CHAPTER 2
CONCEPTUAL ARGUMENTS FOR AN ASYMMETRY IN THE COMPOSITION OF PHRASE STRUCTURE

2.0 Introduction

This chapter presents conceptual arguments for a theory of the composition of phrase structure which claims that there is an asymmetry with respect to merger. I will propose the Immediate Checking Principle (ICP) on uninterpretable formal features (UFFs) and the Earliness Principle (EP) on Select. The ICP on UFFs states that UFFs must be checked immediately when they become accessible to a computation. The EP on Select states that lexical items must be selected from a numeration N as early as possible. I will argue the ICP and the EP are language-specific computational devices which contribute to the reduction of globality in the theory of language. It is shown that it is the need for satisfaction of these two principles that gives rise to the asymmetry.

Section 1 considers problems of computational complexity in the theory of language. It is shown that globality, with its "look-ahead" or "look-back" properties that necessarily induce computational complexity, should be reduced to local properties. Section 2 presents Chomsky's (1993) attempt to eliminate D-structure and S-structure. Chomsky (1993) argues that D-structure and S-structure properties in the EST should be reformulated as conditions on the interface levels. It is shown that Chomsky's (1993) interface condition approach contributes to the elimination of these two theory-internal linguistic levels and thus counts

as a step toward the goal of the MP. Section 3 argues against Chomsky's (1993) approach to the elimination of D-structure and S-structure. It is shown that Chomsky's (1993) approach raises a serious conceptual problem in that it sneaks in an element of globality into the theory of language. I will present Chomsky's (1995) derivational interpretation of strong features, which is intended to capture the displacement property, one of S-structure properties in the EST. It is shown that Chomsky's (1995) derivational constraint approach induces less globality than Chomsky's (1993) interface condition approach. Section 4 proposes the ICP on UFFs. The ICP reformulates D-structure and S-structure properties as constraints which apply throughout derivations. I will argue that our ICP approach is conceptually more desirable than Chomsky's (1993, 1995) in that the former captures D-structure and S-structure properties in a local fashion while the latter does not. Section 5 investigates consequences of the ICP for the theory of language. Section 6 discusses a consequence of the ICP concerning the composition of phrase structure. I propose the EP on Select, which reduces the fundamental globality induced by a condition on an N to local properties. It is shown that if we conform to the ICP coupled with the EP during derivations, the terms required by UFFs are merged cyclically whereas those not required by any UFFs are merged postcyclically. It then follows that arguments, which are required by UFFs, are merged cyclically. On the other hand, typical adjuncts, which are not required by any UFFs, are merged postcyclically. Section 7 makes concluding remarks.

2.1 Computational Complexity and the Theory of Language

Under the MP where the BOC-driven optimal design of language is assumed, each linguistic expression (SD) is the optimal realization of BOCs. The language therefore can be regarded as a procedure of finding a solution for the problem of optimization. The problem of optimization has been extensively studied in the field of theoretical computer science. Optimization problems can be classified in various ways. One classification, called computational complexity, is based on the amount of time, space, or other resources needed to solve a computational problem. The theory of computational complexity classifies computational problems into general complexity classes. Some computational problems are not only decidable but also feasible. Other computational problems, though decidable and thus computationally solvable in principle, are not solvable in practice because their solution requires an inordinate amount of time, memory, or other resources. In the development of generative grammar, we have come to the stage at which we can seriously ask which complexity class human language belongs to. In other words, the MP makes it possible to bring the problem of computational complexity in the theory of language to the research agenda for the first time in the history of generative grammar. This section considers the relation between the theory of computational complexity and human language. The discussion to follow, especially the implication of computational complexity for the theory of language, is largely based on Fukui (1996).

2.1.1 The Theory of Computational Complexity

This subsection overviews the theory of computational complexity. We must admit that our discussion here is quite brief and informal. To inquire further into the matter would lead us into that specialized area of computational complexity, and such a digression would undoubtedly obscure the outline of our argument. The reader should refer to works such as Hopcroft and Ullman (1979) and Johnson (1990) for more precise explanations.

The theory of computational complexity investigates time, memory, or other resources required for solving computational problems. Complexity theory classifies the problems which are decidable and thus computationally solvable in principle into general complexity classes. Among those complexity classes, those based on time are relevant to the present discussion. It has been claimed that when processing n data items, the solutions requiring no more than some polynomial involving n may be feasible. Those requiring more steps than can be described by any polynomial, on the other hand, are not feasible, though computationally solvable in principle.

If a problem has some solution where the number of steps to process n data items is no more than some polynomial involving n, the problem is defined to be in Class P. The problems in Class P are considered to have a feasible solution. In other words, they are computationally tractable. There are, however, problems whose solutions all require more than a polynomial amount of work. Those problems, which require an exponential or factorial amount of work or worse, are defined to be in Class NP. The problems in Class NP do not have any feasible solution. In other words, they are computationally

intractable. The next subsection considers the relation between these two complexity classes and the theory of language.

2.1.2 Computational Complexity in the Theory of Language

Before turning to an examination of computational complexity in the MP, let us first define the notions of globality and locality in the theory of language.[1] Suppose that we come to a stage Σ in a derivation D where we have an option of applying an operation OP. Suppose further that the decision about whether OP applies to Σ is made by a condition C. Then the notions of globality and locality are defined as follows:

(1) a. C is local if it can determine whether to apply OP or not only on the basis of information available in Σ.

b. C is global if it cannot determine whether to apply OP or not only on the basis of information available in Σ.

According to the definitions in (1), the crucial difference between global and local conditions resides in the fact that while the former has the "look-ahead" or "look-back" property, the latter does not. In other words, global conditions require us to make reference to other entities than Σ in D. Some global conditions require us to make reference to Σ and Σ' that is in D or Σ' that is in another derivation D'. For example, the principle of Procrastinate proposed by Chomsky (1993) belongs to this type, since it refers to Σ and PF. Other global conditions require us to make reference to more than one derivations and inspect each of them as a whole. The shortest derivation requirement advocated by, among others, Chomsky (1991a, 1993), Epstein (1992), and Kitahara (1995, 1997), is an instance of

[1]For a discussion of this subject, see, among others, Chomsky (1995, 1996), Collins (1997), and Fukui (1996).

global conditions of this type, since it compares the number of steps involved in more than one derivations.

Returning to computational complexity in the MP, let us consider the relation between the global/local nature of conditions and computational complexity. We take two conditions proposed in the literature as examples, Greed and Suicidal Greed. Let us first consider the principle of Greed proposed by Chomsky (1994):

(2) Move raises α to a position β only if morphological properties of α itself would not otherwise be satisfied in the derivation.

(Chomsky 1994:14)

The principle of Greed (2) has the "look-ahead" property. Accordingly, it is global in nature. In order to decide whether to apply Move to α at a stage Σ in a derivation D, we have to know whether morphological properties of α would be satisfied at Σ' that is in another derivation D' where α does not move.

Let us next consider Suicidal Greed proposed by Chomsky (1996), a modification of the principle of Greed , which is part of the definition of Attract/Move:

(3) An uninterpretable formal feature (UFF) in the extended lexical item (ELI) seeks the closest matching feature F in its c-command domain and attaches it to ELI, UFF then erasing if the match is successful (where ELI is an object formed from a lexical item by attaching other features to it).

(adapted from Chomsky 1996:9)

Unlike the principle of Greed, the principle of Suicidal Greed is local in nature. Suicidal Greed enables us to decide whether to attract F at a stage Σ in a derivation D only on the basis of information available at Σ.

Considering these two principles from the viewpoint of the theory of computational complexity, let us first investigate the principle of Greed, which is a global condition. Its corresponding optimization problem is computationally intractable. To be specific, let us consider the following optimization problem. At a stage Σ in a derivation D, we have three elements, α, β, and γ, which have options of moving to X, Y, and Z, respectively. In order to solve this problem, 2^3 derivations must be inspected to see whether they converge or not. More generally, for *n* elements which have an option of movement, 2^n derivations must be reviewed. If the review of one derivation requires one step of work, this optimization problem requires 2^n steps of work. Since this optimization problem only has a solution which requires an exponential amount of work, it belongs to Class NP. Hence, it is computationally intractable.

Turning to the principle of Suicidal Greed, its corresponding optimization problem is computationally tractable. To be specific, let us consider the following situation. At a stage Σ in a derivation D, we come across a UFF. Suppose that there are *m* formal features within the c-command domain of the UFF. We scan all those formal features to see whether they match the UFF. Suppose that we have found *l* formal features which match the UFF within the c-command domain of the latter. Then, we have to inspect which formal feature is closest to the UFF. Let us assume following Chomsky (1995) that the notion of "closeness" is defined in terms of the notion of c-command. Since c-command is a transitive relation, for *l* features, we have to inspect the c-command relation between two features *l-1* times to decide which feature is closest to the UFF. If the inspection of a c-commanding feature and that of c-command relation between two elements each requires one step of work,

this problem requires $m + (l - 1)$ steps of work. More generally, for n UFFs, this optimization problem requires $n\ (m + (l - 1))$ steps of work. Since this optimization problem has a solution where the number of steps to process n items is no more than a polynomial involving n, it belongs to Class P. Hence, it is computationally tractable. From the above discussion, we can conclude (4) concerning the relation between the global/local nature of conditions and computational complexity:

(4) Global conditions necessarily induce computational intractability while local conditions do not.

