Odkrivanje enačb in predznanje

Ljupčo Todorovski

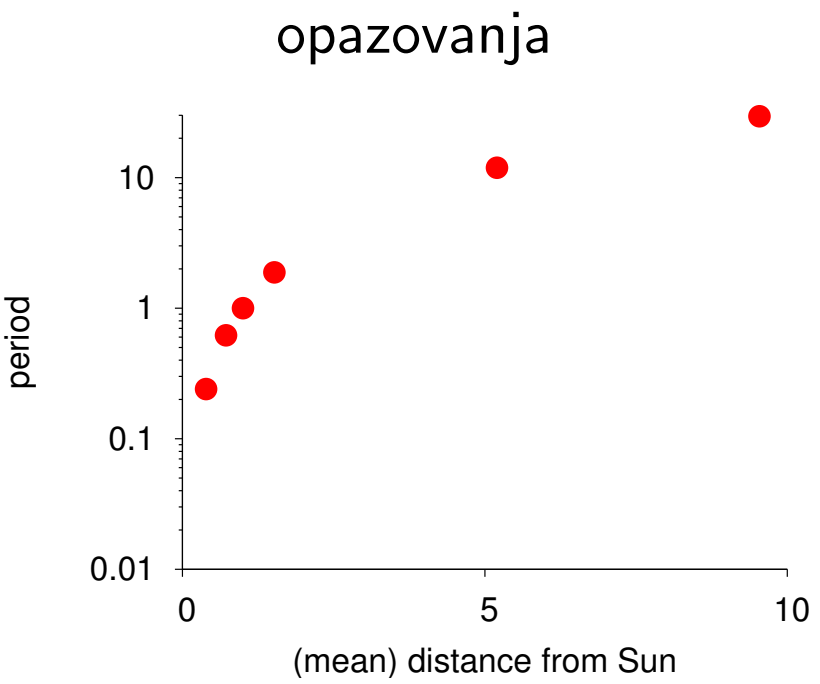
Univerza v Ljubljani, Fakulteta za matematiko in fiziko
Institut Jožef Stefan, Odsek za tehnologije znanja (E8)

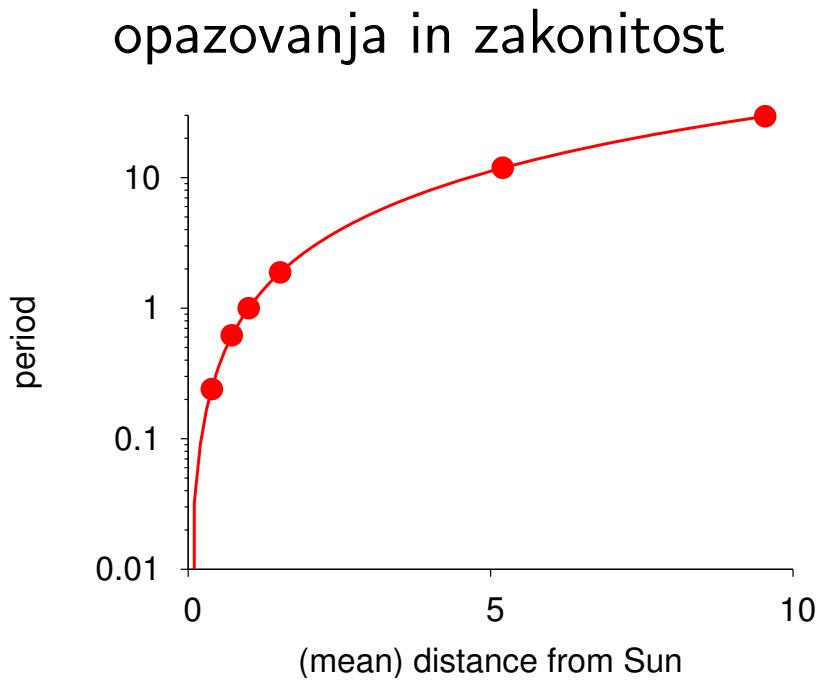
Maj 2023

Odkrivanje enačb: tretji Keplerjev zakon

Rekonstrukcija Keplerjevega tretjega zakona iz podatkov

$$d^3/p^2 = const$$





Pregled predavanja

1. Motivacija

2. Kontekstno-neodvisne gramatike
   - Deterministične gramatike
   - Tvorjenje stavkov
   - Verjetnostne gramatike

3. Od predznanja do gramatik
   - (Ne)Omejeni prostori enačb
   - Znanje o merskih enotah spremenljivk
   - Znanje s konkretnega področja uporabe

Ustvarjanje strukture $E(X)$: Generate

Več možnosti

1. Stohastični evolucijski pristop (običajna, široko uporabljena možnost)
2. Sistematični pristop: deterministične gramatike
3. Stohastični pristop: verjetnostne gramatike

Formalne gramatike in jeziki

Formalna gramatika G določa

- Kako tvoriti nize znakov (besede) iz podane abecede Σ
- Ki so pravilni oz. sledijo predpisani sintaksi jezika

Nizi znakov, besede in jeziki

- Množico vseh možnih besed nad abecedo Σ označimo z Σ^*
- Dolžina niza $w \in \Sigma^*$ je število znakov v w, oznaka $|w|$
- Jezik gramatike G nad abecedo Σ je množica vseh pravilnih besed, t.j., besed, ki sledijo sintaksi, oznaka z $L(G) \subseteq \Sigma^*$

Definicija kontekstno-neodvisne gramatike

Gramatike uporabljamo za definiranje sintakse jezikov.

