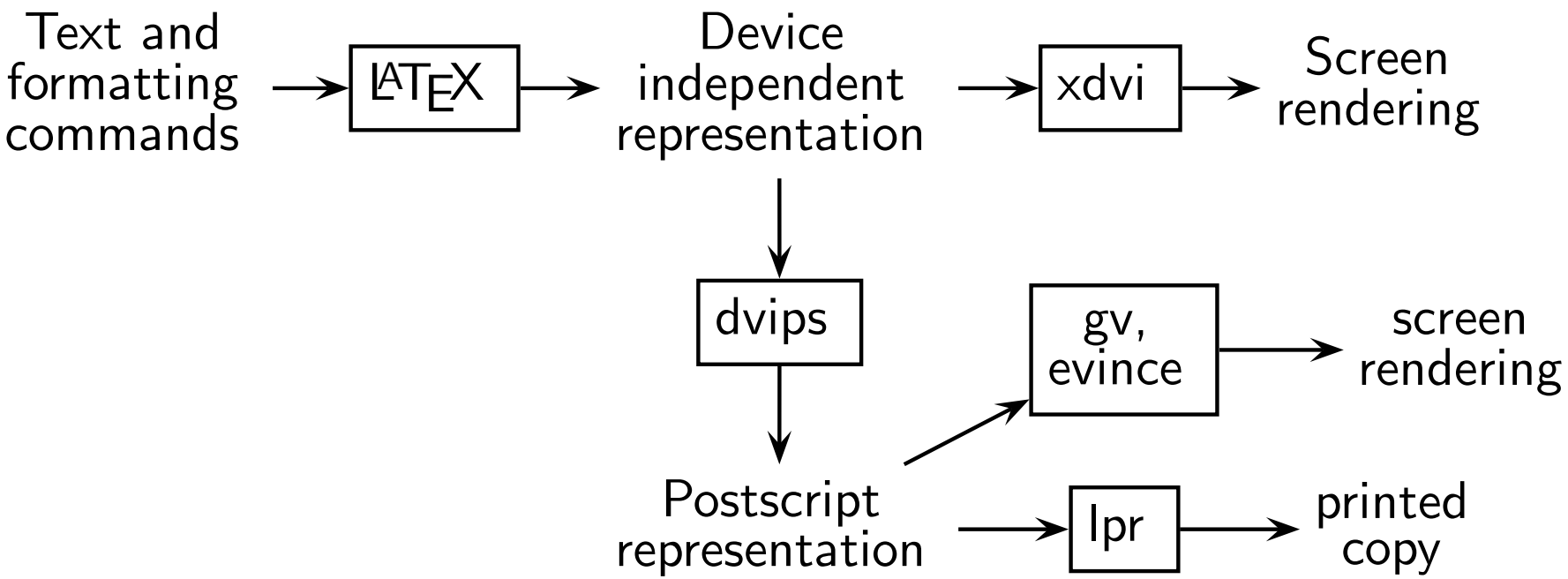
Document Preparation with LaTeX



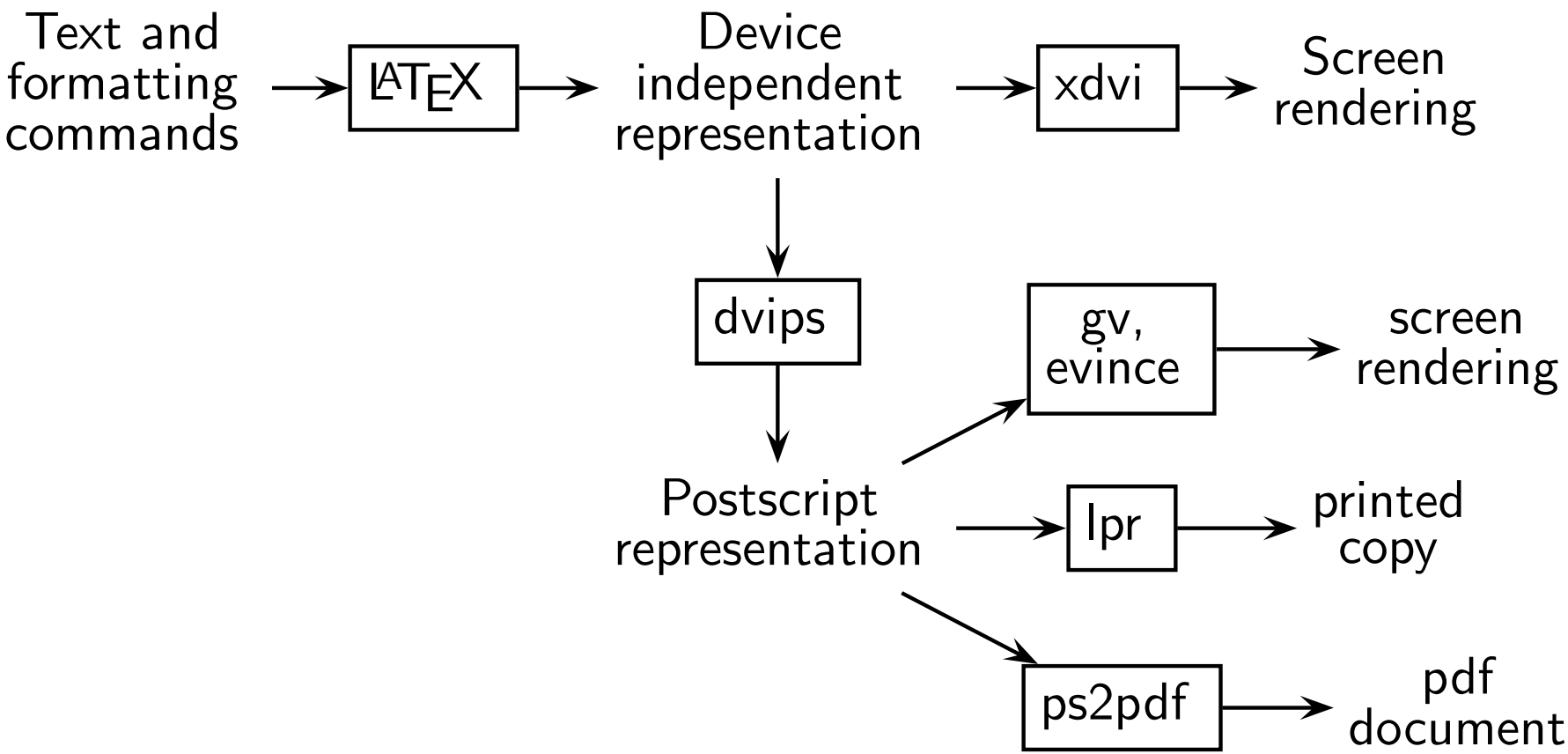
Document Preparation with LaTeX



Document Preparation with LaTeX



Document Preparation with LaTeX



Using LaTeX

- Create `file.tex`
- "`latex file.tex`" produces `file.dvi`
- "`dvips -o file.ps file`" produces `file.ps`
- Can be viewed using "`gv file.ps`"
- Practical tips for Linux users
  - Use of makefile, simultaneous editing and background viewing.
  - Almost interactive

Types of Formatting Commands

- Environment : Contains text to be typeset with a specific logical structure.
  Figures, tables, lists, equations, etc.
- Command : Produces some text in a specific way
  Section headings, footnotes etc.
- Declaration : Customizes the formatting of the text in the scope

Environments

- Environments explicate a logical structure
  Figures, tables, lists, equations, etc.
  - Names : `document, itemize, tabular, table, figure, ...`
  - Scope : `\begin{env } ... \end{env }`

  Example `\begin{document} ... \end{document}`

Commands

- Commands carry out a certain formatting
  (May have side effects)
  - `\chapter{Introduction}`
    Begins a new page.
    Changes the numbering of sections, figures, equations etc.
  - `\foilhead{Commands}`
  - `\textbf{Text to be typeset in bold face}`
  - `\texttt{Text to be typeset in typewrite font}`
  - `\footnote{Text to be typeset as a footnote}`

Types of Formatting Commands

- Declarations
  - Customization of fonts, shape, thickness, numbering, etc.

    \tt indicates `typewriter` font
    \bf indicates **boldface** letter
    \em indicates *emphasized* letters

  - Scope
    Delimited by "{" and "}", "\begin" and "\end" pairs, or ...

