Stack-Based Agree

Sandiway Fong
University of Arizona
sandiway@email.arizona.edu

Jason Ginsburg
Osaka Kyoiku University
jginsbur@cc.osaka-kyoiku.ac.jp

ELSJ 7th International Spring Forum, Doshisha University, Kyoto, April 19, 2014

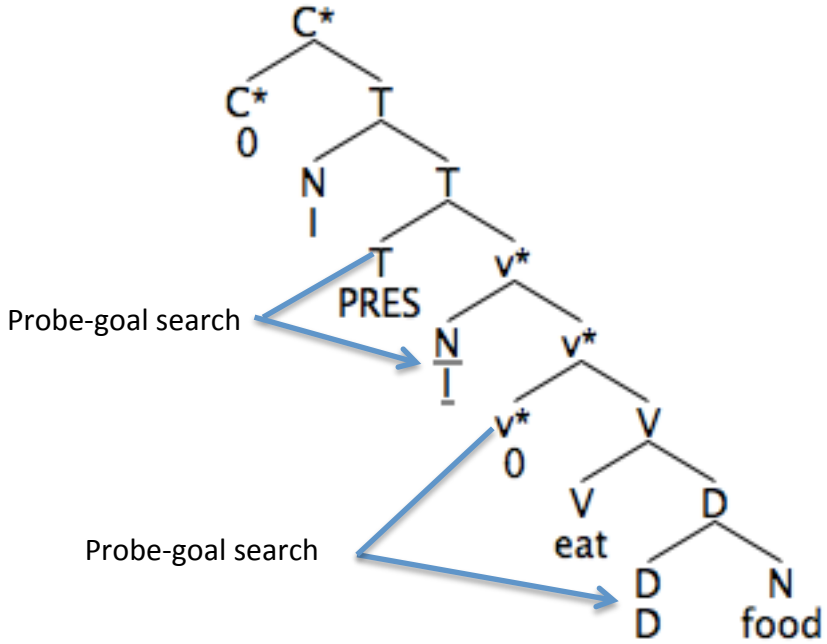
Background

- **Agree**
- Agree is a complex operation in which an uninterpretable feature (uF) functions as a probe that searches through its search domain for a matching interpretable feature (iF).
- Probe-goal search is computationally expensive:
  - search requires that a probe evaluate all of the features in its search domain until it finds a matching goal, if present.
- We propose that a novel "search-free" stack mechanism for Agree relations can replace standard probe-goal search.
- We develop a method of evaluating the cost of:
  - Standard probe-goal search vs. Stack-based Agree
- We attempt to demonstrate that the stack-based Agree method is more economical than probe-goal search.

Agree

- Consider how Agree works.
  - Assume that v* assigns Case (Chomsky 2001, etc.)
  - v* must check the V label, the V head, and the D label, before it finds D (assuming that D has uCase)*
  - T checks the v* label, and N (T doesn't have too far to search in this case)

(1) I eat food.



We use Bare-Phrase structure style trees. For example, a maximal projection labeled v equals vP, etc. Unpronounced copies are underlined.

Agree

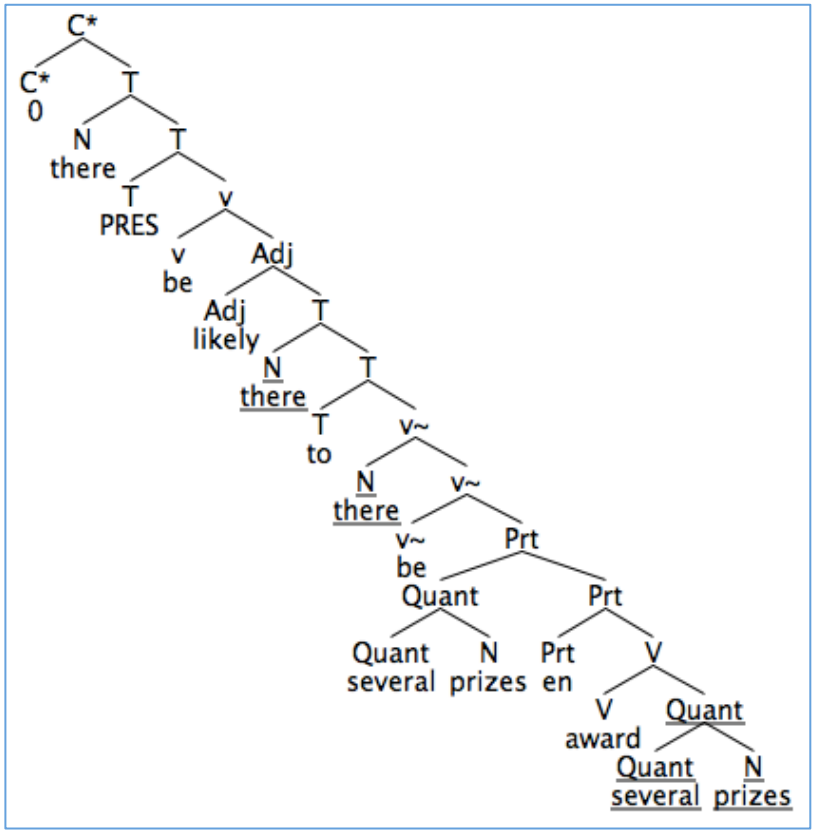
- Consider how Agree works.

(2) There are likely to be several prizes awarded. (Modified version of 4bi)

(4bi) There are likely to be awarded several prizes. (Chomsky 2001:7)

- (2) with thematic extraction, is more natural than (4bi).

- Some assumptions that we make.
- Expletive *there* is Merged at a position below T (cf. Richards 2007, Sobin To Appear*)*
- Chomsky (2001) suggests that Thematic extraction (TH/EX) is a PF operation.
- Deal (2009) and Sobin (To Appear) have analyzed TH/EX as being part of narrow syntax.
  - We follow this, with some deviations.
- Prt has an Edge Feature (EF)* – needed to get the correct word order.
- There is a type of v, referred to as v~ (Deal 2009, Sobin To Appear) that occurs in passive and unaccusative constructions.
- v~ has a split EPP feature (Sobin, To Appear) essentially consisting of [uTheta] and [uN/D]
  - [uTheta] must be checked by a theta-role bearing NP/DP. This determines where the NP/DP that undergoes TH/EX is visible
  - [uN/D] must be checked by a DP/NP (Merge of expletive is OK)



