

Fuzzy parsing and fuzzy branchings

Scott MacLean

ORCCA JLM: December 4, 2008

Context

In the pen-math project at UW we are building software to recognize handwritten math expressions.

We use two main tools for recognition:

1. relational context-free grammars for modeling mathematical syntax, and
2. fuzzy sets for modeling the uncertainty in the recognition problem.

To avoid re-writing expressions due to errors in recognition we allow many simultaneous semantic interpretations of the user's writing.

To display results to the user, these interpretations must be extracted from the parser's output and ranked in order of decreasing confidence.

Here, we

1. Formalize the uncertainty underlying our recognition strategy, and
2. Describe a practical algorithm for ranking semantic interpretations.

Fuzzy sets

A **fuzzy set** is a pair (X, μ) where X is a set and $\mu : X \rightarrow (0, 1)$ (called the **membership function**) gives the grade of membership in the fuzzy set for each $x \in X$.

We typically write \tilde{X} or \tilde{X}_μ rather than (X, μ) .

A **fuzzy binary relation on** (X, Y) is a fuzzy set $(X \times Y, \mu)$ for some μ .

We say $a \in \tilde{A}_\mu$ if $\mu(a) > 0$.

We say $\tilde{A}_\mu \subseteq \tilde{B}_\sigma$ if $\mu(a) \leq \sigma(a) \forall a \in \tilde{A}_\mu$.

We can extend the usual set operations:

- $\tilde{A}_\mu \cup \tilde{B}_\sigma = (A \cup B, \rho)$ (typically $\rho(x) = \max\{\mu(x), \sigma(x)\}$)
- $\tilde{A}_\mu \cap \tilde{B}_\sigma = (A \cap B, \rho)$ (typically $\rho(x) = \min\{\mu(x), \sigma(x)\}$)
- $\tilde{A}_\mu \times \tilde{B}_\sigma = (A \times B, \rho)$ (typically $\rho(x) = \min\{\mu(x), \sigma(x)\}$)

(Aside: we use $\rho(x) = \alpha(1 + \beta(\beta - \alpha))$ for \cap and \times , where $\alpha = \min\{\mu(x), \sigma(x)\}$, $\beta = \max\{\mu(x), \sigma(x)\}$.)

Note that we can view a crisp (ie. non-fuzzy) set S as a fuzzy set with membership function $\mu(x) = 1 \forall x \in S$.

Fuzzy relational context-free grammars

A **fuzzy r-CFG** is a tuple $G = (\Sigma, N, S, T, R, r_\Sigma, P)$ where

- Σ, N, S are as for “normal” context-free grammars,
- T is a set of observables (ie. possible objects comprising the input),
- R is a set of fuzzy binary relations on (\mathbb{T}, \mathbb{T}) ,
- \tilde{r}_Σ is a fuzzy binary relation on (T, Σ) , and
- P is a set of productions of the form $A_0 \rightarrow A_1 \tilde{r} A_2 \tilde{r} \cdots \tilde{r} A_n$ where $A_0 \in N; n > 0; A_i \in N \cup \Sigma, 0 < i \leq n$; and $\tilde{r} \in R$.

For simplicity, we will assume that G is a fuzzy r-CFG in the natural extension of Chomsky normal form; ie. every production in P is either of the form

$A \rightarrow a$ for some $a \in \Sigma$, or

$A \rightarrow B \tilde{r} C$ for some $B, C \in N, \tilde{r} \in R$.

The fuzzy set $\tilde{\mathbb{T}}$ referenced above is the set of all parses in G , recursively defined by:

$$\begin{aligned}\tilde{\mathbb{T}}_A &= (T \times \{a\}) \cap r_\Sigma \text{ for all } A \rightarrow a \in P \\ \tilde{\mathbb{T}}_A &= \left(\tilde{\mathbb{T}}_B \times \tilde{\mathbb{T}}_C \right) \cap \tilde{r} \text{ for all } A \rightarrow B \tilde{r} C \in P \\ \tilde{\mathbb{T}} &= \bigcup_{A \in N} \tilde{\mathbb{T}}_A\end{aligned}$$

Example

The following is a toy grammar which we will use later. This example shows the results of grammar normalization and the correspondence between derivation steps and elements of the set \tilde{T} .