LaTeX: Basic Concepts

- Document Classes (`article, report, book` etc)
- Use of packages
- Fonts and Colors
- Sectioning: Chapters, sections, appendix etc
- Lists and enumerations

LaTeX: Basic Concepts

- Paragraphs
- Formatting of Math formulae
- Tables and Figures
- Page formatting
- Footnotes

LaTeX: Basic Concepts

- Multiple input files
- Defining new commands
- Importing files
- Citations and references

LaTeX: Advanced Concepts

- Formatting programs/algorithms
- Bibtex
- Pictures
- Slides

Part 3

Using Pstricks for Drawing Pictures

Preparing Pictures using Pstricks

- Environment `pspicture`
- Line and curve drawings
- Frames, circles, ovals,
- Nodes and Node connectors
  Relative to the placement of nodes
- Labeling node connectors

The Power of Pstricks

- Logical components of pictures and relationships between them.
  ⇒ Easy refinements/updates/corrections
  - `xfig` does not recognise node-connectors.
    ⇒ If you move a node, a node connector does not move with it.
  - `dia` recognises node-connectors but not the relationship between nodes.
    ⇒ A node connector moves with a node but positioning of two nodes remains independent.
- Very good quality of pictures.
- Free mixing of graphics and text

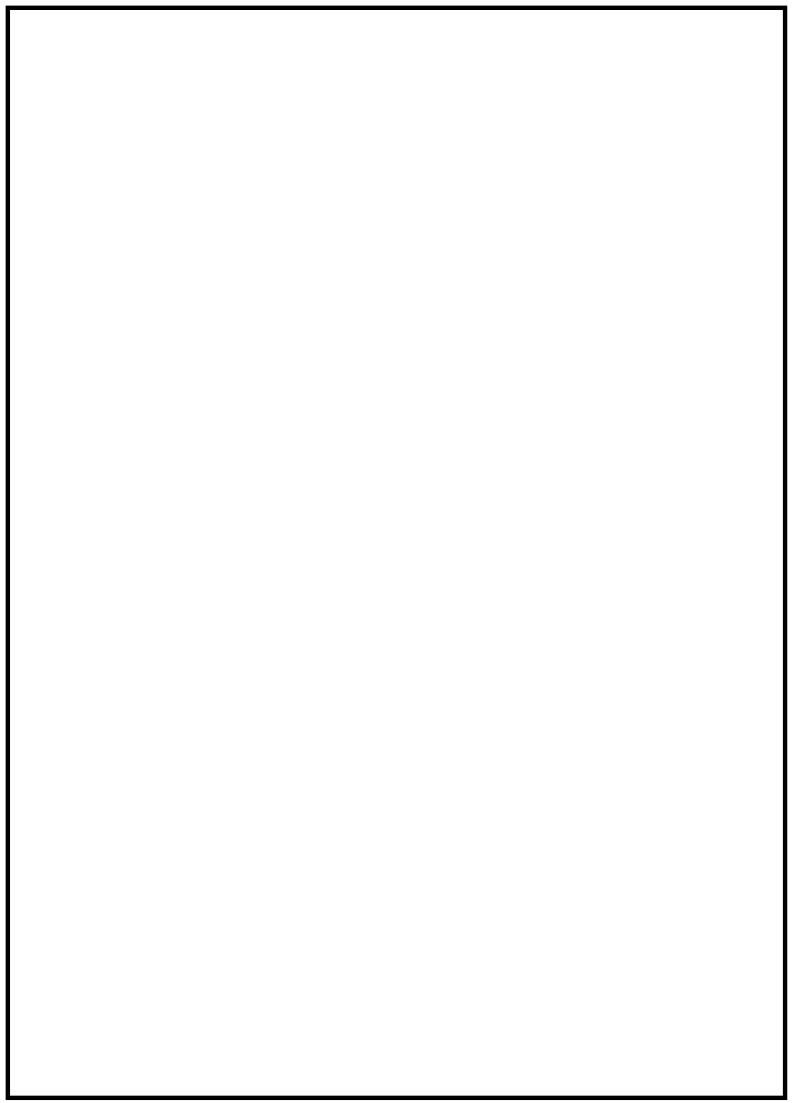
Adding to the Power of Pstricks

- A limitation of `pstricks`
  Absolute coordinates have to be calculated by the user.
- Solution : package `pst-rel-points` available at
  `http://www.cse.iitb.ac.in/uday/latex`.
- Defines command

```
\putnode[l/r]{new}{old}{delta x}{delta y}{stuff}
```

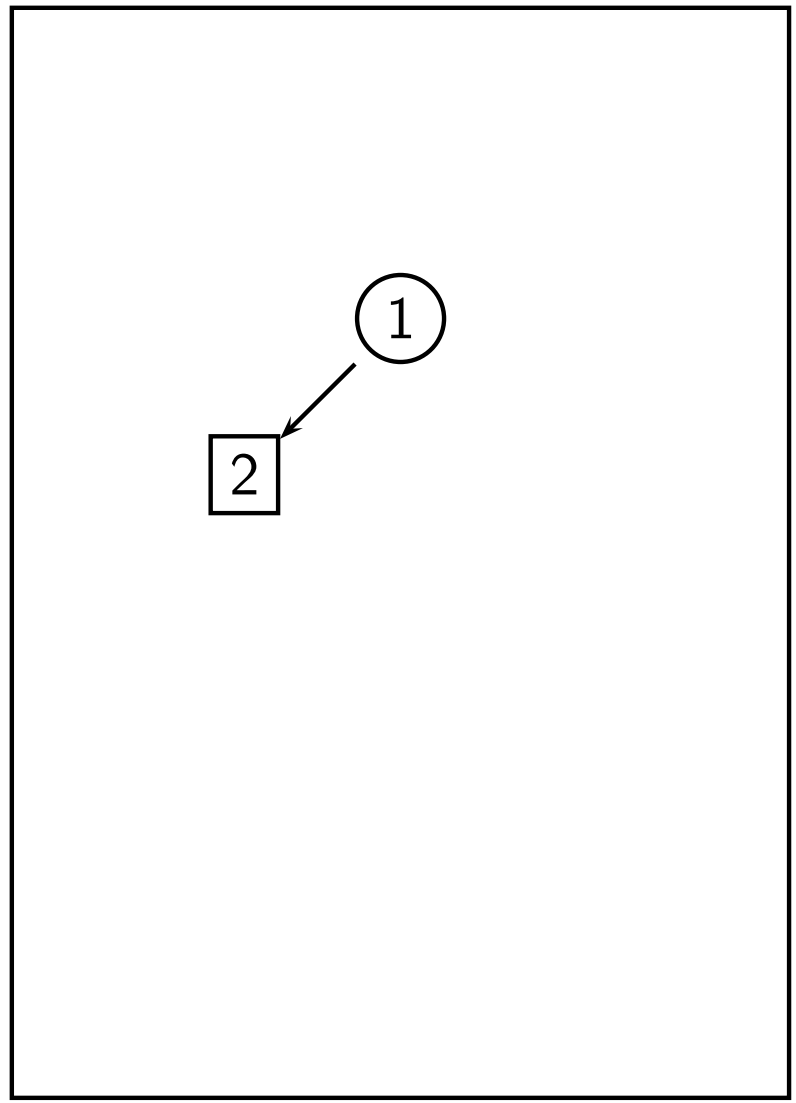

Drawing Pictures Using Pstricks

```
\usepackage{pstricks}
\usepackage{pst-node}
\usepackage{pst-text}
\usepackage{etex}
\usepackage{pst-rel-points}
%%
\psset{unit=1mm}
\begin{pspicture}(0,0)(50,70)
\psframe(0,0)(50,70)
\end{pspicture}
```



