Athugun á fýsileika sjálfvirkrar vörpunar liðgerðargreiningu íslensks trjábanka yfir í venslagreiningu

Örvar Kárason

16. desember 2015

Í þessari ritgerð er lýst frumathugun á fýsileika þess að varpa sjálfvirkt liðgerðargreiningu íslenska trjábanka yfir í venslagreiningu. Lýst er fyrri aðferðum við samsvarandi varpanir fyrir ensku. Sagt er frá tilraunum með að beita þeim aðferðum til að varpa einföldum íslenskum setningum og skoðað er hvort eitthvað í framsetningaraðferðunum kynni að torvelda framkvæmdina. Eins og gefur að skilja þá er mikið ógert en útkoman lofar góðu að svo stöddu og framhald verkefnisins er fýsilegt.

1 Inngangur

Til eru nokkrar mismunandi formleg framsetningarkerfi til að lýsa formgerðum setninga. Þar er helst að nefna framsetningu sem byggir á stofnhlutum eða liðgerðum annarsvegar (Chomsky 1956) og svo venslum orða hinsvegar (Tesniere 1959). Með liðgerðagreiningu er myndað stigveldi liða en með venslagreiningu net sem sýnir innbyrðis vensl orðapara innan setningar.

Ýmislegt er þó sameiginlegt með þessum ólíku nálgunum á mállýsingu. Tesniere vísar til dæmis til oft til stofnhluta í greiningaraðferð sinni (Osborne 2013:269). Einnig er óhætt að segja að þessar tvær aðferðir hafi nálgast hvora aðra þónokkuð í gegnum árin. Viðbót X-liðakerfisins og INFL-liðarins við liðgerðargreininguna færa hana t.d. nær venslagreiningunni. Framsetning venslagreiningar viðheldur nú línulegri orðaröð en gerði það ekki áður, og venslin sjálf eru nú mörkuð. Þar sem þetta eru í raun eingöngu mismunandi nálganir á lýsingu setningarfræðilegrar greiningu þá ætti að vera hægt að varpa framsetningu af annarri tegundinni yfir í hina, og svo öfugt.

Verkefni þetta er frumathugun á fýsileika þess að varpa sjálfvirkt stofnhluta- eða liðgerðargreiningu hins Sögulega íslenska trjábanka (Wallenberg o.fl. 2011) yfir í venslagreiningu sem fylgir Universal Dependencies staðlinum (héreftir UD). Sagt er frá aðferðum sem beitt hefur verið við samsvarandi varpanir fyrir ensku. Lýst er svo tilraunum sem gerðar voru með að nýta þær til að varpa einföldum íslenskum setningum og skoðað er hvort eitthvað í framsetningaraðferðunum kynni að torvelda framkvæmdina.

1.1 Hvað þarf að vera til staðar svo vörpun sé möguleg?

Bakmállýsing vörpunarinnar verður að geyma nauðsynlegar upplýsingarnar fyrir markmállýsinguna. Annaðhvort eru þær þá beinlínis gefnar eða lesa má þær óbeint af framsetningaraðferðinni sjálfri. Í liðgerðargreiningunni eru undirskipun liðanna sýnd beint en höfuð liðarins eingöngu óbeint. Ef nauðsynlegar upplýsingar eru ekki gefnar þá þarf að vera hægt að fletta þeim upp eða greina þær út frá samgengi.

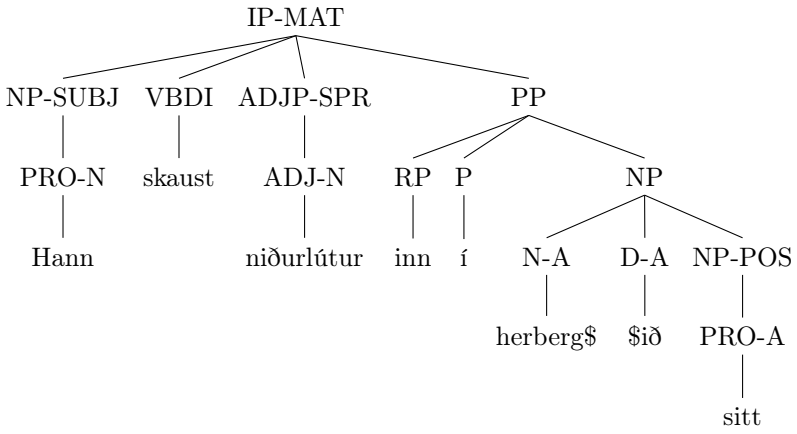
Samkvæmt Pál Uzonyi er hægt að varpa milli liðgerðargreiningar og venslagreiningar að því gefnu að eftirfarandi upplýsingar séu til staðar (2003:240-5, tilvitnun Klotz 2015:1009): (i) höfuð hluta, (ii) línuleg röðun eininda, (iii) flokkun/mörkun eininda, (iv) afstaða milli höfuða og undirskipaðra eininda (fyllilið eða viðhengi) og (v) eðli samtengingar (einindi + einindi; einindi + liður; liður + liður).

