Millstream Systems and Graph Transformation for Complex Linguistic Models

Frank Drewes

Based on joint work with S. Bensch, H. Björklund, H. Jürgensen, and B.v.d. Merwe

DCFS 2013

Introduction

Introduction

Analysis of natural language has to deal with numerous aspects of language, most notably

- morphology
- syntax
- semantics
- pragmatics

The aspects, called language modules, are interlinked by interfaces.

This talk is about Millstream systems

Millstream systems are our attempt to formalize this situation for algorithmic analysis and implementation.

There is a mutual dependency between the language modules (but not a hierarchy).

Our view:

- Each language module defines its own models.
- Such a model should be a set of trees (or maybe DAGs).
- The analysis of a sentence combines k trees, one from each module, by linking them together in a specified way.
- Conditions for placing the links should be formulated in logic.

This reduces the descriptional complexity of the language model as a whole by modularization.

Remainder of this talk:

1. Millstream systems in (slightly) greater detail
2. An example
3. The completion problem
4. Readers for incremental sentence analysis
5. Some final remarks

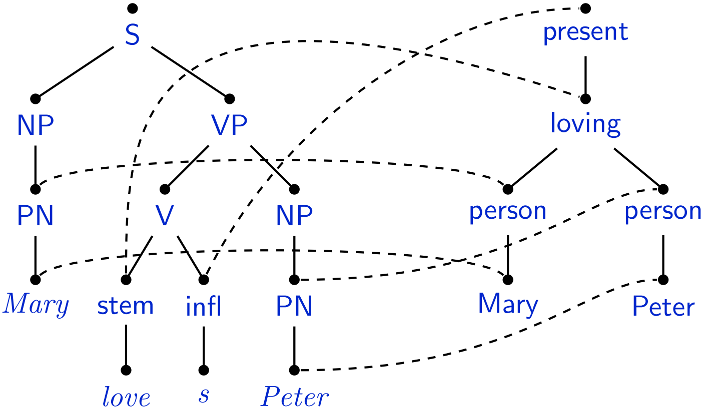
Millstream Systems

Suppose we want to consider k language modules (e.g., syntax and semantics, where $k = 2$).

The analysis of a sentence consists of

- k ranked, ordered and labelled trees and
- a number of relations that represent links between (some of) the nodes of these trees.

This yields a graph or logical structure which I will call an analysis.



In general, there can be links of several types and arities.

⇒ We use ranked alphabets of tree symbols and link symbols.

Components of a Millstream system:

- a finite number of modules $M_1, \dots, M_k$, each specifying a tree language $L(M_i)$ over tree symbols, and
- an interface, which is a set Φ of logical formulas.

A (well-formed) analysis consists of

- k trees $t_1 \in L(M_1), \dots, t_k \in L(M_k)$ and
- additional links over labelled with link symbols

such that Φ is satisfied.

Millstream Systems

To specify a class of Millstream systems, we have to say

- from which class C the modules are taken and
- which logic Λ is used.

Notation: $MS(C, \mathit{\Lambda})$, e.g., $MS(REG, FO)$ or $MS(REG, MSO)$.

- REG: regular tree grammars
- FO: first order
- MSO: monadic second order

An Example
(Concerning Descriptive Complexity)

