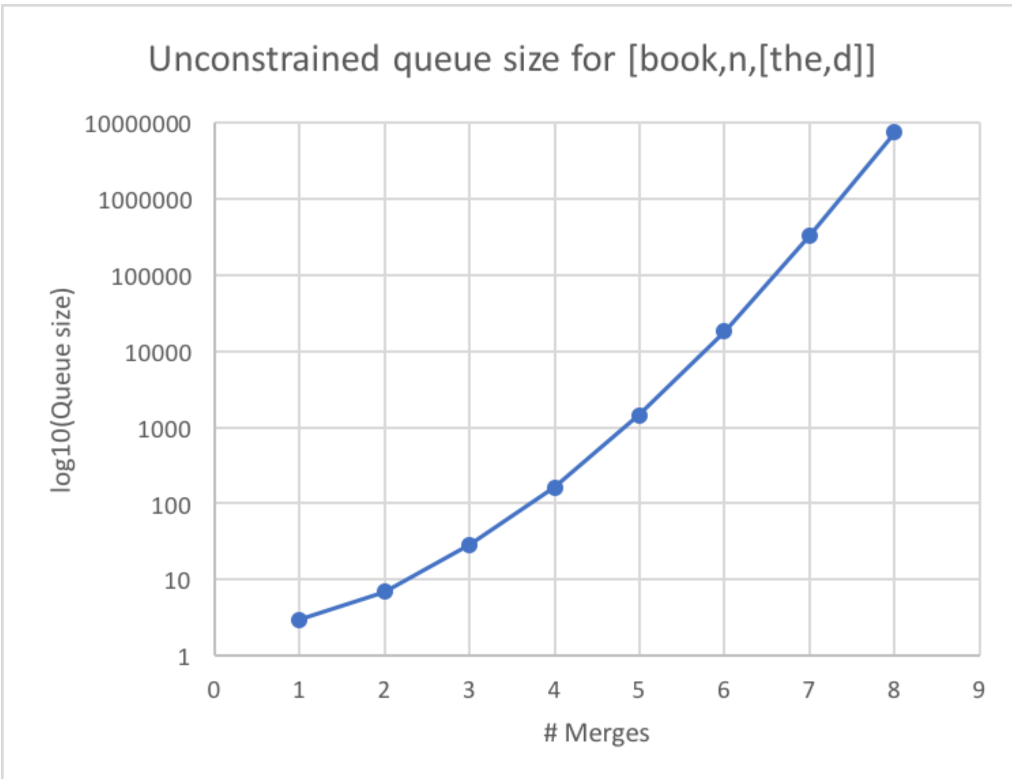
LING 696G: Lecture 3

Sandiway Fong

[*book*, n, [*the*, d]]



