

An Interactive Mathematical Handwriting Recognizer for the Pocket PC

Bo Wan

**Statistics Canada
Ottawa, Ontario**

Stephen Watt

**Ontario Research Center for
Computer Algebra (ORCCA)**

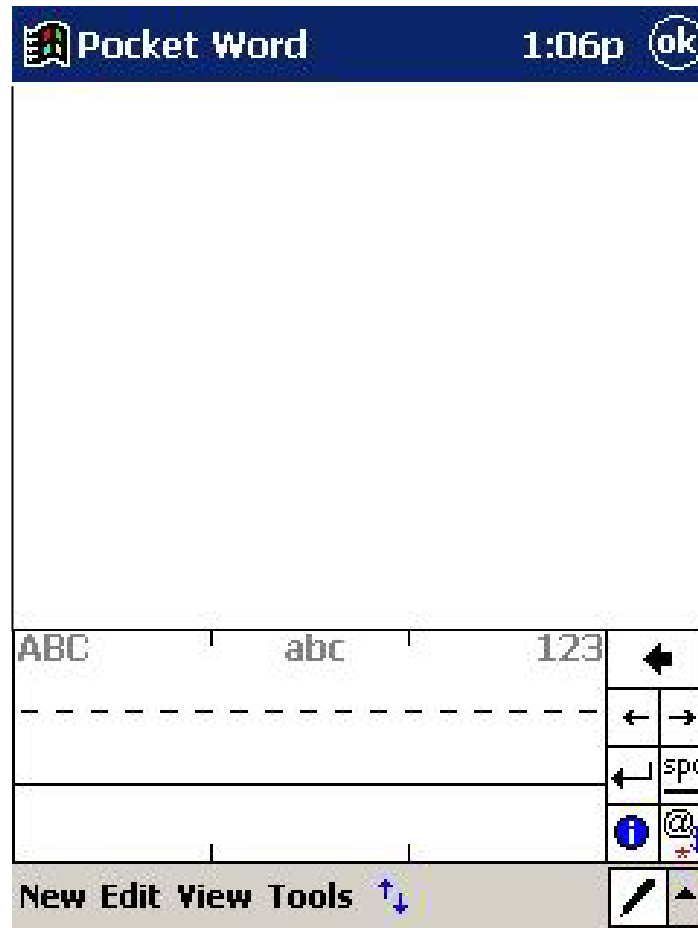
London, Ontario

**The Second MathML International Conference
June 2002, Chicago, Illinois**

Why Handwriting Math?

- ❖ Handwriting is the primary input method for PDAs
- ❖ Handwriting math is the preferred way for user to interact with computer algebra systems on PDAs
- ❖ Limitations of existing handwriting recognizer
 - Unable to recognize mathematical symbols
 - Only understands linear structures
 - Handwriting math is a cross between drawing and handwriting – non-linear structure

Windows CE Default Handwriting Recognizer on Pocket PC



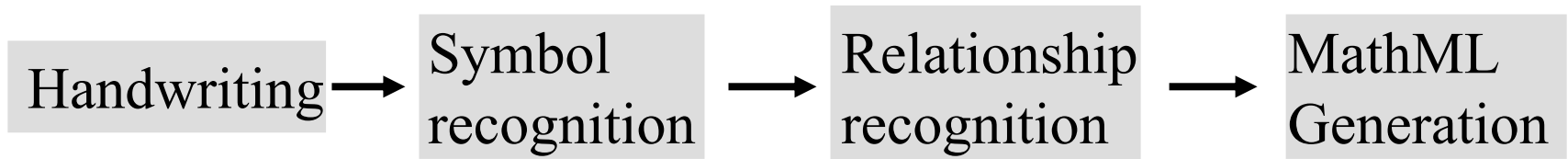
Program Outline

Developed an on-line mathematical
handwriting recognizer for pocket PC

- ❖ Recognizes handwritten symbols on-line
- ❖ Recognizes spatial relationship between symbols on-line
- ❖ Generates *presentation MathML* as output