Before we leave this subsection, it is important to consider the relation between the problem of computational complexity and the design of language. Chomsky (1996) claims that the problem of computational complexity only arises in the theory of language as far as the following assumptions are supported:

(5) a. There is an empirical difference between "derivational" and "representational" interpretations of the recursive procedure constituting the I-language.
b. Language uses the derivational approach.
c. Considerations of computational complexity matter for a cognitive system (a "competence system," in the technical sense of this term).

(Chomsky 1996:10)

Let us first consider (5a) and (5b). A representational approach is the one which takes the recursive procedure to be nothing more than a convention for enumerating a set of linguistic expressions (SDs). Under the representational approach, rules are applied freely, but illegitimate outputs are excluded by output conditions as ill-formed. Note that under

the representational approach where rules are allowed to apply optionally, we do not have to make the decision about whether to apply an operation during a derivation. If we make a wrong decision about the application of an operation during a derivation, its output representation is simply ruled out by an output condition. Hence, the problem of computational complexity never arises.[2]

A derivational approach, on the other hand, takes the recursive procedure literally, forming linguistic expressions (SDs) step-by-step by applying operations to features. At each step of a derivation, we have to make the decision about whether to apply an operation. Hence, the problem of computational complexity may arise given that a computation is sequential, not parallel. Recall that the MP which this thesis adopts as its theoretical foundation is derivational in nature. This is because the MP does not allow any superfluous step in a derivation. Operations are applied whenever necessary, not otherwise. At no point of a derivation, will there be an optional application of an operation. The problem of computational complexity therefore may arise in the MP.

Although it is typically possible to recode one approach in terms of the other, these two approaches are still empirically different. Chomsky (1995) argues that there is evidence which suggests that the derivational approach is on the right track.[3] Chomsky discusses the head movement constraint (HMC), which was first proposed by Travis (1984). The HMC states that head movement cannot pass over the closest c-commanding head, excluding examples like the following:

---

[2] See, among others, Brody (1995), Cinque (1990), Koster (1978a, 1986), and Rizzi (1986, 1990) for arguments in favor of the representational view.

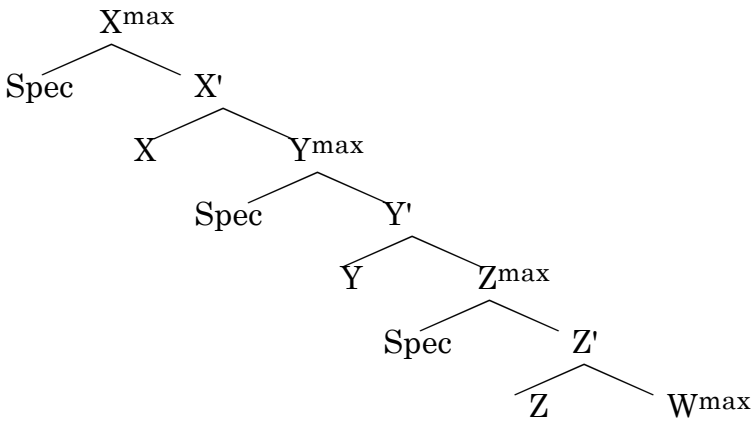
[3] See Chomsky and Lasnik (1993) for further discussion of this subject.

(6) *read John will *t* the book

In (6), *read* raises to the clause-initial position, crossing over *will*. Hence, (6) is excluded as illegitimate by the HMC. Chomsky argues that the HMC imposed on the step-by-step computation is not always reflected on the output representation.[4]

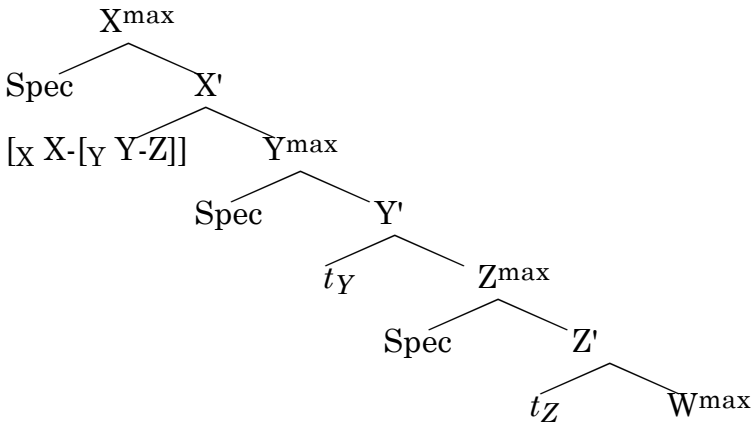
Let us consider the following structure as an example:

(7)



Suppose that Z adjoins to Y, leaving the trace of Z, and the amalgamated form [$_Y$ Y-Z] further adjoins to X, leaving the trace of Y:

(8)



[4]As argued by Lasnik (1994) and Robert (1994), however, Chomsky's (1995) argument for the derivational approach is inconclusive. In V-to-C constructions like (6), there is no feature of either V or C that is satisfied by the raising of V to C itself. Rather, since only a finite verb raises to C, Tense must be involved. Hence, it is possible to claim that the raising of *read* to C in (6) is excluded by the economy condition which bans a superfluous step in a derivation without any recourse to the HMC.

In the output representation (8), the chain (Z, t_Z) violates the HMC although each head movement obeys the HMC derivationally. In other words, the locality property of the adjunction of Z to Y is obscured in the output representation by being "wiped out" by the later operation. This dissertation agrees with Chomsky that there is an empirical difference between derivational and representational approaches and the former approach is on the right track, presenting further empirical evidence in support of the derivational interpretation of the recursive procedure.

Turning to (5c), suppose that considerations of computational complexity matter for fundamental aspects of language. It would then follow that the problem of computational complexity arises in the theory of language given the derivational interpretation of the recursive procedure. Hence, this view would require that global conditions, which necessarily induce computational intractability, should be reduced to local conditions, whose corresponding optimization problems may be feasible. Collins (1997) takes this view, making an attempt to reformulate global economy conditions as local economy conditions.

This dissertation, however, does not share the above view. I argue that considerations of computational complexity do not matter for fundamental aspects of language. Globality is one of the fundamental properties of language although it would necessarily induce "exponential blow-up" in construction and evaluation of derivations. As Chomsky (1991a, 1991b, 1993, 1994, 1995, 1996) argues, since there is no a priori reason to suppose that language is "usable" or "conducive to efficient use," language can be fundamentally computationally intractable due to its fundamental global properties. Its corresponding optimization problem therefore belongs to Class NP. Language, however, is usable in practice.

I argue following Chomsky that for purposes of normal life, usability of language is facilitated by what Chomsky calls "computational tricks," the biological counterparts of "heuristic algorithms," which enable us to obtain approximate solutions to the computationally intractable optimization problems. Such language-specific computational devices reduce fundamental globality to local properties, enabling language to be used in practice. To recapitulate, although language yields computationally intractable problems, they can often be overcome by "heuristic algorithms" ("computational tricks"). Only when such language-specific computational devices are available, expressions can be used. Unusable parts of language are simply not used. Hence, although considerations of computational complexity do not matter for fundamental aspects of language, they do matter for usable parts of language. For usable parts of the language, therefore, the problem of computational complexity arises given the derivational interpretation of the recursive procedure. Global conditions, which necessarily induce computational intractability, should be reduced to local "heuristic algorithms" ("computational tricks") for usable parts of language.

This dissertation presents empirical arguments for this view, proposing the ICP on UFFs and the EP on Select. I will argue that these principles serve as language-specific computational devices which reduce the globality induced by interface conditions on UFFs and numerations Ns to local properties, facilitating usability of language in practice. It is shown that these principles lead to an asymmetry in the composition of phrase structure, which receives strong empirical support from a wide range of facts.

The following sections are devoted to showing that the ICP on UFFs and the EP on Select serve as local "heuristic algorithms" ("computational tricks") for computationally intractable problems. I will first consider the ICP, investigating minimalist approaches to the elimination of the two theory-internal linguistic levels, i.e., D-structure and S-structure.

2.2 Chomsky's (1993) Interface Condition Approach

This section reviews Chomsky's (1993) attempt to reduce D-structure and S-structure properties in the EST to interface conditions. As mentioned in the previous chapter, under the MP where everything is within the domain of virtual conceptual necessity, an SD consists not of the traditional four levels, i.e. D-structure, S-structure, PF, and LF, but rather only of the latter two interface levels. It then follows that every principle which constrains a derivation applies either at the interface levels or at every step of a derivation where it is relevant. The principles and conditions which were assumed to apply at D-structure or S-structure in the EST now have to be captured in a different way. They have to be reformulated either as conditions on the interface levels or as constraints which apply throughout derivations. I call the former the interface condition approach and the latter the derivational constraint approach. Apart from the economy conditions, Chomsky (1993) pursues the interface condition approach, namely, reformulating the principles and conditions which were assumed to apply at D-structure and S-structure in the EST as conditions on the interface levels.

2.2.1 D-structure Properties

Let us first consider D-structure properties in the EST. Within the framework of the EST, D-structure is a representation of GF-θ, i.e., a pure representation of θ-structure. It is created by the operation SATISFY, which selects an array of items from the lexicon and presents it in a format satisfying the conditions of the X-bar theory. SATISFY is assumed to be an "all-at-once" operation in that all items that function at LF are drawn from the lexicon before a computation proceeds. Among the principles and conditions which were assumed to apply at D-structure is the θ-criterion, which states that each argument bears one and only one θ-role, and each θ-role is assigned to one and only one argument. The θ-criterion itself is a criterion of adequacy for LF. When coupled with the Projection Principle, however, the θ-criterion virtually applies at D-structure as well. This is because the Projection Principle requires that every syntactic representation, i.e., D-structure, S-structure, and LF, should be a projection of θ-structure.