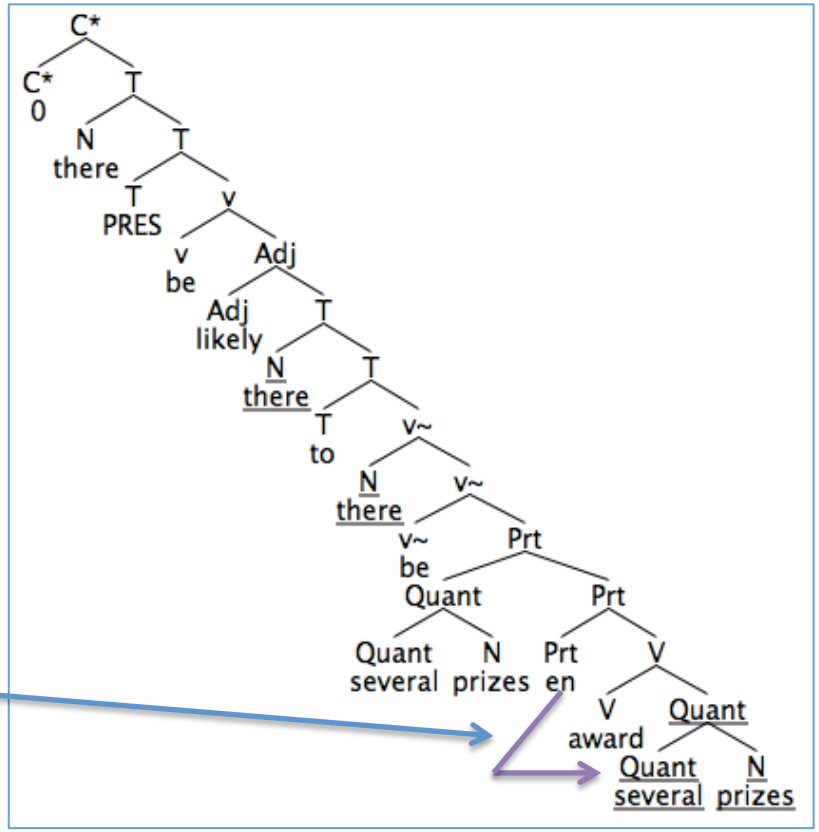
* An Edge Feature (EF) can be likened to an EPP feature (Chomsky 2006)

Agree

- Consider how Agree works (we present the main Agree relations, and not necessarily in the correct order).

  (2) There are likely to be several prizes awarded. (Modified version of 4bi)

- Prt Agrees with 'several prizes' – uPhi on Prt are checked

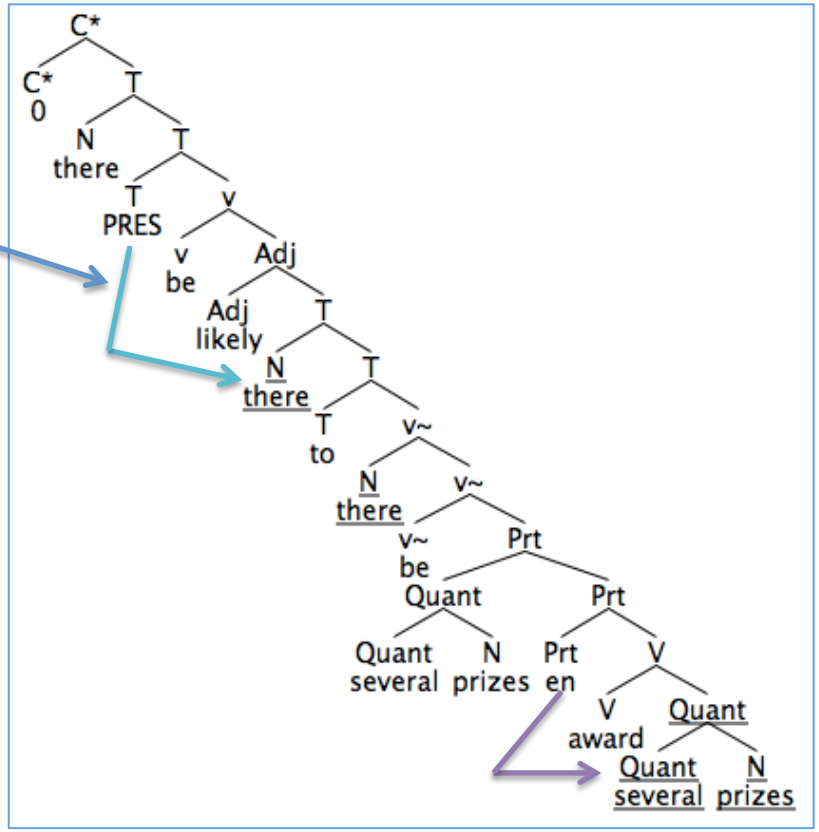


Agree

- Consider how Agree works.

  (2) There are likely to be several prizes awarded. (Modified version of 4bi)

- T Agrees with Expl – uPhi on T check uPerson on Expl

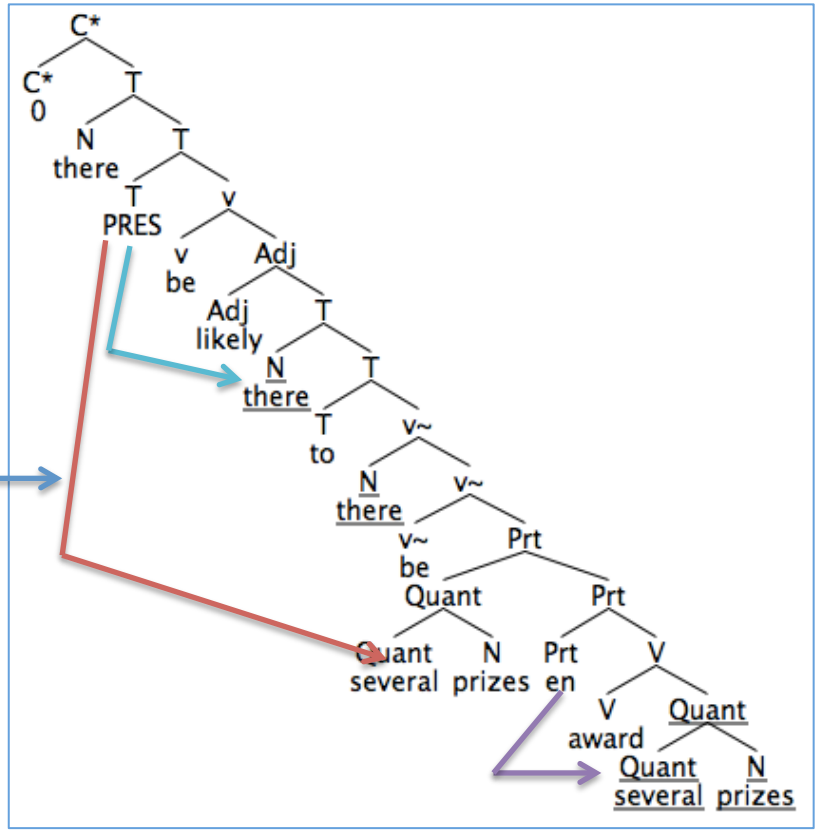


Agree

- Consider how Agree works.

