**Background**

Colorless green ideas sleep furiously

By Chomsky

Looks incorrect - Makes no sense

But something looks right too - has perfectly right “syntax” – ordering of components in phrases and sentences – perfec

Like this sentence: two adjectives followed by a noun.preced verb and finally hs adverb after verb

Even though no semantic meaning, has syntax.

This is the problem being solved

People have been trying to represent the syntax of languages correctly since forever.

Dependency structures represent dependencies between words

**Phase structure**

Lexical categories(we popularly know it as parts of speech)

Noun

Adjective

Verb

Adverb

Pronoun

Determiner – articles, many, one and so on, his, hers

Adposition – includes preposition, pointing to a direction or location or something – includes postposition (like Japanese)

These are very important to define syntax rules

Phrase structure rules:

S -> np vp

Np -> (det) (adj) N (PP)

PP -> p np

Vp -> V (NP) (PP) (Adv)

Adv can be before V

Phrase structure rules aren’t a global thing. We’ve just chosen one set

How do we use hese different components to form a whole set or a sentence.

We will use smething called a tree diagram – top down

S – sentence

He left quickly

The big man left quickly

The man hit the ball

The man hit the ball with the bat

The student wrote his thesis on acid

**Issues**

It might seem like building grammars is so much more easier and it’s a more global form and making of dependency trees is so much more slow and cumbersome. But grammars have a lot of disadvantages

Reusability of grammars. Everyone does it differently. Have different ideas (dependency trre are more reusable). Dependency trees are reusable. Can build on top of this – like pos taggers and stuff like that.

Better way to represent all kinds f things that might happen in the language. Cfgs cant