1.2 Bakmengið

Sögulegi íslenski trjábankinn (IcePaHC) er safn þáttaðra texta frá 12. til 21. aldar sem telja alls ein milljón lesmálsorða úr 61 textaheimild. Í verkefninu sem hér er lýst var látið nægja að horfa til tveggja texta frá 21. öld. Setningarfræðilega greiningin byggir á þáttunarskema Penn-trjábankans fyrir sögulega trjábanka (sjá Santorini 2010) og eru trén nokkuð flatari en venja er nú fyrir í fræðilegri setningarfræði.

Eins og sést á trjáhrískunni hér fyrir neðan þá bera liðgerðarmörkin í Penn-skemanu einnig upplýsingar um hlutverk liðana: -SBJ (frumlög), -OB1, -OB2, -OB3 (bein og óbein andlög), -LOC (staðarmerking), -TMP (tímamerking) o.sfrv.. Þessar upplýsingar koma að góðum notum. Einnig eru tómir liðir og tilvísanir notaðar til að sýna vensl milli fjarlægra liða. Ekki er þó hægt að fullyrða um að svo stöddu hvort öll nauðsynleg vensl séu beinlínis greinanleg í skemanu.

```
( (IP-MAT (NP-SBJ (PRO-N Hann-hann))
  (VBDI skaust-skjóta)
  (ADJP-SPR (ADJ-N niðurlútur-niðurlútur))
  (PP (RP inn-inn)
      (P í-í)
      (NP (N-A herberg$-herbergi)
  (D-A $ið-hinn)
  (NP-POS (PRO-A sitt-sinn)))))
  (ID 2008.MAMMA.NAR-FIC,.1464))
```



Hrísla 1: Liðgerðagreining úr IcePaHC

1.3 Markmengið

Hróður venslagreiningar hefur vaxið fiskur um hrygg seinustu árin vegna þess hversu vel sú tegund formgerðar hentar við sjálfvirka setningafræðilega greiningu, þ.e. þáttun í máltækniverkefnum. Flækjustig þáttunarinnar vex línulega miðað við inntakið, O(n) svo hún hröð og nýtist í verkefnum sem krefjast svörunar í rauntíma, s.s. sjálfvirkar þýðingar og svarvélar. Þá þykir hún einnig henta betur í máltækniverkefnum fyrir tungumál með tiltölulega frjálsa orðaröð.

Enginn venslagreindur íslenskur trábanki er til í dag. Setningafræðileg greining samfelldra texta er tímafrekt og snúið verkefni og því er kostnaður við gerð trjábanka hár. Oftar en ekki er vélræn greining nýtt við frumgreiningu til að flýta fyrir. En vörpun frá liðgerðagreiningu yfir í venslagreiningu eins og hér er til umræðu myndi ná enn lengra því hin handvirka liðgerðargreiningarvinna trjábankans myndi nýtast að mestu eða öllu leyti. Endanlegt markmið væri svo venslagreindur íslenskur trjábanki af þokkalegri stærð sem hægt væri að nýta til að þjálfa venslaþáttara fyrir íslensku.

Ákveðið var að nota UD venslaskemað sem er afrakstur verkefnis til að smíða slíkt skema óháðu ákveðnu tungumáli. Byggt er m.a. á hinu almenna Stanford venslamarkamengi (de Marneffe o.fl. 2014) og almennu málfræðilegu markamengi frá Google (Petro vo.fl. 2012).