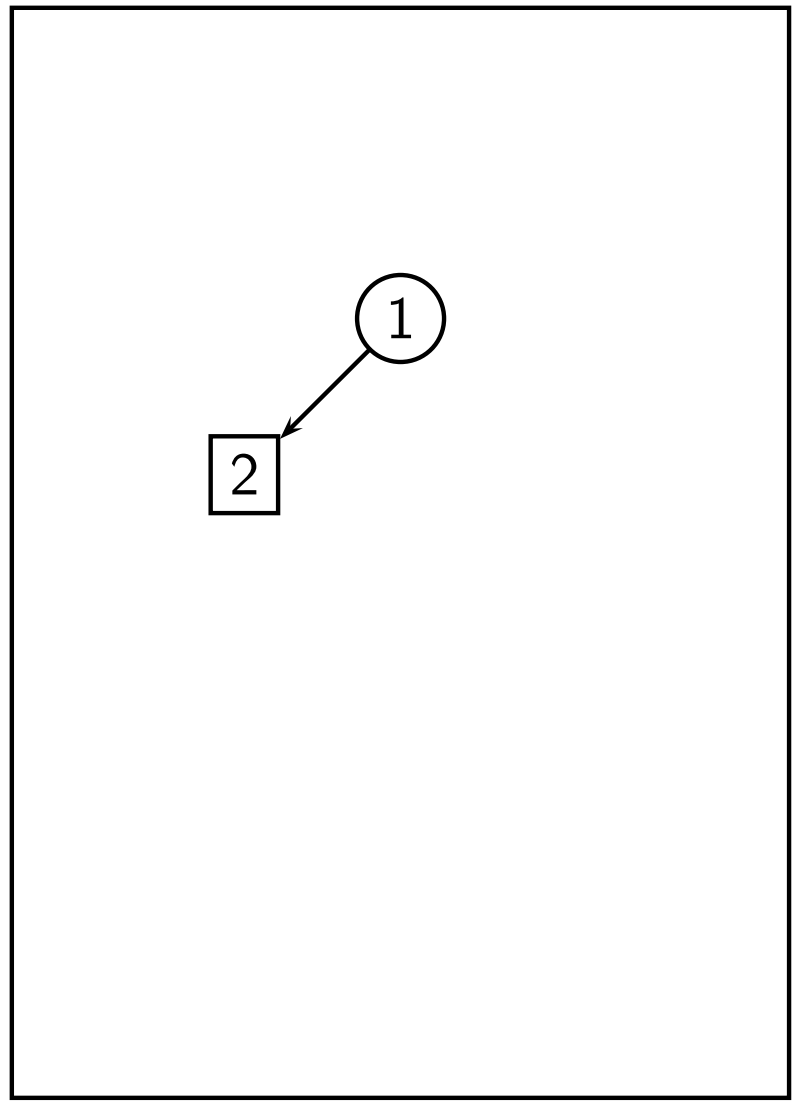
Drawing Pictures Using Pstricks

```
\usepackage{pstricks}
\usepackage{pst-node}
\usepackage{pst-text}
\usepackage{etex}
\usepackage{pst-rel-points}
%%
\psset{unit=1mm}
\begin{pspicture}(0,0)(50,70)
\psframe(0,0)(50,70)
\putnode{n1}{origin}{25}{50}{%
     \pscirclebox{1}}
\putnode{n2}{n1}{-10}{-10}{%
     \psframebox{2}}
\ncline{->}{n1}{n2}
\end{pspicture}
```



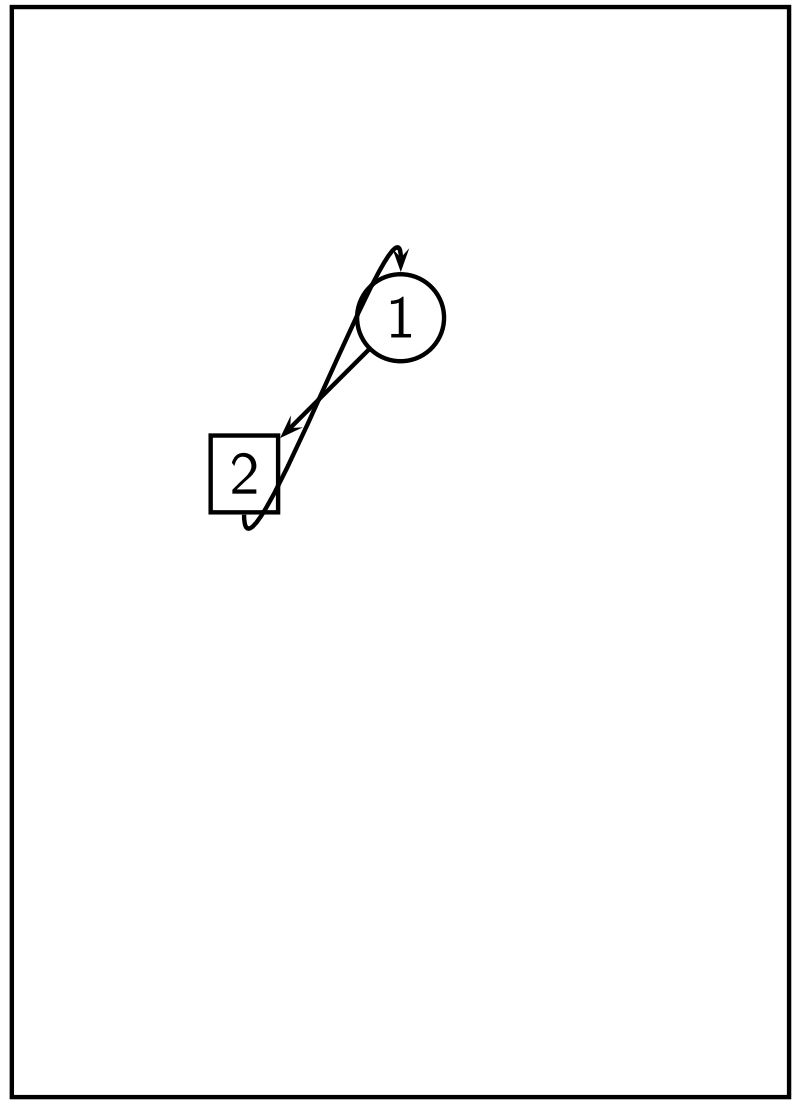
Drawing Pictures Using Pstricks

```
\usepackage{pstricks}
\usepackage{pst-node}
\usepackage{pst-text}
\usepackage{etex}
\usepackage{pst-rel-points}
%%
\psset{unit=1mm}
\begin{pspicture}(0,0)(50,70)
\psframe(0,0)(50,70)
\putnode{n1}{origin}{25}{50}{%
     \pscirclebox{1}}
\putnode{n2}{n1}{-10}{-10}{%
     \psframebox{2}}
\ncline[nodesepA=-1]{->}{n1}{n2}
\end{pspicture}
```



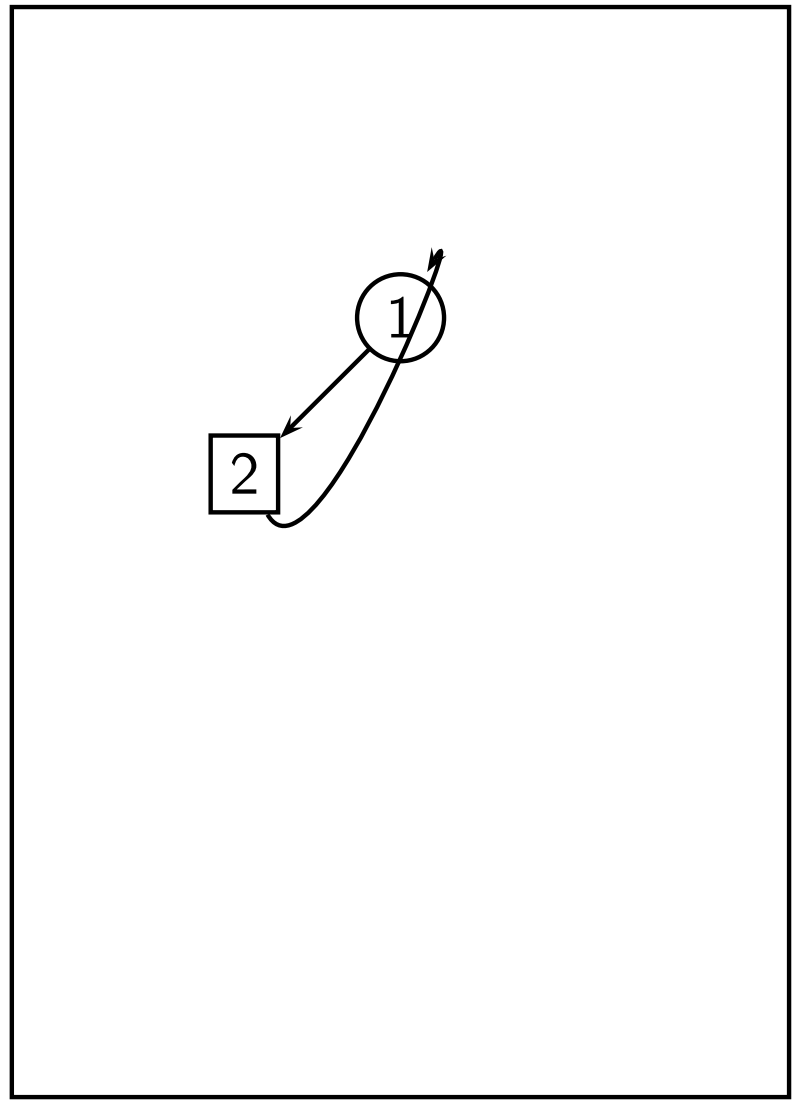
Drawing Pictures Using Pstricks

```
\usepackage{pstricks}
\usepackage{pst-node}
\usepackage{pst-text}
\usepackage{etex}
\usepackage{pst-rel-points}
%%
\psset{unit=1mm}
\begin{pspicture}(0,0)(50,70)
\psframe(0,0)(50,70)
\putnode{n1}{origin}{25}{50}{%
     \pscirclebox{1}}
\putnode{n2}{n1}{-10}{-10}{%
     \psframebox{2}}
\ncline[nodesepA=-1]{->}{n1}{n2}
\nccurve[angleA=270,angleB=90]%
      {->}{n2}{n1}
\end{pspicture}
```