Kontekstno neodvisna (*context-free*) gramatika $G = (N, \Sigma, R, S)$

- N je množica nekončnih simbolov, tudi spremenljivk
- Σ je abeceda, množica končnih simbolov, črk
- $R \subseteq N \times (N \cup \Sigma)^*$ je množica produkcijskih pravil oblike $A \to \alpha$:
  - $A \in N$ je nekončni simbol
  - $\alpha \in (N \cup \Sigma)^*$ je niz končnih in nekončnih simbolov
- $S \in N$ je začetni nekončni simbol

Končni in nekončni simboli

Končni simboli

- Ustrezajo črkam abecede za tvorjenje besed
- Običajno jih pišemo z majhno začetnico
- Osnovni simboli za tvorjenje matematičnih izrazov, npr. x, y, c, $+$, $\cdot$
- Spomnimo se: c je oznaka za konstantni parameter

Nekončni simboli

- Ustrezajo nizom znakov, ki so (ponavljajoči se) sestavni deli besed
- Običajno jih pišemo z veliko začetnico
- Matematični podizrazi, kot so npr. monomi, $c \cdot x$ ali $c \cdot x \cdot y$

Besede in izpeljave

Beseda

- Zaporedje končnih simbolov $w \in \Sigma^*$
- Primer: linearni matematični izraz $c + c \cdot x$

Korak izpeljave z gramatiko G ustreza relaciji $\Rightarrow_G$

$$\beta A \gamma \Rightarrow_G \beta \alpha \gamma$$

- $\alpha, \beta, \gamma \in (N \cup \Sigma)^*$ so nizi končnih in nekončnih simbolov
- $A \in N$ je nekončni simbol
- $A \to \alpha \in R$ je produkcijsko pravilo

Izpeljava besede in jezik gramatike

Izpeljava besede $w \in L(G)$

$$S \Rightarrow_G^* w$$

- S je začetni nekončni simbol gramatike G
- $\Rightarrow_G^*$ je tranzitivna ovojnica relacije $\Rightarrow_G$
- Predstavlja torej zaporedje korakov izpeljave od S do w

Jezik $L(G)$ gramatike G

$$L(G) = \{w : S \Rightarrow_G^* w\}$$

To je množica vseh besed, ki jih lahko izpeljemo z gramatiko G.

Gramatika za linearne matematične izraze: končni simboli

1. Spremenljivke: $x_1, x_2, \ldots, x_p$
2. Konstantni parameter: c
3. Operatorja: $+$ in $\cdot$

Gramatika za linearne izraze: nekončni simboli in pravila

Nekončni simbol za spremenljivke V

- p produkcijskih pravil oblike $V \to x_i,\ i = 1, 2, \ldots p$
- Krajši zapis $V \to x_1 \mid x_2 \mid \ldots \mid x_p$

Nekončni simbol za linearne člene, monome M

- Enostavno pravilo $M \to c \cdot V$
- Produkt konstantnega parametra in spremenljivke

Nekončni simbol za linearne matematične izraze E

- Rekurzivno pravilo $E \to E + M$
- Enostavno pravilo, robni pogoj rekurzije $E \to c$

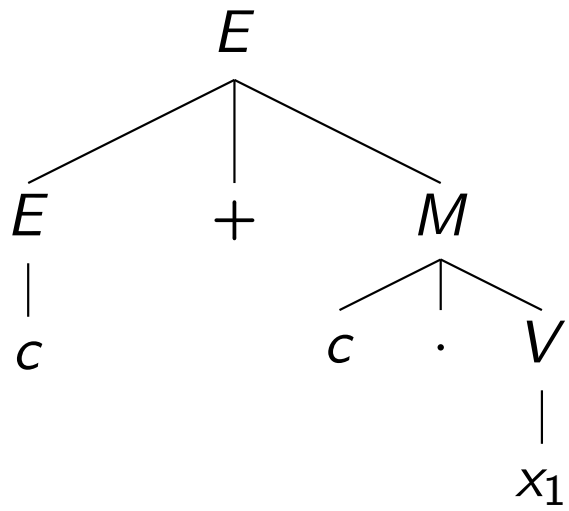
Gramatika za linearne izraze: celotna definicija

$$\Sigma = \{+, \cdot, c, x_1, x_2, \ldots, x_p\}$$
$$N = \{E, M, V\}$$
$$R = \left\{ \begin{array}{lcl} E & \to & E + M \\ E & \to & c \\ M & \to & c \cdot V \\ V & \to & x_1 \\ V & \to & x_2 \\ \vdots & & \\ V & \to & x_p \end{array} \right\}$$
$$S = E$$