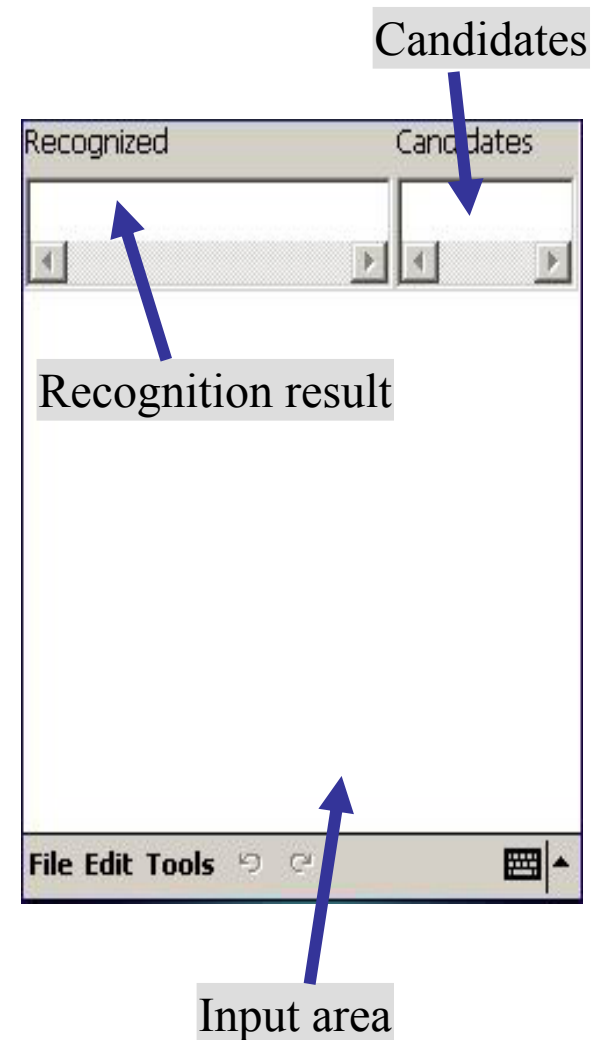
Structure of the Program

- ❖ Handwriting recognizer: Segment and identify individual handwritten symbols
- ❖ Structural analyzer: interpret the 2-D relationship between handwritten symbols
- ❖ Generate corresponding presentation MathML for the mathematical expression
- ❖ Data flow:



General Designs

- ❖ Development Tool: Microsoft embedded visual C++
- ❖ Support Unicode
- ❖ Allow both single-stroke and multi-stroke symbols



Handwriting Recognition - Review

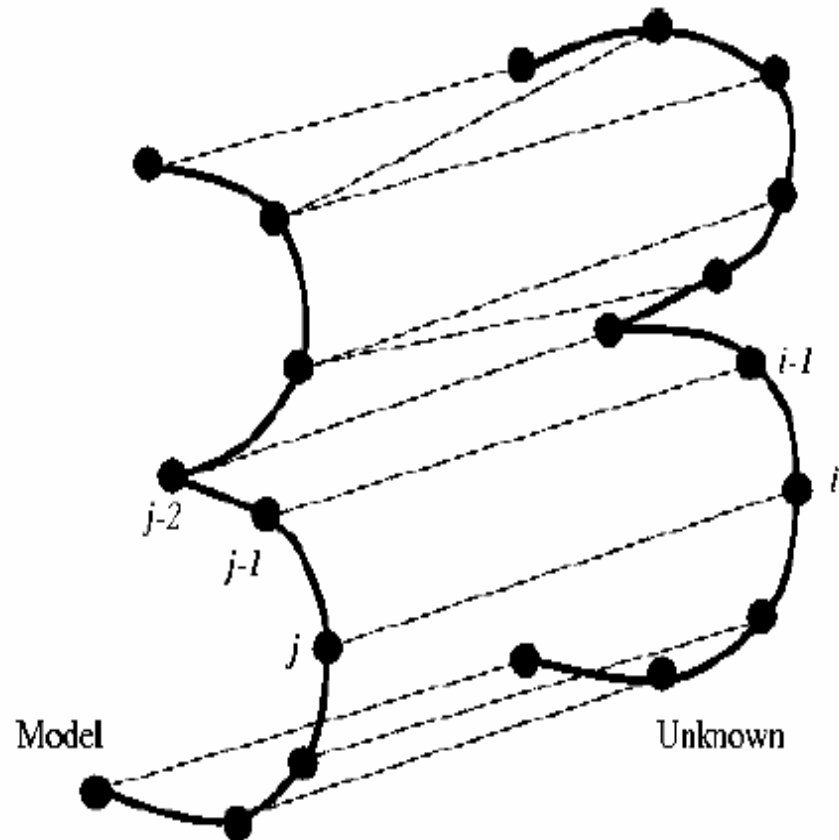
- ❖ Studied for nearly forty years
- ❖ Many open issues
- ❖ Two categories of recognition methods
 - Off-line recognition: *e.g.* OCR
 - On-line recognition
 - **Pattern recognition:** *e.g.* Neural Networks
 - **Statistical method:** *e.g.* Hidden Markov Model
 - **Structural and Syntactical Approaches**
 - **Elastic Matching**