- Infinite supra-exponential expansion observed
- At each Merge step, in principle (subject to availability), we may have a choice of:
  1. External Set Merge (ESM)
  2. Internal Set Merge (ISM)
  3. External Pair Merge (EPM)
  4. Internal Pair Merge (IPM)

```
Example: [book,n!case,[the,d]]
SO: book, Input: [n!case,[the,d]]
1 esm SO: {book,n!case}, Input: [[the,d]]
1 1 ism SO: {book,{book,n!case}}, Input: [[the,d]]
1 1 1 dws SO: the, Input: [d]
1 1 1 1 esm SO: {the,d}, Input: []
1 1 1 1 1 ipm SO: <{the,d},the>, Input: []
1 1 1 1 1 1 uws SO: {book,{book,n!case}}, Input: [<{the,d},the>]
1 1 1 1 1 1 1 ism SO: {{book,n!case},{book,{book,n!case}}}, Input: [<{the,d},the>]
1 1 1 1 1 1 1 1 esm *mergeR SO: {{{book,n!case},{book,{book,n!case}}},<{the,d},the>}, Input: []
1 1 1 1 1 1 2 esm *mergeR SO: {{book,{book,n!case}},<{the,d},the>}, Input: []
1 1 1 1 2 ism SO: {the,{the,d}}, Input: []
1 1 1 1 2 1 uws SO: {book,{book,n!case}}, Input: [{the,{the,d}}]
1 1 1 1 2 1 1 ism SO: {{book,n!case},{book,{book,n!case}}}, Input: [{the,{the,d}}]
1 1 1 1 2 1 1 1 esm *mergeR SO: {{{book,n!case},{book,{book,n!case}}},{the,{the,d}}}, Input: []
1 1 1 1 2 1 2 esm *mergeR SO: {{book,{book,n!case}},{the,{the,d}}}, Input: []
1 1 1 1 3 uws SO: {book,{book,n!case}}, Input: [{the,d}]
1 1 1 1 3 1 epm *mergeR SO: <{the,d},{book,{book,n!case}}>, Input: []
1 1 1 1 3 2 ism SO: {{book,n!case},{book,{book,n!case}}}, Input: [{the,d}]
1 1 1 1 3 2 1 epm *mergeR SO: <{the,d},{{book,n!case},{book,{book,n!case}}}>, Input: []
1 1 1 1 3 2 2 esm *mergeR SO: {{{book,n!case},{book,{book,n!case}}},{the,d}}, Input: []
1 1 1 1 3 3 esm *mergeR SO: {{book,{book,n!case}},{the,d}}, Input: []
1 2 dws SO: the, Input: [d]
1 2 1 esm SO: {the,d}, Input: []
1 2 1 1 ipm SO: <{the,d},the>, Input: []
1 2 1 1 1 uws SO: {book,n!case}, Input: [<{the,d},the>]
1 2 1 1 1 1 ism SO: {book,{book,n!case}}, Input: [<{the,d},the>]
1 2 1 1 1 1 1 esm *mergeR SO: {{book,{book,n!case}},<{the,d},the>}, Input: []
1 2 1 1 1 2 esm *mergeR SO: {{book,n!case},<{the,d},the>}, Input: []
1 2 1 2 ism SO: {the,{the,d}}, Input: []
1 2 1 2 1 uws SO: {book,n!case}, Input: [{the,{the,d}}]
1 2 1 2 1 1 ism SO: {book,{book,n!case}}, Input: [{the,{the,d}}]
1 2 1 2 1 1 1 esm *mergeR SO: {{book,{book,n!case}},{the,{the,d}}}, Input: []
1 2 1 2 1 2 esm *mergeR SO: {{book,n!case},{the,{the,d}}}, Input: []
1 2 1 3 uws SO: {book,n!case}, Input: [{the,d}]
1 2 1 3 1 epm *end SO: <{the,d},{book,n!case}>
1 2 1 3 2 ism SO: {book,{book,n!case}}, Input: [{the,d}]
1 2 1 3 2 1 epm *mergeR SO: <{the,d},{book,{book,n!case}}>, Input: []
1 2 1 3 2 2 esm *mergeR SO: {{book,{book,n!case}},{the,d}}, Input: []
1 2 1 3 3 esm *unlabeled SO: {{book,n!case},{the,d}}, Input: []
```

- Finite expansion observed with restrictions (TBD)
- There's only one way to assemble:

```
<{the, d}, {book, n}>
```

LIs:[book,n!case,[the,d]] Derivation #1			
Step	**Branch**	**Op**	**SO**
1	-	-	book
2	1	esm	{book,n!case}
3	2	dws	the
4	1	esm	{the,d}
5	3	uws	{book,n!case}
6	1	epm	<{the,d},{book,n!case}>
Spellout heads: [the]			
Final output: [the]			

What restrictions make sense?

Research Program:

- Let's entertain my hopeful hypothesis that basic assumptions about grammar + 3rd factor constraints will be sufficient to make Free Merge viable …

Root R

Categorizer k

- all roots must be categorized
- all categorizers must categorize (exactly once)
- locality

Possible cases:

{R, k}	
{k, {R, XP}}	k c-commands R

Illicit cases:

<R, k>	Root invisible to k
<k, R>	k cannot categorize
{k, {x, R}}	no intervening head x

Restrictions from grammar

- Lexicon:
  - roots: *friend*, *john*, *like*
  - categorizers: n, d, v*, v
- Merge Restrictions:
  - (a) roots must be categorized (as soon as possible)
  - (b) each categorizer must find its root (with no intervening heads)
  - (c) categorizers can only categorize once
    - e.g. $*$`{c, {R, {c, R}}}`
      formed with only two LIs, c and R (R=root, c=categorizer)

    n has unvalued features uCase, uTheta

What 3rd factor restrictions make sense?