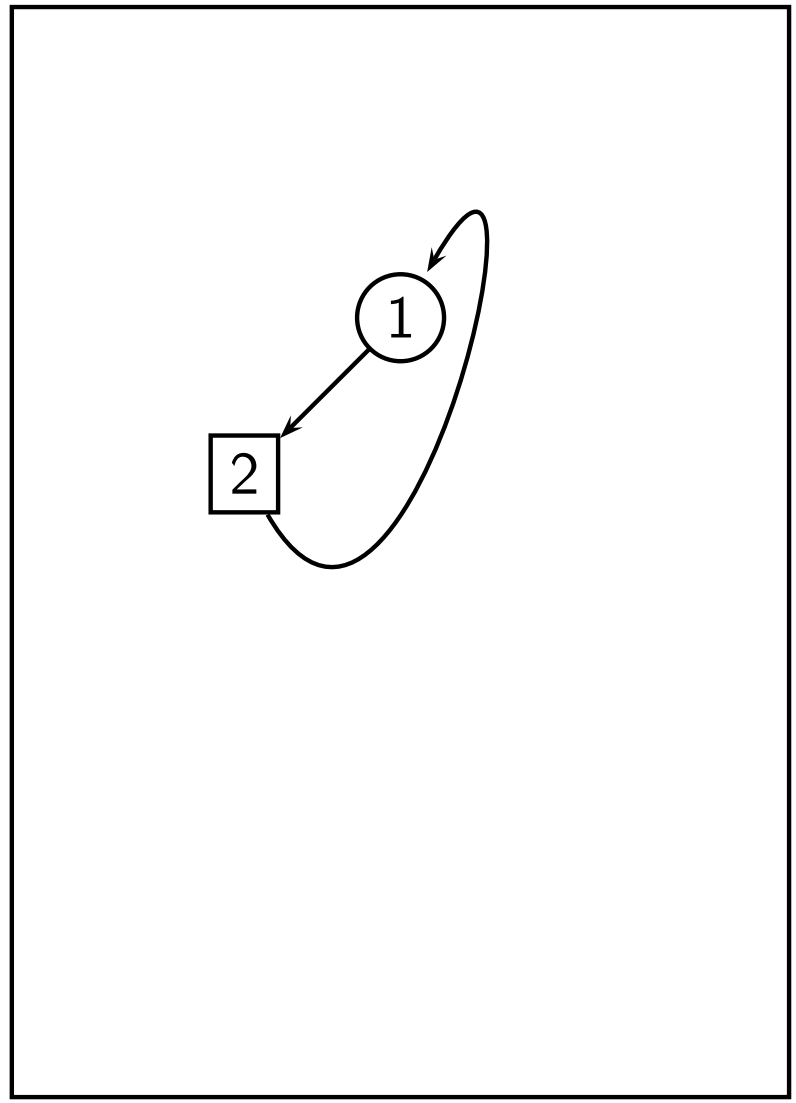
Drawing Pictures Using Pstricks

```
\usepackage{pstricks}
\usepackage{pst-node}
\usepackage{pst-text}
\usepackage{etex}
\usepackage{pst-rel-points}
%%
\psset{unit=1mm}
\begin{pspicture}(0,0)(50,70)
\psframe(0,0)(50,70)
\putnode{n1}{origin}{25}{50}{%
     \pscirclebox{1}}
\putnode{n2}{n1}{-10}{-10}{%
     \psframebox{2}}
\ncline[nodesepA=-1]{->}{n1}{n2}
\nccurve[angleA=300,angleB=60]%
      {->}{n2}{n1}
\end{pspicture}
```



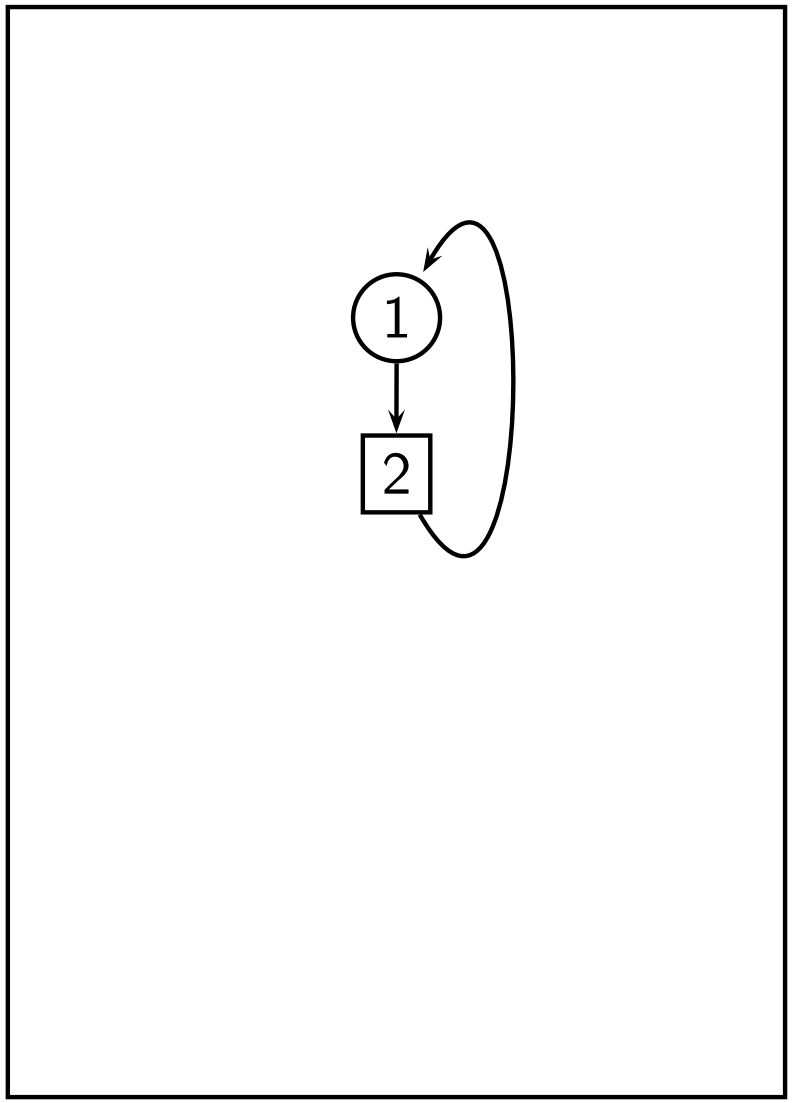
Drawing Pictures Using Pstricks

```
\usepackage{pstricks}
\usepackage{pst-node}
\usepackage{pst-text}
\usepackage{etex}
\usepackage{pst-rel-points}
%%
\psset{unit=1mm}
\begin{pspicture}(0,0)(50,70)
\psframe(0,0)(50,70)
\putnode{n1}{origin}{25}{50}{%
     \pscirclebox{1}}
\putnode{n2}{n1}{-10}{-10}{%
     \psframebox{2}}
\ncline[nodesepA=-1]{->}{n1}{n2}
\nccurve[angleA=300,angleB=60,%
      ncurv=2]{->}{n2}{n1}
\end{pspicture}
```



Drawing Pictures Using Pstricks

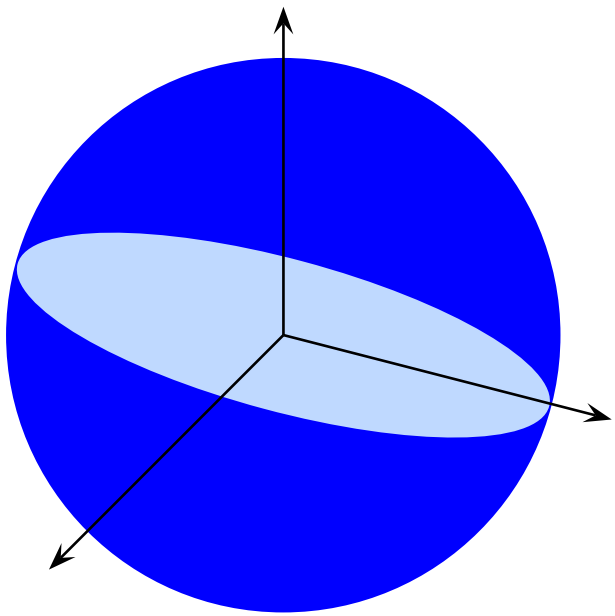
```
\usepackage{pstricks}
\usepackage{pst-node}
\usepackage{pst-text}
\usepackage{etex}
\usepackage{pst-rel-points}
%%
\psset{unit=1mm}
\begin{pspicture}(0,0)(50,70)
\psframe(0,0)(50,70)
\putnode{n1}{origin}{25}{50}{%
     \pscirclebox{1}}
\putnode{n2}{n1}{0}{-10}{%
     \psframebox{2}}
\ncline[nodesepA=-1]{->}{n1}{n2}
\nccurve[angleA=300,angleB=60,%
     ncurv=2]{->}{n2}{n1}
\end{pspicture}
```