Original grammar	Normalized grammar	Derivations with corresponding elements of \tilde{T}
$E \rightarrow T + E$ $E \rightarrow T$ $T \rightarrow \frac{E}{E}$ $T \rightarrow N$ $N \rightarrow 1 2 3$	$E \rightarrow TE_2$ $E_2 \rightarrow PE$ $T \rightarrow \frac{E}{T_2}$ $T_2 \rightarrow \frac{L}{E}$ $N \rightarrow 1 2 3$ $T \rightarrow 1 2 3$ $E \rightarrow 1 2 3$ $E \rightarrow \frac{E}{T_2}$ $P \rightarrow +$ $L \rightarrow -$	$N \rightarrow 1 \text{ with input } $ $(, 1)$ $T_2 \rightarrow \frac{L}{E} \text{ with input } \overline{3}$ $((_, _), (\overline{3}, 3))$ $T \rightarrow \frac{E}{T_2} \text{ with input } \frac{2}{\overline{3}}$ $((\overline{2}, 2),$ $((_, _), (\overline{3}, 3))$ $)$ $E \rightarrow TE_2 \text{ with input } \frac{ +2}{\overline{3}}$ $(, 1),$ $((+, +),$ $((\overline{2}, 2),$ $((_, _), (\overline{3}, 3))$ $))))$

Fuzzy parsing

Let $I \subseteq T$ be a finite set of input observables. We consider the restriction of $\tilde{\mathbb{T}}$ to observables in I , ie. the fuzzy set $\tilde{T}(I)$ of all parses in G on I .

Due to constraints on what constitutes a valid parse (not described here), there exists a set $\Gamma \subseteq 2^I$ of all sets of input elements on which parses exist such that $|\Gamma| = \mathcal{O}(|I|^3)$.

The fuzzy set $\tilde{T}(A, X)$ of all parses in G derived from $A \in N$ using exactly the input elements in $X \in \Gamma$ is characterized by the productions having LHS A , as follows:

Let $p \in P$ have LHS A . If p is of the form $A \rightarrow a$, then let $T_p(A, \{t\}) = (I \times \{a\}) \cap \tilde{r}_\Sigma$.

Otherwise p is of the form $A \rightarrow B \tilde{r} C$. Let

$$\tilde{T}_p(A, X) = \bigcup_{X_B, X_C \in \Gamma; X_B \cup X_C = X; X_B \cap X_C = \phi} \left(\left(\tilde{T}(B, X_B) \times \tilde{T}(C, X_C) \right) \cap \tilde{r} \right)$$

Now

$$\tilde{T}(A, X) = \bigcup_{p \in P \text{ with LHS } A} \tilde{T}_p(A, X)$$

From these sets, we can construct $\tilde{T}(I) = \bigcup_{A \in N, X \in \Gamma} \tilde{T}(A, X)$.

We can describe our interpretation extraction problem as outputting members of the $\tilde{T}(X, A)$ fuzzy sets in order of decreasing grade of membership.

Fuzzy parse graphs

A **split in** Z for any set Z is a pair $(x, (y, z))$ where $x, y, z \in Z$.

A **fuzzy parse graph of** G **on** I is a pair $M = (V, \tilde{E}_\mu)$ where $V = (N \times \Gamma) \cup I$ and \tilde{E} is a fuzzy set of edges and splits in V .

$x, y \in V$ are **adjacent in** M if the edge (x, y) or one of the splits $(x, (y, \cdot)), (x, (\cdot, y))$ is in \tilde{E} .

