M-monoid parsing and reduct generation

Richard Mörbitz

15th December 2017

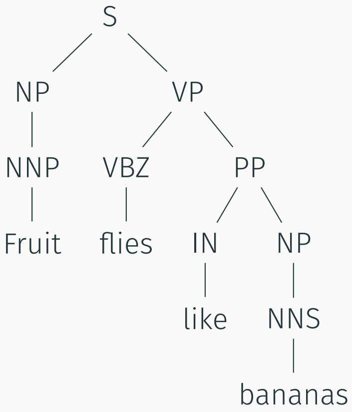
Parsing?

Given $G = (N, \Sigma, S, R)$

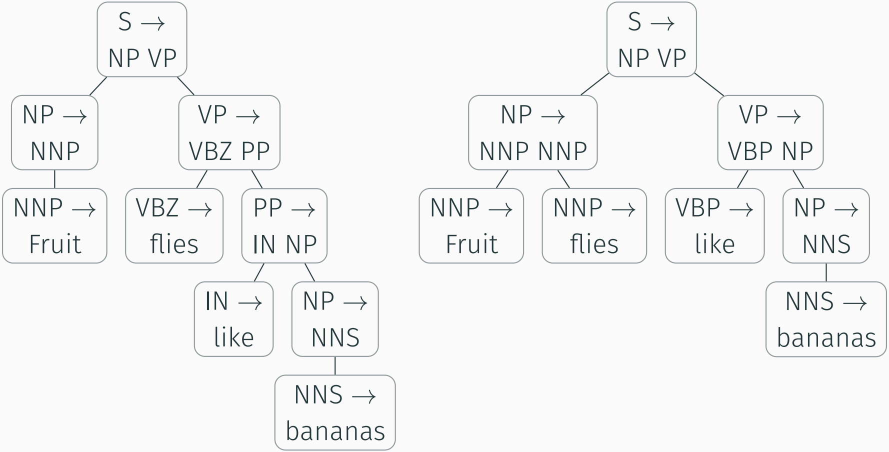
- $N = \{\text{S}, \text{NP}, \text{VP}, \text{NNP}, \dots\}$
- $\Sigma = \{\text{Fruit}, \text{bananas}, \dots\}$
- $R = \{\text{S} \rightarrow \text{NP VP}, \text{NNP} \rightarrow \text{Fruit}, \dots\}$

To parse $e = \text{Fruit flies like bananas.}$

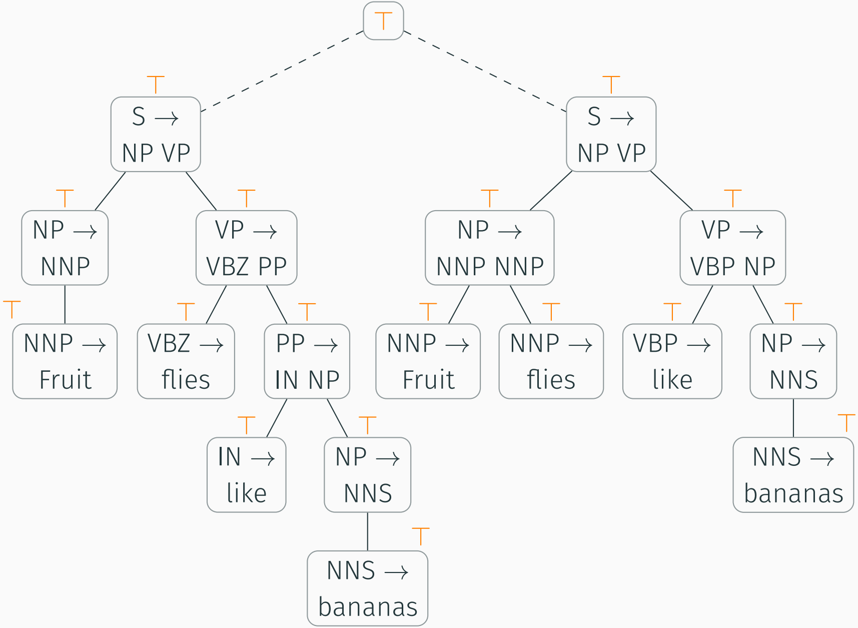
Parse trees


S
NP
VP
NNP
VBZ
PP
Fruit
flies
IN
NP
like
NNS
bananas
S
NP
VP
NNP
NNP
VBP
NP
Fruit
flies
like
NNS
bananas