- Infinite Loops:
  - caused by internal Merge (set and pair) only
  - enlarges current syntactic object (SO) without bound
  - Note: phase-based labeling is not sufficient to limit the damage: i.e. SO can be enlarged indefinitely before reaching v* or C.
- **Hypothesis**:
  - Suppose FL always (attempts to) block infinite loops (**computational minimalism**)
- Implementation:
  - potential infinite loops are **always** blocked at the **first opportunity**
  - a pattern π is a sequence of Merge operations; e.g. ISM(a), ISM(b), ISM(a), ISM(b)
  - use an IM pattern repetition detector: $*\pi\pi = *\pi^{2+}$
  - there is only one kind of repetition permitted (i.e. none),
    e.g. no rule $*\pi^{5+}$ (i.e. *you can repeat up to 4 times but not more*)

What 3rd factor restrictions make sense?

Example:

- consider {a,b} with Internal Set Merge (ISM)
- block repetitive patterns π^{2+} (which all lead to infinite loops)

- e.g. {a,b} = ISM(a)*2 => {a,{a,{a,b}}} =*=> {a,{a,..{a,b}..}}

- e.g. {a,b} = ISM(a,b)*2 => {b,{a,{b,{a,{a,b}}}}} =*=> {b,{a,..{b,{a,{a,b}}}..}}

Other Infinite Loops

- IM pattern: $*\pi\pi = *\pi^{2+}$
- There are more complicated types of infinite loops we can choose to block…
- **Example**:

1. {a,b} (ESM)
2. {a, {a,b}} (ISM of a)
3. {{a,b}, {a,{a,b}}} (ISM of {a,b})
4. {{a,{a,b}}, {{a,b},{a,{a,b}}}} (ISM of {a,{a,b}})
5. {{{a,b},{a,{a,b}}},{{a,{a,b}},{{a,b},{a,{a,b}}}}} (ISM of {{a,b},{a,{a,b}}})

   and so on…

- this not a simple pattern, see below (but it can be blocked programmatically):
  - ISM(a) ISM({a,b}) ISM({a,{a,b}}) ISM({{a,b},{a,{a,b}}})

Restrictions from grammar

- Yet another kind: **lemmas**: (can be applied proactively)
  - Let uF = unvalued feature F
  - **rule**: unvalued features must be valued
  - e.g. can't External Pair Merge (EPM) β[uF] to α forming <β, α>, where β is an adjunct
  - since β is no longer accessible to operations, adjunct with uF can never get valued

What 3rd factor restrictions make sense?

No duplicate SOs

- In just three steps, Internal Set Merge (ISM) with blind selection can create **duplicates SOs**
- **Example**:
  in {x,{ x, y}}
  by selecting either copy of x,
  ISM can create same SO { x,{ x,{ x, y}}}

- Derivation Tree:

Start: SO: x, Input: [y]

1.ESM, SO: {x, y}, Input: []

1.1.ISM, SO: {x,{x, y}}, Input: []

1.1.1.ISM, SO: {x, {x, {x, y}}}, Input: []

1.1.2.ISM, SO: {{x, y},{x,{x, y}}}, Input: []

1.1.3.ISM, SO: {x, {x, {x, y}}}, Input: []

1.1.4.ISM, SO: {y, {x, {x, y}}}, Input: []

1. 2.ISM, SO: {y, {x, y}}, Input: []

1. 2. 1.ISM, SO: {y, {y, {x, y}}}, Input: []