The Handwriting Recognizer

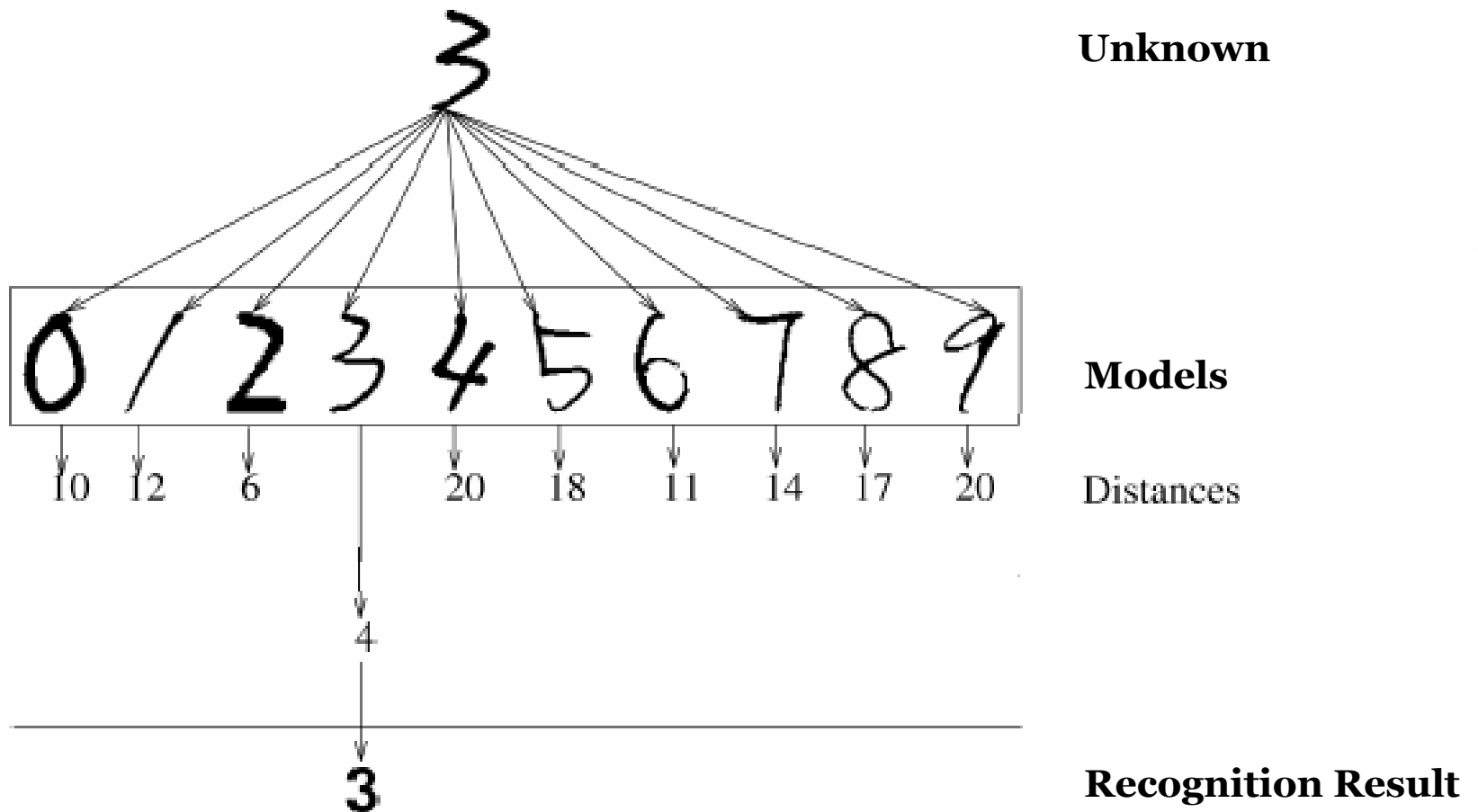
- ❖ Recognizes multi-stroke symbols
 - Distinguishes symbols by time and distance difference
- ❖ Uses *Elastic Matching* algorithm
 - Normalize data by preprocessing
 - size normalization
 - point distance normalization
 - smoothing
 - Recognizing unknown by *Elastic Matching*