Chomsky (1993) argues that there are conceptual and empirical problems with assuming the θ-criterion and the Projection Principle as D-structure conditions. Let us first look at their conceptual problem. Within the EST framework, the θ-criterion and the Projection Principle are required to apply at D-structure in order to ensure that D-structure has the basic properties of LF. At LF, these conditions are trivial, since if they are not met, the linguistic expression receives some deviant interpretation at that interface level. At D-structure, on the other hand, these conditions are only needed to make the EST picture coherent. If we abandon the EST picture, then these D-structure conditions lose their roles. Hence, they are dubious on conceptual grounds.

Among empirical problems with assuming the θ-criterion and the Projection Principle as D-structure conditions, let us look at the complex adjectival construction as an example:

(9) the man who is reading a book is easy to please

According to Chomsky (1981, 1986a, 1986b), (9) is assigned S-structure representation (10):

(10) [the man who is reading a book] is easy [***Op*** [PRO to please ***t***]]

In (10), *the man who is reading a book* is directly inserted in its surface position. The adjective *easy* takes as its complement a clause which has undergone an empty operator movement. As argued by, among others, Chomsky (1977, 1981, 1986a, 1986b), this empty operator movement analysis is supported by the bounding condition effects.

Under the empty operator movement analysis of the complex adjectival construction, we are led to assume that *the man who is reading a book* in (9) occupies a θ-position, since it is directly inserted in its surface position. There is, however, evidence to suggest that the subject position of the adjective *easy* is not a θ-position:

(11) it is easy to please the man who is reading a book

In (11), the expletive *it* appears in the subject position of the adjective *easy*, which suggests that the subject position of the adjective *easy* is a non-θ-position. Here we have a paradoxical situation unless we assume a dual lexical entry of the adjective *easy*. If we consider (11), where the expletive *it* appears in the subject position of the adjective *easy*, the subject position should be empty at D-structure in accordance with the θ-criterion and the Projection Principle. If we consider (10), where *the man who is reading a book*, being an argument, appears in the subject position

of the adjective *easy*, the θ-criterion and the Projection Principle requires that the subject position should be filled at D-structure. This paradoxical situation suggests that assuming the θ-criterion and the Projection Principle as D-structure conditions should be on the wrong track.

These conceptual and empirical problems have led Chomsky (1993) to give up assuming the θ-criterion and the Projection Principle as D-structure conditions. The Projection Principle is totally eliminated while the θ-criterion is reformulated as a condition which only applies at LF. Hence, SATISFY does not have the property of "all-at-once" any more. Instead, a lexical item is selected from the lexicon freely at any point as a computation proceeds.

2.2.2 S-structure Properties

Let us turn to S-structure properties in the EST. Among principles and conditions which were assumed to apply at S-structure in the EST, let us consider structural Case assignment, which was instrumental in explaining the displacement property regarding NP-movement.

It was assumed in the EST that structural Case should be assigned at S-structure. The arguments for assuming structural Case assignment as a S-structure property were as follows. In some languages, Case is morphologically realized while in others it is not. It had been assumed, however, that Case should be assigned in a uniform way whether it is morphologically realized or not. Hence, Case features must appear at PF. In Chomsky (1981, 1986b), the Case Filter, which applies in the PF-component, was intended to capture this fact:

(12) *NP if NP has phonetic content and has no abstract Case.

(adapted from Chomsky 1981:49)

Even in languages where Case is not morphologically realized, Case was assumed to be assigned at S-structure though it is abstract.

Furthermore, it was claimed that Case features must be "visible" at LF (see, among others, Aoun (1985a) and Chomsky (1981, 1986b)). Aoun and Chomsky propose the visibility condition on θ-role assignment, which informally states that only those elements that are assigned Case features are "visible" and thus able to retain θ-roles at LF. They present some empirical arguments for the visibility condition. Among the arguments is the Case requirement of variables. Let us look at the following example:

(13) ***who** does it seem [***t*** to see Mary]

If a variable appears in the non-Case-marked position as in (13), the result is deviant. The Case Filter (12), being a PF requirement, cannot accommodate this fact, since variables do not have any phonetic content and thus should be exempted from the Case Filter. Under the visibility condition approach, on the other hand, variables as in (13) are not assigned any Case. Hence, they cannot retain any θ-role at LF. This violates the θ-criterion. We can correctly predict that examples like (13) are deviant.

One might argue, however, that the Case Filter (12) suffices to rule out examples like (13). One could argue that what needs Case is not the variable, but the *wh*-phrase. The *wh*-phrase must inherit a Case feature from the variable it binds. In (13), since the variable is not assigned any Case feature, *who* does not inherit any Case feature from the variable. Hence, the *wh*-phrase *who* violates the Case Filter (12). There are,

however, cases which even the Case Filter (12) coupled with the Case inheritance mechanism cannot accommodate. Let us consider the following example:

(14) *the man [that [you tried [***t*** to win]]]

Let us assume following, among others, Chomsky (1977) that wh-movement is involved in the formation of relative clauses. In (14), the *wh*-phrase originates in the subject position of *win*. It is raised to the Spec of CP and then deleted there. Since the *wh*-phrase is deleted in (14), it is not submitted to the Case Filter (12). Hence, the Case Filter with the inheritance mechanism cannot provide any basis for excluding examples like (14). The visibility condition, on the other hand, explains why examples like (14) are deviant. This is because since the variable in (14) is not assigned any Case feature, it is "invisible" at LF and thus unable to retain any θ-role at LF; this violates the θ-criterion. These arguments suggest that Case features should appear both at PF and LF. Hence, Case features must be present at the time when derivations reach S-structure.

Chomsky (1993) argues, however, that once we assume the checking theory, the above arguments for assuming structural Case assignment as a S-structure property collapse. Structural Case assignment, which explains the displacement property regarding NP-movement, can be reformulated as conditions on the interface levels. Under Chomsky's (1993) checking theory, lexical items are fully inflected when selected from the lexicon. Fully inflected lexical items take the following form, where α is the morphological complex [R-$Infl_1$, ..., $Infl_n$], R is a root and $Infl_i$ is an inflectional feature:

(15) LI = (α, $Infl_1$, ..., $Infl_n$)

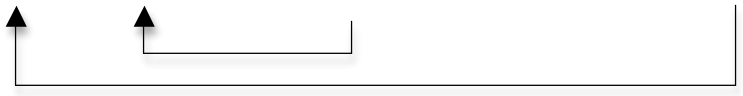
$Infl_i$ in (15) is an inflectional feature, which is removed from the LI when it is checked. There is assumed to be a strong/weak distinction regarding features. Strong features are illegitimate objects at PF. If they do not enter into checking relations before Spell-Out and thus remain at PF, the derivation crashes. Weak features, though legitimate at PF, are illegitimate at LF. They must be checked off before the derivation reaches LF; otherwise the derivation crashes.

Let us look at how the checking theory works, taking (16) as an example:

(16) John saw Mary.

(16) is assigned the following structure at the time of Spell-Out:

(17) $[_{AGRsP}$ **John** $[[$**T** AGRs$]_{AGRs}$ $[_{TP}$ ***t*** $[_{AGRoP}$ Spec $[$AGRo $[_{VP}$ ***t***



[saw Mary]]]]]]]

John has a Nominative Case feature and ϕ-features. *Mary* has an Accusative Case feature and ϕ-features. The verb *saw* has an Accusative Case feature, a Tense feature, and ϕ-features. AGR and T, being functional heads, have V-features as well as N-features. V-features are those which check the inflectional features of verbs like Tense features and ϕ-features. N-features, on the other hand, are those which check the inflectional features of nouns like Case features and ϕ-features. Let us first consider the Nominative Case assignment. Since the N-feature of T, i.e., the Nominative Case feature, is strong in English, T overtly raises to AGRs in order to have its Case feature checked off through entering into a checking relation with a noun. This is because it was assumed that formal features can only be checked through the mediation of AGR.

Then, *John* overtly raises to the Spec of AGRsP to enter into a checking relation with the amalgamated form [$_{AGRs}$ T-AGRs]. If these operations do not apply overtly, the strong Nominative Case feature of T remains at PF and thus the derivation crashes.

Turning to the Accusative Case assignment, the N-features of V, including an Accusative Case feature, are weak in English. V does not have to raise overtly to AGRo to have its Accusative Case feature checked off. The principle of Procrastinate, which prefers covert operations to overt operations, requires that V should raise to AGRo covertly, not overtly. Then, *Mary* raises covertly to the Spec of AGRoP to enter into a checking relation with the amalgamated form [$_{AGRo}$ *saw*-AGRo], resulting in the following structure:

(18) [$_{AGRsP}$ John$_i$ [[T$_j$-AGRs]$_{AGRs}$ [$_{TP}$ t_j [$_{AGRoP}$ **Mary**

[[**saw**-AGRo]$_{AGRo}$ [$_{VP}$ t_i [***t*** ***t***]]]]]]]

The amalgamated form [$_{AGRo}$ *saw*-AGRo] further adjoins to T and then to AGRs in order to check the V-features of AGRs. If these operations do not apply, the features remain at LF and thus the derivation crashes. Therefore, if we assume the checking theory, structural Case assignment, which was assumed to apply at S-structure in the EST, can be reformulated as conditions at the interface levels.