More Pictures Using Pstricks

```
\newcommand{\sphere}{%
\psset{unit=1mm,arrowsize=6pt}
\begin{pspicture}(0,5)(120,110)
\rput(30,60){%
    \pscirclebox*[fillcolor=blue]{%
    \rule{5.7cm}{0cm}}}
\rput{-15}(30,60){%
    \psovalbox*[fillcolor=lightblue]{%
    \rule{4cm}{0cm}\rule{0cm}{1cm}}}
\psline{->}(30,60)(70,50)
\psline{->}(30,60)(30,100)
\psline{->}(30,60)(0,30)
\end{pspicture}}
%%

\scalebox{.6}{\sphere}
```



A Demo of Using Pstricks

- \ncline, \nccurve \ncloop
- Optional arguments
- Minipage and footnote
- \rnode and connectors between text and picture

Part 4

Using Beamer for Preparing Slides

An Overview of Beamer

- Presentations based on frames consistings of slides
- In beamer terminology, "slides" refers to overlays appearing in a frame
  Facilitate animations
- Convenient overlay mechanism
- Same source can be compiled to presentations, handouts, documents
- Multiple themes or templates

Instantiating a Template

- \title[short title]{long title}
- \subtitle[short subtitle]{long subtitle}
- \author[short name]{long name}
- \date[short date]{long date}
- \institution[short name]{long name}

Template Instantiation for this Presentation

```
\usetheme{iitb}

%%

\title[LaTeX]{Preparing Slides Using \\ LaTeX, Pstricks,
              and Beamer}
\author[Aug 2010]{Uday Khedker}
\institute[Uday Khedker, IIT Bombay]{Department of
              Computer Science and Engineering, \\
              Indian Institute of Technology, Bombay}
\titlegraphic{\scalebox{.4}{\includegraphics{IITBlogo.epsi}}}
\date[Prabhat Workshop]{August 2010}
```

Frames

- A separately numbered page in the presentation
- All overlays (i.e. slides) in a frame share the same page number
- Created by the following options

`\begin{frame}[options]`
`\frametitle{Title}`
 	
`%% LaTeX commands for`
`%% frame contents`
 	
`\end{frame}`
	`\frame{`
	`\frametitle{Title}`
	
	`%% LaTeX commands for`
	`%% frame contents`
	
	`}`

Useful Options for Frames

- `[plain]`. No header, title or footer
- `[fragile]`. Required for using `verbatim` environment

Using `verbatim` Environment

- Use option `[fragile]` for a frame
- Use `minipage`

```
\begin{minipage}{width}
\begin{verbatim}

\end{verbatim}
\end{minipage}
```


Using `semiverbatim` Environment

- LaTeX commands can be used but text is typeset like verbatim
- Example uses: changing color or size of text

Creating Overlays

- Common Commands: `\only`, `\onslide`, `\pause`
- Common Environments: `\begin{onlyenv} ...\end{onlyenv}`
- Common Range Specification:
  - From n to m: `<n-m>`
  - From n onwards: `<n->`
  - After the previous one and until m: `<+-m>`
  - From beginning until m: `<-m>`
  - On m, n, and i: `<m,n,i>`

Overlays in a List

- Explicitly ordered

```
\begin{itemize}
\item<1-> This is the first item
\item<2-> This is the second item
\item<3-> And this is the third
\end{itemize}
```

Overlays in a List

- Explicitly ordered

```
\begin{itemize}
\item<1-> This is the first item
\item<2-> This is the second item
\item<3-> And this is the third
\end{itemize}
```

- Implicitly ordered

```
\begin{itemize}
\item<+-> This is the first item
\item<+-> This is the second item
\item<+-> And this is the third
\end{itemize}
```


More on Overlays and Themes

- Excellent examples at
  http://www.uncg.edu/cmp/reu/presentations/Charles Batts - Beamer Tutorial.pdf
  (include spaces in the file name and replace new line by a space)

Converting Slides to Handouts

- Step 1: Modify the range specifications
  - If slides that appear between 1 to 5 should appear on handout slide 2
    `<1-5|handout:2>`
  - Slide 6 to 8 should appear only in the presentation but not in the handout
    `<6-8|handout:0>`
  - Slide 9 onwards should appear only in the handout but not in the presentation
    `<0|handout:9->`

Converting Slides to Handouts

- Step 2: Add handout declarations in the preamble

```
\usepackage{pgf,pgfarrows,pgfnodes,pgfautomata,pgfheaps}
\mode<handout>
{
  \usepackage{pgfpages}
  \pgfpagesuselayout{4 on 1}[a4paper,landscape,%
        border shrink=5mm]
}
```

- Step 3: Change `\documentclass[dvips]beamer` to `\documentclass[handout]beamer`

Part 5

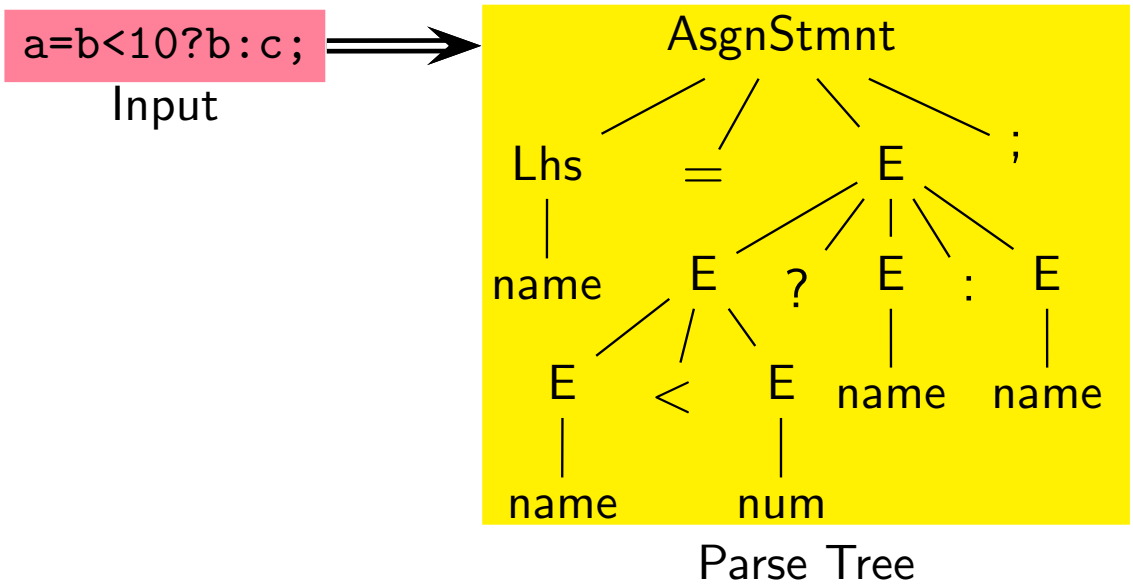
Some Sample Slides

Translation Sequence in Our Compiler: Parsing

```
a=b<10?b:c;
```

Input

Translation Sequence in Our Compiler: Parsing



Parse Tree

Issues:

- Grammar rules, terminals, non-terminals
- Order of application of grammar rules

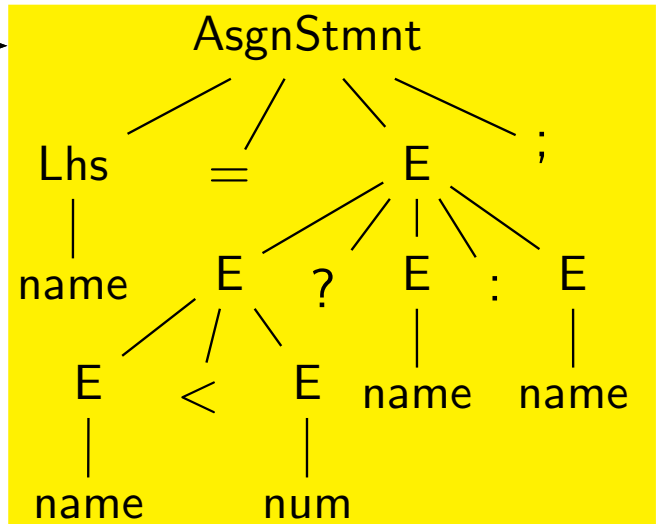
  eg. is it `(a = b<10?)` followed by `(b:c)`?
- Values of terminal symbols

  eg. string "10" vs. integer number 10.

