X-bar theory continued...

(1) XP
   Spec      X'
   specifier  YP      X'
   adjunct    X      ZP
   head       complement

Substitution and adjunction

Two kinds of movement: **Substitution** and **Adjunction**.

**Substitution**: Replaces an empty (but existing) position with a moved element.

Passives:
(2) a. [IP e [r I [VP was [VP solved the problem]]]
    b. [IP the problem, [r I [VP was [VP solved t]]]]

Wh-questions:
(3) a. I wonder… [CP e [C C[+Q] [CP John bought what]]]
    b. I wonder… [CP what [C C[+Q] [CP John bought t]]]

(4) XP
   ZP_1  X'
   X  YP
   ...t...

**Adjunction**: Creates a new position by attaching to an existing node.

In (5a) there is only one real XP node; in (5b) there is only one real X' node.

The fact that two are written is a **convention**. Maybe closer to this:

(6) ZP, XP
    Spec  X'
    X  ...

Recall the tree concepts of **dominance** and **sisterhood**.
In (6) and (5a): Spec and X’ are sisters. ZP has no sister.
XP dominates ZP, Spec, X’ (and X and …)

In (4), ZP and X’ are sisters,
XP dominates ZP, X’ (and X and YP)

Topicalization:

(7) I know… CP
    Spec  C'
    C (that)
    IP
    NP, this problem
   Spec
   I
   VP
   solve
   t_i

Extraposition and Heavy NP shift are also adjunction movements, right-adjointing to VP.

**Head movement** (movement of terminal categories) is also adjunction, where the moving head adjoins to the target head.

(8) C
    C'
    IP
    Spec
    I'
    VP
    t_i
    ...

(although: we will later need to make use of the notation convention used in (5))
Speaking of adjunction… We can also think of attachment to X’ this way:

\[
(9) \quad \text{VP} \\
\quad \text{V}' \\
\quad \text{PP} \\
\quad \text{Adv} \quad \text{quickly} \\
\quad \text{V} \\
\quad \text{NP} \\
\quad \text{the ball}
\]

(base-adjunction—not by movement)

Where we are

X-bar theory says that trees are all built on this template (for any category X):

\[
(10) \quad \text{XP} \leftarrow \text{maximal projection}
\]

\[
\text{specifier} \rightarrow \text{XP} \rightarrow \text{X’} \rightarrow \text{intermediate projection (X-bar)}
\]

\[
\text{head} \rightarrow \text{X} \rightarrow \text{ZP} \rightarrow \text{complement}
\]

We also have the ability to adjoin phrases to each level of the representation.

\[
(11) \quad \text{adjunct} \rightarrow \text{XP} \rightarrow \text{adjunction to XP}
\]

\[
\text{Spec} \rightarrow \text{X’} \rightarrow \text{adjunction to X’}
\]

\[
\text{X complement} \rightarrow \text{Y} \rightarrow \text{X} \rightarrow \text{adjunction to X°}
\]

Adjoining allows for iteration (where there is only one specifier and one complement).

**Binary Branching**

A node can dominate at most two branches.

---

**Small clauses**

(12) a. John considers [IP [NP Bill] [VP be [AP incompetent]]].
   b. John considers Bill incompetent.

(13) \[
\text{AP} \\
\quad \text{Spec} \\
\quad \text{NP} \\
\quad \text{Bill} \\
\quad \text{A}’
\]

\[
\text{incompetent}
\]

(14) a. The captain expects the drunken sailor off the ship (immediately).
   b. John made Bill read the whole book.

(15) \[
\text{PP} \\
\quad \text{Spec} \\
\quad \text{NP} \\
\quad \text{the drunken sailors} \\
\quad \text{off} \\
\quad \text{the ship}
\]

(16) \[
\text{VP} \\
\quad \text{Spec} \\
\quad \text{NP} \\
\quad \text{Bill} \\
\quad \text{V} \\
\quad \text{NP} \\
\quad \text{read} \\
\quad \text{the whole book}
\]

**Subjects, IP, and the Extended Projection Principle**

**Extended Projection Principle (EPP)**

Clauses must have a subject (that is, ‘The specifier of IP must be filled.’).

(17) a. *(It) seems that Mary has solved the problem.
    b. Mary, it seems to have solved the problem.

(18) a. A unicorn is in the garden.
    b. *(There) is a unicorn in the garden.

In (18b), the meaningless element *there* is required. What makes it required is the EPP.
Nonfinite clauses and PRO

Nonfinite clauses lack tense and agreement features in Infl:

(24) IP
    Spec I'
    I
    VP
    V'
    V
    PP
    NP P'
    a unicorn
    P
    NP
    in the garden

Not all nonfinite clauses have to—gerundive clauses are nonfinite but lack to:

(25) a. John dislikes [CP [IP PRO eating in public]].
    b. [CP [IP PRO reading detective stories]] is fun.

Chapter 7: θ-theory

The structure of the framework we are developing is something like this:

X’ Theory

θ-Theory

Case Theory

Binding Theory

(26) a. The goalie kicked the ball.
    b. Pat likes pizza.
    c. Joe gave food to the cat.