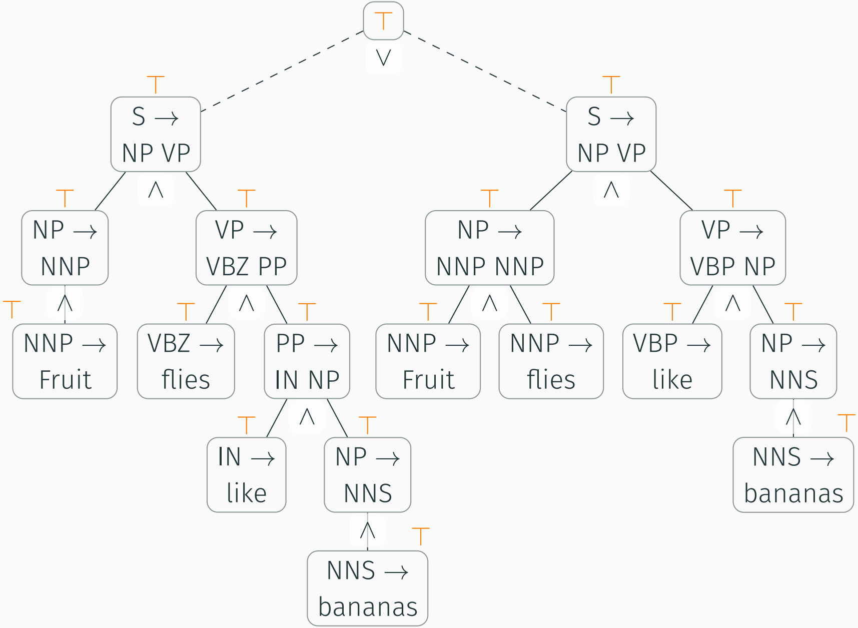
Abstract syntax trees



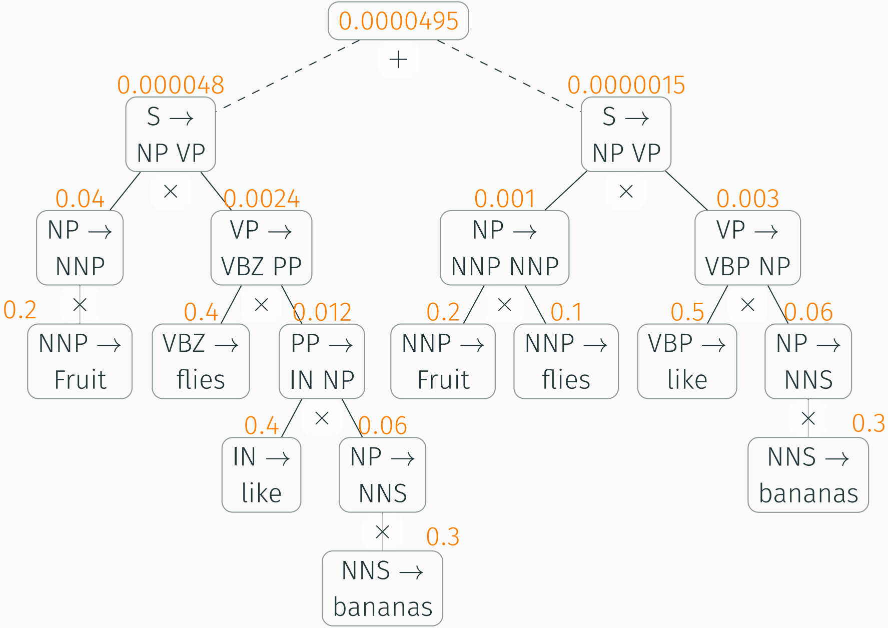
Recognition



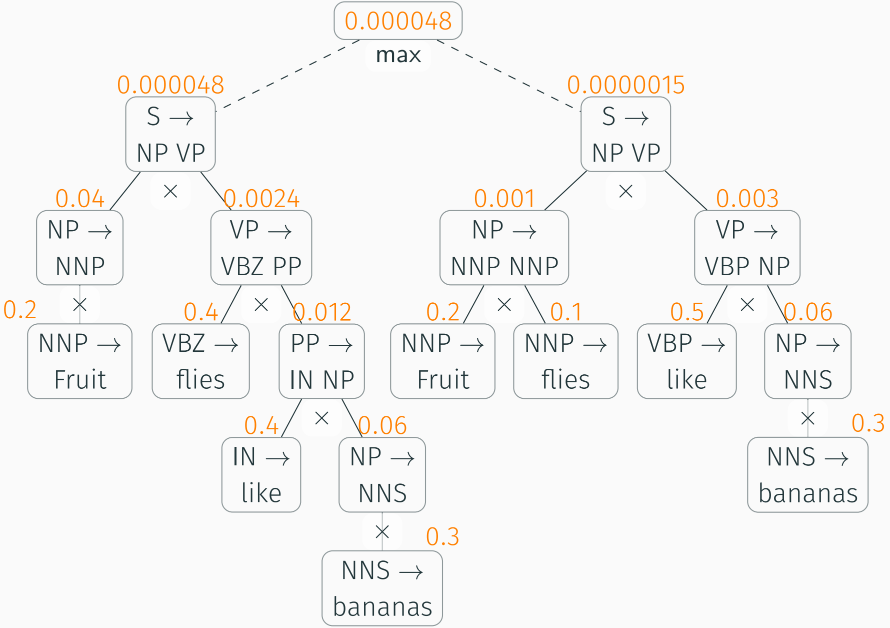
Recognition



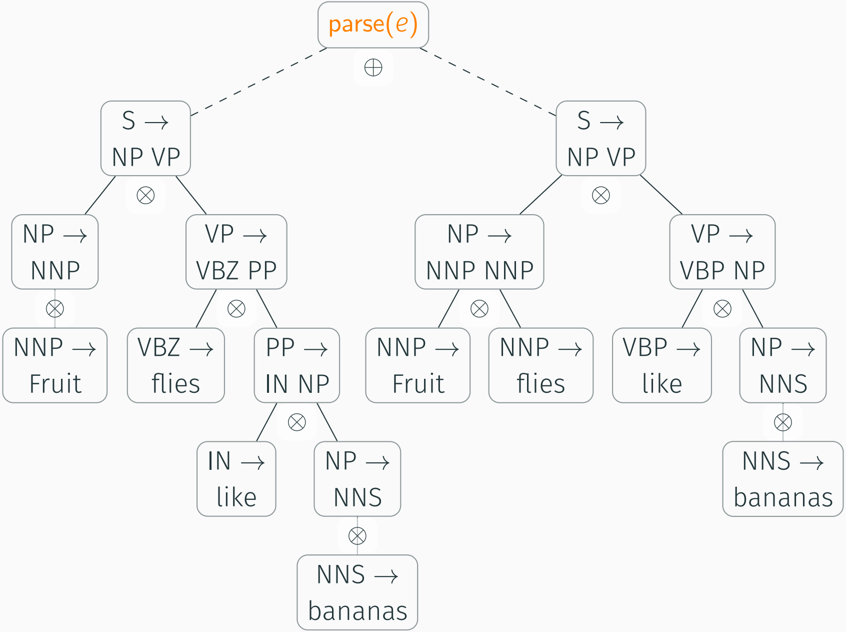
String probability



Probability of the most likely derivation



Generic computation



Semiring parsing

Semiring

Definition (Semiring)

A semiring $(S, \oplus, \otimes, 0, 1)$ is an algebraic structure, such that

- $(S, \oplus, 0)$ is a commutative monoid
- $(S, \otimes, 1)$ is a monoid,
- $\otimes$ is left-distributive and right-distributive over $\oplus$, and
- $a \otimes 0 = 0 = 0 \otimes a$ for every $a \in S$.

S is *complete* if $\sum^{\oplus}$ exists.

Semiring parsing

Algorithm for generic complete semiring $(S, \oplus, \otimes, 0, 1)$ [Goo99]

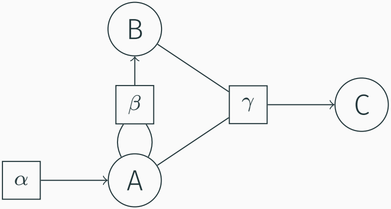
Instances:

Recognition	$(\{\top, \bot\}, \vee, \wedge, \bot, \top)$
String probability	$(\mathbb{R}_0^\infty, +, \times, 0, 1)$
Probability of best derivation	$(\mathbb{R}_0^1, \max, \times, 0, 1)$
Derivation forest	$(2^{\mathbb{E}^1}, \cup, \cdot, \emptyset, \{\varepsilon\})$

[1] set of derivations (elements of R^*)

M-monoid parsing

Knuth's generalization of Dijkstra's algorithm [Knu77]



$G = (N, \Sigma, C, R)$, with

- $N = \{\mathrm{A}, \mathrm{B}, \mathrm{C}\}$