  (2) There are likely to be several prizes awarded. (Modified version of 4bi)

- T Agrees with 'several prizes' – uCase on object checked as a result of phi-feature agreement

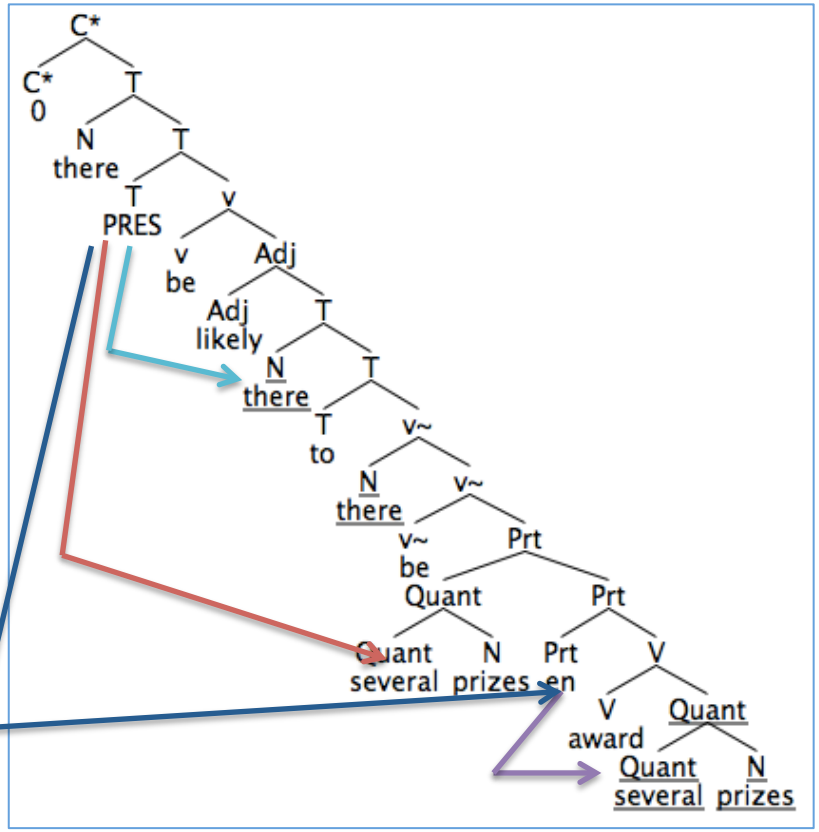


Agree

- Consider how Agree works.

  (2) There are likely to be several prizes awarded. (Modified version of 4bi)

- T Agrees with Prt – uCase on Prt is checked as a result of phi-feature agreement

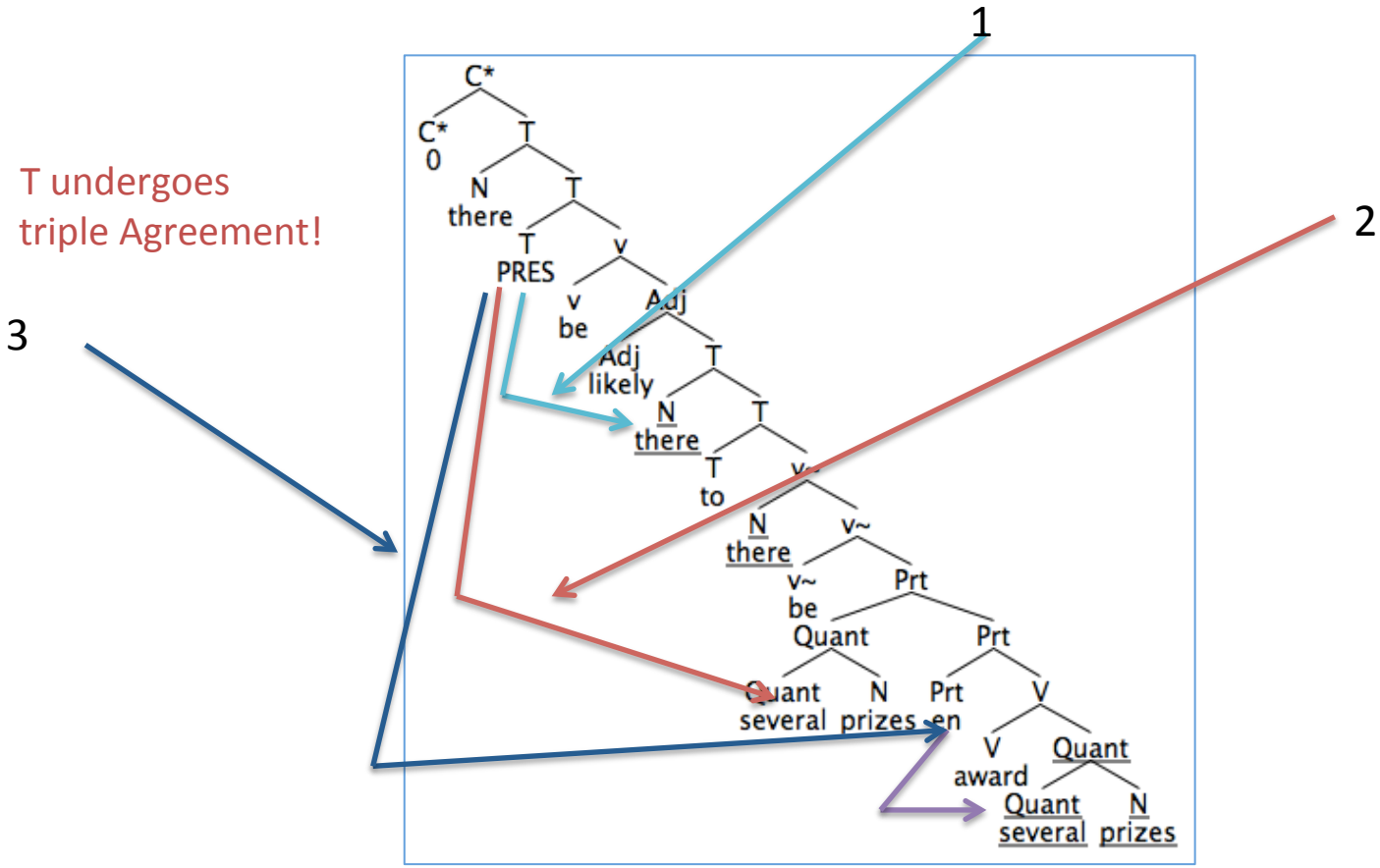


Agree

- Consider how Agree works.

  (2) There are likely to be several prizes awarded. (Modified version of 4bi)

- T undergoes triple Agreement!

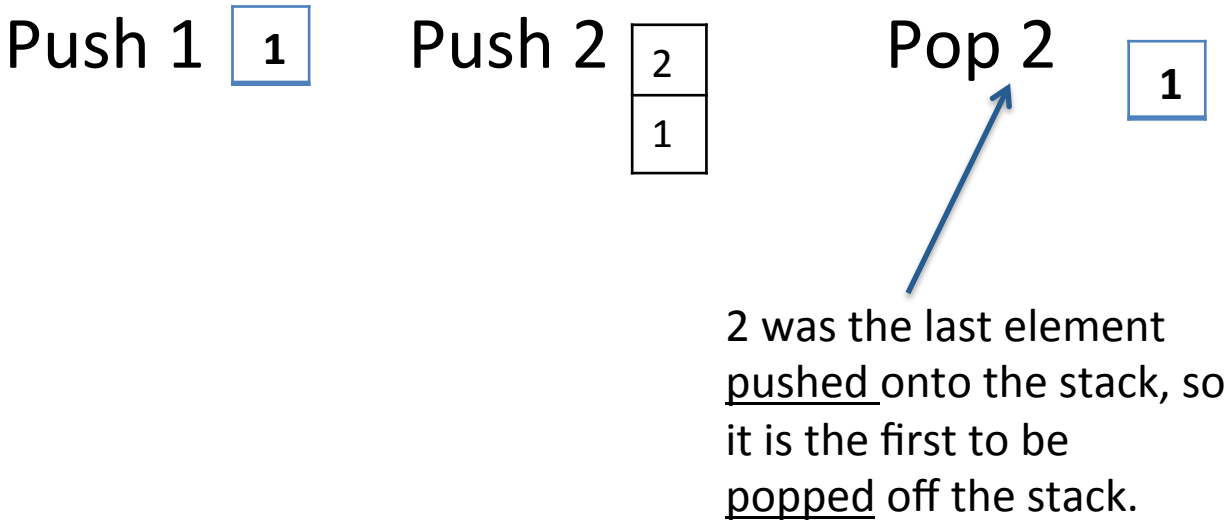


Proposals

- We propose that there is a simpler search mechanism than typical probe-goal search.
  - There is no need to check all of the labels within a search domain.
    - Search is confined to a stack.