So: The subject of kick has the θ-role of Agent.
The object of kick has the θ-role of Patient.
The subject of like has the θ-role of Experiencer.
The object of like has the θ-role of Theme.
The recipient of give has the θ-role of Goal.

The θ-role is the semantic role (thematic role) played by the argument in the event.

An argument is a ‘referring expression’—corresponds to an individual or entity (perhaps abstract). Verbs select for arguments on the basis of their syntactic category (c-selection), and on the basis of their semantic properties (s-selection).

(27) kick: Agent, Patient
    like: Experiencer, Theme

Important concept: The verb is thought of as assigning roles to its syntactic arguments.
The goalie kicked the ball.

<table>
<thead>
<tr>
<th>Agent</th>
<th>Patient</th>
</tr>
</thead>
</table>

Internal and external arguments:

External means “external to the VP”

Internal means “internal to the VP”

Verbs **directly** θ-mark the internal argument, and **indirectly** θ-mark the external argument.

θ-positions vs. θ’-positions.

Structural positions where a θ-role is assigned are called **θ-positions**. Positions where no θ-role is assigned are called **θ’-positions**.

(29) IP

External

I’

External means “external to the VP”

I

VP

Internal

V

θ-positions vs. θ’-positions.

The θ-criterion

i) Each argument chain must be assigned exactly one a θ-role.

ii) Each θ-role must be assigned to exactly one argument chain.

Requires that we consider unmoved arguments to be **trivial chains**.

Note: This means that you can never move an argument into a θ-position. Arguments can only move from a θ-position to a θ’-position.

Arguments as referring expressions and covert wh-movement.

Arguments, recall, are referring expressions.

(30) kick: Agent <Patient>

smile: Agent <Ø>

(31) a. It seems [CP that [IP Mary has solved the problem]].

b. Mary, seems [IP t to have solved the problem]].

(32) seem: Ø <proposition>

The θ-criterion (first statement)

i) Each argument must be assigned a θ-role.

ii) Each θ-role must be assigned to an argument.

(33) a. * John seems that Mary has solved the problem.

b. * There solved a problem.

c. * Mary solved there.

Chains: The collection of positions occupied by a single argument.

(34) Mary, seems [IP t to have solved the problem]].

Chain: {Mary, t, }
Quantifiers, QR, and c-command

(41) a. John suspects everyone.
   b. SS: [IP John suspects everyone].
   c. LF: [IP everyone, [IP John suspects ti ]].

(42) For every person x: [John suspects x].

(44) Everyone suspects someone.
   a. For every person x [ there is a person y [ x suspects y ]].
      ‘For every x, you can find a person y such that x suspects y.’
   b. There is a person y [ for every person x [ x suspects y ]].
      ‘There is a person y such that y is suspected by everyone.’

Two logical scope readings—we match the logical structure to syntactic structure:

(45) For every person x: [John suspects x].
   Everyone suspects someone.

(46) Scope
   The scope of α is the set of nodes α c-commands in the LF representation.

(48) C-command
   a c-commands b iff:
   i) the first branching node dominating α also dominates β.
   ii) α does not dominate β.

Informally: To find what a node c-commands, go up one level, and it is everything below it except the original node.
In (50), someone c-commands everyone, and in (51) everyone c-commands someone.

Segments count for determining c-command.

A-positions vs. A′-positions; A-chains vs. A′-chains

An A-position (argument position) is a structural position where an argument can be found at LF.

An A′-position is a structural position where a non-argument can be found at LF.

It is also useful to distinguish movement chains into A-chains (movement to an A-position) and A′-chains (movement to an A′-position).

Nonstandard arguments

Passives

(53) a. Mary ate the sandwich.  
     b. The sandwich was eaten.

The passive verb does not assign an external θ-role (to SpecIP).

Derivation of eaten to take place in the lexicon (prior to insertion into X′ trees). Attaching -en suppresses the external θ-role:

(54) eat: Agent <Theme> \rightarrow eaten: — <Theme>

SpecIP is an A-position. SpecIP is a θ′-position. \{the sandwich, t_i\} is an A-chain.

EPP (SpecIP must be filled) forces movement of the sandwich to SpecIP.

Unaccusatives

There is a class of intransitive verbs that work in a very similar way to the passive.

(56) a. The vase broke.  
     b. John broke the vase.

(57) break: Ø <Theme>

(58) [IP [the vase], [VP broke t_i]]

(59) [IP John [VP CAUSE+break the vase]]

(60) CAUSE+break: Agent <Theme>
### Adjectival passives

(61)  
   a. The island was uninhabited.  
   b. The performance was interrupted.  

(62)  
   a. CBS employees inhabited the island.  
   b. John interrupted the performance.  

(63)  
   a. The uninhabited island  
   b. The uninterrupted performance  

(64)  
   a. The island seemed uninhabited.  
   b. The performance remained uninterrupted.  

 Derived in the lexicon with a category changing suffix (also -en, -ed): 

(65)  
   \[ \text{inhabit: } [+V -N] \quad \rightarrow \quad \text{inhabited: } [+V +N] \]  
   Agent <Theme> \quad \emptyset <\text{Theme}>