Our parser outputs a fuzzy parse graph with \tilde{E} constructed such that

1. An edge $((A, \{t\}), t)$ is in \tilde{E} iff there exists $a \in \Sigma$ such that $(t, a) \in \tilde{T}(A, \{t\})$, and
2. A split $((A, X_B \cup X_C), ((B, X_B), (C, X_C)))$ is in \tilde{E} iff there exists $(x, y) \in \tilde{T}(A, X_B \cup X_C)$ such that $x \in \tilde{T}(B, X_B), y \in \tilde{T}(C, X_C)$.

There is a direct correspondence between edges and derivation steps of the form $A \rightarrow a$.

Similarly between splits and steps of the form $A \rightarrow B \tilde{r} C$.

A **branching in** M **from** $v \in V$ **to** $W \subseteq V$ is a subset $\tilde{F} \subseteq \tilde{E}$ such that there exists exactly one path from v to each $w \in W$ in the graph (V, \tilde{F}) .

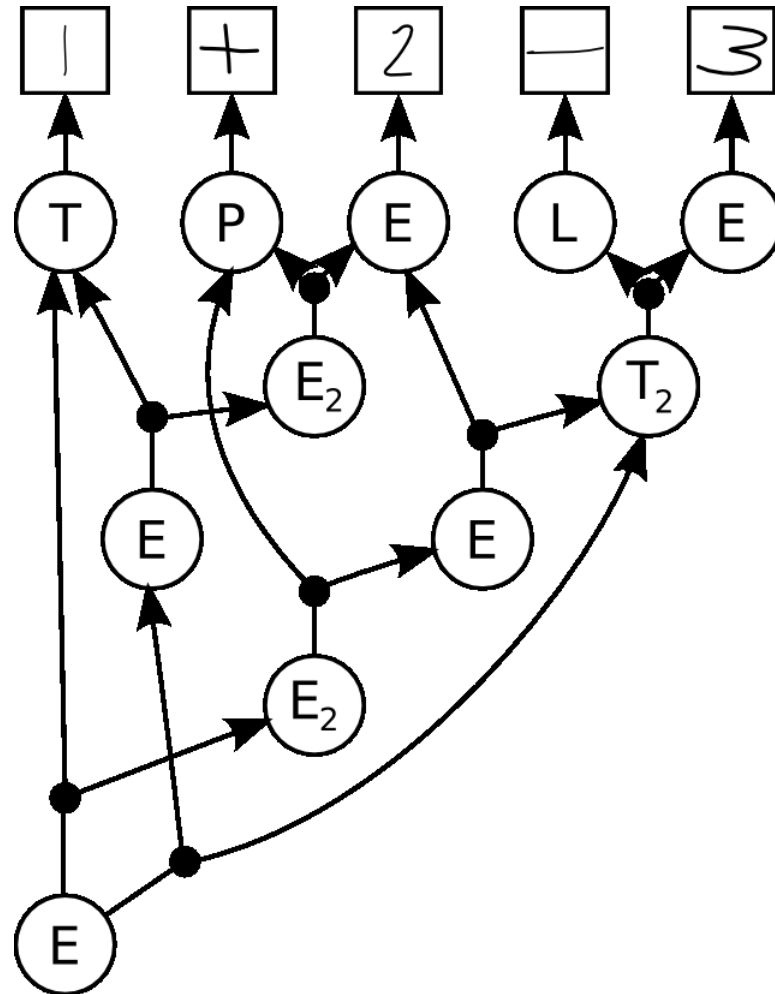
There is a bijection between the set of branchings in M from (A, X) to X and the set $\tilde{T}(A, X)$. A fuzzy parse graph is a compact representation of $\tilde{T}(I)$.

Now our problem is to extract ranked branchings from the parse graph.

Example

Consider parsing a small input using the grammar from the previous example. On the left is the input; on the right is a partial parse graph. The graph captures two interpretations, namely $1 + \frac{2}{3}$ and $\frac{1+2}{3}$.