Izpeljava linearnega izraza $c + c \cdot x_1 + c \cdot x_2$

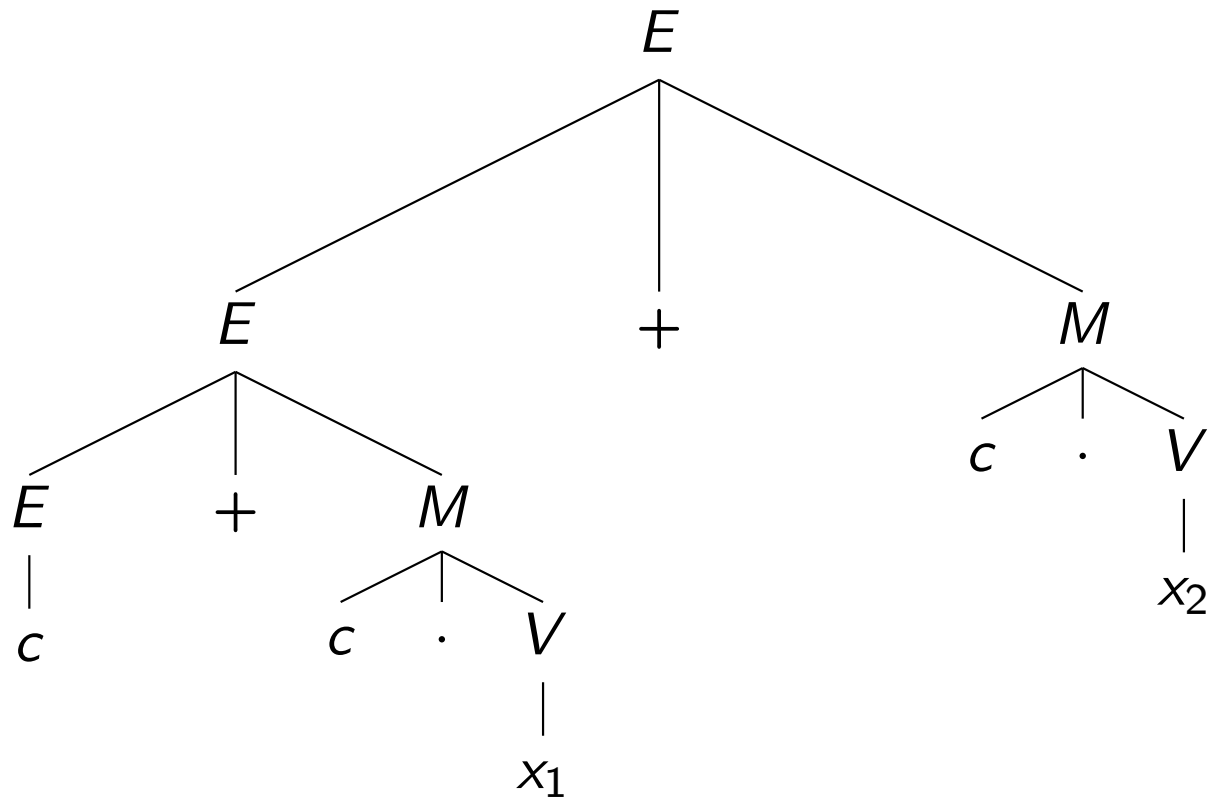
$$
\begin{aligned}
E &\to \\
[E \to E + M] &\to E + M \\
[E \to E + M] &\to E + M + M \\
[E \to c] &\to c + M + M \\
[M \to c \cdot V] &\to c + c \cdot V + M \\
[M \to c \cdot V] &\to c + c \cdot V + c \cdot V \\
[V \to x_1] &\to c + c \cdot x_1 + c \cdot V \\
[V \to x_2] &\to c + c \cdot x_1 + c \cdot x_2
\end{aligned}
$$

Drevo izpeljave linearnega izraza $c + c \cdot x_1$



- Korensko vozlišče je $S \in N$
- Notranja vozlišča so nekončni, končna vozlišča pa končni simboli
- Vsako notranje vozlišče razvejano s produkcijskim pravilom
- Izpeljana beseda je sprehod po končnih vozliščih od leve proti desni

Drevo izpeljave linearnega izraza $c + c \cdot x_1 + c \cdot x_2$



Dva načina uporabe gramatik

Običajna: Razčlenjevanje besed na osnovi sintakse, *parsing*

- Od podane besede w do (drevesa) izpeljave
- Preverjanje pravilnosti sintakse podane besede

Pri odkrivanju enačb: Tvorjenje besed

- Sistematično naštevanje ali (naključno) tvorjenje besed
- Lahko tvorimo strukture izrazov za prvo fazo odkrivanja enačb

Tvorjenje (drevesa izpeljave) besede

Require: $G = (N, \Sigma, R, S)$ je kontekstno-neodvisna gramatika
Require: $A \in N$ je nekončni simbol, pri prvem klicu je $A = S$
Ensure: t je drevo izpeljave v G s korenskim vozliščem A

function GenerateParseTree(G, A)
    iz R izberi pravilo $A \to X_1 X_2 \dots X_k$
    for $i \in \{1, 2, \dots k\}$ **do**
        if $X_i \in N$ **then**
            $t_i =$ GenerateParseTree(G, X_i)
        else
            $t_i = list\ X_i$
        end if
    end for
    return $tree(koren\ A, nasledniki\ t_1, t_2, \dots t_k)$
end function

Sistematično naštevanje dreves izpeljave

Enostaven algoritem

1. Začni z najbolj enostavnim drevesom izpeljave
2. Na vsakem koraku zamenjaj najbolj desno poddrevo z naslednjim

Potrebujemo delno ureditev produkcijskih pravil gramatike

- Bolj enostavna produkcijska pravila: bolj plitva drevesa
- Bolj zapletena produkcijska pravila: bolj globoka drevesa

Globina simbola $X \in N \cup \Sigma$

Globina najbolj plitvega drevesa izpeljave s korenom X.

Globine končnih simbolov $a \in \Sigma$

Očitno velja $d(a) = 0$.

Globine nekončnih simbolov $A \in N$

$$d(A) = \min_{r \in R:\, r = A \to \alpha} : d(r)$$

Globina pravila $d(r)$ je definirana na naslednji prosojnici.

Globina produkcijskega pravila $r \in R$

Naj bo produkcijsko pravilo $r = A \to X_1 X_2 \dots X_k$

$$d(r) = 1 + \max\{d(X_1), d(X_2), \dots, d(X_k)\}$$

Globina najbolj plitvega drevesa izpeljave s korenom A v katerem uporabimo pravilo r.

Primer izračuna globine, prvi korak, globina $d = 0$

Vsi končni simboli

- $d(x_i) = 0,\ i = 1, 2, \ldots, p$
- $d(c) = 0$
- $d(+) = 0$
- $d(\cdot) = 0$

Primer izračuna globine, drugi korak, globina $d = 1$

Produkcijska pravila, ki imajo na desni simbole z znano globino

- $d(V \to x_i) = 1 + \max(0) = 1,\ i = 1, 2, \ldots, p$
- $d(E \to c) = 1 + \max(0) = 1$

Nekončni simboli s pravili znane globine

- $d(V) = \min(1, 1, \ldots, 1) = 1$
- $d(E) = \min(d(E \to c) = 1, d(E \to E + M) > 1) = 1$

Primer izračuna globine, drugi korak, globina $d = 2$

Produkcijska pravila, ki imajo na desni simbole z znano globino

- $d(M \to c \cdot V) = 1 + max(0, 0, 1) = 2$

Nekončni simboli s pravili znane globine

- $d(M) = \min(d(M \to c \cdot V) = 2) = 2$

Primer izračuna globine, tretji korak, globina $d = 3$

Produkcijska pravila, ki imajo na desni simbole z znano globino

- $d(E \to E + M) = 1 + max(1, 0, 2) = 3$

Nekončni simboli s pravili znane globine

Vsi nekončni simboli že imajo znano globino, ni več izračunov.