- $\Sigma = \{\alpha, \beta, \gamma, \hat{(}, \hat{)}, \hat{,}\}$
- $R = \{\mathrm{C} \to \gamma(\mathrm{A}, \mathrm{B}), \mathrm{B} \to \beta(\mathrm{A}, \mathrm{A}), \mathrm{A} \to \alpha\}$

Mapping $\mathsf{val} : \Sigma^* \to \mathbb{R}_0^\infty$,
each $\sigma \in \Sigma$ is a mapping $\sigma : (\mathbb{R}_0^\infty)^k \to \mathbb{R}_0^\infty$

Generalization $(\mathbb{R}_0^\infty, \leq) \rightsquigarrow (S, \preceq)$ [Jun06]

Multioperator monoid

Definition (Multioperator monoid)

An M-monoid is an algebraic structure $(S, \oplus, 0, \Omega)$, such that

- $(S, \oplus, 0)$ is a commutative monoid,
- Ω is a set of operations on S, such that $\forall \omega \in \Omega : \omega(s_1, \ldots, s_k) = 0$ if $\exists i : s_i = 0$
- $0^k \in \Omega$ for all $k \in \mathbb{N}$, $0^k : S^k \to S$, such that $0^k(s_1, \ldots, s_k) = 0$.

S is *complete* if $\sum^{\oplus}$ exists.

M-monoid parsing problem

Given

1. a complete M-monoid $(S, \sum^{\oplus})$ with $(S, \oplus, 0, \Omega)$,
2. a weighted LCFRS (G, wt) over S where $G = (N, \Sigma, Z, R)$ is an LCFRS over Δ and $\mathsf{wt} : R \to \Omega$, and
3. a sentence $e = e_1 \ldots e_n$ with $n \geq 1$ and $e_i \in \Delta$

Compute $\mathsf{parse}(e) = \sum^{\bigoplus}_{d \in (T_R)_Z : [\![\pi_\Sigma(d)]\!] = e} h(d)$,

where $h : T_R \to S$ is the homomorphism from T_R to (S, Ω).