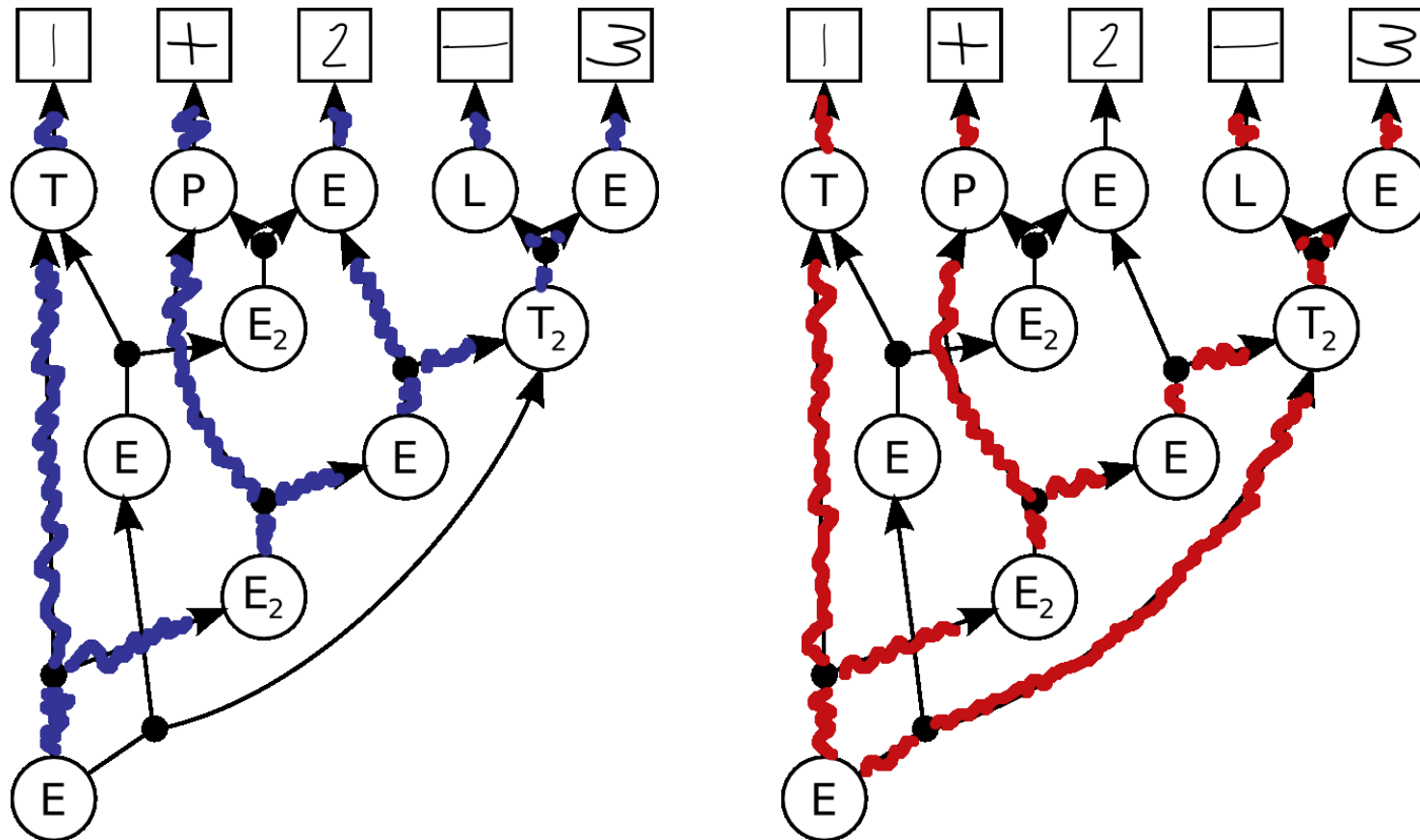
1 + 2
3



Example (cont'd)

On the left is a branching from one of the E nodes to all the terminals, corresponding to the last parse given in the previous example.