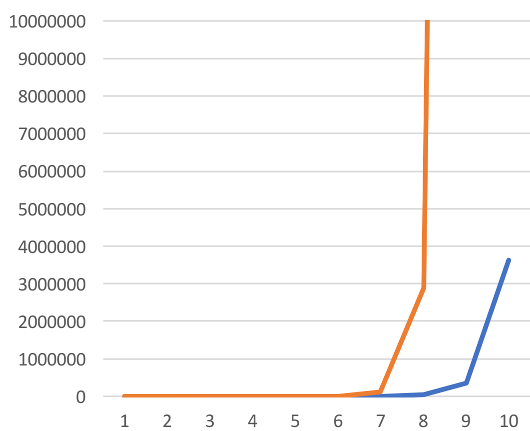
1.2.2.ISM, SO: {{x, y}, {y ,{x, y}}}, Input: []

1.2.3.ISM, SO: {x, {y, {x, y}}}, Input: []

1.2.4.ISM, SO: {y, {y, {x, y}}}, Input: []

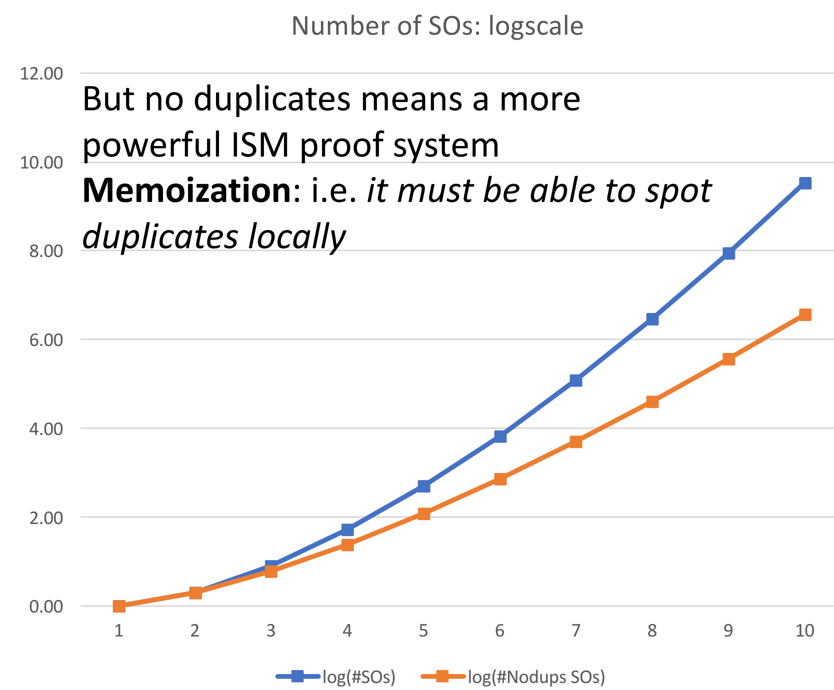
Duplicates vs. No Duplicates

- Eliminating duplicate SOs:
  - X-axis: number of Set Merges (SM)
  - Y-axis ☟ : number of SOs built
  - Y-axis ☞: **log** number of SOs built
  - Orange line: allowing duplicates
  - Blue line: with no duplicates



But no duplicates means a more powerful ISM proof system

Memoization: i.e. *it must be able to spot duplicates locally*



Restrictions from grammar

- It’s tempting to limit the range of Internal Merge (IM); but such stipulations require justification
- **Example**: Internal Set Merge (ISM)
  - SELECT (proper) sub-SO
- means:
  - SO:{a, {b, c}}

1. {a, {b, c}} NO
2. a
3. {b, c}
4. b
5. c

- For Pair Merge (PM), in <x,y>, x is invisible to SELECT
  - assume same SELECT is used for both IM operations
- **Example**: SELECT sees SO as {b,<c,{d,e}>} below:
  - SO: <a,<z,{b,<c,{d,e}>}>>
  - <a,<z,{b,<c,{d,e}>}>> NO
  - <z,{b,<c,{d,e}>}> NO
  - {b,<c,{d,e}>} NO
  - b
  - <c,{d,e}>
  - {d,e} NO
  - d
  - e