Translation Sequence in Our Compiler: Semantic Analysis

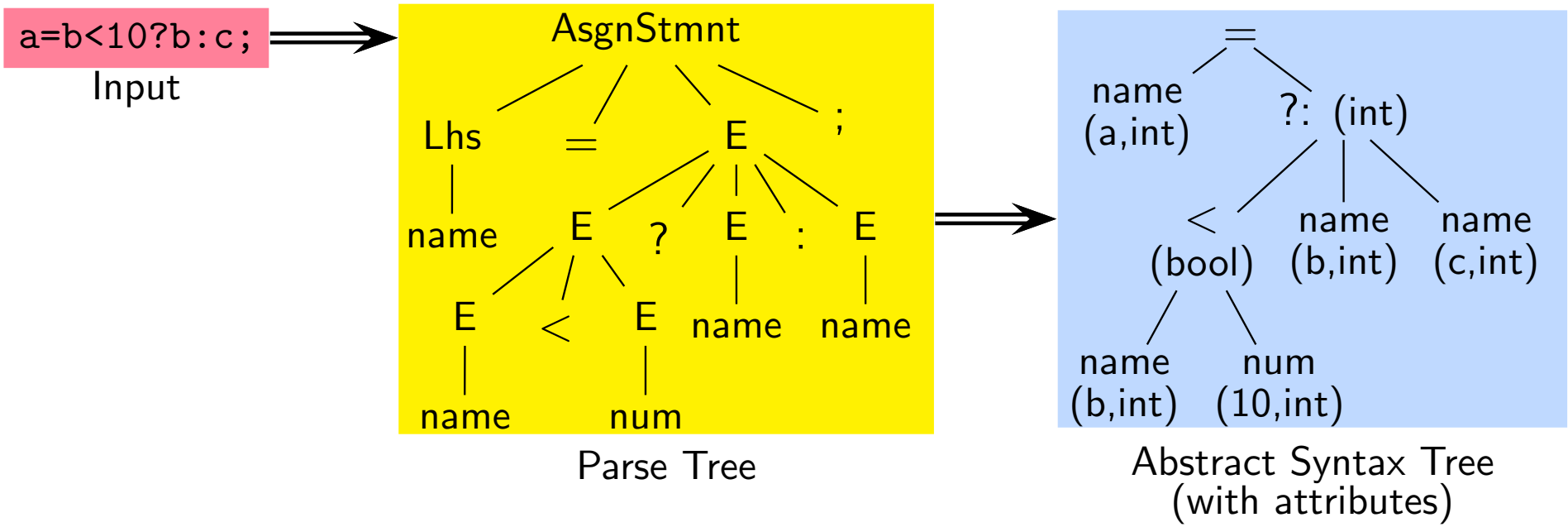
```
a=b<10?b:c;
```

Input



Parse Tree

Translation Sequence in Our Compiler: Semantic Analysis



Issues:

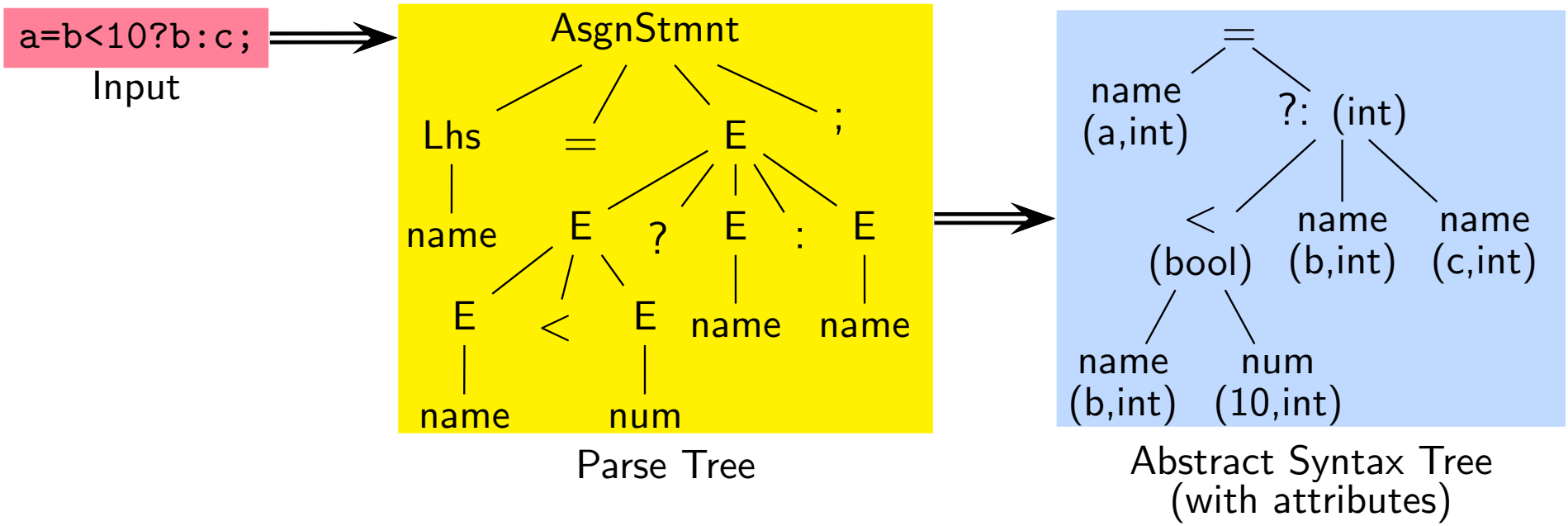
- Symbol tables

  Have variables been declared? What are their types? What is their scope?

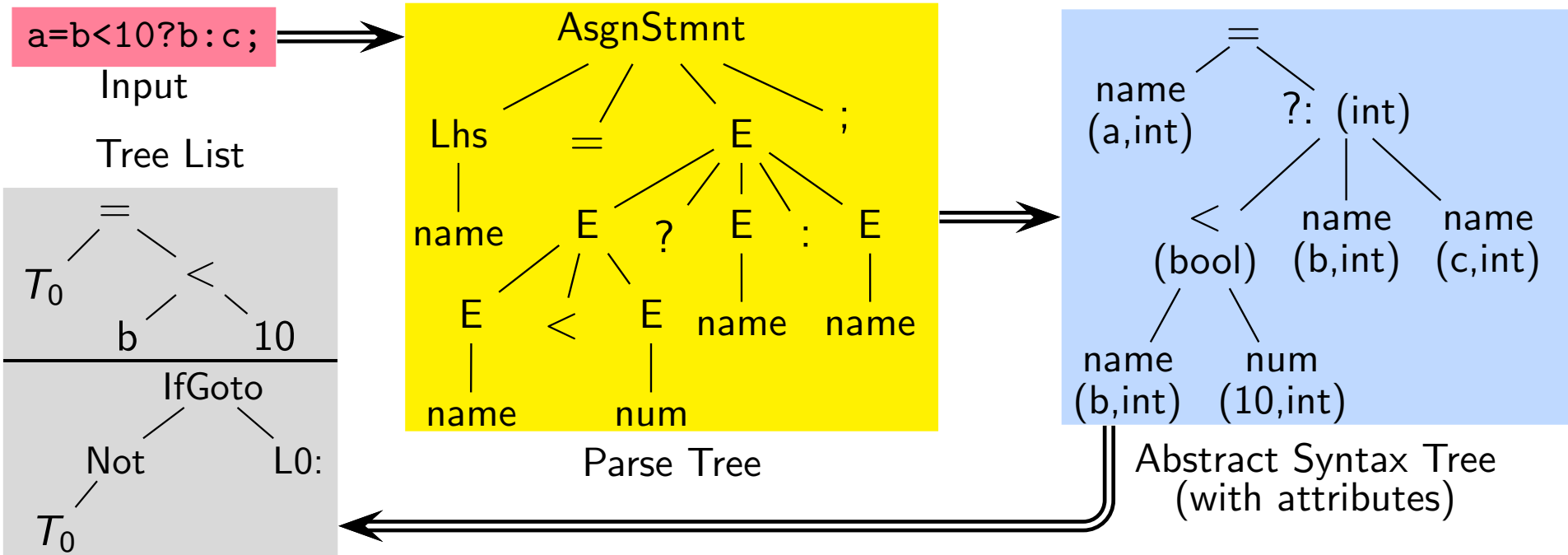
- Type consistency of operators and operands