Elastic Matching

- ❖ An application of dynamic programming
- ❖ Analog to String matching
- ❖ Robust, handles writing mistakes quite well



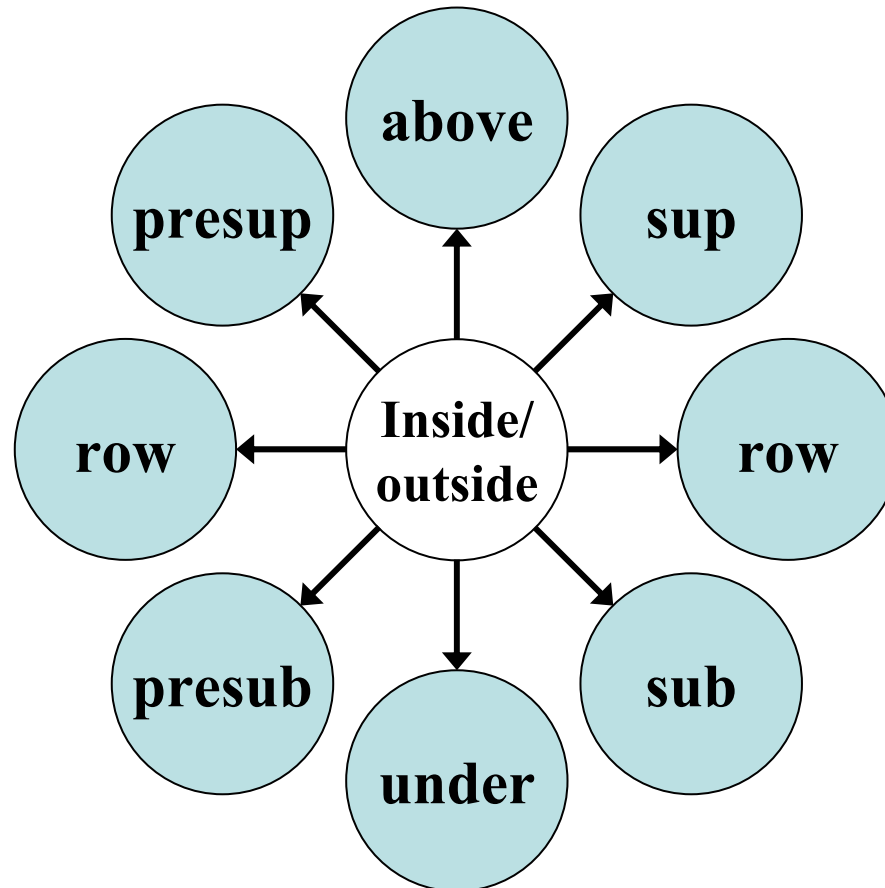
Recognition with *Elastic Matching*



Properties of Mathematical Expressions

- ❖ Two-dimensional hierarchical structure
- ❖ Spatial relationship implies mathematical meaning
- ❖ Complicated grouping rules
 - Basic symbols
 - Binding, fence symbols
 - Operator symbols
 - Context sensitive roles

Possible Spatial Relationships in Mathematics



Properties of Mathematical Expressions (2)

Basic symbols and their grouping:

❖ Digits

- 123 vs 12^3

❖ Roman letters

- \sin vs abc

❖ Other basic symbols

- Always as separate unit

Properties of Mathematical Expressions (3)

Binding, fence and operator symbols

❖ Binding symbols dominate their neighbors

- $\Sigma, \int, \sqrt{}, \text{ etc.} \longrightarrow \sum_{i=0}^k i, \sqrt{a+b}$