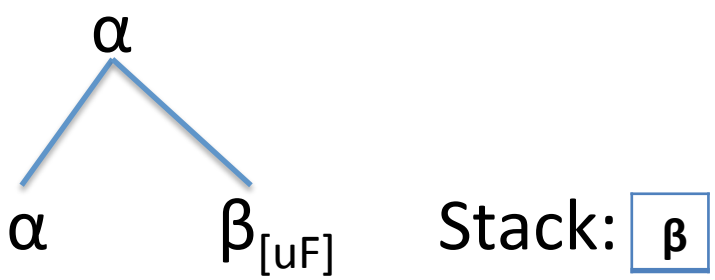
Proposals

- Stack.
  - A stack is a last-in, first out list.



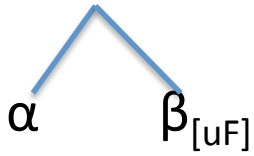
Proposals

- Stack.
  - Merge α and β, α is the label.
  - β has an uninterpretable feature uF.
  - B, with a uF, is pushed onto the stack.



Proposals

- Stack.
  - α



Stack: **β**

  - The first element that a probe can view is the element at the top of the stack, the TOS (Top of the Stack).
  - When a stack is populated by more than one element, the TOS is the first element that is visible to a probe.
  - If all the features of the TOS are checked, then the TOS is popped off, and the next element in the stack, if present, moves to the top position, thus becoming available to future Agree operations.
  - The TOS must be popped before any other elements in the stack can be popped.

Derivations

- We examine the derivations of

  (1) I eat food.

  (2) There are likely to be several prizes awarded.

- We demonstrate how these derivations work with the stack mechanism
- We compare the costs of these derivations as calculated via stack-based Agree and typical probe-goal search

Derivations

- We examine the derivations of

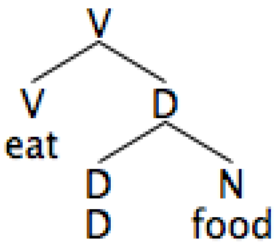
  (1) I eat food.

  (2) There are likely to be several prizes awarded.

Derivations

(1) I eat food.

- Merge V & D (object)
- Object has uCase
  - Push Object onto Stack



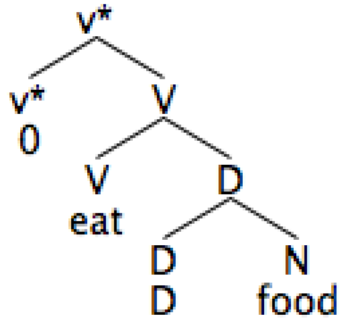
Stack:

$[_D$ D food$_{[uCase]}]$

Derivations

(1) I eat food.

- Merge v* & V
- uPhi on v* searches into stack
- uPhi on v* find the TOS *food*
- *v** and *food* Agree
  - *uPhi on v** are checked and *uCase* on food is checked
- *Food* is popped from the stack

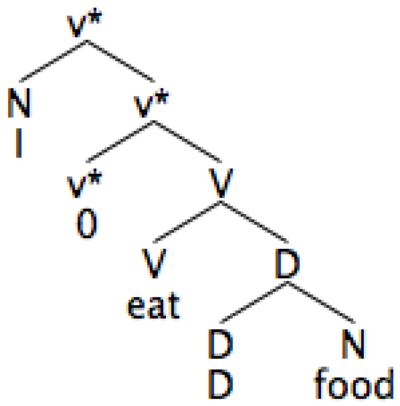


Stack:
Pop *food*

Derivations

(1) I eat food.

- Merge v* & N (subject)
- Subject has uCase
  - Push subject onto Stack



Stack:

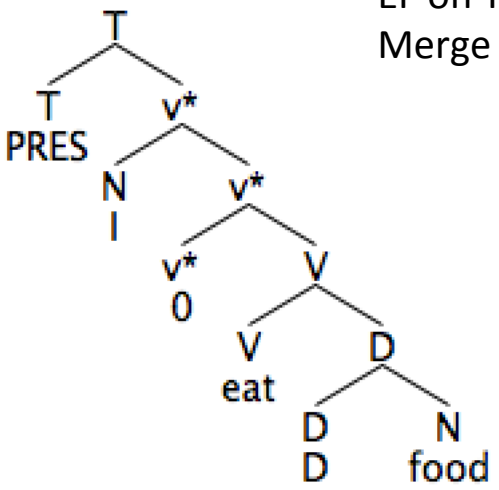
Push *I*

$[_N\ I_{[uCase]}]$

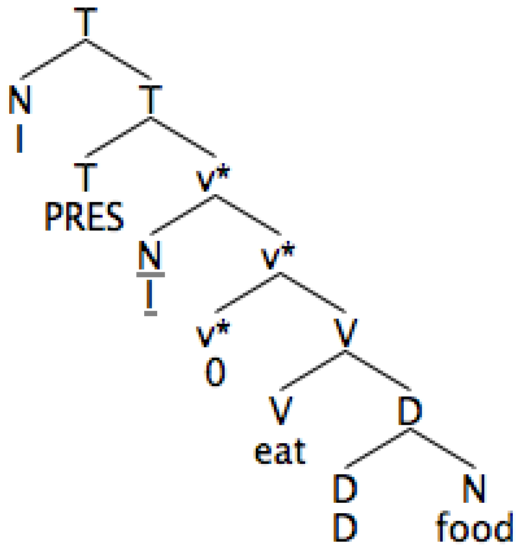
Derivations

(1) I eat food.

- Merge T & v*
- uPhi on T searches into stack
- uPhi finds the TOS *I*
- *T* and *I* Agree
  - *uPhi on T* are checked and *uCase* on I is checked
  - EF (Edge Feature) on T forces *I* to Merge with T
- *I* is popped from the stack



EF on T forces *I* to Merge with T

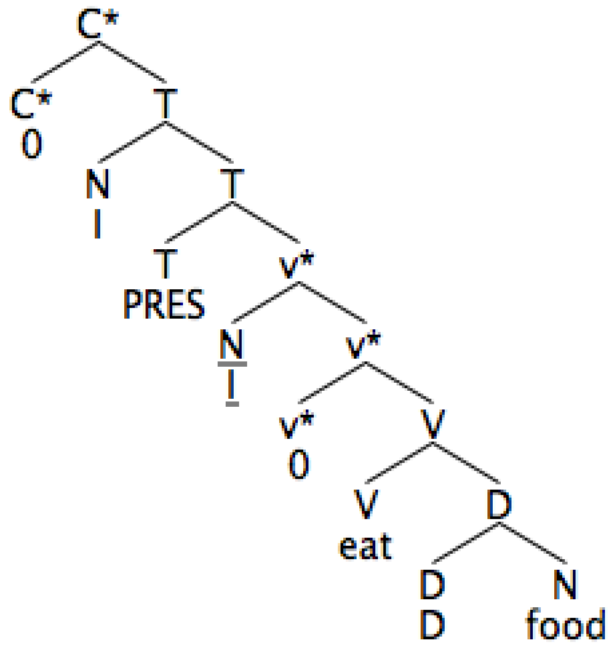


Stack:
Pop *I*

Derivations

(1) I eat food.