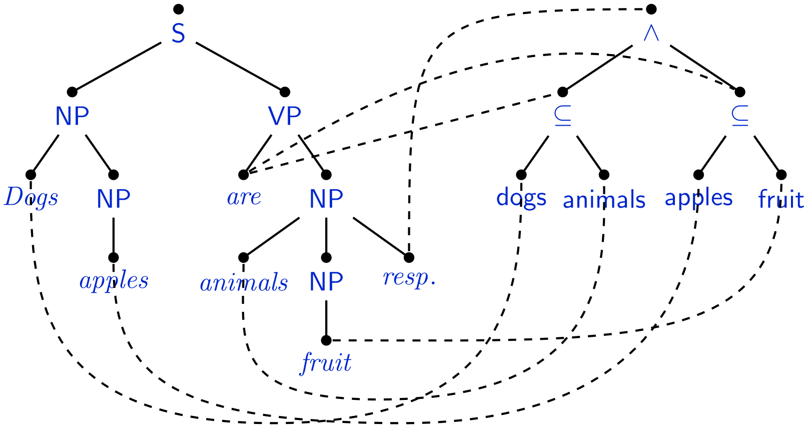
An Example

"Dogs, apples, [. . .] and potatoes
are animals, fruit, [. . .] and vegetable, resp."

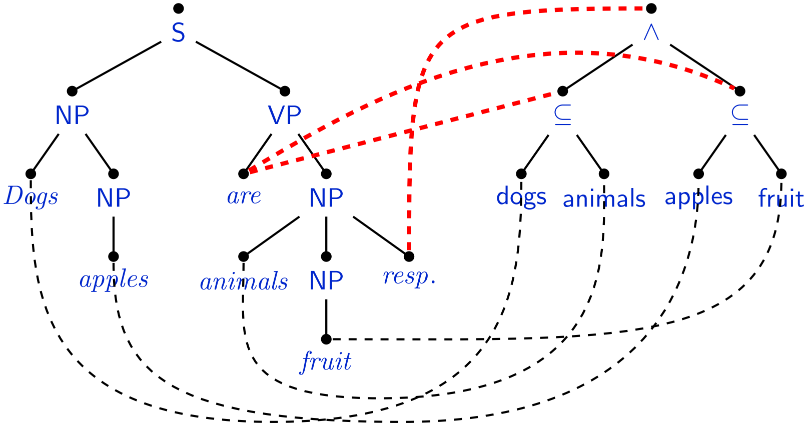
exhibits a structure similar to $w\$w$, but we can capture it using $MS(REG, FO)$.

An Example

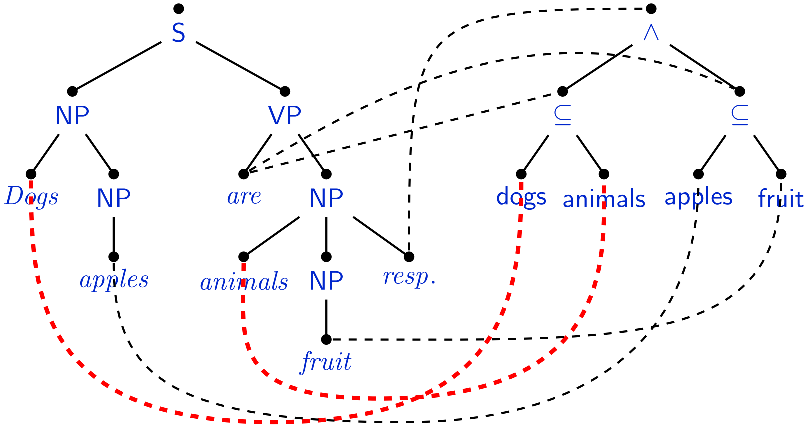
"Dogs and apples are animals and fruit, respectively."



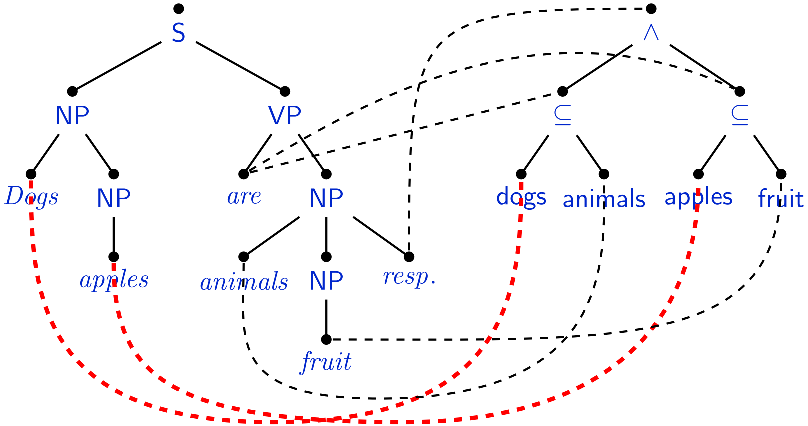
An Example



The node *resp.* is linked to a right-comb of $\wedge$ nodes, and the corresponding *are* node is linked to the $\wedge$ nodes' $\subseteq$ children.