Weighted deductive parsing [Ned03]

Items $\mathcal{I} = \{[A, \vec{\kappa}] \mid A \in N, \vec{\kappa}$ range vector over $e\}$

Inference rules

SCAN: $\frac{}{[A,(i-1,i)]}$ if $\rho = (A \to \langle e_i \rangle)$ in R

RULE: $\frac{[B_1,\vec{\kappa}_1] \quad \ldots \quad [B_k,\vec{\kappa}_k]}{[A,\sigma(\vec{\kappa}_1,\ldots,\vec{\kappa}_k)]}$ if $\rho = (A \to \sigma(B_1, \ldots, B_k))$ in R

Goal: $[Z, (0, |e|)]$

M-monoid parsing algorithm

Input

1. an M-monoid $(S, \oplus, 0, \Omega)$,
2. an LCFRS$^-$ $G = (N, \Sigma, Z, R)$ over Δ, and $\mathsf{wt} : R \to \Omega$,
3. a function select : $2^{\mathcal{I}} \to \mathcal{I}$ specific to the M-monoid, and
4. a sentence $e = e_1 \dots e_n$ with $n \geq 1$ and $e_i \in \Delta$

Variables $V : \mathcal{I} \to S$ mapping

Output $\mathsf{parse}(e)$

Algorithm 2.1 M-monoid parsing for LCFRS^{-}

$\mathcal{A}, \mathcal{C} \leftarrow \varnothing$

for each $A \in N$ and $\vec{\kappa}$ range vector over e **do**

    $V([A, \vec{\kappa}]) \leftarrow 0$

for each $\rho = (A \to \sigma)$ in R and $[A, \vec{\kappa}]$ generated by SCAN $\frac{}{[A,\vec{\kappa}]}$ **do**

    $V([A, \vec{\kappa}]) \leftarrow V([A, \vec{\kappa}]) \oplus \mathsf{wt}(\rho)()$

    $\mathcal{A} \leftarrow \mathcal{A} \cup \{[A, \vec{\kappa}]\}$

while $\mathcal{A} \neq \emptyset$ **do**

    $[A, \vec{\kappa}] \leftarrow \mathrm{select}(\mathcal{A})$

    $\mathcal{A} \leftarrow \mathcal{A} \setminus \{[A, \vec{\kappa}]\}$

    $\mathcal{C} \leftarrow \mathcal{C} \cup \{[A, \vec{\kappa}]\}$

    for each $\rho = (B \to \sigma(B_1, \ldots B_k))$ in R and $[B, \vec{\eta}]$ deduced by RULE $\frac{*}{[B,\vec{\eta}]}$ from $[A, \vec{\kappa}]$ and other items from $\mathcal{C}$ **do**

        $V([B, \vec{\eta}]) \leftarrow V([B, \vec{\eta}]) \oplus \mathsf{wt}(\rho)(V([B_1, \vec{\kappa}_1]), \ldots, V([B_k, \vec{\kappa}_k]))$

        if $[B, \vec{\eta}] \notin \mathcal{C}$ **then**

            $\mathcal{A} \leftarrow \mathcal{A} \cup \{[B, \vec{\eta}]\}$

return $V([Z, (0, n)])$

Reduct generation with M-monoid parsing

Reduct of a grammar and a sentence

Given

- an LCFRS $G = (N, \Sigma, Z, R)$ over Δ (RTG-notation)
- a sentence $e \in \Delta^*$

Compute LCFRS $G \triangleright_\psi e = (N', \Sigma, Z', R')$, such that

1. $[\![L(G \triangleright_\psi e)]\!]^{\text{LCFRS}} = [\![L(G)]\!]^{\text{LCFRS}} \cap \{e\}$, and
2. with the mapping $\psi : N' \to N$ there exists a bijective mapping from the ASTs of $G \triangleright_\psi e$ to the ASTs of G

Preliminary definitions

Prototype rules:

$$P_R = \{[A, \vec{\kappa}] \rightarrow \sigma([B_1, \vec{\kappa}_1], \ldots [B_k, \vec{\kappa}_k]) \mid$$
$$(A \rightarrow \sigma(B_1, \ldots, B_k)) \in R \wedge$$
$$\vec{\kappa}, \vec{\kappa}_1, \ldots, \vec{\kappa}_k \text{ are range vectors over } e\}$$

Prototype nonterminals:

$$P_N = \{[A, \vec{\kappa}] \mid A \in N \wedge \vec{\kappa} \text{ is a range vector over } e\}$$

The reduct problem as an instance of M-monoid parsing

Given:

1. $(\mathbb{G}, \sum^{\cup})$, where
   - $\mathbb{G} = (2^{P_N} \times 2^{P_R}, \cup, \emptyset \times \emptyset, \Omega_{\mathrm{REDUCT}})$ is the *reduct M-monoid*
   - $\sum^{\cup}_{i \in I} s_i = \bigcup_{i \in I} s_i$ is used as the infinitary sum operation
2. (G, wt) for an arbitrary G over Δ and $\mathsf{wt} : R \to \Omega_{\mathrm{REDUCT}}$
3. an arbitrary $e \in \Delta^*$

Compute: $\mathsf{parse}(e) = (\{\mathrm{A} \in N' \mid \mathrm{A} \text{ is of the form } [\mathrm{Z}, \vec{\kappa}]\}, R')$