2 Fyrri verkefni

Eins og áður sagði er nauðsynlegt að greina vensl milli orðapara þar sem annað yfirskipar hitt og kallast höfuð þess. Fyrri verkefni sem hafa fengist við vörpun frá liðgerðargreiningu yfir í venslagreiningu grundvallast öll á aðferð við að velja höfuð sem Magerman (1994) kynnti. Aðferðin hefur svo verið bætt af Collins (1999) sem og Yamada og Matsumoto (2003), Johansson og Nugues (2007).

Aðferðin byggir á höfuðgreiningarreglum sem eru notaðar til að velja eitt orð liðarins sem höfuð hans og skilgreina þannig vensl þess við hin orðin í liðnum. Reglurnar hafa eftirfarandi skema, þar sem fyrst kemur kemur mark liðarins, svo merki um hvort leitin frá vinstri til hægri eða öfugt (l/r) og svo mörk orðanna sem koma til greina í forgangsröð.

```
IP-MAT      r      VB.*   RD.*   BE.*   N.*
```

Penn2Malt vörpunartólið[1] (Nivre 2006) er útfærsla á aðferð Yamada og Matsumoto þar sem bætt er við greiningu á venslamörkum. Eins og nafnið gefur til kynna er vörpunin frá Penn-skemanu yfir í venslaskema MaltParser-þáttarans. Það hefur einnig verið notað fyrir kínverska Penn trjábankann.

Fyrir LTH vörpunartólið[2] bæta Johansson og Nugues (2007) hvorutveggja höfuðgreiningarreglurnar og gera venslamörkunina nákvæmari. Vörpunin er frá enska Penn-skemanu yfir í CoNLL-X venslaskemað.

Greiningin á venslamörkununum byggir a.m.k. á mörkun liðsins sem höfuðið er í, mörkun höfuðsins og mörkun yfirskipaða liðsins. Einnig nýtast upplýsingar hlutverka liðanna sem liðgerðamörkin bera í Penn-skemanu. Einfaldar þau mjög val á frumlagi og andlögum, og gera sérstaka greiningu á þeim óþarfa.

3 Aðferð

Gerð var tilraun til að nýta samsvarandi vörpunaraðferðir og þær sem lýst var hér á undan fyrir íslensku. Einföld Python skrifta[3] var útbúin í því augnarmiði og verður útfærsla hennar rakin í megindráttum. NLTK máltæknipakkinn (Bird, Loper og Klein 2009) var nýttur til að flýta fyrir þar sem þess var kostur.

Tree klasinn úr nltk.tree getur þáttað liðgerðartré samkvæmt Penn-skemanu. Liðirnir eru úrfærðir sem listar af listum eða strengjum (fyrir lesmálsorð) ásamt liðgerðarmarki fyrir listann. Skilgreindur var afleiddur klasi `IndexedTree` sem bætti við raðnúmerum fyrir lesmálsorðin og aðgerðum fyrir þau. Með númerinu er hægt að merkja hvaða orð er höfuð liðsins.

Klasinn `DependencyGraph` úr `nltk.parse` nýtist til að byggja venslanet. Skilgreindur var afleiddur klasi sem bætir við aðferð til að birta netið samkvæmt CoNLL-U venslaskemanu, `to_conllU()`. Venslamörkin fjörtíu sem notuð eru í UD verkefninu byggja á Universal Stanford markaskemanu (de Marneffe et al. 2014) með smávægilegum breytingum.

Vörpunin fer mestmegnis fram í klasanum `Converter` í fallinu `toDep()` (sjá viðauka). Það fer í gegnum alla liðina og myndar venslanet og á `_conllU_tag()` fallið til að finna mörkin fyrir orðin. Notast er við almennt markamengi sem byggir markamengi frá Google.[4] Samhliða því er tekinn saman listi yfir liði sem þarf að finna höfuð fyrir (þ.e. liði sem innihalda fleiri en eitt orð). Næst er

[1] Sjá http://stp.lingfil.uu.se/ nivre/research/Penn2Malt.html
[2] Sjá http://nlp.cs.lth.se/software/treebank_converter/
[3] Sjá https://github.com/OKarason/venzl/blob/master/converter.py
[4] Sjá https://code.google.com/p/universal-pos-tags

þessum lista raðað þannig að liðir sem eru neðar í trénu komu fyrst og svo er farið í gegnum hann og kallað á fallið `_select_head()` fyrir hvern lið. Höfuðin synda þannig upp tréð ef þannig má komast að orði. Valin höfuð eru næst tengd í venslanetinu og venslamörk valin fyrir þau með fallinu `_relation()`, höfuð efsta liðsins fær þó markið 'root'.