To summarize Chomsky's (1993) approach to the elimination of D-structure and S-structure, the principles and conditions which were assumed to apply at either of these levels within the EST are reformulated as conditions on the interface levels. The θ-criterion, which

was a D-structure condition within the EST, is reformulated as a condition at LF. Structural Case assignment, which was a S-structure property within the EST that subsumes the displacement property regarding NP-movement, is reformulated as conditions at PF and LF. Such an interface condition approach to the elimination of D-structure and S-structure contributes to the elimination of these two theory-internal linguistic levels and thus counts as a step toward the goal of the MP. In the next section, I will first present Chomsky's (1995) derivational interpretation of strong features, which is intended to reformulate the displacement property, one of S-structure properties in the EST, as a condition which applies through derivations. It is shown that Chomsky's (1995) derivational constraint approach should be preferred over Chomsky's (1993) interface condition approach on conceptual grounds.

2.3 Chomsky's (1995) Derivational Constraint Approach

The last section has considered Chomsky's (1993) interface condition approach to the elimination of D-structure and S-structure and pointed out that it contributes to the elimination of the two theory-internal linguistic levels. In this section, however, I will first point out that such an interface condition approach raises a serious conceptual problem in that it sneaks in an element of globality in the theory of language. I will then present Chomsky's (1995) derivational interpretation of strong features, which is intended to capture the displacement property, one of S-structure properties in the EST. It is shown that Chomsky's (1995) derivational constraint approach is more

desirable than Chomsky's (1993) interface condition approach in that the former reduces computational burden.

Under Chomsky's (1993) approach to the elimination of D-structure and S-structure, the properties of these two levels are reduced to conditions on the interface levels. Although Chomsky's (1993) interface condition approach counts as a step toward the goal of the MP in that it eliminates the two theory-internal linguistic levels, it raises a serious conceptual problem. Let us consider structural Case assignment as an example. Under Chomsky's (1993) checking theory, we should apply the raising operation for checking a Case feature only if the derivation would otherwise reach the interface levels with an illegitimate object. Hence, at the stage of the derivation where we have an option of raising DP to check its Case feature, we cannot decide whether to apply the raising operation or not only on the basis of information available at that stage. We must look ahead to see whether the application/non-application of the raising operation would yield the interface levels only with legitimate objects. In other words, global considerations are needed for the decision about whether to apply the raising operation or not. As argued in section 1, globality, which necessarily induces computational intractability, should be reduced to local properties.

It should be noted that we are arguing against globality not against interface conditions. Interface conditions can be formulated not to trigger any operations during derivations but to simply rule out illegitimate interface representations. As will be argued later, the binding theory, the uniformity condition on chains, and the bans against vacuous quantification and free variables could be formulated as such

local interface conditions. Such local interface conditions are not conceptually problematic, since they do not induce any globality.

Reformulating the previous D-structure and S-structure principles and conditions as constraints which apply throughout derivations greatly contributes to the reduction of globality in the theory of language. This is because such constraints can be formulated in such a way that they only need local considerations, but not global considerations. Chomsky's (1995) treatment of strong features goes along this line. As mentioned in the previous chapter, strong features are those which trigger overt category movement. Recall that in Chomsky (1993), strong features are characterized as illegitimate objects at PF and thus required to be checked off before Spell-Out for PF-convergence. Hence, they trigger overt category movement. As I have argued, however, this analysis needs very global considerations in the sense that when we come to a stage of a derivation where we have an option of checking a strong feature, we cannot decide whether to check it only on the basis of information available at the stage. This is because we must look ahead to see whether the checking/non-checking of the strong feature would yield PF consisting only of legitimate objects. Chomsky (1995), on the other hand, defines strong features as those that derivations "cannot tolerate" in the sense stated in (19):

(19) Suppose that a derivation D has formed a structure containing α with a strong feature F. Then, D is canceled if α is in a category not headed by α.

(adapted from Chomsky 1995:234)

This condition gives us a less global interpretation of strong features.

Before considering how (19) works, let us clarify two points on which Chomsky (1995) departs from Chomsky (1993). First, among the functional categories, Chomsky (1995) eliminates AGR, which exists only for theory-internal reasons. The Case features and ϕ-features of T and Verb are now assumed to enter into checking relations with those of nominals without any mediation of AGR. Second, Chomsky (1995) assumes that a strong feature always calls for a certain category in its checking domain. Hence, the parametric variations among languages regarding overt category movement are attributed to the existence/nonexistence of strong categorial features. Overt subject raising to the Spec of TP, for instance, is triggered not by the strong Nominative Case feature of T as in Chomsky (1993) but by the strong D-feature of T.

For an illustration of Chomsky's (1995) derivational interpretation of strong features, let us consider overt subject raising to the Spec of TP, taking the embedded clause in (20) as an example:

(20) Bill believes that John saw Mary.

Under Chomsky's (1995) theory, the embedded clause in (20) is analyzed as follows:

(21) [$_{v}$P John [v [$_{\text{VP}}$ saw Mary]]]

The v is a light verb which is assumed in the transitive and unergative constructions but not in the unaccusative construction. The subject *John* appears in the Spec of vP where it receives an external θ-role in the v-VP configuration. Then, we come to the following structure:

(22) [T$_{[\text{D}]}$ [$_{v}$P John [v [$_{\text{VP}}$ saw Mary]]]]

Recall that under Chomsky's (1995) theory, what triggers overt subject raising is the existence of the strong D-feature of T. In English, where subjects are raised overtly, T is assumed to have a strong D-feature.

There are logically two possible continuations of (22); we either move *John* to the Spec of TP in order to check the D-feature of T or merge C with (22). Between these two possibilities, we must choose the former option, yielding the following structure:

(23) [$_{TP}$ **John** [T [$_{\upsilon P}$ ***t*** [υ [$_{VP}$ saw Mary]]]]]

This is because if we merged C with (22) as in (24), then the derivation would be canceled:

(24) [C [T$_{[D]}$ [$_{\upsilon P}$ John [υ [$_{VP}$ saw Mary]]]]]

In (24), the strong D-feature of T is contained in the category which is not headed by T; this violates (19). Hence, (19), which is a condition which applies throughout derivations, triggers overt subject raising in English without recourse to any interface conditions. More generally, the displacement property, an S-structure property in the EST, can be reduced to the derivational constraint (19).

Chomsky's (1995) derivational constraint approach to strong features is conceptually more attractive than Chomsky's (1993) interface condition approach. This is because the former needs less global considerations than the latter. In order to decide whether to apply a checking operation of a strong feature, Chomsky's (1995) derivational constraint approach does not have to look at the interface levels, but only the next stage of the derivation. If the inspection of the next stage tells us that the non-checking of the strong feature would result in a structure which violates (19), we should apply the operation to check the strong feature at the present stage.

2.4 A Strictly Derivational Constraint Approach

2.4.1 Reinterpretation of Strong Features

The previous section has shown that Chomsky's (1995) interpretation of strong features (19) is conceptually more attractive than Chomsky's (1993) interpretation. It was pointed out that the former needs less global considerations than the latter. I will argue, however, that Chomsky's (1995) interpretation, though more plausible than Chomsky's (1993), still has conceptual and empirical problems.

Conceptually, Chomsky's (1995) interpretation of strong features (19) cannot reduce globality to local properties. This is because it has to look ahead to inspect the next stage in a derivation to make the decision about whether to apply an operation. To be specific, suppose that we come to a stage Σ_i in a derivation D where we have an option of applying an operation OP. Suppose further that the inspection of the next stage Σ_{i+1} tells us that the non-application of OP at Σ_i would yield a structure which violates the derivational constraint (19) at Σ_{i+1}. Then, we know that it is necessary to apply OP at Σ_i. Hence, Chomsky's (1995) interpretation of strong features (19), which has the "look-ahead" property, is still global, since it cannot decide whether to apply OP only on the basis of information available at Σ_i. Accordingly, the corresponding optimization problem is computationally intractable. To be specific, suppose that we come across *n* strong features at Σ_i in D. In order to decide whether to apply operations to check these features, it is necessary to make reference to Σ_{i+1} in D and the derivations which share a sequence of syntactic objects up to Σ_i with D. Overall, 2^n derivations must be reviewed to see whether they violate (19). If the review of one derivation

requires one step of work, this optimization problem requires 2^n steps of work (apart from the work required to search matching formal features within the c-command domain of the strong features). Since this optimization problem only has a solution which at least requires an exponential amount of work, it belongs to Class NP. Hence, it is computationally intractable. We can see from the above discussion that Chomsky's (1995) derivational constraint approach, though it needs less global considerations than Chomsky's (1993), is still global. Its corresponding optimization problem is computationally intractable. Such global conditions should be reduced to local properties.[5]

Chomsky's (1995) interpretation of strong features (19) also has an empirical problem. It cannot trigger root overt movement like overt wh-movement in the matrix clause. Let us consider (25) as an example:

(25) **what** did you read ***t***

During the derivation of (25), we come to the stage where the strong Q-feature of C can be checked off:

(26) $[C_{[Q]}$ [you read what]]

In order to derive (25), we have to move the *wh*-phrase *what* to the Spec of CP and check the strong Q-feature of C. Derivational constraint (19),

---

[5]Note in passing that exactly like Chomsky's (1995) interpretation of strong features, the local economy conditions proposed by Collins (1997) are also global, contrary to what Collins himself claims. Let us consider his Last Resort condition:

(i) Last Resort
An operation OP involving α may apply only if some property of α is satisfied.