Then $G \rhd_\psi e = (N', \Sigma, Z', R')$, where

- $R' = v$ where $(u, v) = \mathsf{parse}(e)$,
- $N' = \{[A, \vec{\kappa}] \mid [A, \vec{\kappa}] \text{ is the left-hand side of a rule in } R'\}$,
- $Z' = [Z, (0, |e|)]$

The operations of Ω_{REDUCT}

- if ρ is of the form $A \to \langle w \rangle$, then

$$\omega_\rho() = \langle \{[A,(i-1,i)] \mid e_i = w\}, \\ \{[A,(i-1,i)] \to \langle w \rangle \mid e_i = w\} \rangle.$$

- if ρ is of the form $A \to \sigma(B_1, \ldots, B_k)$, then

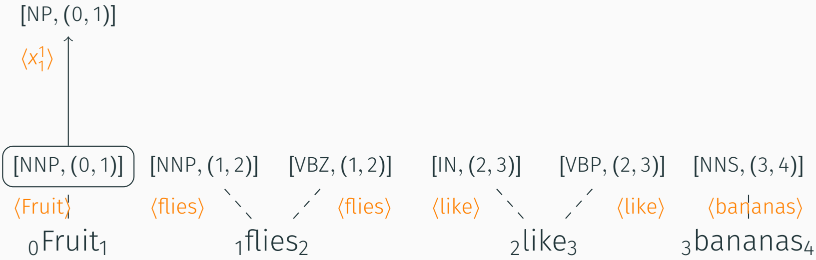
$$\omega_\rho((U_1,V_1),\ldots,(U_k,V_k)) = \langle U, V \rangle, \text{where}$$

$$U = \{[A, \sigma(\vec{\eta}_1,\ldots,\vec{\eta}_k)] \mid (\vec{\eta}_1,\ldots,\vec{\eta}_k) \in \mathrm{fit}_\sigma(U_1,\ldots,U_k)\}$$

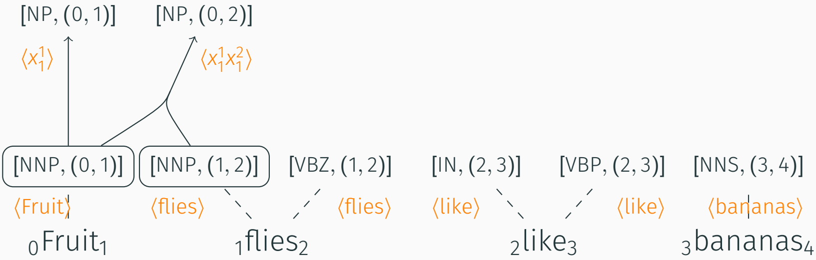
$$V = \bigcup_{1\le i\le k} V_i \cup \{[A,\sigma(\vec{\eta}_1,\ldots,\vec{\eta}_k)] \to \sigma([B_1,\vec{\eta}_1],\ldots,[B_k,\vec{\eta}_k]) \mid \\ (\vec{\eta}_1,\ldots,\vec{\eta}_k) \in \mathrm{fit}_\sigma(U_1,\ldots,U_k)\}$$