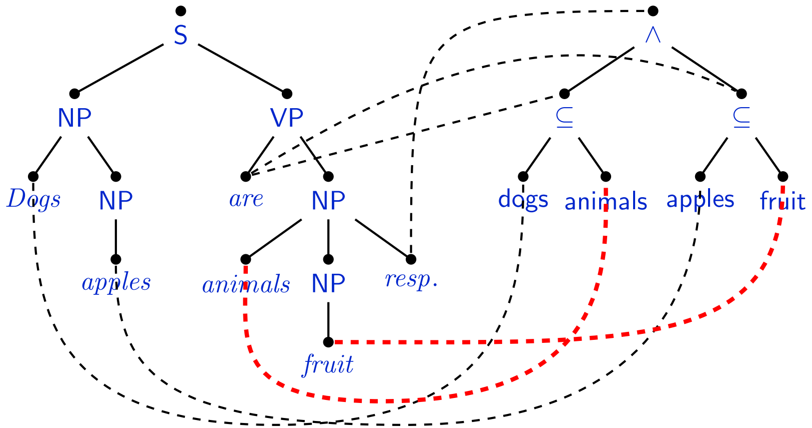


The children of the left-most $\subseteq$ are linked (only) to the left-most children of the first and second NP-combs, resp.

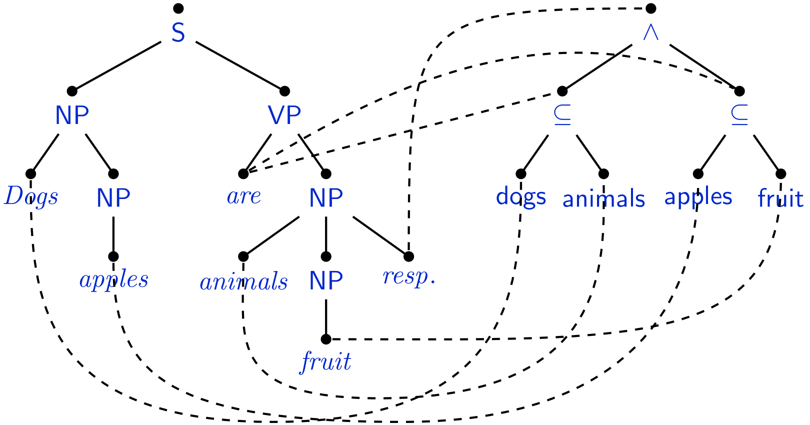


If the 1st child of an NP is linked to a 1st (2nd) child of a $\subseteq$, then the 1st child of the preceding NP is linked to the 1st (2nd, resp.) child of the preceding $\subseteq$.

An Example



If the 1st child of an NP is linked to a 1st (2nd) child of a $\subseteq$, then the 1st child of the preceding NP is linked to the 1st (2nd, resp.) child of the preceding $\subseteq$.



Note: This could even be done with the first tree alone. The links give us a certain “second-order power”.

The Completion Problem

The Completion Problem

We consider a (fixed) Millstream system MS with k modules and a set $K \subseteq [k] = \{1, \ldots, k\}$.

The K-completion problem

Instance: Trees κ_i for $i \in K$.
Question: Does MS admit an analysis whose trees $t_1, \ldots, t_k$ are such that $t_i = \kappa_i$ for all $i \in K$?