(Collins 1997:9)

It should be noted that in order to decide whether an operation OP involving α satisfies some property of α, it is necessary to make reference to the next stage in a derivation. The Last Resort condition (i) cannot decide whether to apply OP only on the basis of information available at the present stage. Hence, it is a global condition. Accordingly, its corresponding optimization problem is computationally intractable.

however, cannot trigger this overt wh-movement. Let us consider why (19) does not work in such a case. If the *wh*-phrase *what* does not move to the Spec of CP, then the strong Q-feature remains. According to (19), however, this derivation is not canceled. This is because the CP is the root clause and thus never contained in another category. (19) would thus claim that the strong Q-feature in (26) does not have to be checked at this stage by the application of overt wh-movement. According to the principle of Procrastinate, which prefers covert operations to overt operations, *what* should move in the covert component rather than in the overt component. Hence, there is no way to derive (25).

In order to solve these conceptual and empirical problems, I propose the following constraint on strong features while pursuing the derivational constraint approach:

(27) Strong features must be checked immediately when they become accessible to a computation.

(27) requires that strong features should be checked and deleted immediately when they become accessible to a computation; otherwise, the derivation is canceled. I call this approach the strictly derivational constraint approach to strong features.

Let us look at how the strictly derivational constraint approach works, considering the overt subject raising to the Spec of TP as an example. I will take the embedded clause in (20) (repeated here as (28)) as an example:

(28) Bill believes that John saw Mary.

Let us assume that the embedded clause in (28) is analyzed as follows:[6]

(29) [$_{VP}$ John [see Mary]]

The subject *John*, which is base-generated in the Spec of VP, receives an external θ-role in that position. Then, we come to the following structure where T with a strong D-feature is introduced:

(30) [$T_{[D]}$ [$_{VP}$ John [see Mary]]]

Recall that under Chomsky's (1995) approach, there are logically two possible continuations of (30); we either move *John* to the Spec of TP in order to check the D-feature of T or merge C with structure (30). Between these two possibilities, we choose movement of *John* to the Spec of TP, since merger of C with (30) would violate derivational constraint (19) at the next stage and thus make the derivation canceled. Under the strictly derivational constraint approach, on the other hand, there exists only one option available at this stage, i.e., the raising of *John* to the Spec of TP in order to check off the strong D-feature of T. This is because (27) requires that the checking operation of the strong D-feature of T should be applied prior to any other operations. This correctly yields the following structure:

(31) [$_{TP}$ **John** [T [$_{VP}$ ***t*** [see Mary]]]]

It should be noted that the strictly derivational constraint approach to strong features can solve the conceptual and empirical problems which Chomsky's (1995) derivational constraint approach faces. Let us first consider the conceptual problem. Unlike Chomsky's (1995) approach, the

---

[6]I do not assume Chomsky's (1995) light verb analysis of the transitive and unergative constructions. It should be pointed out, however, that the arguments to follow are valid even under Chomsky's (1995) light verb analysis.

strictly derivational constraint approach is local, since at a stage Σ of a derivation D, it can decide whether to apply an operation OP only on the basis of information available at Σ. Let us look at (30) again as an example. Under the strictly derivational constraint approach, at stage (30) where the strong D-feature of T is introduced into the derivation, the information available at this stage tells us that we should apply the raising operation to check the strong D-feature. Unlike Chomsky's (1995) derivational constraint approach, the strictly derivational constraint approach does not has to look ahead to inspect the next stage to make the decision about whether to apply the raising operation. Accordingly, the corresponding optimization problem is computationally tractable. To be specific, suppose that we come across *n* strong features at Σ in D. Suppose further that there are *m* formal features within the c-command domain of those strong features. For each strong feature, we scan all those c-commanded formal features to see whether it matches them. If the inspection of one c-commanded feature requires one step of work, this problem requires *nm* steps of work. Since this optimization problem has a solution where the number of steps to process *n* items is no more than some polynomial involving *n*, it belongs to Class P. Hence, it is computationally tractable.

Turning to the empirical problem, let us consider (25) (repeated here as (32)) again as an example:

(32) **what** did you read ***t***

During the derivation of (32), we come to the stage where C is introduced into the derivation:

(33) $[C_{[Q]}$ [you read what]]

C has a strong Q-feature. The strictly derivational constraint approach requires that this strong feature should be checked immediately by the raising of the *wh*-phrase *what* to the Spec of CP, correctly deriving (32). Hence, unlike Chomsky's (1995) approach, the strictly derivational constraint approach can correctly trigger root overt movement.

In this subsection, I have proposed the strict derivational constraint approach to strong features and argued that it should be preferred over Chomsky's (1995) approach on both conceptual and empirical grounds. It is shown that the displacement property, an S-structure property in the EST, can be reduced to the strictly derivational constraint. In the next subsection, I will argue that other D-structure and S-structure properties in the EST can also be subsumed under the strictly derivational constraint.

2.4.2 The Immediate Checking Principle

2.4.2.1 Strong Features

In the previous subsection, I have proposed the strictly derivational constraint approach to strong features, which claims that strong features must be checked immediately when they become accessible to a computation. It was shown that the displacement property, an S-structure property in the EST, can be reduced to the strictly derivational constraint (27). I have argued that the strictly derivational constraint approach to the elimination of this S-structure property is more attractive than the previous approaches on both conceptual and empirical grounds. The question to be addressed at this point is why strong features are subject to the strictly derivational constraint. I argue that the notion [+/-Interpretable] in Chomsky's (1995) sense plays a crucial role in

characterizing features which are subject to the strictly derivational constraint. I argue that the uninterpretable characteristic of strong features makes them subject to the strictly derivational constraint. In other words, strong features, because they are [- Interpretable] and thus illegitimate at LF, must be checked immediately when they become accessible to a computation.[7]

It is then natural to claim that the strictly derivational constraint not only applies to strong features but also the other uninterpretable formal features (UFFs). I propose the ICP on UFFs, arguing that all UFFs are subject to the strictly derivational constraint:[8]

(34) The Immediate Checking Principle (ICP)
Uninterpretable formal features (UFFs) must be checked immediately when they become accessible to a computation.

The ICP requires that UFFs must be checked and deleted (made invisible at LF) immediately when they become accessible to a computation; otherwise the derivation is canceled.[9] According to the ICP, UFFs appear in derivations just to be deleted. I will argue that the ICP gains support from the fact that it subsumes other D-structure and S-structure properties than the displacement property in the EST. I will first consider two S-structure properties, i.e., structural Case assignment and

---

[7]Recall that our characterization of a strong feature is different from Chomsky's (1993, 1994, 1995, 1996). There is no theoretical notion of strong feature in our system. The expression "strong feature" is used just for sake of presentation to identify the uninterpretable categorial feature of a functional element which triggers overt category movement. According to our system, therefore, the Q-feature of C, which triggers overt wh-movement, is uninterpretable. This is in contrast with Chomsky's system where the strong Q-feature of C is assumed to be interpretable.

[8]Conceptual plausibility of the ICP, especially its status in the theory of language, will be discussed in detail in section 2.5.1.

[9]I will later argue that there is no operation of checking. Accordingly, the ICP will be reformulated without recourse to the operation of checking.

agreement relations. I will then consider selectional restrictions, which were D-structure properties in the EST.

Before turning to the ICP approach to the elimination of D-structure and S-structure, let us explicate the operation of checking. Following Chomsky (1995), let us assume the following formulation of the checking operation:

(35) A checked feature is deleted when possible.

(Chomsky 1995:280)

Deleted features are invisible at LF, but still accessible to a computation. (35) claims that a checked feature is deleted only when it would not contradict the principle of recoverability of deletion, which states that unrecoverable items may not be deleted. Interpretable features, which receive interpretation at LF, cannot be deleted when they are checked. This is because if checked interpretable features are deleted, it would violate the principle of recoverbility of deletion. Uninterpretable features, on the other hand, are deleted when they are checked. Since uninterpretable features do not contribute to any content at the LF interface, their deletion does not violate the principle of recoverbility of deletion.

I also assume with Chomsky (1995) that checking relations can be established either by Attract/Move or Merge between a head H and an element in its neighborhood. Although Chomsky assumes that checking relations are only established between H and an element in its Spec position, I claim that they are established between H and an element in its Spec or complement position.

2.4.2.2 Structural Case Assignment

Within the framework of the EST, structural Case assignment was assumed to apply at S-structure. As I have presented in section 2.2.2, the arguments for the S-structure property of structural Case assignment were based on the fact that Case features appear at both PF and LF and thus must be present at the time when derivations reach S-structure. I have presented Chomsky's (1993) checking theory of Case, according to which the S-structure property of structural Case assignment can be reduced to the interface conditions. It was shown that it counts as a step toward the goal of the MP in that it contributes to the elimination of S-structure. As I have argued above, however, Chomsky's (1993) approach raises a conceptual problem, since it would sneak in an element of globality in the theory of language. In order to solve the globality problem, I argue that structural Case assignment should be subsumed under the ICP. Under the ICP approach, where structural Case assignment is reduced to a local condition, the problem of globality does not arise.

Recall that since Case features are uninterpretable no matter where they may appear, they are always subject to the ICP. The ICP then requires that when Case features become accessible to a computation, they should immediately enter into checking relations either by Attract/Move or Merge. After Case features enter into checking relations, they are deleted and made invisible at LF, since they are uninterpretable.