Example: reduct generation

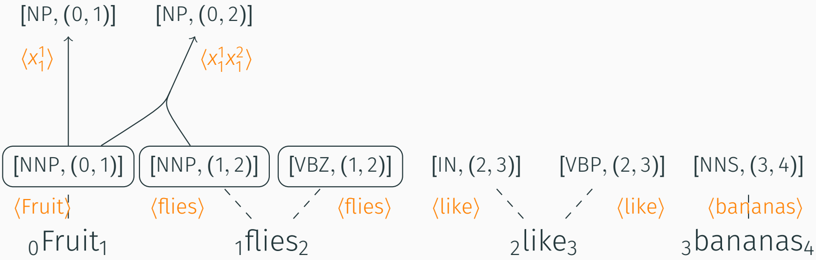
Example: reduct generation



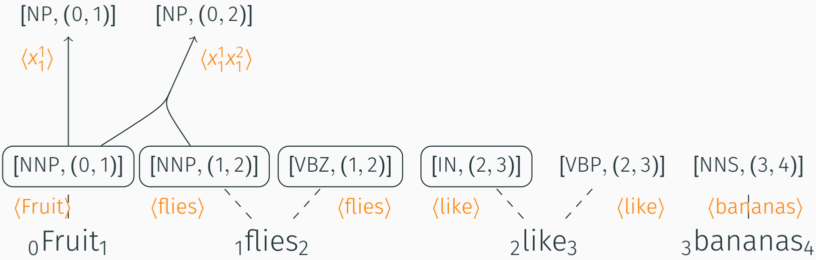
Example: reduct generation



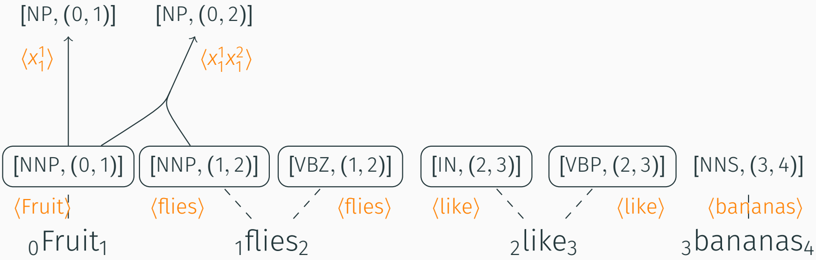
Example: reduct generation



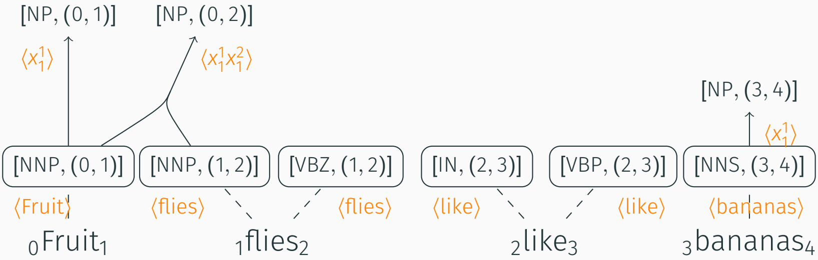
Example: reduct generation



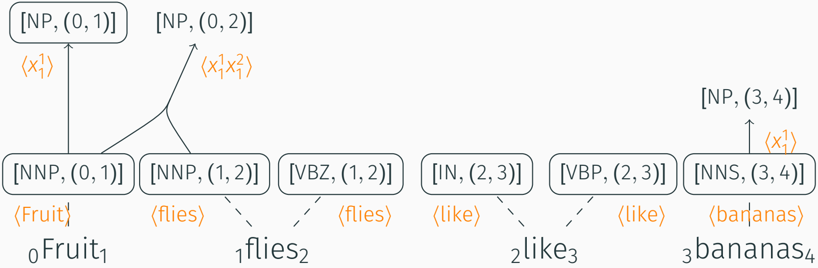
Example: reduct generation



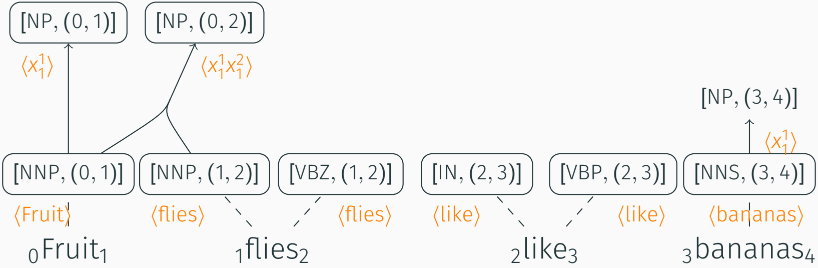
Example: reduct generation



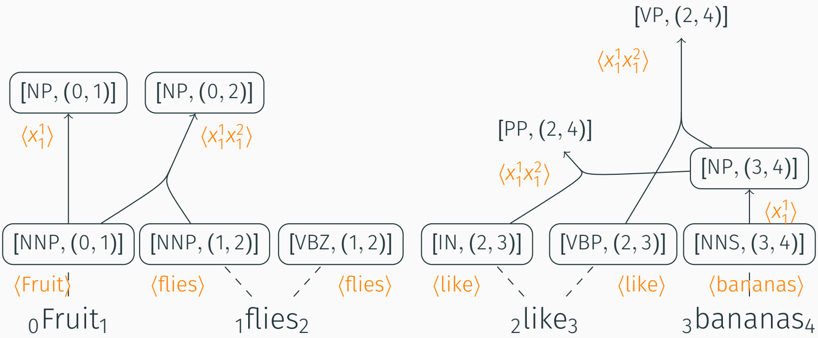
Example: reduct generation



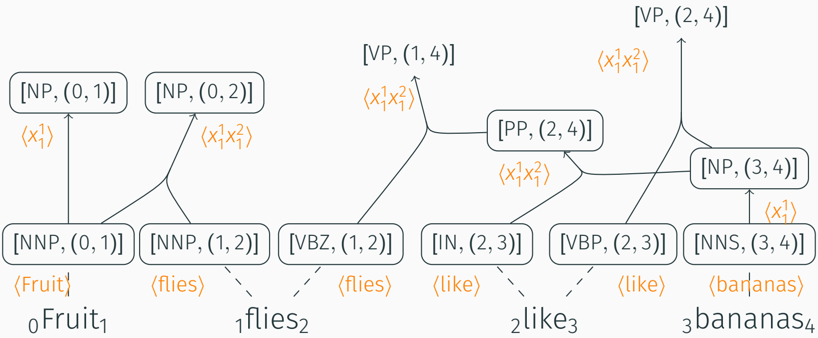
Example: reduct generation



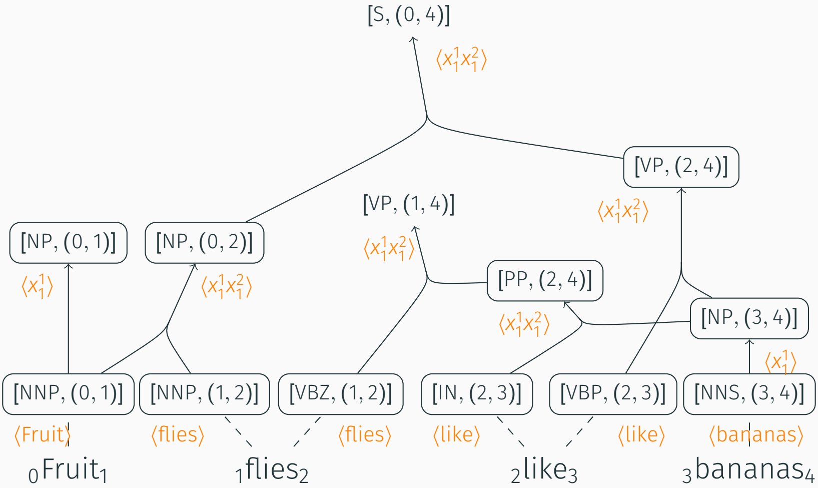
Example: reduct generation



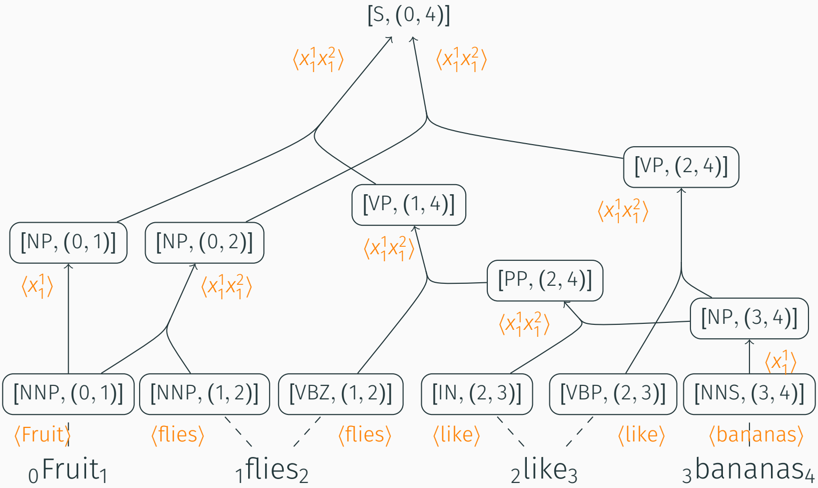
Example: reduct generation



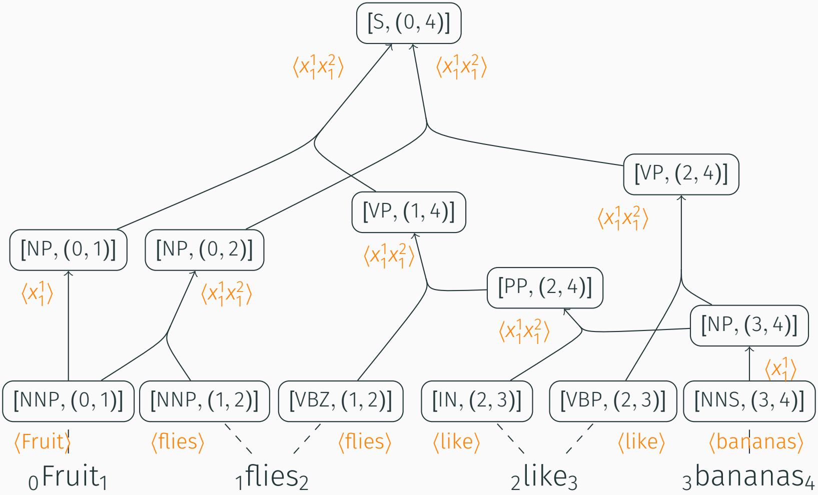
Example: reduct generation



Example: reduct generation



Example: reduct generation



Properties of the algorithm

Termination and correctness

Termination

- the algorithm always terminates

Termination and correctness

Termination

- the algorithm always terminates

Correctness

- not correct in general