Some particular cases:

- $K = \emptyset$ is the emptiness problem.
- $K = [k]$ asks whether there exist suitable links.
- Sentence understanding ($K = \{\text{semantic module}\}$)
- Sentence generation ($K = \{\text{syntactic module}\}$)

UMEÅ UNIVERSITET

The Completion Problem

Some results:

- Emptiness (the $\emptyset$-completion problem) is undecidable even for regular FO Millstream systems. Reduction of PCP.
- The $[k]$-completion problem is trivially decidable by enumeration if the logic is decidable.
- The $[k]$-completion problem is decidable for regular MSO Millstream systems without direct links between modules in $[k] \setminus K$.
- The K-completion problem is also decidable for so-called nested regular MSO Millstream systems.

Complexity is an open question.

Readers

Readers

How can we construct the analysis of a given sentence?

First approach: parse syntactically and use a completion procedure.

Can be useful, but has disadvantages:

- expected to be inefficient in most cases
- gives priority to syntax (against the "equal rights" spirit of Millstream systems).

Readers

How can we construct the analysis of a given sentence?

Second approach: “readers” consider all modules in parallel.

Inspired by the way in which humans analyse a sentence:

- from left to right
- taking all aspects (= modules) into account in parallel.

Readers

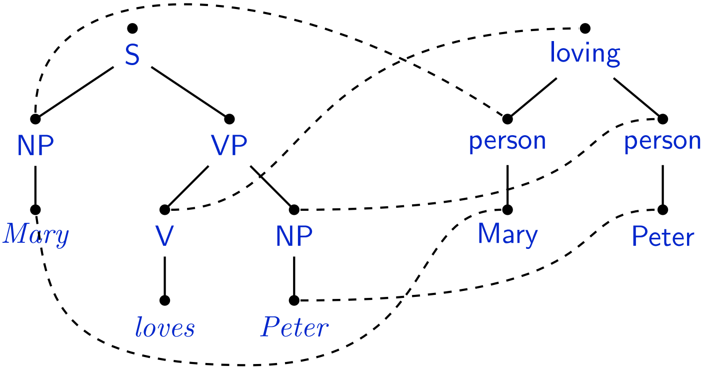
Our suggestion: use graph transformation (graph grammars)

- A sentence is "read" from left to right (as humans do).
- Every word w has a set of graph transformation rules $\mathcal{R}(w)$ associated with it.
- Reading sentence $w_1 \cdots w_n$ means to make a derivation

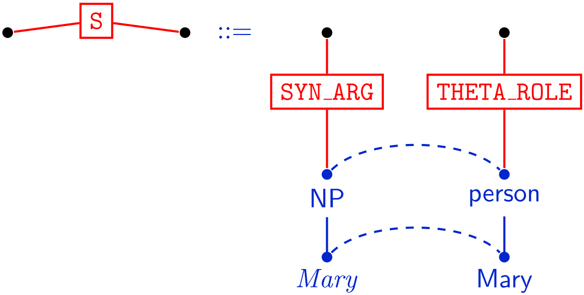
$$START \underset{\mathcal{R}(w_1)}{\Longrightarrow} G_1 \underset{\mathcal{R}(w_2)}{\Longrightarrow} G_2 \cdots \underset{\mathcal{R}(w_n)}{\Longrightarrow} G_n.$$

- The result G_n is the analysis of the sentence (if it is terminal).
- We call these graph transformation systems readers.

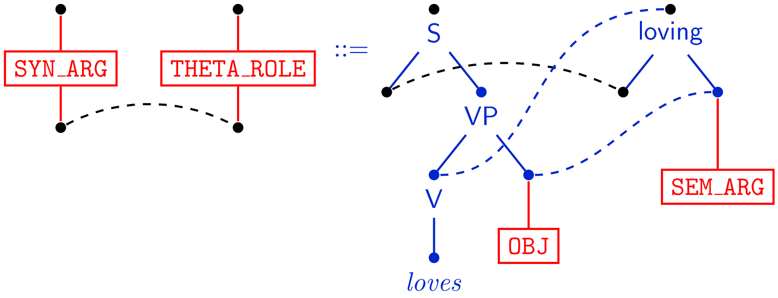
A (very simple) example: "Mary loves Peter."