- Merge C* & T



Derivations and Cost

- The total cost of computing a derivation =
  - (1) # Merges + (2) # Agrees
- (1) is uncontroversial
  - Since the models compute the same tree, we have equal # Merges and cost.
- (2) is unclear
  - We have two models of Agree: one stack-based and a (standard) search-based model.
  - Issue: how to fairly compare the models from the point of cost when they use different mechanisms?
  - The most economical approach should be favored from the perspective of Minimalism.

Derivations and Cost

Stack Model

Operation	Cost
Agree	+1 for each Agree operation
Push, Pop	+1 for each Push operation (Push onto stack) or Pop (Pop off stack)
Stack Depth	+1 each time that the stack has an element in it

- We calculate Cost of the Stack Model in 3 ways:
  1. Agree + Push, Pop + Stack Depth
  2. Agree + Stack Depth
  3. Agree + Push, Pop

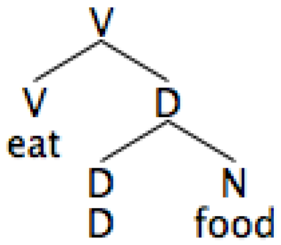
Typical Probe-goal Search

Operation	Cost
Search	+1 for each node checked

- Note that there are more operations associated with the Stack than with typical probe-goal search.
  - We are not sure if this is meaningful.
- It may be if there is a stack, the operations of push and pop come for free.
- It is not clear if there should be a cost associated with stack depth.
- It may also be that the typical probe-goal search operation has various sub-components.
- At this point, the Cost value is most relevant.

Derivations and Cost

(1) I eat food.



Stack: **[food$_{[uCase]}$]**

Stack Models: Tables 1-3

Table 1: Agree + Push, Pop + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	0	0	0
Push, Pop	**0**	**+1**	**1**
Stack Depth	**0**	**+1**	**1**
Totals	0	2	2

Table 2: Agree + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	0	0	0
Stack Depth	**0**	**+1**	**1**
Totals	0	1	1

Table 3: Agree + Push, Pop

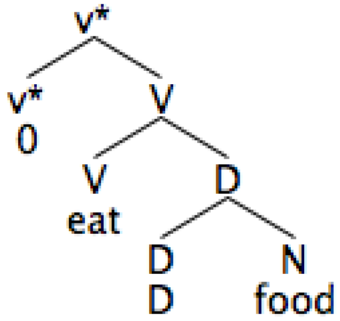
Operation	Previous Cost	Current Cost	Total Cost
Agree	0	0	0
Push, Pop	**0**	**+1**	**1**
Totals	0	1	1

Table 4: Probe-goal Model

Operation	Previous Cost	Current Cost	Total Cost
Search	0	0	0

Derivations and Cost

(1) I eat food.



Stack:

Stack Models: Tables 1-3

Table 1: Agree + Push, Pop + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	0	+1	1
Push, Pop	**1**	**+1**	**2**
Stack Depth	**1**	**0**	**1**
Totals	2	2	4

Table 2: Agree + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	0	+1	1
Stack Depth	**1**	**+1**	**2**
Totals	1	1	3

Table 3: Agree + Push, Pop

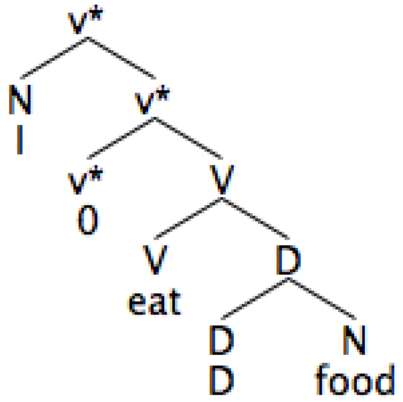
Operation	Previous Cost	Current Cost	Total Cost
Agree	0	+1	1
Push, Pop	**1**	**+1**	**2**
Totals	1	1	3

Table 4: Probe-goal Model

Operation	Previous Cost	Current Cost	Total Cost
Search	0	+4	4

Derivations and Cost

(1) I eat food.



Stack: $[I_{[uCase]}]$

Stack Models: Tables 1-3

Table 1: Agree + Push, Pop + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	1	0	1
Push, Pop	**2**	**+1**	**3**
Stack Depth	**1**	**+1**	**2**
Totals	4	2	6

Table 2: Agree + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	1	0	1
Stack Depth	**2**	**+1**	**3**
Totals	3	1	4

Table 3: Agree + Push, Pop

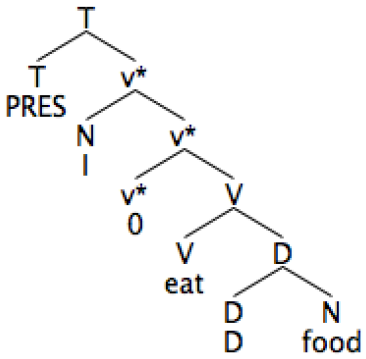
Operation	Previous Cost	Current Cost	Total Cost
Agree	1	0	1
Push, Pop	**2**	**+1**	**3**
Totals	3	1	4

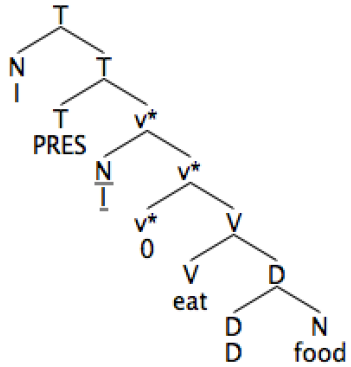
Table 4: Probe-goal Model

Operation	Previous Cost	Current Cost	Total Cost
Search	4	0	4

Derivations and Cost

(1) I eat food.





Stack:

Stack Models: Tables 1-3

Table 1: Agree + Push, Pop + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	1	+1	2
Push, Pop	**3**	**+1**	**4**
Stack Depth	**2**	**0**	**2**
Totals	6	2	8

Table 2: Agree + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	1	+1	2
Stack Depth	**3**	**0**	**3**
Totals	4	1	5

Table 3: Agree + Push, Pop

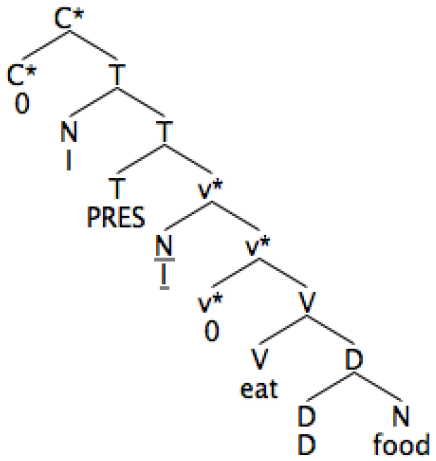
Operation	Previous Cost	Current Cost	Total Cost
Agree	1	+1	2
Push, Pop	**3**	**+1**	**4**
Totals	4	2	6

Table 4: Probe-goal Model

Operation	Previous Cost	Current Cost	Total Cost
Search	4	+2	6

Derivations and Cost

(1) I eat food.