Termination and correctness

Termination

- the algorithm always terminates

Correctness

- not correct in general
- correct for *not cyclic* LCFRS
- correct for *inferior* M-monoids
- correct for the reduct generation

Termination

Lemma

The M-monoid parsing algorithm always terminates.

Proof.

- termination of while loop if $\mathcal{A} = \emptyset$
- set of all possible elements of $\mathcal{A}$ is finite
- every element is added to $\mathcal{A}$ at most once
- only a finite number of elements is added to $\mathcal{A}$
- in each iteration, one element is removed from $\mathcal{A}$

□

Definition (Inferior operation)

Let $(S, \preceq)$ be a totally ordered set and $\omega : S^k \to S$ ($k \in \mathbb{N}$) be an operation. We call ω $\preceq$-*inferior* if for every $s_1, \ldots, s_k, s \in S$ and for every $i \in \{1, \ldots, k\}$ the following properties hold:

1. if $s \preceq s_i$ then $\omega(s_1, \ldots, s_{i-1}, s, s_{i+1}, \ldots, s_k) \preceq \omega(s_1, \ldots, s_{i-1}, s_i, s_{i+1}, \ldots, s_k)$
2. $\omega(s_1, \ldots, s_k) \preceq \min\{s_1, \ldots, s_k\}$