❖ Fence symbols change precedence

- $(,), [,], \text{ etc.} \longrightarrow a \times (b + c)$

❖ Operators

- Explicit and implicit operators

Properties of Mathematical Expressions (4)

❖ Operator symbols

- Explicit operators: $+$, $-$, \times , $-$, $/$, *etc.*
- Implicit operators:

- $2\underset{\uparrow}{\frac{3}{4}}$, $a\underset{\uparrow}{\frac{b}{c}}$, $\underset{\uparrow\uparrow}{abc}$, *etc.*

❖ Context sensitive roles

- $\int x\underline{dx}$ vs $ay + \underline{\underline{dx}}$

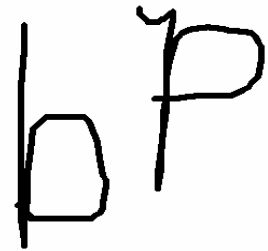
Structural Analysis Problems

- ❖ No clear separation between horizontal adjacency and superscript/subscript relations

$a_2 \rightarrow a_2 \rightarrow a_2 \rightarrow a_2 \rightarrow a_2 \rightarrow a^2 \rightarrow a^2 \rightarrow a^2$

- ❖ Ascenders and descenders

• b, d, h, p, q, g, y , etc. →

Handwritten examples of ascenders and descenders: 'b' and 'p'.

- ❖ Baseline must be determined globally

$x^y \rightarrow z x^y$

Structural Analysis Considerations

Generally used approaches are not suitable for pocket devices:

- ❖ Computationally intensive – need faster processor
- ❖ Require large memory and physical storage
- ❖ Pocket device only has limited resource
- ❖ Procedure coding is used

Structural Analysis Approaches- Overview

- ❖ Determine relationship of sub-expressions with bounding box information
 - Distance between symbols:
 - Centroid distance of bounding boxes
 - Direction between symbols or sub-expressions:
 - Centroid direction of bounding boxes
- ❖ Maintain relationships in a tree structure

Bounding Boxes

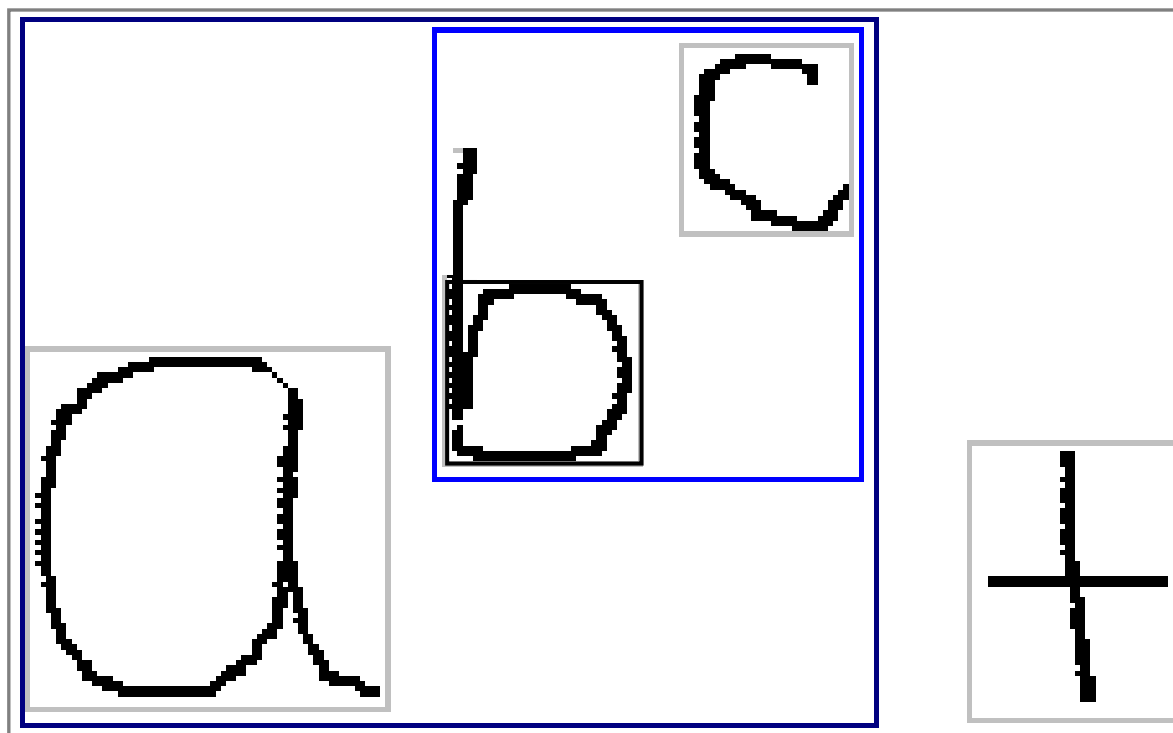
❖ Symbols:

- The minimum rectangle surrounding a symbol
- Exceptions: *Ascenders* and *Descenders*

❖ Sub-expressions:

- Union of the bounding boxes of involved symbols or sub-expressions

Bounding Boxes of Symbols and Sub-Expressions

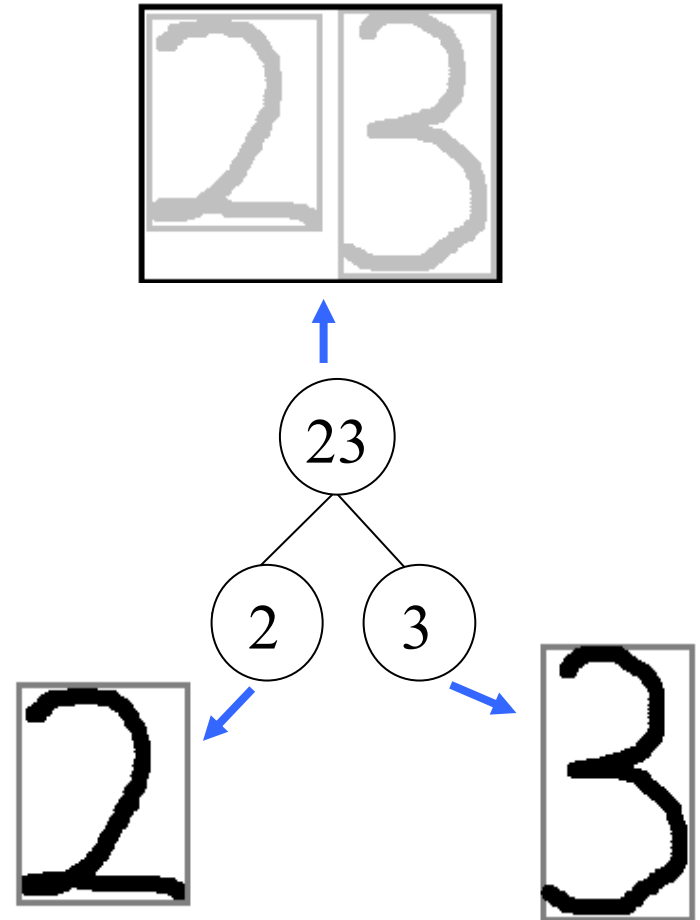


Structural Analysis Approaches - Steps

- ❖ Create a new node for each new symbol
- ❖ Locate the nearest neighbor (NN) node
- ❖ Find a proper grouping unit
 - Special cases: numbers, function names, *etc.*
 - General cases: Long vs Short range grouping
- ❖ Integrate the new node into expression tree

The Expression Tree

- ❖ *Label*
- ❖ *Scribble*
- ❖ *Bounding box*
- ❖ *Flag*
- ❖ *Children node list*