  The result of computing b<10? is bool and not int

Translation Sequence in Our Compiler: IR Generation



Translation Sequence in Our Compiler: IR Generation

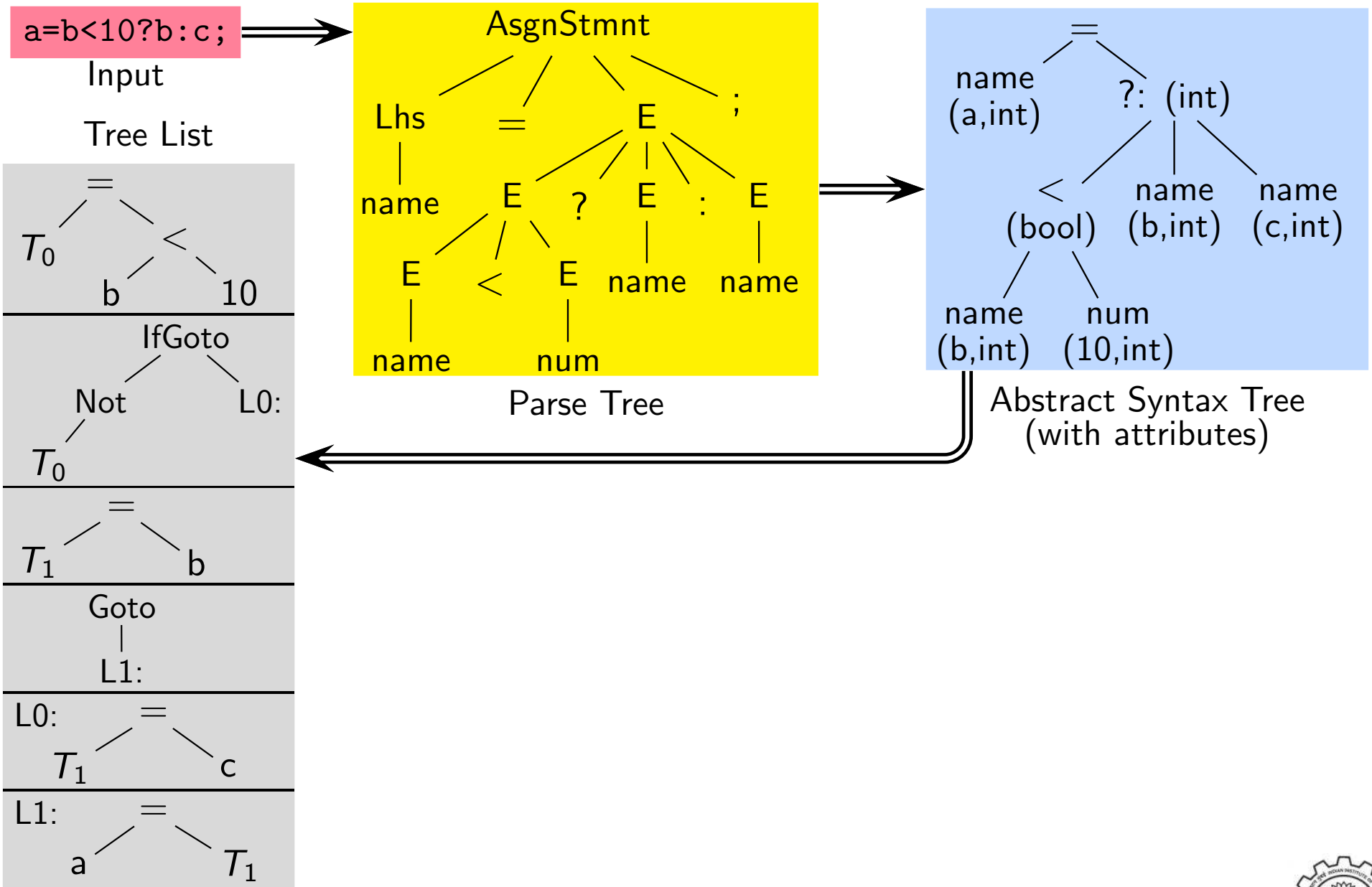


Issues:

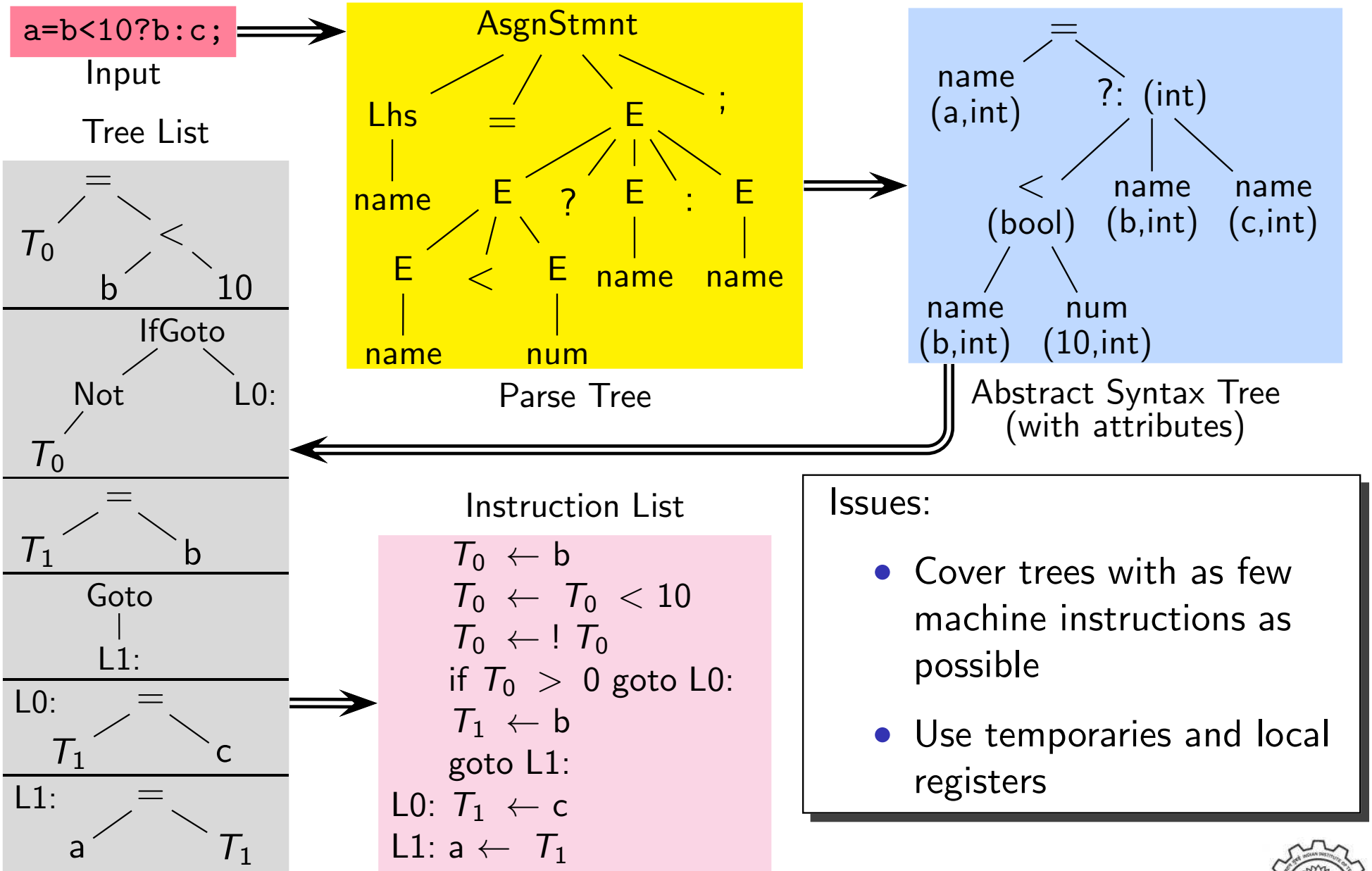
- Convert to maximal trees which can be implemented without altering control flow

  Simplifies instruction selection and scheduling, register allocation etc.
- Linearise control flow by flattening nested control constructs