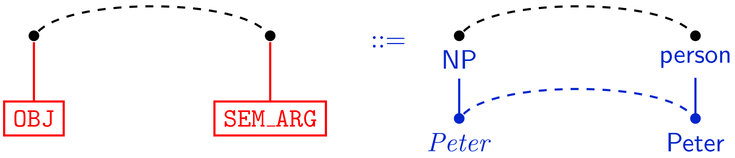
Rule for reading *Mary* at the beginning of a sentence.



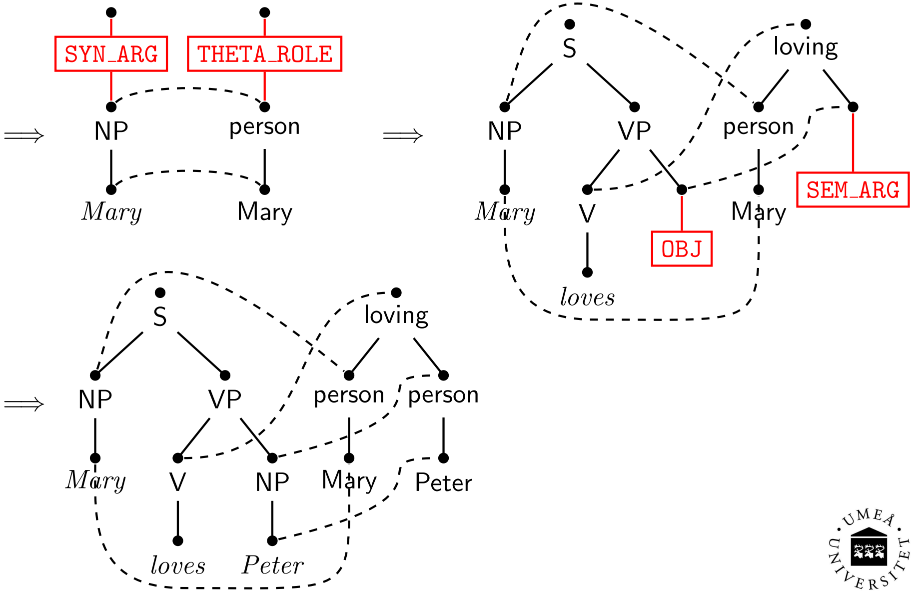
Rule for reading *loves* when a syntactic argument and a theta role are "looking for their verb".



Rule for reading *Peter* when an object and a semantic argument are sought.



Readers


⟹
SYN_ARG
THETA_ROLE
NP
person
Mary
Mary
⟹
S
loving
NP
VP
person
Mary
V
Mary
SEM_ARG
OBJ
loves
⟹
S
loving
NP
VP
person
person
Mary
V
NP
Mary
Peter
loves
Peter

The first nontrivial result:

Theorem (correctness of readers)

For $MS \in MS(REG, MSO)$ and nonterminal bounded readers R, it is decidable whether all output analyses of R are well formed according to MS.

Proof idea: From R, construct a hyperedge replacement grammar. Then use the known fact that MSO properties of hyperedge replacement languages are decidable.

Note: You would not want to implement this algorithm, much less execute it.

Final Remarks

Final Remarks

- Not much is known about Millstream systems and readers yet.
- Both linguistic application and theoretical investigation are of interest.
- Descriptive power and complexity of $MS(REG, FO)$ seems to be an interesting theoretical subject.
- What is the relation between the size of a Millstream system and the size of a corresponding reader?
- Not only via readers, graph transformation may contribute a lot to computational linguistics.
- Together with Senjuti Kundu, first steps are being made towards implementing readers on top of GrGen.net.

If you are interested, please contribute and collaborate!

Thank you very much
for listening!