Delna urejenost produkcijskih pravil na osnovi globine

- E: $M \to c \preccurlyeq E \to E + M$
- M: $M \to c \cdot V$
- V: $V \to x_1 \preccurlyeq V \to x_2 \preccurlyeq \ldots \preccurlyeq V \to x_p$

Zdaj je vse pripravljeno za naštevanje dreves izpeljave.

Dva algoritma za naštevanje dreves omejene globine

Najbolj enostavno drevo izpeljave

- GenerateParseTree: izbiramo najbolj enostavna pravila
- Produkcijska pravila z najnižjim indeksom v delno urejenem seznamu

Naslednje drevo izpeljave

- Vsakemu notranjemu vozlišču drevesa pripišemo indeks pravila
- Za vsako vozlišče v seznamu obratnega pregleda drevesa
  - Če je vozlišče končno, gremo naprej
  - Če je vozlišče notranje in je indeks produkcijskega pravila manjši od števila pravil, zamenjamo pravilo z naslednjim
  - Sicer pa zbrišemo poddrevo notranjega vozlišča
- Po potrebi dokončamo drevo z najbolj enostavnimi poddrevesi

Najbolj enostavno drevo izpeljave

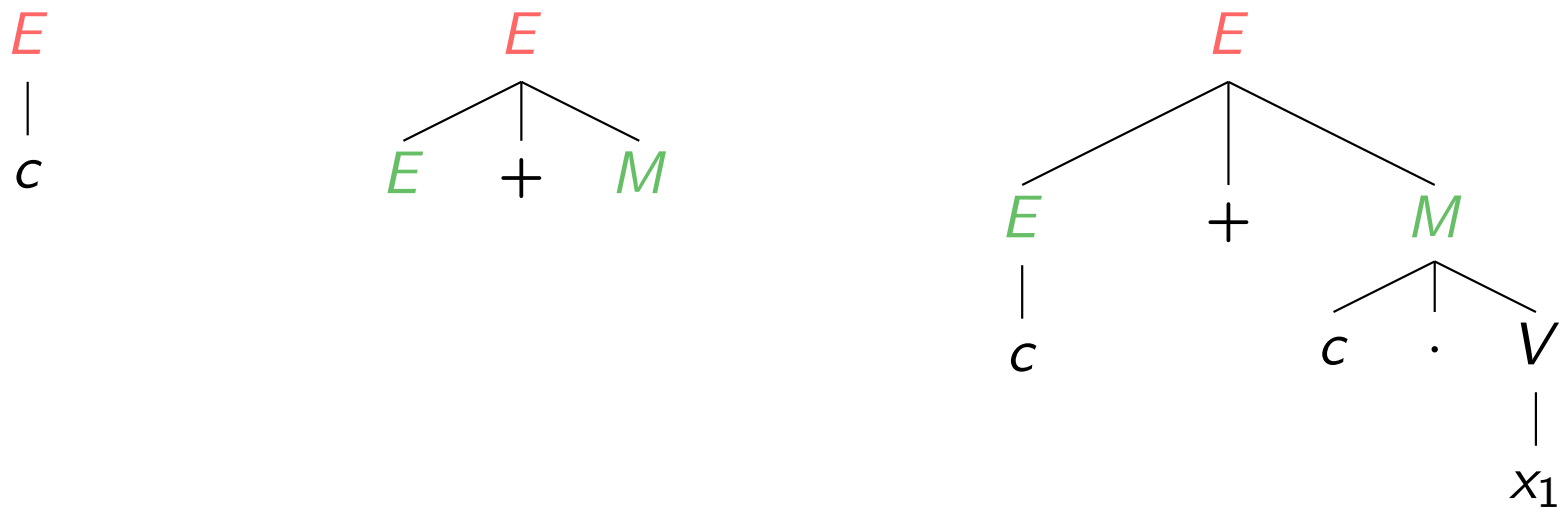
GenerateParseTree: izbiramo najbolj enostavna pravila.

$$\begin{array}{c} E \\ | \\ c \end{array}$$

Izpeljemo torej izraz c.

Naslednje drevo izpeljave (1)

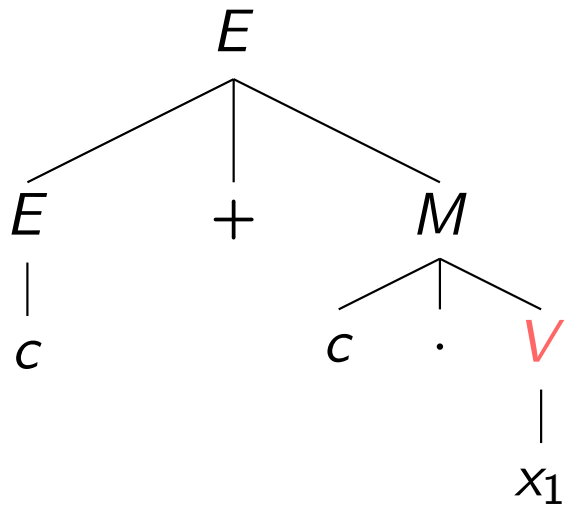
- Edino notranje vozlišče je E
- Produkcijsko pravilo za E zamenjamo z naslednjim
- Dokončamo drevo z najbolj enostavnimi drevesi izpeljave