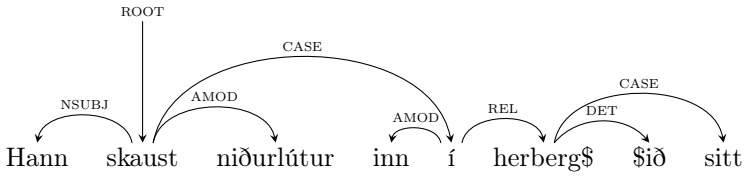
Fallið `_select_head()` sem velur höfuðin nýtir höfuðgreiningarreglur eins og lýst var hér á undan. Ef það fyrirfinnst regla fyrir viðkomandi lið en ekkert af orðamörkunum á við þá er fyrsta eða seinast orð liðarins sjálfkrafa valið (skv. r/l dálknum). En ef ekki er til regla fyrir liðinn þá er fyrsta orðið frá vinstri valið sem höfuð þess liðar.

3.1 Útkoman

Útkoma vörpunarinnar er `UniversalDependecyGraph` sem hægt er að vinna frekar með í NLTK. Einnig er hægt að prenta það út með `to_conllU()` fallinu. Í CoNLL-U sniðinu eru setningar eru skráðar með eitt orð ásamt greiningu í hverri línu og línubili á milli setninga. Undirstrik merkir að upplýsingar séu ekki til staðar. Hver lína inniheldur eftirfarandi dálka:

ID: Raðnúmer orðs, talið frá 1 við upphaf hverrar setningar.
FORM: lesmálsorð eða greinarmerki.
LEMMA: grunnmynd/stofn orðs.
UPOSTAG: orðhlutamark úr UD markamenginu.
XPOSTAG: orðhlutamark úr markamengi viðkomandi tungumáls.
FEATS: frekari málfræðileg greining.
HEAD: höfuð viðkomandi lesmálsorðs, annaðhvort ID-gildi eða núll (0).
DEPREL: UD venslamark sem sýnir afstöðu til höfuðs ('root' eff HEAD = 0).
DEPS: listi yfir undirskipuð orð og viðkomandi venslamerki (head-deprel pör).
MISC: aðrar upplýsingar.

```
# 2008.MAMMA.NAR-FIC,.1464
1    Hann        hann        PRON  PRO-N  _    2    nsubj   _    _
2    skaust      skjóta      VERB  VBDI   _    0    root    _    _
3    niðurlútur  niðurlútur  ADJ   ADJ-N  _    2    amod    _    _
4    inn         inn         RP    RP     _    5    amod    _    _
5    í           í           P     P      _    2    case    _    _
6    herberg$    herbergi    NOUN  N-A    _    5    rel     _    _
7    $ið         hinn        DET   D-A    _    6    det     _    _
8    sitt        sinn        PRON  PRO-A  _    6    case    _    _
```



Hrísla 2: Venslagreining skriftunnar á sömu setningar og í hríslu 1

3.2 Hvað er ógert?

Höfuðgreiningarreglurnar í `Convert` klasanum og útfærslan í heild sinni byggir á greiningu á tiltölulega litlu safni einfaldra setningu úr textum frá 21. öld í Sögulega íslenska trjábankanum. Ekki þyrfti að bæta við mörgum setningum áður en uppfæra þyrfti reglurnar og jafnvel laga skriftuna. Að sjálfsögðu þyrfti takast kerfisbundið og skipulega við útfærslu höfuðgreiningarreglanna, sem og reglanna sem velja mörkun vensla og orða.

Ekkert hefur verið hugað hér að flóknari setningarskipan, né heldur samteningar eða aðrar tengingar milli liði (hvað þá innan liða). Nýta þyrfti einnig upplýsingar í Penn-skemanu um tóma liði og tilvísanir í forvinnslu setninganna svo þær varðveitist. Það er svo sannarlega af nógu að taka enda var þessu verkefni eingöngu ætlað að verða einhverskonar frumathugun eins og áður segir.