Translation Sequence in Our Compiler: Instruction Selection



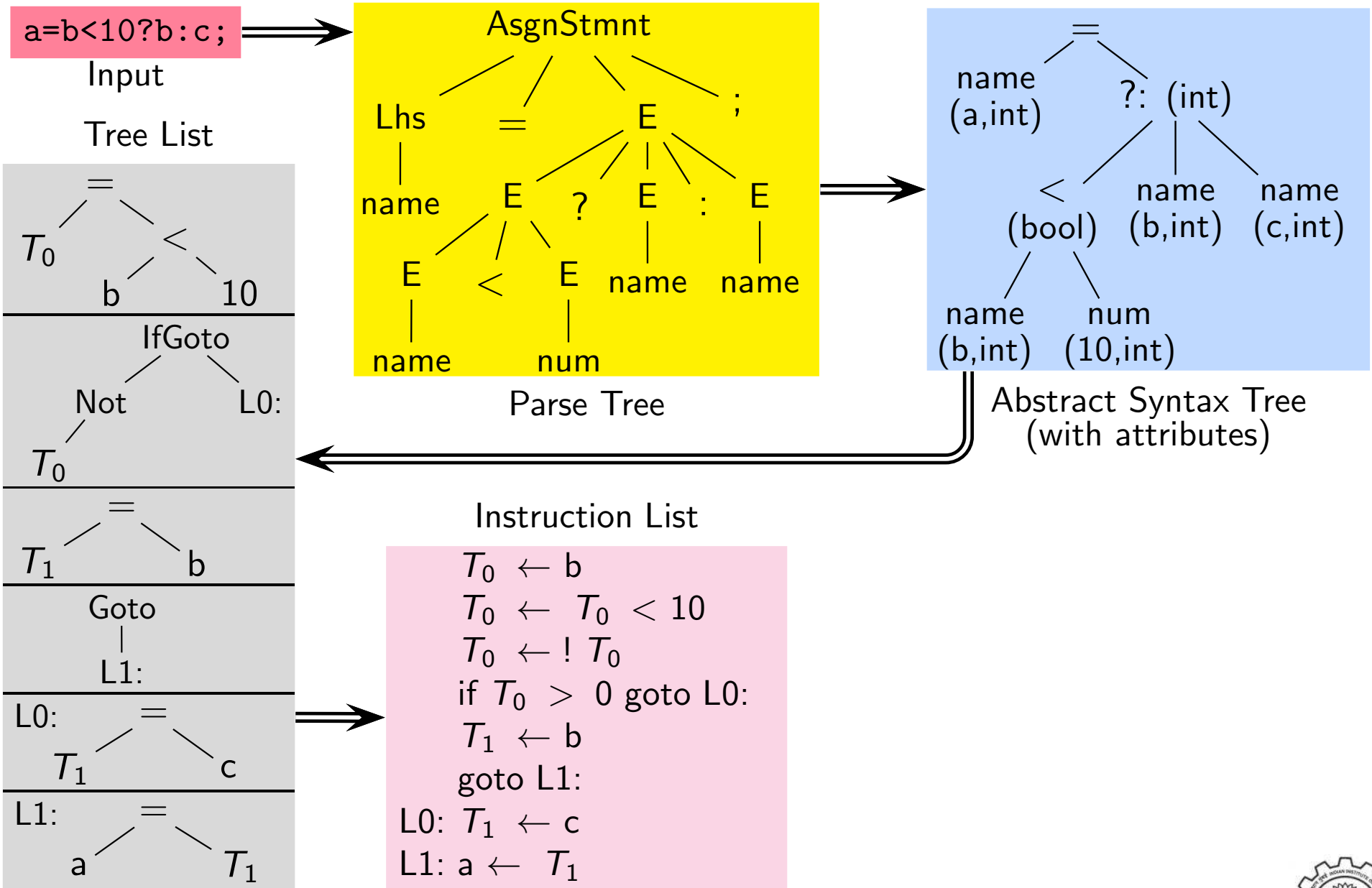
Translation Sequence in Our Compiler: Instruction Selection



Issues:

- Cover trees with as few machine instructions as possible
- Use temporaries and local registers

Translation Sequence in Our Compiler: Emitting Instructions



Translation Sequence in Our Compiler: Emitting Instructions

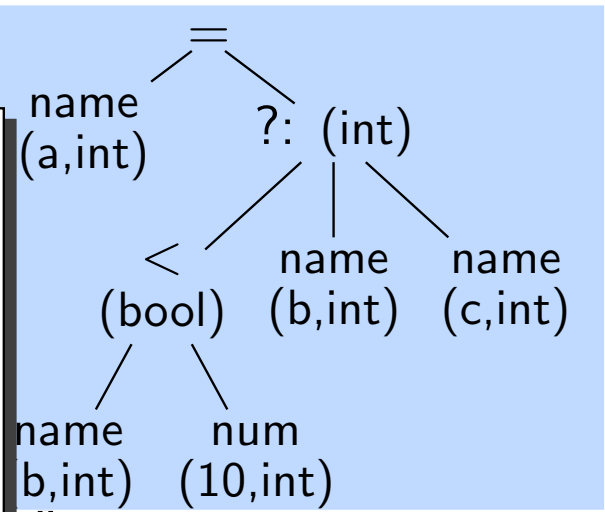
```
a=b<10?b:c;
```

Input

AsgnStmnt

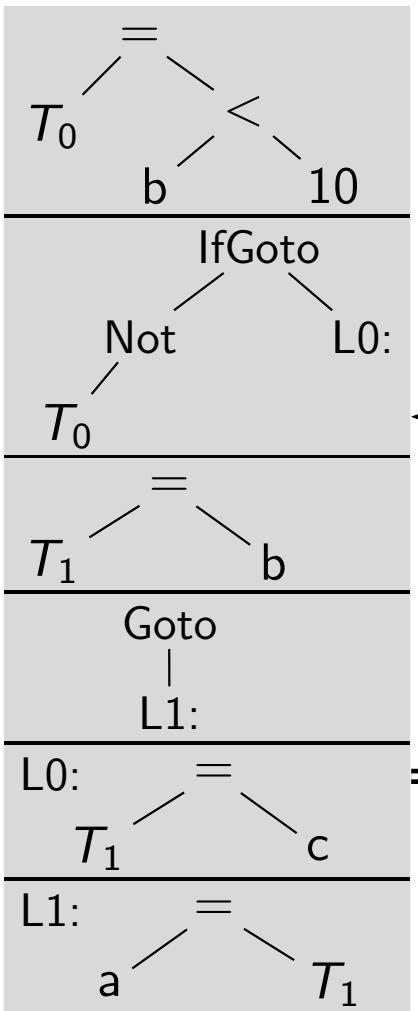
Issues:

- Offsets of variables in the stack frame
- Actual register numbers and assembly mnemonics
- Code to construct and discard activation records



Abstract Syntax Tree (with attributes)

Tree List



Instruction List

$T_0 \leftarrow$ b
$T_0 \leftarrow T_0 < 10$
$T_0 \leftarrow !\ T_0$
if $T_0 > 0$ goto L0:
$T_1 \leftarrow$ b
goto L1:
L0: $T_1 \leftarrow$ c
L1: a $\leftarrow T_1$

Assembly Code

```
    lw   $t0, 4($fp)
    slti $t0, $t0, 10
    not  $t0, $t0
    bgtz $t0, L0:
    lw   $t0, 4($fp)
    b    L1:
L0: lw   $t0, 8($fp)
L1: sw   0($fp), $t0
```

i386 Assembly

Dump file: `test.s`

```
        jmp   .L2
.L3:
        addl   $1, -4(%ebp)
.L2:
        cmpl   $7, -4(%ebp)
        jle   .L3
        cmpl   $12, -4(%ebp)
        jg    .L6
        movl   -8(%ebp), %edx
        movl   -4(%ebp), %eax
        leal   (%edx,%eax), %eax
        addl   -12(%ebp), %eax
        movl   %eax, -4(%ebp)
.L6:
```

```
while (a <= 7)
{
    a = a+1;
}
if (a <= 12)
{
    a = a+b+c;
}
```

i386 Assembly

Dump file: `test.s`

```
        jmp   .L2
.L3:
        addl  $1, -4(%ebp)
.L2:
        cmpl  $7, -4(%ebp)
        jle   .L3
        cmpl  $12, -4(%ebp)
        jg    .L6
        movl  -8(%ebp), %edx
        movl  -4(%ebp), %eax
        leal  (%edx,%eax), %eax
        addl  -12(%ebp), %eax
        movl  %eax, -4(%ebp)
.L6:
```

```
while (a <= 7)
{
    a = a+1;
}
if (a <= 12)
{
    a = a+b+c;
}
```

i386 Assembly

Dump file: `test.s`

```
        jmp   .L2
.L3:
        addl   $1, -4(%ebp)
.L2:
        cmpl   $7, -4(%ebp)
        jle   .L3
        cmpl   $12, -4(%ebp)
        jg    .L6
        movl   -8(%ebp), %edx
        movl   -4(%ebp), %eax
        leal   (%edx,%eax), %eax
        addl   -12(%ebp), %eax
        movl   %eax, -4(%ebp)
.L6:
```

```
while (a <= 7)
{
    a = a+1;
}
if (a <= 12)
{
    a = a+b+c;
}
```

i386 Assembly

Dump file: `test.s`

```
        jmp   .L2
.L3:
        addl   $1, -4(%ebp)
.L2:
        cmpl   $7, -4(%ebp)
        jle   .L3
        cmpl   $12, -4(%ebp)
        jg    .L6
        movl   -8(%ebp), %edx
        movl   -4(%ebp), %eax
        leal   (%edx,%eax), %eax
        addl   -12(%ebp), %eax
        movl   %eax, -4(%ebp)
.L6:
```

```
while (a <= 7)
{
    a = a+1;
}
if (a <= 12)
{
    a = a+b+c;
}
```

Part 6

Conclusions

Conclusions

- LaTeX + Pstricks + Beamer: Magic Potion for Making Presentations
- We have barely scratched the surface
- Initial learning seems difficult but the payoffs are immense
- Excellent guides and tutorials are available
- All sources and slides of this presentation will be soon uploaded on http://www.cse.iitb.ac.in/~uday/latex/

Last But Not the Least

Thank You!