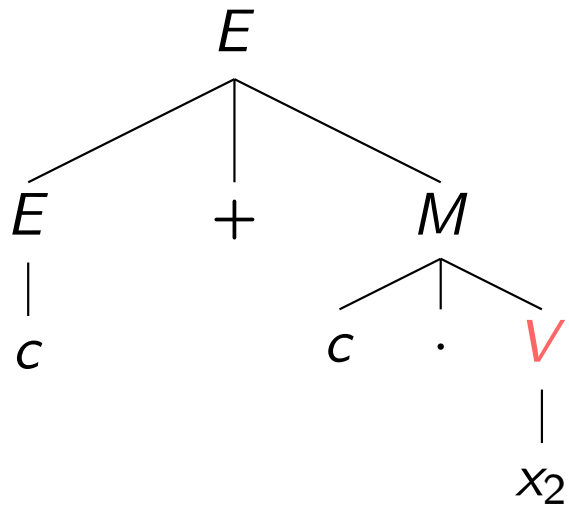


Izpeljemo torej izraz $c + c \cdot x_1$.

Naslednje drevo izpeljave (2a)

- Prvo notranje vozlišče v obratnem pregledu je V
- Produkcijsko pravilo $V \to x_1$ zamenjamo z naslednjim $V \to x_2$

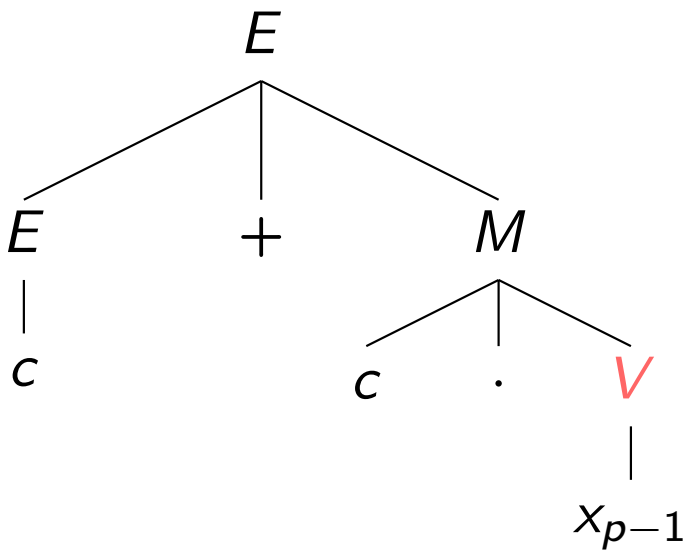


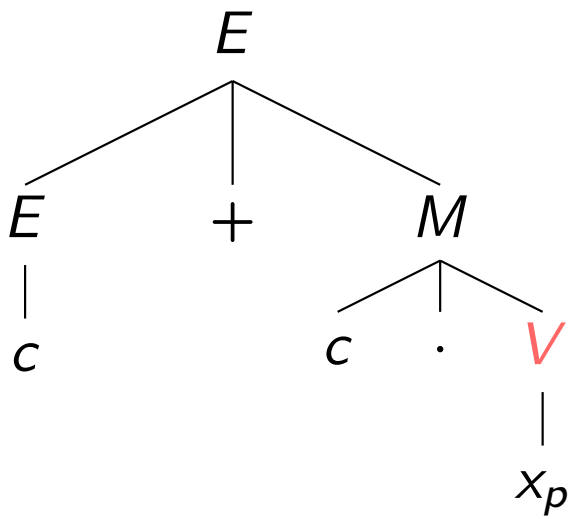


Izpeljemo torej izraz $c + c \cdot x_2$.

Naslednje drevo izpeljave (2b)

- Prvo notranje vozlišče v obratnem pregledu je V
- Produkcijsko pravilo $V \to x_{p-1}$ zamenjamo z naslednjim $V \to x_p$

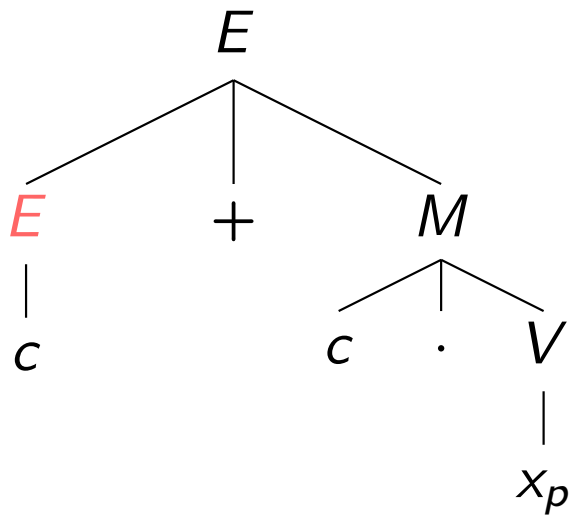


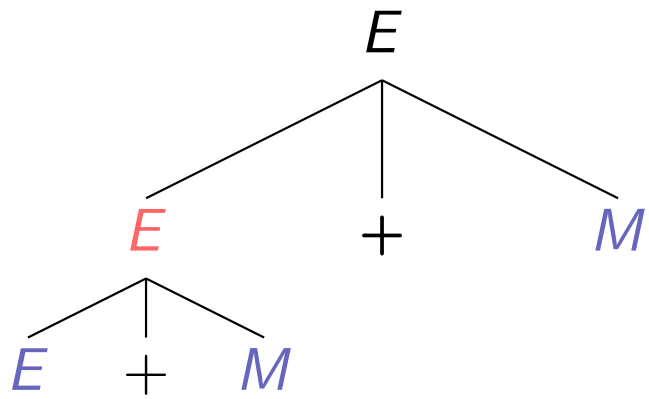


Izpeljemo torej izraz $c + c \cdot x_p$.