Can use for evaluating tools by makignt reebanks

Frequency and distribution information

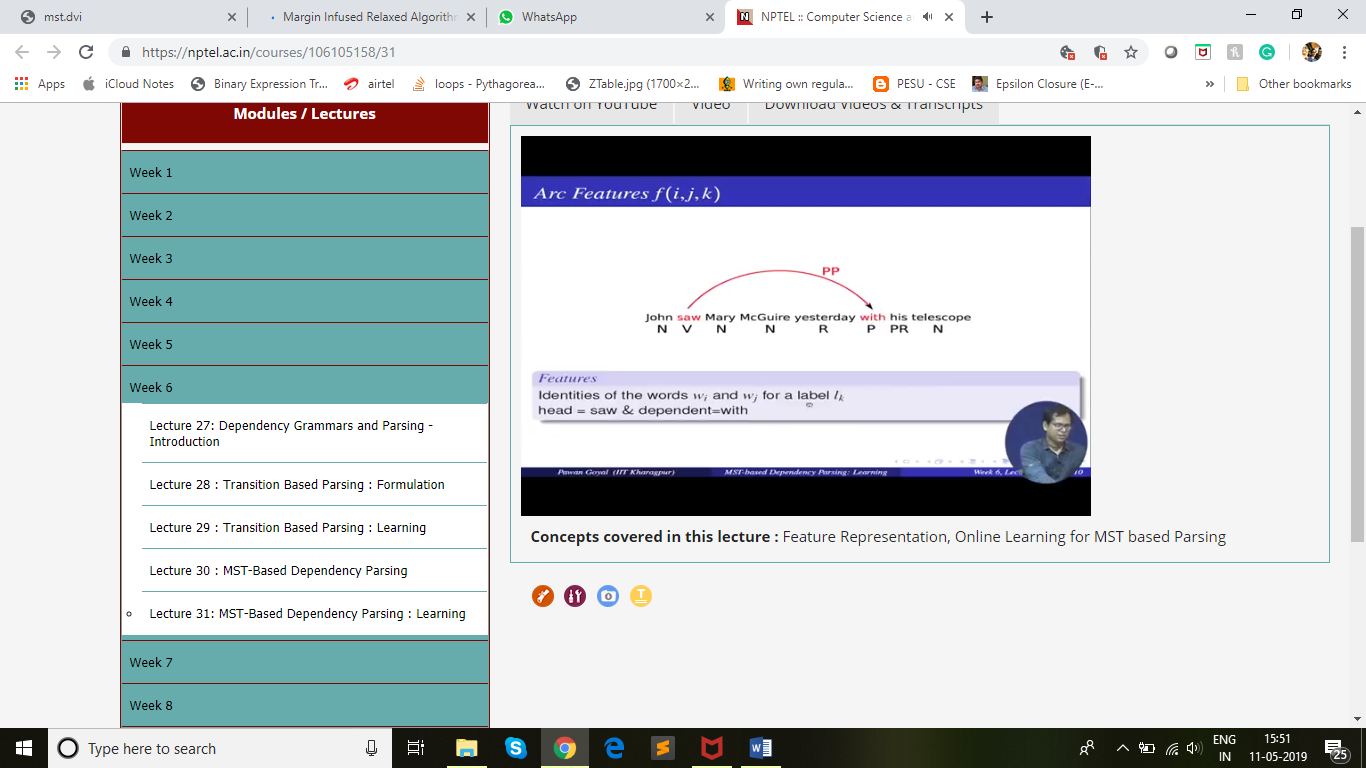
**MIRA**

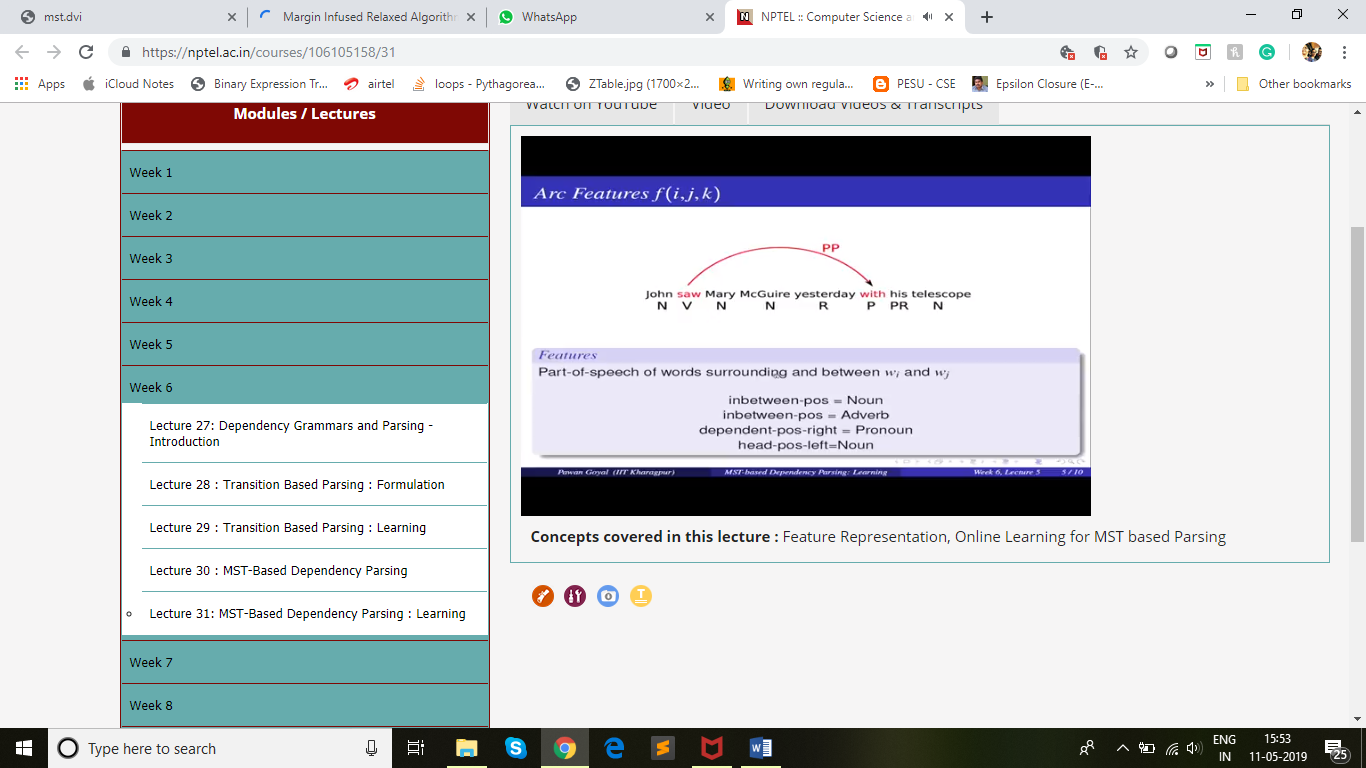
In maximum spanning tree we covered that given some weights, we can get the maximum spanning tree and hence the dependency tree. But how do we get those weights?

Once we get the weights, using these weights and chiu lu we are done. So how do we construct these edge weights?

Edge weights: so first we get a feature vector over nodes and labels. We define feature and cal calculate this vector. Its just an eq that can be formulated. But each feature will also have a weight defined.

And this weight is difficult to find. Need learning for that. What weights will help us get the best spanning tree.





In between part of speech

No of words in between

What occurs before

Edge weight is feature vector \* weight vector

Training set T = {x\_t, y\_t} ^T. sentence and its correct dependency trees

Dt(x) are possible dependency trees

New weight vector is is w(i+1). Current weight vector is w(i)

Minimize ||w(i+1) – w(i)||

Subjected to the constraint:

S(xt,yt) – s(xt – yt’) >= L(yt, yt’) for all yt’ belonging to dt(x).

L = loss is the number of words with incorrect parents

Experiment:

They use the Czech POS Tagging and test on Czeck prague dependency treebank

Have additional tags like gender, case and stuff also

Since czeck has more flexible word order, a lot of non projective dependencies exist

Results:

Metrics – accuracy (correctly identified their parent in the tree)

Complete(no. of sentences in which resulting tree was completely correct)

If we take a dataset that has atleast one non projective dependency, then the impact is even more

Chu Liu is n^2 while Eisner is n^3

Factored MIRA gives better accuracy than single-best MIRA

It might seem like an overkill for pure projective langauges, and inefficien but it doesn’t matter because the weights would adapt such that non projective trees are ignored

Even though chu liu has better efficiency it gives worse results than eisner

Summary:

Dependency trees equivalence to maximum spanning tree

Parsing – Eisner’s or Chu Lius and can use online learning after that

Unlike previous models, non projective is simpler here