Let us consider (36) as an example:

(36) John saw Mary

We first select the verb *see* from the numeration N:

(37) $see_{[ACC]}$

In (37), the Accusative Case feature is represented as ACC. Recall that transitive verbs like *see* have Accusative Case features as their intrinsic features. The ICP requires that the Accusative Case feature of *see* should be checked immediately. It is checked by selecting *Mary*, which has a Case feature as its optional feature, and merging *see* with *Mary*, as shown below:[10]

(38) [V^{max} see Mary]

Note that if the Case feature of *Mary* does not coincide with that of *see*, the derivation is canceled due to a violation of the ICP.

One might say that this derivation violates the ICP, since Select *Mary* intervenes between Select *see*, which makes its uninterpretable Accusative Case feature accessible to the computation, and merger of *see* with *Mary*, which checks the uninterpretable feature. I argue that the ICP only regulates operations which manipulate terms in phrase structures, i.e., Merge and Attract/Move. The operation Select selects a lexical item from an N and introduces it into a derivation. It does not manipulate terms. Select is therefore immune from the ICP. Hence, in (38), although we apply Select *Mary* before merger of *see* with *Mary*, there is no violation of the ICP.

Another important point to note concerns the introduction of optional features. Chomsky (1995) claims that they are added

---

[10]Essentially following Chomsky (1995) and Muysken (1982), we define the notion of maximal projection derivationally. In (38), for instance, the dominating node is assigned the categorial status of V^{max}, since it is the top node of the V projection at this stage. If it further projects up, its categorial status will change to an intermediate projection of the V rather than remain as a maximal projection of the V.

arbitrarily as a lexical item enters an N. I depart from Chomsky, proposing that optional features are added at any point of a derivation:[11]

(39) Optional features are added arbitrarily at any point of a derivation.

In (38), recall that the Case feature of *Mary* is optional. Hence, *Mary* does not have any Case feature when it is selected from the N. We add an Accusative Case feature to *Mary* after it is selected from the N. We then merge *see* with *Mary* and check their Accusative Case features, constructing (38). Note that the operation of adding an optional feature to a lexical item is immune from the ICP, since it does not manipulate terms in phrase structures.

After having constructed *see Mary*, we merge *John* and *see Mary*:

(40) [$_V$^{max} John [see Mary]]

---

[11]The question that arises here is whether optional features are still elements associated with an N. I claim that lexical items have slots for its optional features (if it has ones) which are to be filled by specific features during a derivation. For example, a noun has a slot for its optional Case feature. The slot is to be filled by a specific Case feature during the derivation. Under this view, lexical items in an N only have slots for its optional features but not the features themselves. I claim that although optional features are not part of lexical items within an N, they are in an N as independent elements ("floating" in a sense) and later added to lexical items at any stage of a derivation in order to fill their slots. Note that it is plausible to assume that optional features exist as independent elements in an N, since, to be precise, lexical items themselves are collections of features. Alternatively, it might be possible to say that optional features are not included in an N. This view, however, would increase reference sets and hence induce computability problems. It is also against the idea that an N is what specifies an input for a derivation.

Chomsky (1995) considers two possible ways of introducing optional features. The one is to add optional features to a lexical item when an N is formed. The other is to add optional features when a lexical item is selected from an N and introduced into a derivation. He claims that both of these approaches are compatible with the MP. Between these two approaches, an N is the place where a lexical item and its optional features get together only in the former but not in the latter. Our view therefore can be regarded as a further extension of the latter approach . I will argue in chapter 5 that this interpretation of optional features is also instrumental in accounting for the distribution of *wh*-elements in-situ in languages like Japanese. I am indebted to Lisa Cheng (personal communication) for bringing this subject to my attention.

Note that the Case feature of *John*, being optional, has not been added at this point of the derivation yet.

The next step is to select the finite T. It has a Nominative Case feature, which is to be checked by an element in its specifier position. Apart from the Case feature, the finite T also has a D-feature and V-feature. As will be discussed in detail in section 2.4.2.4, these features are both uninterpretable. The V-feature is checked by merger of T with a projection of V. The D-feature is checked by the raising of D^{max} to the Spec of T^{max}. I argue that these formal features, i.e., the Nominative Case feature, the D-feature, and the V-feature, are not just listed in an unordered fashion in the lexical entry of the finite T. There is a hierarchical structure among these formal features within the lexical entry, which ensures a specific ordering among the applications of their checking operations. The finite T has the following hierarchical structure concerning its formal features, where NOM, D, and V represent the Nominative Case feature, the D-feature, and the V-feature, respectively:

(41) $T_{[[NOM,\ D]\ V]}$

Let us assume that only the structurally highest feature is accessible to a computation. Embedded features, which are "covered" by the higher features, are not accessible to a computation. In (41), the computation is only accessible to the V-feature, which is structurally highest, but not to the embedded Nominative Case feature and D-feature. It is only after the V-feature is eliminated by the checking operation that the Nominative Case feature and the D-feature become accessible to the computation. This ensures that the V-feature is checked by merger of T with its complement while the Nominative Case feature and the D-feature

are checked by the raising of D^{max} to the Spec of T^{max}. Putting it in traditional terms, the feature hierarchy (41) tells us that the finite T takes a projection of V as its internal argument and a D^{max} with a Nominative Case feature as its external argument.[12]

The hierarchical structure among formal features like (41) corresponds to the traditional argument structure, which is advocated by, among others, di Sciullo and Williams (1987), Grimshaw (1990), Levin and Rappaport (1986), Marantz (1984), Williams (1981, 1994), and Zubizarreta (1987), though the hierarchical ordering between external and internal arguments advocated here is opposite of what is assumed in their works. It should be noted that such a hierarchical structure within a lexical entry is needed anyway, since it is necessary to make a distinction between external and internal arguments. Since our feature hierarchy and the traditional argument structure are both meant to express the same fact, it is more desirable if they can be collapsed into one. I claim that the traditional argument structure should be replaced by our feature hierarchy. Feature-wise, an internal argument is higher than an external argument. Because of the ICP, the external argument appears in a structurally higher position than the internal argument in phrase structures.[13]

---

[12]Alternatively, it is conceivable that the notion of closeness comes into play. It has been assumed that the notion of closeness only plays a role when a triggering feature searches for its matching feature in its c-commanding domain within a single phrase structure. Let us assume contra this standard view that the notion of closeness is also relevant when a triggering feature searches for its matching feature within a lexical entry. The structurally higher within the feature hierarchy of a lexical entry a feature is, it is "closer" to the triggering feature. It then follows that at each stage in a derivation, it is only the "closest" feature within a lexical entry which can enter into a checking relation.

[13]I am indebted to Jim Huang (personal communication) for bringing this subject to my attention.

Returning to the derivation of (36), when we select the finite T, its V-feature becomes accessible to the computation. Note that the Nominative Case feature and the D-feature are "covered" by the structurally higher V-feature and thus not accessible to the computation at this stage. The ICP requires that the V-feature of the finite T should be checked immediately by merger of the finite T with the V^{max} *John see Mary*. The V- feature, being uninterpretable, is deleted and made invisible at LF.

Note that at this stage of the derivation, the Nominative Case feature and D-feature of the finite T become accessible to the computation. Recall that both of these features are uninterpretable and thus subject to the ICP. The question then arises as to how to check off more than one UFFs which become accessible to the computation at the same time without violating the ICP. I claim following Chomsky (1993, 1995) that when a certain operation checks one feature, it can also check other features simultaneously as "free riders." Hence, both of these UFFs can be checked off in conformity with the ICP. At this point, we first add an optional Nominative Case feature to *John*. Then, we check the Nominative Case feature and D-feature of T through the raising of *John* to the Spec of T^{max} in accordance with the ICP. The resultant structure is as follows, given the copy theory of Attract/Move:

(42) $[_{T^{max}}$ John [T $[_{V^{max}}$ John [see Mary]]]]

As illustrated above, we can construct (36) in conformity with the ICP. Hence, structural Case assignment, which was assumed to apply at S-structure in the EST, can be subsumed under the ICP.

2.4.2.3 Agreement Relations

Apart from structural Case assignment, agreement relations also counted as S-structure properties within the EST. The arguments for assuming agreement relations as S-structure properties were as follows. In some languages, agreement is morphologically realized while in others it is not. It has been claimed, however, that agreement features, called ϕ-features, should occur in a uniform way whether they are morphologically realized or not. Then, ϕ-features appear at PF. Furthermore, the ϕ-features of nouns receive interpretations at LF. Then, they also appear at LF. Hence, agreement was assumed to occur at S-structure before derivations bifurcate into PF and LF.

Chomsky (1993) argues that once we assume the checking theory, agreement relations can be reduced to conditions at the interface levels. Considering the verb-noun agreement as an example, it was assumed that both verbs and nouns are morphologically fully inflected before the checking operations for the verb-noun agreement take place. The agreement morphology is always realized at PF wherever the checking operations for the agreement take place. Hence, the realizations of agreement morphology at PF do not necessarily count as evidence in support of the view that agreement takes place at S-structure.

Concerning the LF interpretability of the ϕ-features of nouns, ϕ-features have already been assigned to both nouns and verbs within the lexicon. The ϕ-features of nouns therefore appear at LF wherever the checking operations for the verb-noun agreement take place. Hence, the LF interpretability of the ϕ-features of nouns does not constitute evidence for the S-structure property of agreement relations, either.