Naslednje drevo izpeljave (3a)

- Prvi dve notranji vozlišči v obratnem pregledu sta V in M
- Pri nobenem nimamo na voljo naslednjega produkcijskega pravila
- Naslednje notranje vozlišče je E
- Produkcijsko pravilo $E \to c$ zamenjamo z naslednjim $E \to E + M$

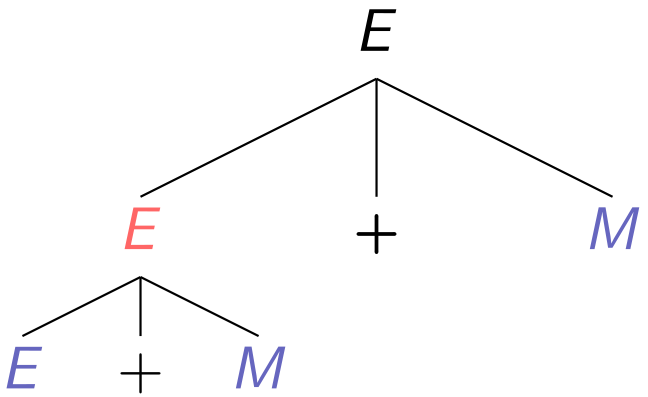


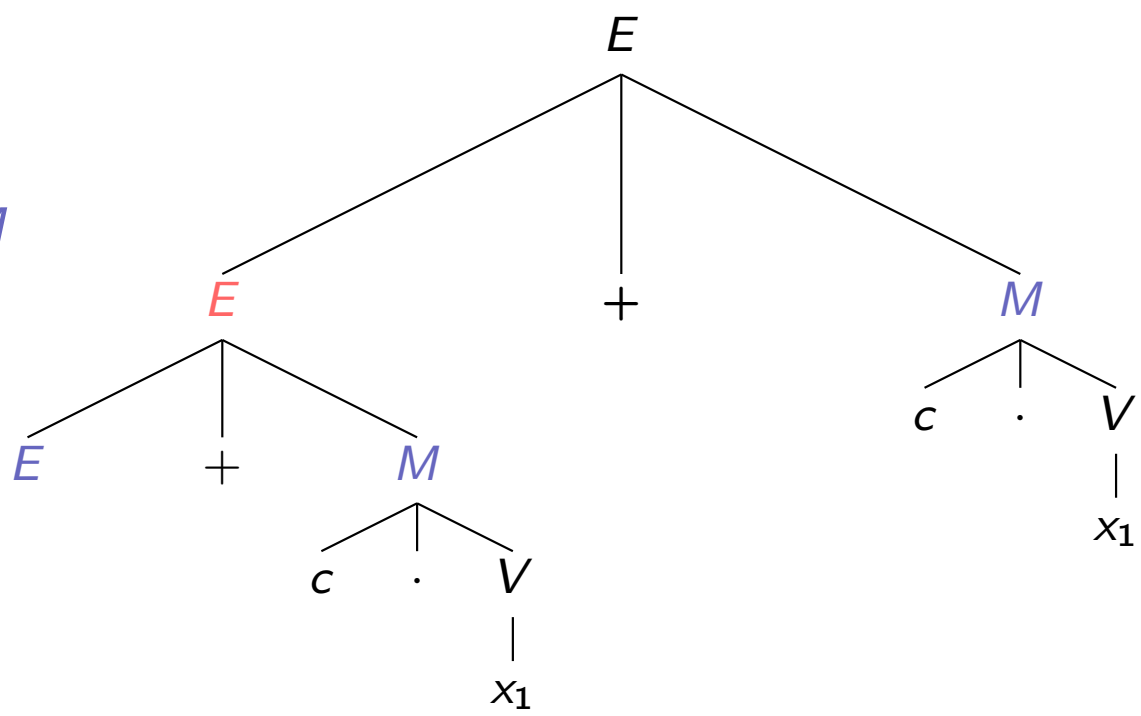


Nato dokončamo drevo, naslednja prosojnica.

Naslednje drevo izpeljave (3b)

- Dokončamo drevo z najbolj enostavnimi drevesi izpeljave





Izpeljemo torej izraz $c + c \cdot x_1 + c \cdot x_1$.

Problem: ogromno število dreves izpeljave

Rešitve

1. Uporabniško podana omejitev globine dreves
2. Različne strategije nepopolnega preiskovanja prostora dreves
3. Naključna tvorba dreves in verjetnostne gramatike

Verjetnosti produkcijskih pravil

Vsakemu produkcijskemu pravilu $r \in R$

Priredimo verjetnost $P(r)$.

Za vsak nekončni simbol A velja

$$\sum_{r \in R:\, r = A \to \alpha} P(r) = 1$$

Vsota verjetnosti produkcijskih pravil za isti nekončni simbol je 1.

Verjetnost $P(w)$ izpeljane besede $w \in L(G)$

Je produkt verjetnosti vseh produkcijskih pravil v drevesu izpeljave w.

$$P(w) = \prod_{r \in R} P(r)^{f(r)}$$

$f(r)$ je število pojavitev pravila r v izpeljavi/ drevesu $S \Rightarrow_G^* w$.

Verjetnost $P_G(S, d)$ dreves izpeljav globine največ d

Je enaka verjetnosti, da pri naključnem vzorčenju dreves izpeljave izberemo drevo globine največ d.