Correctness for inferior M-monoids ii

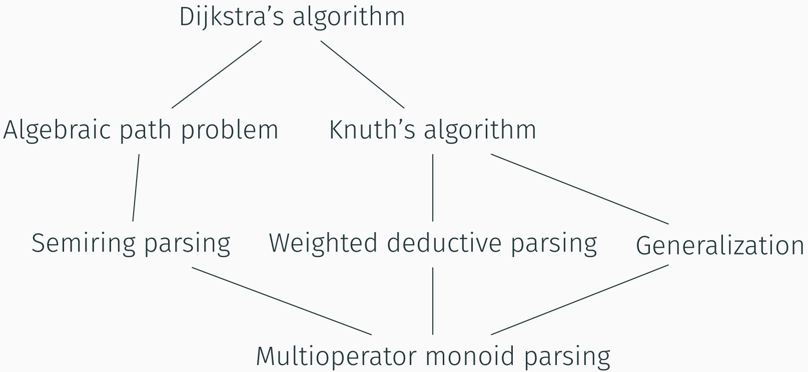
Definition (Inferior M-monoid)

Let $(S, \oplus, 0, \Omega)$ be an M-monoid. Moreover, let $\preceq_\oplus$ be the binary relation on S defined for every $a, b \in S$ as follows: $a \preceq_\oplus b$ if $a \oplus b = b$. If $\preceq_\oplus$ is a total order such that $0 \preceq_\oplus a$ for every $a \in S$ and every $\omega \in \Omega$ is $\preceq_\oplus$-inferior, then we call the M-monoid S *inferior*.

Lemma

Let $\mathcal{S}$ be the class of inferior M-monoids. The algorithm is correct for $\mathcal{S}$ and the select *function* arg max.

Conclusion


Dijkstra's algorithm
Algebraic path problem
Knuth's algorithm
Semiring parsing
Weighted deductive parsing
Generalization
Multioperator monoid parsing

Infinitary sum operation

Definition (Infinitary sum operation)

$\sum^{\oplus}$ is a family $(\sum^{\oplus}{}_I \mid I \text{ countable})$ of mappings $\sum^{\oplus}{}_I : S^I \to S$.

- S^I is represented as a family $(s_i \mid i \in I)$ with $s_i \in S$
- notion: $\sum^{\oplus}{}_{i \in I} s_i$

The mapping fit

$\mathsf{fit}_\sigma : (P_N)^k \to 2^{(\vec{\kappa}^k)}$, where for each $\sigma \in \Sigma$ and arity $k \in \mathbb{N}$:

$$\begin{aligned}\mathsf{fit}_\sigma(U_1, \ldots, U_k) = \{(\vec{\eta}_1, \ldots, \vec{\eta}_k) \mid \\ & [B_1, \vec{\eta}_1] \in U_1, \ldots, [B_k, \vec{\eta}_k] \in U_k, B_1, \ldots, B_k \in N \wedge \\ & \sigma(\vec{\eta}_1, \ldots, \vec{\eta}_k) \text{ is a range vector over } e\}.\end{aligned}$$

Cyclic and weakly cyclic LCFRS$^-$

We call an LCFRS$^-$ G *cyclic for* $e \in \Delta^*$ if there are $A \in N$, range vectors $\vec{\kappa}$ over e, and sentential forms $\alpha, \beta, \alpha', \beta'$ such that

$$Z(\varepsilon) \Rightarrow^* \alpha A(\vec{\kappa}(e))\beta \Rightarrow^+ \alpha' A(\vec{\kappa}(e))\beta' \Rightarrow^* \varepsilon$$

We call an LCFRS$^-$ G *weakly cyclic for* $e \in \Delta^*$ if there are $A \in N$, range vectors $\vec{\kappa}$ over e, and sentential forms α, β such that

$$A(\vec{\kappa}(e)) \Rightarrow^+ \alpha A(\vec{\kappa}(e))\beta \Rightarrow^* \varepsilon$$

We call an LCFRS$^-$ *cyclic* (*weakly-cyclic*) if there is an $e \in \Delta^*$ such that G is cyclic for e (respectively, weakly cyclic for e).

Correctness for not cyclic LCFRS$^-$

Lemma

Let $G = (N, \Sigma, Z, R)$ be a LCFRS$^-$ not cyclic for $e \in \Delta^*$. Then, for every M-monoid S, $\mathsf{wt} : R \to S$, and $\mathsf{select} : 2^{\mathcal{I}} \to \mathcal{I}$ it holds that the algorithm is correct for S, (G, wt), select, and e. In particular, for every LCFRS$^-$ G and every sentence e for which G is not weakly cyclic, after termination, for each $[A, \vec{\kappa}] \in \mathcal{C}$, it holds that

$$V([A, \vec{\kappa}]) = \sum^{\bigoplus}_{d \in (T_R)_A : [\![\pi_\Sigma(d)]\!] = \vec{\kappa}(e)} h(d) \enspace ,$$

Corollary

The algorithm is correct for the class of all not cyclic LCFRS$^-$.

Joshua Goodman.
Semiring parsing.
Computational Linguistics, 25(4):573–605, 1999.

Jean Christoph Jung.
Knuth's generalization of Dijkstra's algorithm.
2006.

D.E. Knuth.
A Generalization of Dijkstra's Algorithm.
Inform. Process. Lett., 6(1):1–5, February 1977.

M.-J. Nederhof.
Weighted deductive parsing and Knuth's algorithm.
Computational Linguistics, 29(1):135–143, 2003.