Heimildir

2014. *Universal Dependencies.* Útgáfa 1. http://universaldependencies.github.io/docs/

Beatrice Santorini (2010). *Annotation manual for the Penn historical corpora and the PCEEC.* [Útgáfa 2. http://www.ling.upenn.edu/ beatrice/annotation/, sótt 16. des. 2015]

Bird, Steven, Edward Loper og Ewan Klein. 2009. *Natural Language Processing with Python.* O'Reilly Media Inc.

Chomsky, Noam. 1956. *The Logical Structure of Linguistic Theory.* Cambridge University Press, Cambridge.

Chomsky, Noam. 1965. *Aspects of the Theory of Syntax.* MIT Press, Cambridge.

Johansson, Richard og Nugues, Pierre. 2008. Extended Constituent-to-dependen cy Conversion for English. *Proceedings of NODALIDA 2007.* Tartú.

Klotz, Micheal. 2015. Foundations of Dependency and Valency Theory. Kiss, Tibor og Artemis Alexiadou (ristj.): *Syntax - Theory and Analysis. An International Handbook,* vol. 2, bls. 1004-1026. Walter de Gruyter, Berlín.

de Marneffe, Marie-Catherine o.fl. 2014. *Universal Stanford Dependencies: A cross-linguistic typology.* [http://nlp.stanford.edu/pubs/USD_LREC14_UD_revision.pdf, sótt 16. des. 2015]

Nivre, Joakim (2006). *Inductive Dependency Parsing.* Springer Verlag, New York.

Petrov, Slav, Das, Dipanjan, og McDonald, Ryan. 2012. A universal part-of-speech tagset. *Proceedings of LREC.*

Osborne, Timothy. 2013. A Look at Tesnière's Éléments through the Lens of Modern Syntactic Theory. *Proceedings of the Second International Conference on Dependency Linguistics (DepLing 2013)*, bls. 262–271. Matfyzpress, Prag.

Tesnière, Lucien. 1959. *Éléments de syntaxe structurale.* Klincksieck, París.

Uzonyi, Pál. 2003. Dependenzstruktur und Konstituenzstruktur. Vilmos Ágel o.fl. (ritstj.): *Dependenz und Valenz/1. Handbücher zur Sprach- und Kommunikationswissenschaft 1*, bls. 230-247. Walter de Gruyter, Berlín.

Wallenberg, Joel, Anton Karl Ingason, Einar Freyr Sigurðsson og Eiríkur Rögnvaldsson. 2011. *Icelandic Parsed Historical Corpus (IcePaHC).* [Útgáfa 0.9. http://www.linguist.is/icelandic_treebank, sótt 16. des. 2015]

Viðauki

```
def toDep(self, psd):
    #...
    t = IndexedTree.fromstring(psd)
    self.dg = UniversalDependencyGraph()
    #...
    for i in t.treepositions():
        if isinstance(t[i], Tree):
            if len(t[i]) == 1:
                #tag + lemma
                tag_list[nr] = t[i].label()
                t[i].set_id(nr)
            else:
                #phrase
                t[i].set_id(0)
                const.append(i)

        else:
            #form-lemma
            if '-' in t[i]:
                form, lemma = t[i].split('-', 1)
            else:
                form = lemma = t[i]
            self.dg.add_node({'address': nr, 'word': form, 'lemma': lemma,
                              'ctag': self._conllU_tag(tag_list[nr]),
                              'tag': tag_list[nr],
                              'feats': self._extract_features(tag_list[nr]),
                              'deps': defaultdict(list), 'rel': None})
            nr += 1

    # go through the constituencies (bottom up) and find their heads
    const.sort(key=lambda x: len(x), reverse=True)

    for i in const:
        self._select_head(t[i])

    for i in const:
        head_tag = t[i].label()
        head_nr = t[i].id()
        for child in t[i]:
```

```
            mod_tag = child.label()
            mod_nr = child.id()
            if head_nr == mod_nr and re.match( "IP-.+|QTP|CP-.+|FRAG", head_tag):
                self.dg.get_by_address(mod_nr).update({'head': 0,
                                                       'rel': 'root'})
                self.dg.root = self.dg.get_by_address(mod_nr)
            else:
                self.dg.get_by_address(mod_nr).update({'head': head_nr,
                    'rel': self._relation(mod_tag, head_tag)})
            if head_nr != mod_nr:
                self.dg.add_arc(head_nr, mod_nr)

    return self.dg
```