Stack:

Stack Models: Tables 1-3

Table 1: Agree + Push, Pop + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	2	0	2
Push, Pop	**4**	**0**	**4**
Stack Depth	**2**	**0**	**2**
Totals	8	0	8

Table 2: Agree + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	2	0	2
Stack Depth	**3**	**0**	**3**
Totals	5	0	5

Table 3: Agree + Push, Pop

Operation	Previous Cost	Current Cost	Total Cost
Agree	2	0	2
Push, Pop	**4**	**0**	**4**
Totals	6	0	6

Table 4: Probe-goal Model

Operation	Previous Cost	Current Cost	Total Cost
Search	6	0	6

Derivations and Cost

(1) I eat food.

- The costs for this simple sentence are <u>similar</u>.

Stack:

Stack Models: Tables 1-3

Table 1: Agree + Push, Pop + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	2	0	2
Push, Pop	**4**	**0**	**4**
Stack Depth	**2**	**0**	**2**
Totals	8	0	8

Table 2: Agree + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	2	0	2
Stack Depth	**3**	**0**	**3**
Totals	5	0	5

Table 3: Agree + Push, Pop

Operation	Previous Cost	Current Cost	Total Cost
Agree	2	0	2
Push, Pop	**4**	**0**	**4**
Totals	6	0	6

Table 4: Probe-goal Model

Operation	Previous Cost	Current Cost	Total Cost
Search	6	0	6

Agree

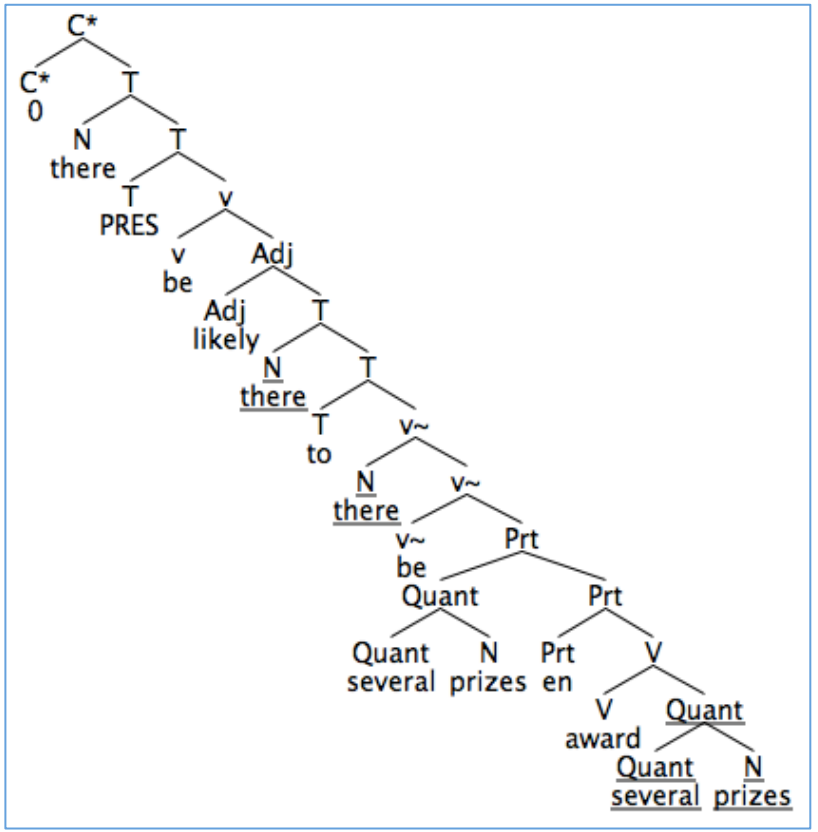
- Consider how Agree works.

(2) There are likely to be several prizes awarded. (Modified version of 4bi)

(4bi) There are likely to be awarded several prizes. (Chomsky 2001:7)

- (2), with thematic extraction, is more natural than (4bi).

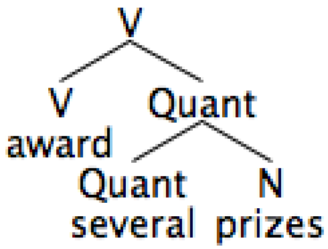
- Some assumptions that we make.
- Expletive *there* is Merged at a position below T (cf. Richards 2007, Sobin To Appear*)*
- Chomsky (2001) suggests that Thematic extraction (TH/EX) is a PF operation.
- Deal (2009) and Sobin (To Appear) have analyzed TH/EX as being part of narrow syntax.
  - We follow this, with some deviations.
- Prt has an Edge Feature (EF)* – needed to get the correct word order
- There is a type of v, referred to as v~ (Deal 2009, Sobin To Appear) that occurs in passive and unaccusative constructions.
- v~ has a split EPP feature (Sobin, To Appear) essentially consisting of [uTheta] and [uN/D]
  - [uTheta] must be checked by a theta-role bearing NP/DP. This determines where the NP/DP that undergoes TH/EX is visible
  - [uN/D] must be checked by a DP/NP (Merge of expletive is OK)



* An Edge Feature (EF) can be likened to an EPP feature (Chomsky 2006)

Derivations and Cost

(2) There are likely to be several prizes awarded.



Stack: [several prizes$_{[uCase]}$]

Stack Models: Tables 1-3

Table 1: Agree + Push, Pop + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	0	0	0
Push, Pop	**0**	**+1**	**1**
Stack Depth	**0**	**+1**	**1**
Totals	0	2	2

Table 2: Agree + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	0	0	0
Stack Depth	**0**	**+1**	**1**
Totals	0	1	1

Table 3: Agree + Push, Pop

Operation	Previous Cost	Current Cost	Total Cost
Agree	0	0	0
Push, Pop	**0**	**+1**	**1**
Totals	0	1	1

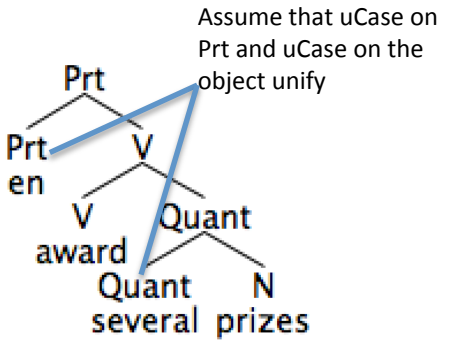
Table 4: Probe-goal Model

Operation	Previous Cost	Current Cost	Total Cost
Search	0	0	0

Derivations and Cost

(2) There are likely to be several prizes awarded.