As I have pointed out repeatedly, Chomsky's checking theory, although it contributes to the elimination of the theory-internal linguistic levels, raises the problem of globality. I argue that agreement relations should also be subsumed under the ICP. Recall that while the ϕ-features of nouns are interpretable, those of verbs are uninterpretable. Hence, only the ϕ-features of verbs, but not those of nouns, are subject to the ICP.

Let us consider how the ICP approach to agreement relations works, taking (36) (repeated here as (43)) as an example. For expository purposes, the discussion of this subsection only explicates how ϕ-features are checked, ignoring Case features:

(43) John saw Mary

We first select the verb *see*. Since the ϕ-features of verbs are optional, *see* does not have any ϕ-features when selected. After selecting *see*, therefore, we add ϕ-features to it. The ϕ-features are hierarchically structured as follows:

(44) $\text{see}_{[[3\text{SM}]\ 3\text{SF}]}$

In (44), 3rd person, singular, male, and female features are represented as 3, S, M, and F, respectively. Since the ϕ-features of verbs are uninterpretable, they must be checked immediately when they become accessible to the computation. Since the ϕ-features of *see* are accessible to the computation at this point, they are checked by selecting *Mary*, which has the matching ϕ-features as its intrinsic property, and combining *see* with *Mary*. The resultant structure is as below:

(45) $[\text{V}^{\text{max}}\ \text{see}_{[3\text{SM}]}\ \text{Mary}_{[3\text{SF}]}]$

Note that if the ϕ-features of *see* do not coincide with those of *Mary*, the derivation is canceled due to a violation of the ICP. While the ϕ-features

of *see*, being uninterpretable, are deleted when checked, those of *Mary*, being interpretable, remain intact.

At this stage of the derivation, the remaining ϕ-features of *see* become accessible to the computation. Since they are uninterpretable, the ICP requires that they should be checked immediately. They are checked by selecting *John*, which has those ϕ-features as its intrinsic property, and combining *John* with *see Mary*:

(46) $[V^{max}$ John$_{[3SM]}$ [see Mary$_{[3SF]}$]]

The ϕ-features of *see*, being uninterpretable, are deleted when checked. Those of *John*, on the other hand, remain intact and receive interpretations at LF. In this way, agreement relations can be subsumed under the ICP.[14]

[14]In our system, the ϕ-features of verbs are deleted and made invisible at LF through checking operations. One might wonder how verbs are interpreted if their ϕ-features are deleted especially in polysynthetic languages (see, among others, Baker (1988, 1996)). It should be noted, however, our system is only claiming that the ϕ-features of verbs, being formal features, are deleted. Apart from formal features, verbs also have pure semantic features. Chomsky (1995) claims that formal features typically have their purely semantic correlates which reflect semantic properties (accusative Case and transitivity, for example). Hence, the nonexistence of the ϕ-features of verbs at LF does not necessarily mean that verbs do not have any features that receive interpretations at LF regarding their agreement. I am indebted to Lisa Cheng (personal communication) for bringing this subject to my attention.

2.4.2.4 Selectional Restrictions

It has been assumed by, among others, Abney (1987), Chomsky (1965, 1981, 1986b), and Fukui (1986) that lexical items can be classified into two types, i.e., thematic elements like Noun, Verb, Adjective, and Preposition and functional elements like Complementizer, Tense, and Determiner. Among several differences between these two types of elements is the way they impose restrictions on their arguments, i.e., the items which appear in their specifier and complement positions.[15]

Thematic items choose the thematic types of arguments they take. For instance, the verb *believe* in (47), being a two-place predicate, assigns the θ-role of Agent to its first argument *John* and the θ-role of Theme to its second argument *that Bill saw Mary*:

(47) John believes that Bill saw Mary.

It should be noted that exactly what kinds of θ-roles each thematic item assigns is irrelevant to the following discussion. We just use them as identifying the arguments of thematic items.

Functional elements, on the other hand, do not assign any θ-role, but only choose the categorial status of their complement. Another salient property of functional elements is that those belonging to the same category share their categorial selection property. All the functional elements belonging to C select T^{max} as their complement. Similarly, all the functional elements belonging to T and D select V^{max} and N^{max}, respectively, as their complements. In the following, I will use the notion selectional restriction as a cover term for the θ-role assignment properties of thematic items and the categorial selection properties of functional

---

[15]See Grimshaw (1979) and Pesetsky (1982) for further discussion of this subject.

items. It is important to note that the present definition of the notion selectional restriction differs from that of Chomsky (1965). The latter specifies the restrictions which verbs impose on the semantic features of their arguments like [+/- Human] and [+/- Abstract].

It was assumed in the EST framework that these selectional restriction properties of functional and thematic items should be satisfied at D-structure. This is because it was assumed that the base component, including the lexicon and the phrase structure component in the EST, should generate D-structure, respecting the selectional restriction properties of functional and thematic items. As shown in section 2.2.1, the θ-criterion coupled with the Projection Principle used to ensure that D-structure is a representation of GF-θ. Hence, the selectional restrictions of thematic items were assumed to be satisfied at D-structure. The selectional restrictions of functional items were also required to be satisfied at D-structure by the Projection Principle.[16] Under the MP, however, selectional restrictions, which were stated at the level of D-structure in the EST, must be reformulated either as conditions on the interface levels or constraints which apply throughout derivations.

Chomsky (1993, 1995) pursues the former possibility, arguing that the selectional restrictions of functional and thematic items should be satisfied at LF. As mentioned above, he totally eliminates the Projection Principle and claims that lexical items are selected from the lexicon freely at any point of a derivation. The θ-criterion, which was responsible for

---

[16] In the earlier stage of the EST where the phrase structure component is still posited, the selectional restriction properties of functional items are specified in terms of phrase structure rules. See, among others, Higginbotham (1983, 1985), Speas (1984) and Stowell (1981) for arguments in favor of the elimination of the phrase structure component.

stating the selectional restrictions of thematic items in the EST, is reformulated as a condition at LF. As I have extensively argued above, however, reformulating the properties of the theory-internal linguistic levels as conditions on the interface levels would raise a conceptual problem in that it needs global considerations. For this reason, I would rather pursue the latter possibility, i.e., reformulating selectional restrictions as constraints which apply throughout derivations, arguing that selectional restrictions should be subsumed under the ICP.

Let us consider how the ICP approach to selectional restrictions works. For expository purposes, the discussion of this subsection restricts itself to explicating how selectional restriction features can be checked in accordance with the ICP, ignoring Case and agreement features unless they become relevant to the discussion. Functional elements have categorial features to be checked through merger with their complements. For instance, C has the categorial feature [T], which is to be checked by its merger with a projection of T. T has the categorial feature [V], which is to be checked by its merger with a projection of V. D has the categorial feature of [N], which is to be checked by its merger with a projection of N. These categorial features of functional items are uninterpretable, since they do not specify the categorial status of the functional items themselves and thus do not receive any interpretations at LF. The categorial features of the complements, on the other hand, are interpretable, since the features specify the categorial status of the complements themselves.

Turning to thematic items, let us assume contra Chomsky (1995) that thematic features are formal features and thus accessible to a computation. Thematic items have thematic features, which are to be

checked by merger with their argument(s). For instance, the verb *see* has thematic features AGENT and THEME. Its lexical entry also specifies that the former thematic feature is checked by merger of the verb with an element in its specifier position while the latter is checked by merger of the verb with an element in its complement position. The verb *see* has the following structure concerning its thematic features, where AGENT and THEME features are represented as A and T, respectively:

(48) [[A] T]

The thematic features of thematic items are uninterpretable, since thematic items themselves do not receive any interpretations regarding thematic types like Agent and Theme at LF. The thematic features of the specifiers and complements, on the other hand, are interpretable, since they receive interpretations regarding their thematic types at LF.

Let us consider the embedded clause of (47) (repeated here as (49)) as an example:

(49) John believes that Bill saw Mary

We first select the verb *see*, which has uninterpretable thematic features AGENT and THEME:[17]

---

[17]This thematic feature hierarchy is exactly the opposite of the thematic hierarchy advocated by, among others, Foley and van Valin (1984), Jackendoff (1972), Nishigauchi (1984), Randoll (1988), Rappaport and Levin (1988), Schwartz (1988), and Wilkins (1988). It might be possible to claim that the thematic hierarchy is defined by our thematic feature hierarchy. Feature-wise, for instance, Theme is structurally higher than Agent. Because of the ICP, however, we derive the thematic hierarchy as a phrase structure generalization. As far as thematic items are concerned, it is possible to achieve the same hierarchical effect without recourse to any feature hierarchy. Let us assume the lexical decomposition approach advocated by, among others, Chomsky (1995), Hale and Keyser (1993), and Huang (1994). Then, each thematic item is only associated with one thematic feature. Regarding the selectional restrictions of functional elements, however, we still need a feature hierarchy to achieve the hierarchical effects in phrase structures. I am indebted to Jim Huang (personal communication) for bringing this subject to my attention.

