Issue 53: 1. Wat gebeurt?

multi [1 : l_1], other atom [l_1+1]
| case: one
atom[1], otheratom[l+1]
...
|
atom[1], other atom[2]

2. Wat is gewenst?

3. Wat is verklaring?
- generatie is top-down bepaald en is nooit < gen. parent
- unfold multi heeft „zijwaarts" effect
- vergelijkbaar met fold van multi: color. gen is symb. en dat heeft invloed op siblings

4. Welke (andere) scenario's kunnen er zijn?

multi [i : j]
| one, i<j
atom[i]

multi [i : j]
| one, i>j
atom[j]

multi [i : j]
| many, i<j
atom[i], multi [i+1 : j]
alle symbolen intact

multi [i : j]
| many, i>j
atom[i], multi [i-1 : j]
alle symbolen intact

5. Waar zit probleem in code?
generational-graph.rkt; 'annotate-level 1' roept 'annotate-unfolding'
als er geen interne multi's zijn (wat hier het geval is)

UPPSALA UNIVERSITET

Issue 53, vervolg: ①

6. Hoe werkt 'annotate-level!' ?

a) maakt partitie unlabeled level: conjuncten met 1 under of meerdere unders, i.e. nieuwe multi's

b) annoteert eerst alle nieuwe multi's en onthoudt conversie bestaande gen. naar nieuw symbool

c1. indien 0 nieuwe multi's: annoteer unfoldings zoals eerder

c2. indien nieuwe multi's: symboliseer verwante generaties

7. Welke fix?

in annotate-level!

- check eerst multi's verdwijnen i.p.v. bijkomen ✓
- manier om te checken: parent/neighbours in multi én heeft multi-kind ✓ geen
- achterhaal welk symbool verdwijnt / niet symbolisch meer zal zijn ✓
- onthoud gelijkstelling v. symbool met andere uitwerking ✓
- pas toe over volledig level

8. Kan apply-multi-mapping dit laatste aan?

'(apply-single! 1 e) was oude type call, maar ook

'(apply-single! sym1 sym2) moet kunnen (bv. er 2 multi's)

→ kan '(apply-single! sym 1) dan ook?

UPPSALA
UNIVERSITET

Issue 53, vervolg ②: 9. 'apply-single!' krijgt een oorspronkelijke (te herhalen) generatie en een subst-punt

a) ouder heeft enkele generatie: bepaal 'translation' adhv parent-gen, gesubst-gen, subst.
   └ opt-gelabelde node

...

=> niet dat veronderstelt dat we symboliseren (behalve var. namen)?

Addendum: jawel: (and gap (not (number? ...))) veronderstelt symbolische som

Key: generatie (incl. origin), value: enkel generatienummer, apply-single! is altijd voor een single-parent conjunct. (en die parent kan een multi zijn)
-> veronderstelling dat hier altijd een gen-range uit voortkomt als parent multi is!