Rekurzivna formula za $P_G(X, d)$, za $X \in N \cup \Sigma$

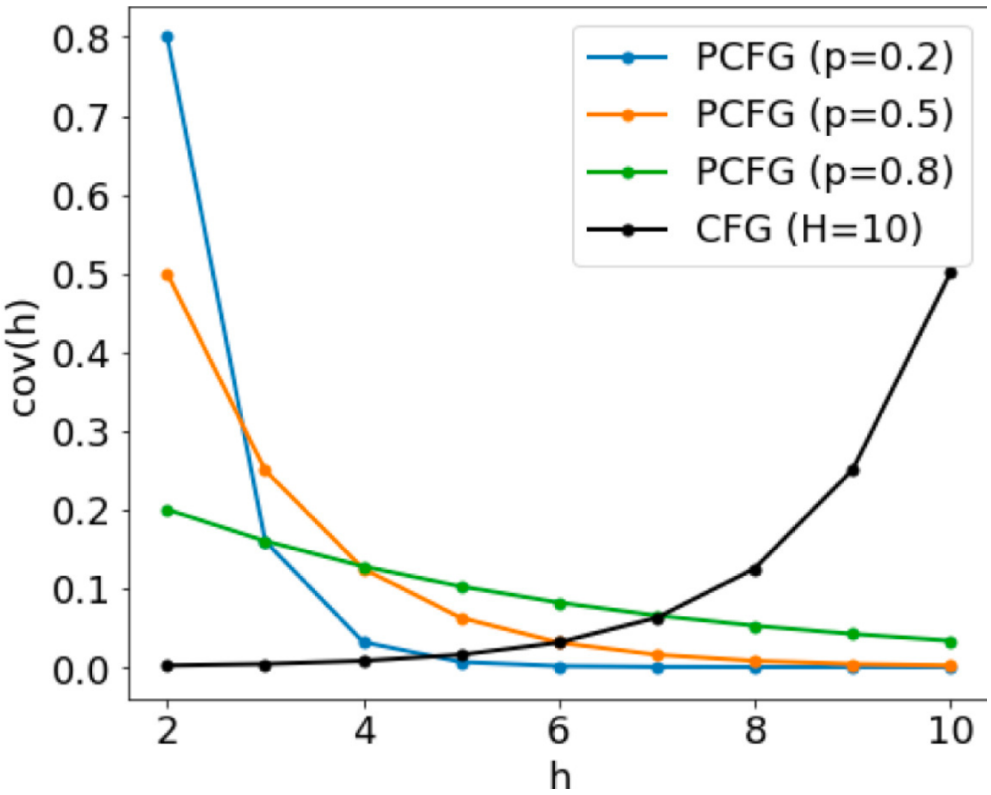
$$P_G(X, d) = \begin{cases} 1 & ; X \in \Sigma \wedge d \geq 0 \\ 0 & ; X \in N \wedge d = 0 \\ \sum_{r \in R: r = X \to X_1 X_2 \ldots X_k} P(r) \prod_{i=1}^{k} P_G(X_i, d-1) & ; X \in N \wedge d \geq 1 \end{cases}$$

Verjetnostna gramatika za linearne izraze

$$\Sigma = \{+, \cdot, c, x_1, x_2\}$$
$$N = \{E, M, V\}$$
$$R = \left\{ \begin{array}{lcl} E & \to & E + M \mid c \\ M & \to & c \cdot V \\ V & \to & x_1 \mid x_2 \end{array} \right\}$$
$$S = LE$$

- $P(V \to x_i) = 1/2,\ i = 1, 2$
- $P(M \to c \cdot V) = 1$
- $P(E \to c) = 1 - p,\ P(E \to E + M) = p$

Primerjava verjetnostne in deterministične gramatike



Vrednost p uravnava verjetnost bolj enostavnih enačb.

Stohastičen algoritem za odkrivanje enačb

Require: Verjetnostna gramatika $G = (N, \Sigma, R, S$ za matematične izraze, število vzorčenj n, podatkovna množica D, ciljna spremenljivka y
Ensure: Seznam enačb *eqns*, urejen po naraščajoči napaki na S

function ProGED(G, n, D, y)
    $eqns = [\,]$
    for $i \in \{1, 2 \ldots, n\}$ **do**
        (e, p) = GenerateParseTree(G, S)
        e_c = CanoncicalForm(e)
        $(eqn, error)$ = EstimateParameters($y = e_c$, D)
        $eqns$.append(eqn, p, $error$)
    end for
    return $eqns$.sort(key=$error$, order=increasing)
end function

Gramatika za linearne izraze

$$N = \{E, M, V\}$$
$$\Sigma = \{+, \cdot, c, x_1, x_2, \ldots, x_p\}$$
$$R = \left\{ \begin{array}{lcl} E & \to & E + M \mid c \\ M & \to & c \cdot V \\ V & \to & x_1 \mid x_2 \mid \ldots \mid x_p \end{array} \right\}$$
$$S = E$$

Gramatika za polinomske izraze

$$N = \{E, M, V\}$$
$$\Sigma = \{+, \cdot, c, x_1, x_2, \ldots, x_p\}$$
$$R = \left\{ \begin{array}{rcl} E & \to & E + c \cdot M \mid c \\ M & \to & M \cdot V \mid V \\ V & \to & x_1 \mid x_2 \mid \ldots \mid x_p \end{array} \right\}$$
$$S = E$$

Gramatika za racionalne izraze

$$N = \{RE, E, M, V\}$$
$$\Sigma = \{+, \cdot, /, (,), c, x_1, x_2, \ldots, x_p\}$$
$$R = \left\{ \begin{array}{rcl} RE & \to & (E)\,/\,(E) \\ E & \to & E + c \cdot M \mid c \\ M & \to & M \cdot V \mid V \\ V & \to & x_1 \mid x_2 \mid \ldots \mid x_p \end{array} \right\}$$
$$S = RE$$

Univerzalna gramatika za poljuben aritmetični izraz

$$N = \{E, F, T, FE, V\}$$

$$\Sigma = \{+, -, \cdot, /, (,), \sin, \cos, \exp, \ldots, c, x_1, x_2, \ldots, x_p\}$$

$$R = \left\{ \begin{array}{rcl} E & \to & E + F \mid E - F \mid F \\ F & \to & F * T \mid F/T \mid T \\ T & \to & (E) \mid FE \mid V \mid c \\ FE & \to & \sin(E) \mid \cos(E) \mid \exp(E) \mid \ldots \\ V & \to & x_1 \mid x_2 \mid \ldots \mid x_p \end{array} \right\}$$

$$S = E$$

Merske enote in sestavljanje

Znane enote opazovanih spremenljivk

- Tri spremenljivke: čas t, hitrost v in pot s
- Osnovne enote: meter $[m]$ in sekunda $[s]$
- Sestavljena enota: meter na sekundo m/s