Short Range and Long Range Groupings

A symbol may not be in a grouping with its NN node

- Example 1: $a^{b+c} d$

- Example 2: $(a+b)^2$

- Example 3: $\frac{a}{b}$
 c

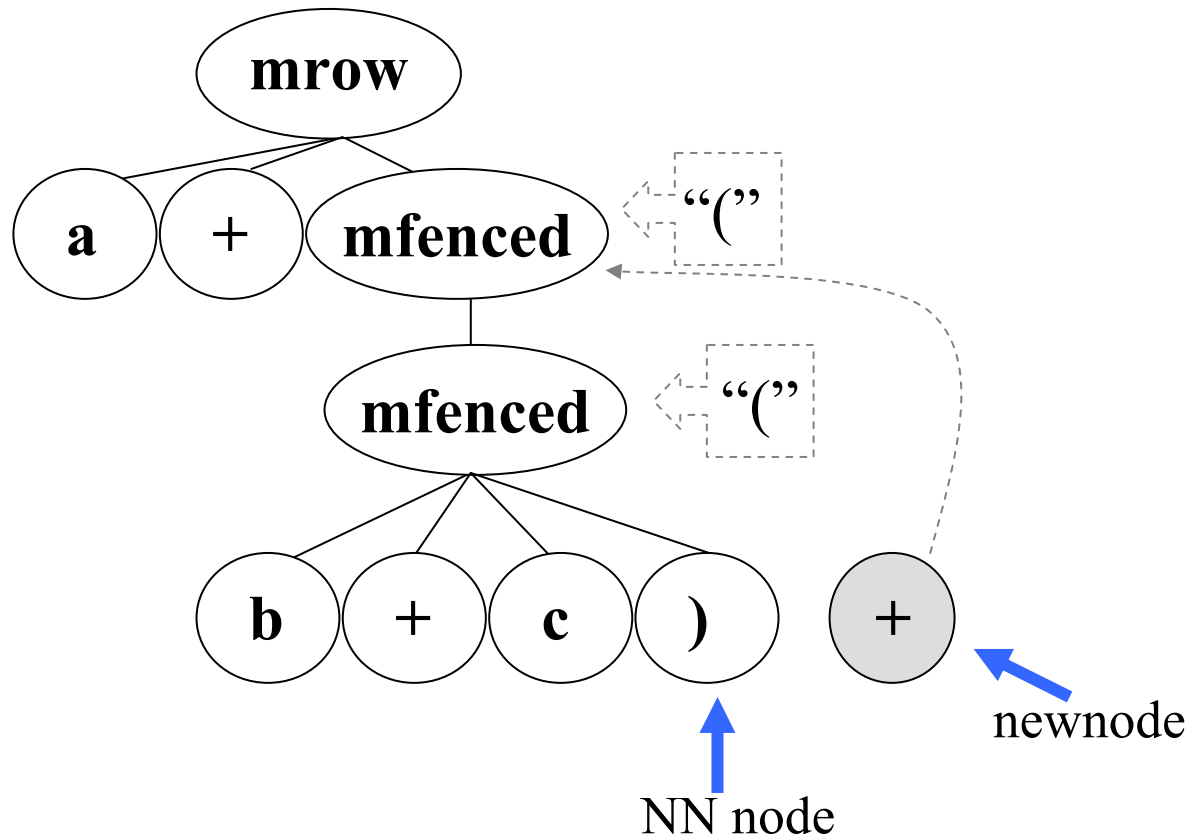
Locate the Proper Grouping

- ❖ Long range grouping dominates short range groupings
 - Row direction check with algorithm *RowParent*
 - Column direction check with algorithm *ColParent*
 - Superscript and Subscript direction check

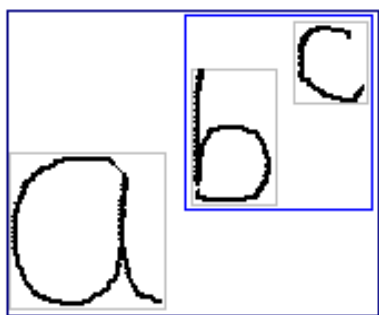
Example: Algorithm *RowParent*

```
Routine RowParent{
  newnode = the new node;
  parent = NN node;
  target = null;
  while(parent != null){
    if(newnode is in a row with parent){
      if(parent is a '(' without a matching ')')
        return parent;
      else
        target = parent;
    }
    parent = the parent node of parent;
  }
  return target;
```