Restrictions from grammar

- Internal Set Merge (ISM)
  - SO x={..{..x'..}..}
  - SELECT x' a (proper) sub-SO of x
  - produce {x',{..{..x'..}..}}
- Internal Pair Merge (IPM)
  - SELECT x' a (proper) sub-SO of x
  - produce <{..{..x'..}..},x'>
  - **what about?** <x',{..{..x'..}..}>
    - think we want to ban this (lack of prominence for selected sub-SO)
    - also would permit flip-flop between adjunct (invisible) and non-adjunct (visible)

- Example:
  - *the professor of John's that he always praises* (Cecchetto & Donati, 2015:71)
  - our proposal using IPM:
    - <professor,[of [John ['s ~~professor~~]]]> counts as a head for relabeling
    - SELECT chooses *professor* from John's professor
    - IPM allows it to be a "new" head and not violate C&D's constraints on relativization (and avoids their Late Merge solution)
  - Other (different) definitions of IPM:
    - Richards (2009) etc.
    - EKS (2016)

Resulting system

Notation:
Set Merge (SM): {α,β}; Pair Merge (IM): <α,β>, α is adjunct. α!F means α has unvalued feature F. Workspace (WS) = Syntactic Object (SO) + unprocessed Lexical Items (LIs). Initial WS: LIs = a list of heads ([...]) to be processed in order. 1st(LIs) denotes the first element. Initial SO = 1st(LIs). Sub-WS: a sub-list defines a sub-WS. Compute a sub-SO that substitutes for the sub-list in the higher WS.

Derivation Tree (DT) examples:	
Each line encodes one step of the DT. Formats: (1) `op SO Input` (non-leaf step) (2) `op *Reason SO Input` (blocked) (3) `op ✓ SO []` (convergent step)	Explanation: op = previous operation resulting in SO; Input = LIs remaining Reason = a restriction (or end) blocking SO (end: labeled SO computed but unvalued features still present)
esm SO: {{friend,n!case},like}, Input: [v*] 1▶ism SO: {{friend,n!case},{{friend,n!case},like}}, Input: [v*] 2▶esm SO: {{{friend,n},like},v*}, Input: [] (Click on ▶ to extend the derivation one step.)	Explanation: from SO {{friend,n!case},like}, there are two possible ways to proceed: (1) ism of {friend,n!case} (object shift), or (2) esm of v*.
SO: book, Input: [n!case,[d,the]] 1 epm *pmR SO: <n!case,book>, Input: [[d,the]] 2 epm *mergeR(ext) SO: <book,n!case>, Input: [[d,the]] 3 esm SO: {book,n!case}, Input: [[d,the]]	Explanation: (greyed out = blocked derivation) from SO book, epm of n!case is blocked (*) in 1 and 2 by restrictions pmR and mergeR, resp.; however, esm of n!case, option 3, is permitted.

Operations:	**Operation**	**Restriction**	**Restriction**	**Restriction**
esm: External SM {SO,1st(LIs)}	epm: External PM <SO,1st(LIs)> or <1st(LIs),SO>	pmR: *<α[uF],β> no unvalued features (uF) within adjunct α	dup: duplicate SOs eliminated	xmit: transmit INFL failure phase head, e.g. C or v*, transmits inflectional (INFL) features to lower head X, triggering Agree(X, β), β a goal
ism: Internal SM {α,SO}, α a sub-SO of SO	dws: Down WS begin computing sub-list	ipmR: disallow <{x,y},x> from {x,y}	loop: IM repetitions disallowed e.g. *ism α ism α, or *ism α ism β ism α ism β	cii: CI interface crash uninterpretable formulae: *<nP,nP>, *<dP,dP> (cf. <dP,nP>)
ipm: Internal PM (IPM) <SO,α>, α a sub-SO of SO	uws: Up WS end sub-list computation; SO to higher WS	mergeR: apply lexical restrictions e.g. a categorizer must be the 1st SM'ed head above a Root	unlabeled (SO): labeling algorithm: non-weak head X labels{X,YP}, R (root) and T weak weak head W labels {W,YP} if strengthened; X labels {X,R}; <ϕ,ϕ> labels {XP,{Y,ZP}} assuming identical ϕ-features for XP and Y, strengthened Y labels {Y,ZP} XP labels {XY,YP} if YP moves Stipulation: n* strengthens R in {n*,{R,XP}}	