Pravila za kombiniranje enot

- Sestavljene enote tvorimo z množenjem in deljenjem
- Seštevamo in odštevamo le enake enote

Merske enote kot vektorji

Predpostavimo nabor osnovnih enot $\mathcal{U}_O = \{[m], [s]\}$

- Sestavljeno enoto lahko predstavimo kot vektor potenc osnovnih enot
- Osnovni enoti $[m]$ in $[s]$ prestavljata enotska vektorja $(1, 0)$ in $(0, 1)$
- Enoto za hitrost $[m/s] = [m \cdot s^{-1}]$ predstavlja vektor $(1, -1)$

Atributna gramatika in enote

Atributna gramatika

- Vsakemu simbolu gramatike priredimo atribut(e)
- Vsakemu pravilu priredimo še atributno pravilo

Vsakemu simoblu priredimo vektorsko predstavitev enote

- Atributna pravila povedo pravila za enote
- Pravila za prirejanje enot
- Pravila za preverjanje ustreznosti enot

Atributna pravila za enote

Pravila za prirejanje enot

- $F.u1 \rightarrow F.u2 + T.u3$: $u1 = u2 + u3$
- $F.u1 \rightarrow F.u2/T.u3$: $u1 = u2 - u3$
- $V.u1 \rightarrow X_{cas}$: $u1 = (0, 1)$
- $V.u1 \rightarrow X_{pot}$: $u1 = (1, 0)$

Pravila za preverjanje ustreznosti enot

- $E.u1 \rightarrow E.u2 + F.u3$: $u1 == u2 == u3$
- $F.u1 \rightarrow T.u2$: $u1 == u2$
- $FE.u1 \rightarrow \sin(E.u2)$: $u1 == u2 == (0, 0)$

Epidemiološki model SIR

Spremenljivke

- t je čas, N je velikost opazovane populacije
- $S(t), s(t) = S(t)/N$, število in delež dovzetnih v opazovani populaciji
- $I(t), i(t) = I(t)/N$, število in delež okuženih v opazovani populaciji
- $R(t), r(t) = R(t)/N$, število in delež ozdravelih v populaciji

Celotna opazovana populacija se ne spreminja s časom

- $S(t) + I(t) + R(t) = N$
- $s(t) + i(t) + r(t) = 1$

Enačba za populacijo dovzetnih $s(t)$

- Populacija dovzetnih nikoli ne narašča
- Pada zaradi kritičnih kontaktov med dovzetnimi in okuženimi
- Predpostavka: v enoti časa, okuženi ima β kritičnih kontaktov
- Delež teh kontaktov z dovzetnimi je $\beta\, s(t)$
- Torej, vsak okuženi okuži $\beta\, s(t)$ dovzetnih na dan

Število okuženih v enoti časa je torej $\beta\, s(t)\, I(t)$

$$\frac{dS(t)}{dt} = -\beta\, s(t)\, I(t) \quad \text{oziroma} \quad \frac{ds(t)}{dt} = -\beta\, s(t)\, i(t)$$

Enačba za populacijo ozdravelih $r(t)$

- Predpostavka: v enoti časa je γ delež okuženih, ki ozdravi
- Če okužba traja d enot časa, je $\gamma = 1/d$

Delež ozdravelih v enoti časa je torej $\gamma\, i(t)$

$$\frac{dr(t)}{dt} = \gamma\, i(t)$$

Enačba za populacijo okuženih $i(t)$

- Iz enačbe $s(t) + i(t) + r(t) = 1$ dobimo
- $ds(t)/dt + di(t)/dt + dr(t)/dt = 0$

Torej

$$\frac{di(t)}{dt} = -\frac{ds(t)}{dt} - \frac{dr(t)}{dt} \text{ oziroma } \frac{di(t)}{dt} = \beta\, s(t)\, i(t) - \gamma\, i(t)$$

Tri diferencialne enačbe modela SIR

$$\frac{ds(t)}{dt} = -\beta\, s(t)\, i(t)$$
$$\frac{di(t)}{dt} = \beta\, s(t)\, i(t) - \gamma\, i(t)$$
$$\frac{dr(t)}{dt} = \gamma\, i(t)$$

Gramatika za epidemiološke modele

$$N = \{SIR, SR, IR, RR, Contact, Recovery\}$$
$$\Sigma = \{+, -, \cdot, c, s, i, r\}$$
$$R = \left\{ \begin{array}{rcl} SIR & \to & SR;\ IR;\ RR \\ SR & \to & -Contact \\ IR & \to & Contact - Recovery \\ RR & \to & Recovery \\ Contact & \to & c \cdot s \cdot i \\ Recovery & \to & c \cdot i \end{array} \right\}$$
$$S = SIR$$

Odkrivanje diferencialnih enačb (DE)

Ciljne spremenljivke

- Niso več osnovne (opazovane) spremenljivke
- Temveč njihovi časovni odvodi

Dva pristopa

1. Numerično izračunamo časovne odvode, algebraične enačbe
2. Med ocenjevanjem parametrov numerična simulacija DE

Reference in implementacije

Odkrivanje enačb in deterministične gramatike

- Lagramge (Todorovski 1998; Todorovski in Džeroski 1997)
- `kt.ijs.si/~ljupco/ed/lagrange.html`

Verjetnostne gramatike

- ProGED (Brence in ost. 2021), merske enote (Brence in ost. 2023)
- `github.com/brencej/ProGED`