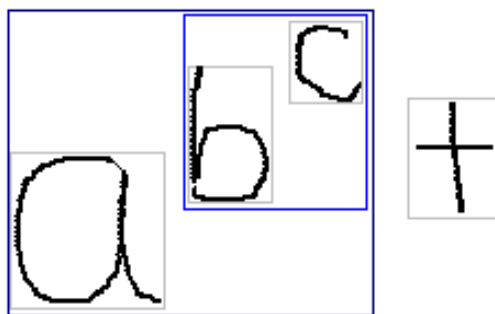
RowParent Example $a + ((b + c) +$



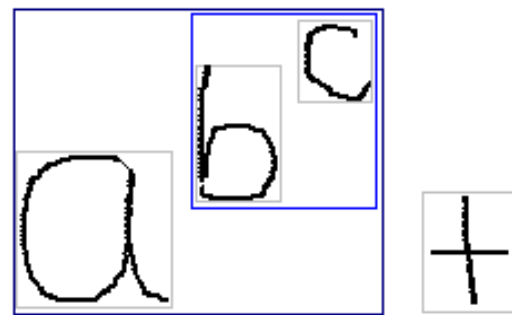
Problem with Bounding Box Operations



$$a^{b^c}$$

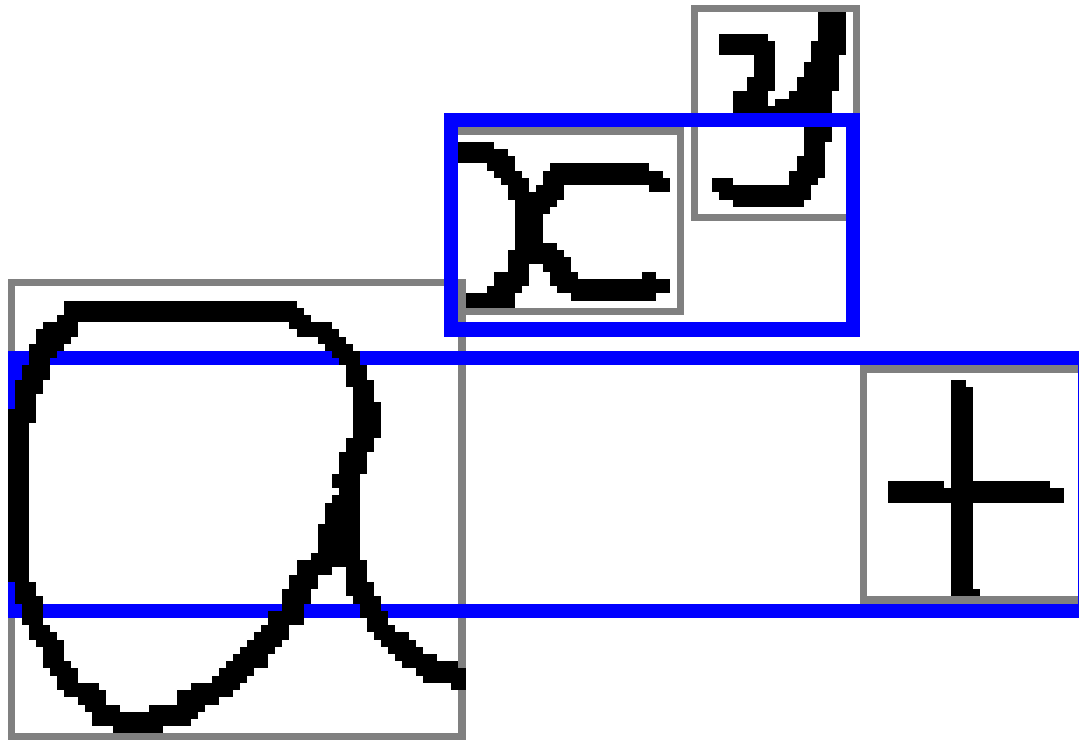


$$a^{b^c} +$$



$$a^{b^c} +$$

Conditional Updating of Bounding Boxes



Generate MathML

❖ Post process expression tree

- Split nodes when necessary: $ab \rightarrow a \times b$
- Merge nodes when necessary: a and $2 \rightarrow a2$
- Add missing implicit operators

❖ Generate MathML

- Preorder tree traversal

Open Issues and Future Work

❖ Symbol Recognition

- Speed is slow for *ElasticRecognizer*
- Relatively high symbol recognition errors
- Improve symbol recognition speed and recognition rate

❖ Structural Analysis

- Cannot recognize square roots and matrices
- Expression tree lacks flexibility
- Make expression tree compatible with DOM and satisfy different rendering needs

Related Work @ORCCA

- ❖ Off-line mathematical handwriting recognition, 2000
 - *Cross Pad* and *IBM Electric Ink*
 - By Arthur Louie
- ❖ Mathematical handwriting database, 2002
 - By Xiaojie Wu
- ❖ Mathematical handwriting recognition on *Microsoft* Tablet PC, 2002