Stack Models: Tables 1-3



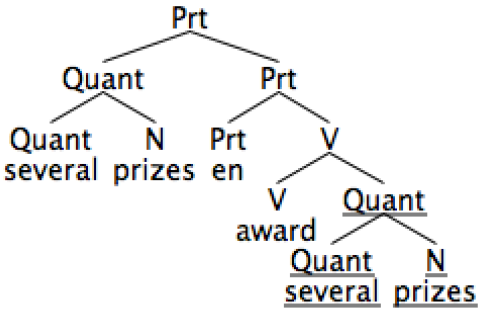


Table 1: Agree + Push, Pop + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	0	1	1
Push, Pop	**1**	**0**	**1**
Stack Depth	**1**	**1**	**2**
Totals	2	2	4

Table 2: Agree + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	0	+1	1
Stack Depth	**1**	**+1**	**2**
Totals	1	1	3

Table 3: Agree + Push, Pop

Operation	Previous Cost	Current Cost	Total Cost
Agree	0	+1	1
Push, Pop	**1**	**0**	**1**
Totals	1	1	2

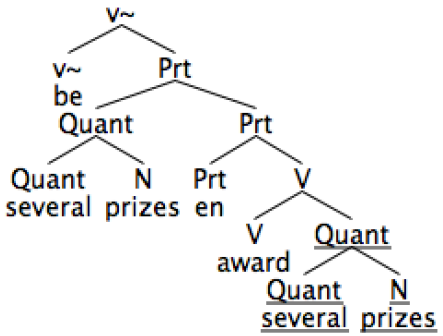
Table 4: Probe-goal Model

Operation	Previous Cost	Current Cost	Total Cost
Search	0	+4	4

Stack: **[several prizes$_{[uCase]}$]**

Derivations and Cost

(2) There are likely to be several prizes awarded.



Stack: [several prizes$_{[uCase]}$]

Stack Models: Tables 1-3

Table 1: Agree + Push, Pop + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	1	0	1
Push, Pop	**1**	**0**	**1**
Stack Depth	**2**	**+1**	**3**
Totals	4	1	5

Table 2: Agree + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	1	0	1
Stack Depth	**2**	**+1**	**3**
Totals	3	1	4

Table 3: Agree + Push, Pop

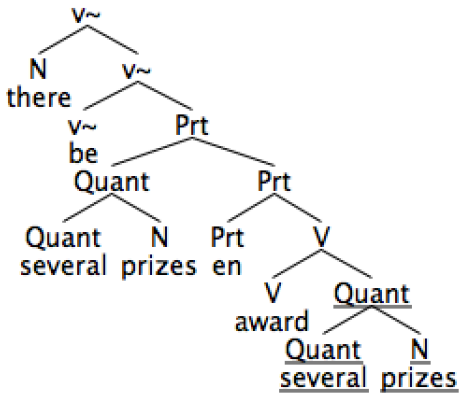
Operation	Previous Cost	Current Cost	Total Cost
Agree	1	0	1
Push, Pop	**1**	**0**	**1**
Totals	2	0	2

Table 4: Probe-goal Model

Operation	Previous Cost	Current Cost	Total Cost
Search	4	0	4

Derivations and Cost

(2) There are likely to be several prizes awarded.



Stack:

there
[several prizes$_{[uCase]}$]

Stack Models: Tables 1-3

Table 1: Agree + Push, Pop + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	1	0	1
Push, Pop	**1**	**+1**	**2**
Stack Depth	**3**	**+1**	**4**
Totals	5	2	7

Table 2: Agree + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	1	0	1
Stack Depth	**3**	**+2**	**5**
Totals	4	2	6

Table 3: Agree + Push, Pop

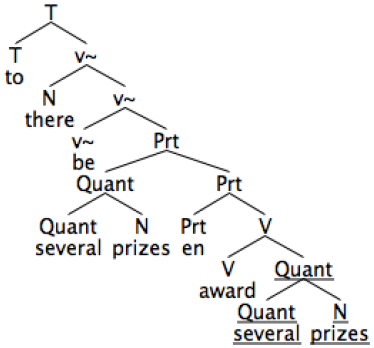
Operation	Previous Cost	Current Cost	Total Cost
Agree	1	0	1
Push, Pop	**1**	**+1**	**2**
Totals	2	0	3

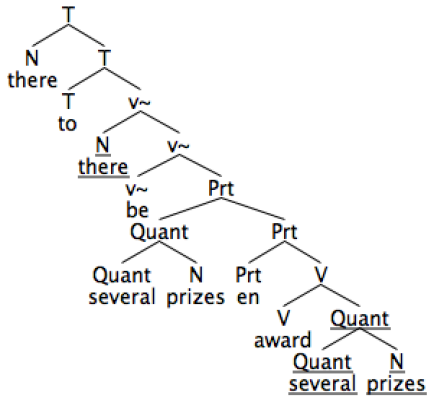
Table 4: Probe-goal Model

Operation	Previous Cost	Current Cost	Total Cost
Search	4	0	4

Derivations and Cost

(2) There are likely to be several prizes awarded.





Stack Models: Tables 1-3

Table 1: Agree + Push, Pop + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	1	+1	2
Push, Pop	**2**	**0**	**2**
Stack Depth	**4**	**+2**	**6**
Totals	7	3	10

Table 2: Agree + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	1	+1	2
Stack Depth	**5**	**+2**	**7**
Totals	6	3	9

Table 3: Agree + Push, Pop

Operation	Previous Cost	Current Cost	Total Cost
Agree	1	+1	2
Push, Pop	**2**	**0**	**2**
Totals	3	1	4

Table 4: Probe-goal Model

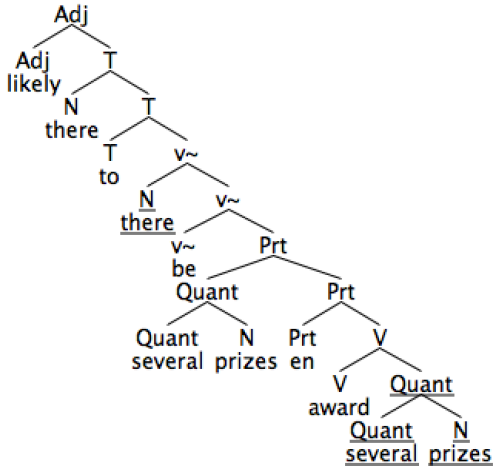
Operation	Previous Cost	Current Cost	Total Cost
Search	4	+2	6

Stack:

there
[several prizes$_{[uCase]}$]

Derivations and Cost

(2) There are likely to be several prizes awarded.



Stack:

there
[several prizes$_{[uCase]}$]

Stack Models: Tables 1-3

Table 1: Agree + Push, Pop + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	2	0	2
Push, Pop	**2**	**0**	**2**
Stack Depth	**6**	**+2**	**8**
Totals	10	2	12

Table 2: Agree + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	2	0	2
Stack Depth	**7**	**+2**	**9**
Totals	9	2	11

Table 3: Agree + Push, Pop

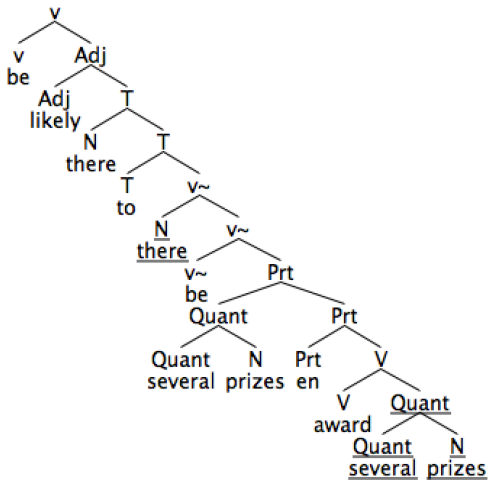
Operation	Previous Cost	Current Cost	Total Cost
Agree	2	0	2
Push, Pop	**2**	**0**	**2**
Totals	4	0	4

Table 4: Probe-goal Model

Operation	Previous Cost	Current Cost	Total Cost
Search	6	0	6

Derivations and Cost

(2) There are likely to be several prizes awarded.



