INF5830, H2009, Obigatorisk innlevering 2

Innleveringsfrist 4.11

1 Oppgave: Unære produksjoner i CKY

For bottom-up parsere, som CKY, har vi forutsatt at grammatikken er på CNF. For de ikke-leksikalske reglene skal det være nøyaktig 2 ikke-terminaler på høyresiden, eller mer findelt

- Ikke en
- Ikke flere enn to
- Ikke ingen

Vi skal se om vi kan løse opp på noen av disse forutsetningene.

Modifiser CKY-algoritmen til å takle grammatikker med unære produksjoner. Løsningen kan enten gjøres ved å modifisere pseudokoden gitt i J & M (figure 13.10, side 440 i min utgave = oppgave 13.3), eller ved å modifisere en implementasjon av algoritmen du selv har laget (for INF4820 ?).

Hint: Du må på den ene siden huske på at unære produksjoner kan itereres: *NP* $\rightarrow$ *Nom* og *Nom* $\rightarrow$ *PN*, og på den andre siden sikre deg at algoritmen terminerer.

2 Oppgave: "Dotted items" i CKY

Vi skl bruke "dotted items" (DIs) for å takle høyresider med mere enn to elementer. Til en regel som *VP* $\rightarrow$ *Verb NP PP* svarer følgende DI-er:

VP $\rightarrow$ •Verb NP PP

VP $\rightarrow$ Verb • NP PP

VP $\rightarrow$ Verb NP • PP

VP $\rightarrow$ Verb NP PP •

Disse er ikke regler. De er generert fra en regel og kan brukes som hjelpemiddel i parsing. Som eksempel, se på analysen

$[_{\text{VP}}\ [_{\text{Verb}}$ prefer$][_{\text{NP}}$ a flight$][_{\text{PP}}$to Houston$]]$

Deler av denne frasen svarer da til DI-ene som følger:

VP $\rightarrow$ Verb $\bullet$ NP PP : prefer

VP $\rightarrow$ Verb NP $\bullet$ PP : prefer a flight

VP $\rightarrow$ Verb NP PP $\bullet$: prefer a flight to Houston

Med andre ord svarer en DI på formen $A \rightarrow \beta \bullet \gamma$ til en streng som tilsammen utgjør β, og kombinert med en streng som utgjør γ, kan de danne en A.

Blant annet fordi de samme ordene kan forekomme flere ganger i en streng, har vi behov for å spesifisere hvilke deler av en streng en DI svarer til. Hvis vi bruker tall for posisjoner, kan vi skrive $\langle A \rightarrow \beta \bullet \gamma, i, j \rangle$ for å uttrykke at strengen fra i til j svarer til $A \rightarrow \beta \bullet \gamma$, og for eksempelet:

$\langle$VP $\rightarrow$ Verb $\bullet$ NP PP, $2, 3\rangle$

$\langle$VP $\rightarrow$ Verb NP $\bullet$ PP, $2, 5\rangle$

$\langle$VP $\rightarrow$ Verb NP PP $\bullet$, $2, 7\rangle$

DI-er med γ tom, altså på formen $A \rightarrow \beta \bullet$ er litt spesielle. Denne vil jo representere en frase av kategori A. Så det å si at en streng svarer til $A \rightarrow \beta \bullet$ medfører å si at den er en A. Slike DI-er vil vi kalle inaktive. De vil ikke "ta opp i seg noe mer". Mens en DI på formen $A \rightarrow \beta \bullet \gamma$ hvor γ ikke er tom, kaller vi aktiv. Den "jakter" på noe som kan fylle γ.

DI-er på formen $A \rightarrow \bullet \gamma$ er også spesielle. En slik svarer til den tomme strengen og kan slik sett forekomme overalt. Det varierer mellom forskjellige parsingalgoritmer om de gjør bruk av slike DI-er og hvilken rolle de spiller. En mulighet i en CKY-algoritme er å ta med diagonalfeltene $[i, i]$ i tabellen og fylle de med slike DI-er. I utgangspunktet skal da alle diagonalfelt innehold alle slike DI-er. Men vi kan alternativt bare legge til slike elementer når vi trenger dem, eller hoppe over bruken av dem. Tenk igjennom hvordan du løser dette i oppgaven!

Noen parsere kombinerer en BU-parsing med en TD-strategi for når slike elementer skal innføres. Det er prinsippet i Earley-algoritmen, som vi ikke skal gå inn på her.