The selected elements on the right do not form such a branching because not all terminals are reachable, there are multiple paths to some terminals, and only half of some splits are selected.



Extracting ranked branchings (1)

For a function ω , let $\max_{\omega} S = \operatorname{argmax}_{s \in S} \omega(s)$.

For $A \in N, X \in \Gamma$, define $C(A, X) = \left\{ (x, y) : ((A, X), (x, y)) \in \tilde{E} \right\}$; ie. the pairs of nodes adjacent to (A, X) by a split (“children of (A, X) ”).

Let \tilde{B}_{ω} be a fuzzy set of branchings in M . Denote the branching from (A, X) to X with k th largest membership grade in \tilde{B} by $[k](A, X)$.

If $A \rightarrow a \in P$, then $[0](A, \{t\}) = \{((A, \{t\}), t)\}$ for all t such that $(t, a) \in \tilde{T}(A, X)$.

If $A \rightarrow B \tilde{r} C \in P$, then $[0](A, X) = \max_{\omega} \{[0]x \cup [0]y : (x, y) \in C(A, X)\}$.

So we have found the best branching. How can we find the next-best?

Suppose $[0](A, X) = [0]x^* \cup [0]y^*$.

Then

$$[1](A, X) = \max_{\omega} (\{[0]x^* \cup [1]y^*, [1]x^* \cup [0]y^*\} \cup \{[0]x \cup [0]y : (x, y) \in C(A, X) \setminus \{(x^*, y^*)\}\})$$

Generally, we can say

$$[k](A, X) = \max_{\omega} \{[i]x \cup [j]y : (x, y) \in C(A, X), (i, j) \in J\}$$

for some set of index pairs J varying with k, A, X .

Extracting ranked branchings (2)

Our approach is to maintain for each $(A, X) \in V$ a list of ranked branchings $[k](A, X), 0 \leq k < n$, a set K of known but unranked branchings from (A, X) to X , and a set $J(x, y)$ of valid index pairs to consider when finding $[n](A, X)$ for each $(x, y) \in C(A, X)$.

Initialization:

$$K \leftarrow \{[0]x \cup [0]y : (x, y) \in C(A, X)\}$$

$$[0](A, X) \leftarrow \max_{\omega} K$$

$$\text{Set } (x^*, y^*) \text{ such that } [0](A, X) = [0]x^* \cup [0]y^*$$

$$K \leftarrow K \setminus \{[0](A, X)\}$$

$$J(x, y) \leftarrow \{(0, 0)\} \quad \forall (x, y) \in C(A, X)$$

$$J(x^*, y^*) \leftarrow \{(1, 0), (0, 1)\}$$

To compute the n th-ranked branching, given that all higher-ranked branchings are known:

$$K \leftarrow K \cup \bigcup_{(x,y) \in C(A,X)} \{[i]x \cup [j]y : (i, j) \in J(x, y)\}$$

$$[n](A, X) \leftarrow \max_{\omega} K$$

$$\text{Set } (i^*, j^*, x^*, y^*) \text{ such that } [n](A, X) = [i^*]x^* \cup [j^*]y^*$$

$$K \leftarrow K \setminus \{[n](A, X)\}$$

$$J \leftarrow (J \setminus \{(i^*, j^*)\}) \cup \{(i^* + 1, j^*), (i^*, j^* + 1)\}$$

Comments

To obtain a parse in G on I , invoke the algorithm on (S, I) and then recurse as required on other nodes.

- In this way, the algorithm is only invoked for those nodes which may actually be part of a branching.

If K is implemented as a heap and the $J(\cdot, \cdot)$ as arrays, then the only non-constant operation is updating K .

In general, at most $\sum_{(x,y) \in C(A,X)} |J(x,y)|$ elements are added to K during the extraction of a branching.

Unfortunately the theoretical bound on $|C(A, X)|$ is $|\Gamma| = \mathcal{O}(|I|^3)$. But in practice it is very small.