Stack:

there
[several prizes$_{[uCase]}$]

Stack Models: Tables 1-3

Table 1: Agree + Push, Pop + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	2	0	2
Push, Pop	**2**	**0**	**2**
Stack Depth	**8**	**+2**	**10**
Totals	12	2	14

Table 2: Agree + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	2	0	2
Stack Depth	**9**	**+2**	**11**
Totals	11	2	13

Table 3: Agree + Push, Pop

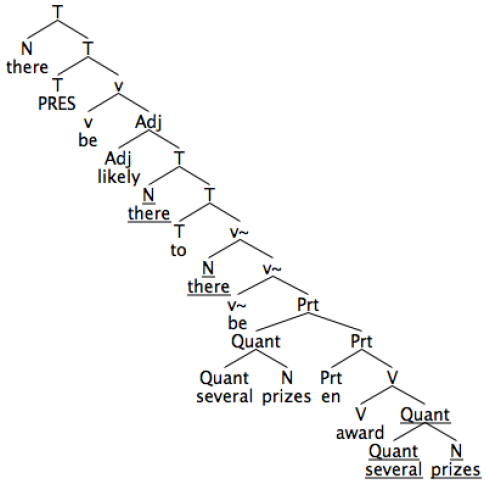
Operation	Previous Cost	Current Cost	Total Cost
Agree	2	0	2
Push, Pop	**2**	**0**	**2**
Totals	4	0	4

Table 4: Probe-goal Model

Operation	Previous Cost	Current Cost	Total Cost
Search	6	0	6

Derivations and Cost

(2) There are likely to be several prizes awarded.



Stack: Pop *there*, Pop *several prizes*

Stack:

Stack Models: Tables 1-3

Table 1: Agree + Push, Pop + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	2	+2	4
Push, Pop	**2**	**+2**	**4**
Stack Depth	**10**	**0**	**10**
Totals	14	4	18

Table 2: Agree + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	2	+2	4
Stack Depth	**11**	**0**	**11**
Totals	13	2	15

Table 3: Agree + Push, Pop

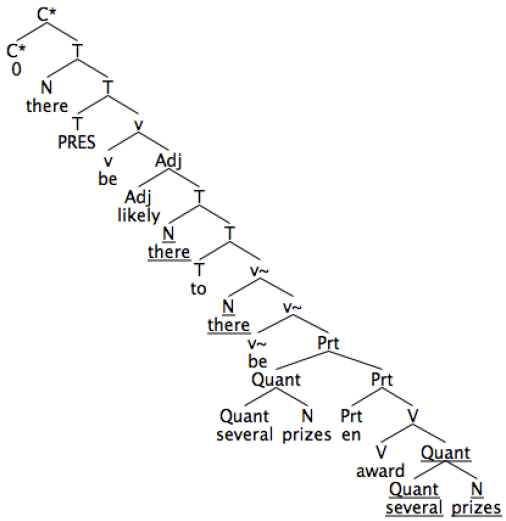
Operation	Previous Cost	Current Cost	Total Cost
Agree	2	+2	4
Push, Pop	**2**	**+2**	**4**
Totals	4	4	8

Table 4: Probe-goal Model

Operation	Previous Cost	Current Cost	Total Cost
Search	6	+17	23

Derivations and Cost

(2) There are likely to be several prizes awarded.



Stack:

Stack Models: Tables 1-3

Table 1: Agree + Push, Pop + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	4	0	4
Push, Pop	**4**	**0**	**4**
Stack Depth	**10**	**0**	**10**
Totals	18	0	18

Table 2: Agree + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	4	0	4
Stack Depth	**11**	**0**	**11**
Totals	15	0	15

Table 3: Agree + Push, Pop

Operation	Previous Cost	Current Cost	Total Cost
Agree	4	0	4
Push, Pop	**4**	**0**	**4**
Totals	8	0	8

Table 4: Probe-goal Model

Operation	Previous Cost	Current Cost	Total Cost
Search	6	+17	23

Derivations and Cost

(2) There are likely to be several prizes awarded.

- The Stack model has a lower cost.
- Table 1 is most conservative, and Table 3 is least conservative
- Note that calculating Stack Depth greatly increases the cost.
- Multiple Agree is simpler with the Stack Model, since a probe only peers into a stack, and does not have to look through all intervening nodes in a tree.

Stack Models: Tables 1-3

Table 1: Agree + Push, Pop + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	4	0	4
Push, Pop	**4**	**0**	**4**
Stack Depth	**10**	**0**	**10**
Totals	18	0	18

Table 2: Agree + Stack Depth

Operation	Previous Cost	Current Cost	Total Cost
Agree	4	0	4
Stack Depth	**11**	**0**	**11**
Totals	15	0	15

Table 3: Agree + Push, Pop

Operation	Previous Cost	Current Cost	Total Cost
Agree	4	0	4
Push, Pop	**4**	**0**	**4**
Totals	8	0	8

Table 4: Probe-goal Model

Operation	Previous Cost	Current Cost	Total Cost
Search	6	+17	23

Conclusion

- We have proposed a Stack-based derivational model.
- The probe-goal model and stack model seem to perform similarly (with a possible slight advantage to the stack-model) for simple sentences
- The stack model performs better than the probe-goal model for complex sentences, even in cases in which a very conservative approach is taken to calculating cost.

Type of sentence	Stack-based model vs. Probe-goal Model
simple	Similar costs under the conservative approaches, stack model has lower costs for less conservative approaches.
complex	Stack based model has a lower cost.

Conclusion

- Stack vs. Probe-goal Search
- Which is more economical?
- We argue that the stack is more economical in terms of actual cost and in terms of the general Minimalist Program goal to eliminate unnecessary search; i.e., economy.
  - Search in the stack-based model is confined to the stack. There is no need to check irrelevant nodes.

- More work is needed to determine exactly how to best calculate the cost of a derivation.

References

Chomsky, Noam. 1995. *The Minimalist Program*. MIT Press, Cambridge, MA: 1-52.

Chomsky, Noam. 2001. Derivation by phase. In *Ken Hale: A life in Language*, ed. by M. Kenstowicz, 1-52. Cambridge, MA: MIT Press.

Chomsky, Noam. 2006. On Phases. In *Foundational issues in linguistic theory; essays in honor of Jean-Roger Vergnaud*, ed. by Freidin, Robert, Otero, Carlos and Maria-Luisa Zubizaretta, 133-166. Cambridge, MA: MIT Press.

Deal, Amy Rose. 2009. The origin and content of expletives: Evidence from "selection". Syntax 12:285-323.

Richards, Marc. 2007. Object shift, phases, and transitive expletive constructions in Germanic. In Linguistics *Variation Yearbook 6,* ed. by Pica, Pierre, Rooryck, Johan, and Jeroen Van Craenenbroeck. Amsterdam: John Benjamins.

Sobin, Nicholas. To Appear. TH/EX, agreement and Case in expletive sentences. *Syntax*.