Helt sentralt i parsing med hjelp av DI-er står det som i chartparsinglitteraturen kalles fundamentalregelen:

(1) Fra: $\langle A \rightarrow \beta \bullet C\, \delta, i, j \rangle$ og $\langle C \rightarrow \gamma \bullet , j, k \rangle$

kan en slutte: $\langle A \rightarrow \beta\, C \bullet \delta, i, k \rangle$

Denne kan erstatte følgende trekk i andre BU-parsere:

(2) Fra: $\langle B, i, j\rangle$, $\langle C, j, k\rangle$ og regelen $A \rightarrow B\,C$

kan en slutte: $\langle A, i, k\rangle$

Vi er nå klar til handling.

a Modifiser CKY-algoritmen til å bruke DI-er, og slik at trekket (2) erstattes av (1). Algoritmen skal kunne virke med grammatikker der det er to eller flere elementer på høyresiden. Algoritmen kan være i pseudokode eller implementert. Du kan bruke ideer fra seksjon 13.4.2 og 13.4.3 i J & M, men løsningen skal ligge nærmere CKY:

- Den skal bruke datastrukturen en todimensjonal tabell.
- Den skal være en ren BU-parser uten noen form for TD-avskjæring.

En av tingene du må tenke på, er hva som skal føre til innførsel av aktive kanter.

b Takler algoritmen din også unære regler? Hvis ikke, modifiser den til å gjøre det.

c Demonstrer algoritmen din på strengen

(3) *I prefer a flight to Houston*

med grammatikken i figur 13.1. Dette kan du gjøre ved å vise en ferdig tabell, der de ulike elementene i tabellen er nummerert i forhold til rekkefølgen de ble lagt inn i tabellen

3 Oppgave: Probabilistisk CNF

I kapittel 14 presenterer J & M en eneste probabilistisk CFG, i figur 14.1, og den er ikke på CNF. De presenterer også bare en parsingalgoritme for probabilistiske CFG-er, i figur 14.3, og den krever at grammatikken er på CNF. Her er det en avstand som må tettes. Vi må endre på minst en av grammatikken og algoritmen. I denne omgangen tar vi grammatikken, i neste oppgave algoritmen.

a Modifiser algoritmen for overgang til CNF fra kapittel 13 til å virke for PCFG (oppgave 14.2 i J& M).

b Omform grammatikken i figur 14.1 med algoritmen slik at du får en PCFG på CNF. Du behøver bare ta med nye regler og regler med endret sannsynslighet i besvarelsen.

c Vis hvordan algoritmen i figur 14.3 vil parse setning (3) med den modifiserte grammatikken. Dette kan du gjøre ved å vise en ferdig tabell, der de ulike elementene i tabellen er nummerert i forhold til rekkefølgen de ble lagt i tabellen.

4 Oppgave: Modifisert CKY for PCFG

a Modifiser algoritmen i figur 14.3 på liknende måte som du modifiserte CKY-algoritmen til å takle regler med ett eller flere elementer på høyresiden.

b Vis hvordan den modifiserte algoritmen vil parse setning (3) med den originale grammatikken fra figur 14.1. Dette kan du gjøre ved å vise en ferdig tabell, der de ulike elementene i tabellen er nummerert i forhold til rekkefølgen de ble lagt i tabellen. Sjekk at resultatet blir det samme som i forrige oppgave.

5 Oppgave: Shift-reduce parser og CNF

På forelesning gjennomgikk vi shift-reduce parser (SR-parser) i detalj. Kort repetisjon:

- Den forutsetter en grammatikk på CNF.
- Den har to datastrukturer REST(en av inputstrengen) og
- En STACK med ikke-terminaler.
- Den starter med tom STACK og strengen som skal undersøkes i REST.
- Den er suksessfull hvis den stopper med tom REST og hovedsymbolet alene på STACK
- Det er to typer trekk, SHIFT: hvis neste ord i REST er w, og det er en leksikalsk regel $A \rightarrow w$, så fjern w fra REST og legg A på toppen av STACK.

- REDUCE: Hvis det øverste elementet på stacken er C og det neste er B og det er en regel $A \rightarrow B\,C$, så fjern de to øverste elementene på stacken og legg A på toppen av stacken.

For generelle grammatikker er algoritmen ikkedeterministisk. Det vil være et valg mellom SHIFT og REDUCE. Det kan være flere alternative SHIFT hvis ordet har alternative tagger, osv. Det skal vi ikke se på her.

Vi skal se på om vi kan modifisere på kravet til CNF. Vi vil bruke "dotted items" DI-er. Vi vil bruke en liknende datastruktur som i standard SR-parser, men med den endringen at stacken er befolket med DI-er. SHIFT-trekket kan være nesten uendret, mens REDUCE-trekket vil erstattes av to trekk. Det ene vil fra en topp på STACK på formen $B \rightarrow \gamma \bullet$ og en regel $A \rightarrow B\,\delta$ kunne bytte toppen av stacken med $A \rightarrow B \bullet \delta$. Fyll ut detaljene i algoritmen. Den skal altså virke for grammatikker med to eller flere ikke-terminaler på høyresiden i ikke-leksikalske regler.

Lykke til!

-slutt