(50) $see_{[[A]\ T]}$

When the verb *see* is selected from the N, the ICP requires that THEME should be checked immediately. It should be noted that AGENT, being embedded in the structure of thematic features and thus not accessible to the computation, cannot be checked at this point of the derivation. Since the thematic features of nominals are optional, we add THEME feature to *Mary*. The THEME feature of *see* is checked by selecting *Mary* and combining *see* with *Mary*, in accordance with the ICP:

(51) $[V^{max}\ see_{[A]}\ Mary_{[T]}]$

The THEME feature of *see*, being uninterpretable, is deleted when checked. The THEME feature of *Mary*, on the other hand, remains, since it is interpretable. At this point of the derivation, the AGENT feature of *see* becomes accessible to the computation. The ICP requires that it should be checked immediately. We select *Bill* from the N and add an AGENT feature to it, since its thematic feature is optional. We then merge *Bill* with *see Mary*, resulting in the following structure:

(52) $[V^{max}\ Bill_{[A]}\ [see\ Mary_{[T]}]]$

While the AGENT feature of *see*, being uninterpretable, is deleted, that of *Bill*, being interpretable, remains intact. It should be noted that if the thematic types of *see* do not coincide with those of its arguments, the derivation is canceled due to a violation of the ICP.

The next step is to select the finite T from the N. The finite T, being a functional item, has a V-feature as its selectional restriction property as well as a D-feature as its strong feature. These categorial features are hierarchically structured within the lexical entry of the finite T:

(53) $T_{[[D]\ V]}$

This ensures that the finite T takes a projection of V as its internal argument and a projection of D as its external argument. Since both of these categorial features are uninterpretable, the ICP requires that they should be checked immediately when they become accessible to the computation. When T is selected, only its V-feature is accessible to the computation. Hence, the V-feature must be checked immediately by merger of T with V^{max}. After the V-feature of T undergoes deletion through the checking operation, the D-feature of T becomes accessible to the computation. The D-feature must be checked immediately by the raising of *Bill* to the Spec of T^{max} at this point of the derivation in accordance with the ICP. The D-feature, being uninterpretable, is deleted when checked. The resultant structure is as follows:

(54) $[_{T^{max}}$ $Bill_{[A]}$ $[T$ $[_{V^{max}}$ $Bill_{[A]}$ [see $Mary_{[T]}$]]]]

Finally, we select the complementizer *that* from the N. The complementizer *that*, being a functional item, has the categorial feature [T] as its selectional restriction property. This categorial feature is uninterpretable and thus must be checked immediately by merger of *that* with T^{max}. The categorial feature [T] of *that*, being uninterpretable, is deleted when checked:

(55) $[_{C^{max}}$ that $[_{T^{max}}$ $Bill_{[A]}$ $[T$ $[_{V^{max}}$ $Bill_{[A]}$ [see $Mary_{[T]}$]]]]]

It was shown that the selectional restriction properties of lexical items, which were assumed to be D-structure properties in the EST, can be subsumed under the ICP. Unlike Chomsky's (1995) approach, the former D-structure properties are now reduced to the constraint which applies throughout derivations without recourse to any interface conditions.

To summarize section 2.4.2, I have shown that the displacement property, structural Case assignment, agreement relations, and selectional restrictions, which were assumed to be D-structure or S-structure properties in the EST, can be subsumed under the ICP. Left untouched among the properties of the theory-internal linguistic levels in the EST is the binding theory, which I will discuss in the next section.

2.5 Consequences of the Immediate Checking Principle

This section considers theoretical consequences of the ICP. Section 2.5.1 investigates interpretation of the ICP in the theory of language. Section 2.5.2 claims that if we assume the ICP, we can virtually eliminate the notion of LF-convergence. Section 2.5.3 shows that the ICP is a language-specific computational device which greatly contributes to the reduction of globality in the theory of language.

2.5.1 Interpretation of the Immediate Checking Principle

The above discussion has shown that the ICP enables us to capture the D-structure and S-structure properties in the EST. Especially, since the ICP accommodates the displacement property, it subsumes the principle of Last Resort advocated by Chomsky (1991a) and Chomsky and Lasnik (1993) (or its modified principles, Greed proposed by Chomsky (1993, 1995) and Suicidal Greed proposed by Chomsky (1996)), which was intended to capture that property. Chomsky (1995, 1996) argues that Last Resort should not count as an economy condition, since it is

inviolable. It should rather be part of the definition of Attract/Move-F, as can been seen in Chomsky's (1995) definition:[18]

(56) Attract/Move-F

F raises to target K only if F enters into a checking relation with a sublabel of K (where a sublabel of K is a feature of the zero-level projection of the head H(K) of K).

(adapted from Chomsky (1995))

Our analysis claims, on the contrary, that the ICP, which subsumes Last Resort, is an independent principle rather than part of the definition of Attract/Move. This is because, as argued above, the ICP constrains not only Attract/Move but also Merge. If the ICP were part of the definition of Attract/Move, then it would also be part of the definition of Merge. This would be theoretically undesirable, however, since it is redundant to incorporate the ICP into the definitions of the two distinct operations. Concerning Chomsky's argument for incorporating Last Resort as part of the definition of an operation, we should recall that the ICP is inviolable. There is therefore no need to incorporate the ICP into definition of an operation in order to derive the inviolability of the ICP.[19]

---

[18]This definition differs from Chomsky's (1995) original one in the following respects. First, Chomsky (1995) totally eliminates the notion of Move, arguing that the traditional notion of movement should be reinterpreted as Attract-F. Under this view, the locus of the notion is completely shifted from the moved element to the target. We are assuming, on the other hand, that the notion of Move is still needed. The traditional operation of movement is reinterpreted as Attract/Move-F rather than Attract-F. Second, Chomsky incorporates the Minimal Link Condition (MLC) into the definition of Attract/Move-F. I argue contra Chomsky that the MLC is also an independent principle. As will be discussed soon, our analysis claims that the last resort condition on Attract/Move should be subsumed by the ICP, which is an independent principle. Hence, the MLC, which only regulates the operations satisfying the ICP, should also be an independent principle under our analysis.

[19]Collins (1997) also argues that his economy conditions, Last Resort and Minimality, should not be part of the definition of Attract/Move.

Given that the ICP is an independent principle, a question then arises as to interpretation of the ICP in the theory of language. I argue that the ICP is a local "heuristic algorithm" ("computational trick") which gives us an approximate solution to a computationally intractable problem induced by the global interface condition on UFFs. Recall that UFFs are those which are illegitimate at LF. According to the BOC-driven optimal design of language assumed by the MP, we have an interface condition which requires that UFFs, being illegitimate at LF, should not remain at that level. If they remain at LF, the derivation crashes at that level. As argued above, however, the postulation of the interface condition on UFFs alone necessarily leads to computational intractability. This is because triggering a checking operation of a UFF in terms of the interface condition would necessarily need global considerations. It would then follow that the part of language involving UFFs, whose corresponding optimization problem is computationally intractable, is not usable in practice, contrary to fact. Hence, there should exist a local "heuristic algorithm" ("computational trick") which gives us an approximate solution to this computationally intractable problem, making that part of the language usable in practice.

I argue that it is the ICP which plays this role. First, the ICP brings about an approximate solution to this computationally intractable problem. Suppose that we have an option of applying a checking operation of a UFF at a certain stage of a derivation. In most cases, the ICP gives us the same answer to this computational problem as the interface condition on UFFs. In other words, the ICP only requires us to apply the checking operation when the interface condition on UFFs would also require us to do so. For example, when we have an option of

applying overt subject raising to check a strong D-feature of T during a derivation, the ICP requires us to apply that operation, and so would the interface condition on UFFs. Hence, in such cases, the ICP brings about a perfect solution to the computationally intractable problem induced by the interface condition on UFFs.

There are, however, cases where the ICP does not give us the same answer to a computational problem as the interface condition on UFFs. For example, suppose that we select V, which has a UFF that can be checked by its merger with D^{max}, which is in the N. Suppose further that we also have a so called VP-adverb in the N. Here, we have two options. We can select D^{max} and merge it with V. Alternatively, we can select the adverb and merge it with V. Recall that while merger of V with D^{max} is triggered by the UFF of V, merger of V with the adverb is not triggered by any UFF. The interface condition on UFFs would claim that we may apply either of these operations at this stage as long as the UFF of V eventually enters into a checking relation before LF. The ICP, on the other hand, only allows us to apply merger of V with D^{max} at this stage, preventing us from applying merger of V with the adverb. This is because the ICP requires us to apply the checking operation of the UFF of V before any other operations that manipulate terms. In cases like this, the ICP does not gives us the same answer to a computational problem as the interface condition on UFFs. Hence, the solution which the ICP brings about to the computationally intractable problem induced by the interface condition on UFFs is only approximate.

Furthermore, the ICP is local. This is because when we have an option of applying a checking operation of a UFF at a certain stage of a

derivation, it can decide whether to apply the checking operation only on the basis of information available at that stage.

In conclusion, the interface condition on UFFs, which is required by BOCs, fundamentally induces globality and thus its corresponding optimization problem is computationally intractable. The ICP, being a local "heuristic algorithm" ("computational trick") for this computationally intractable problem, reduces computational burden and facilitates usability of language in practice. Hence, it is conceptually plausible to claim that UFFs are subject to the ICP. It should also be noted that the ICP would make a derivation canceled before it reaches LF which would otherwise crash due to a violation of the interface condition on UFFs. If we conform to the ICP throughout derivations, UFFs never remain at LF and the interface condition on UFFs is always satisfied. Hence, the interface condition on UFFs virtually plays no role once we assume the ICP.

It is important to point out that the ICP approach supports the language design that considerations of computational complexity do not matter for fundamental aspects of language. Under this view, globality is one of the fundamental properties of language. Its corresponding optimization problem is fundamentally intractable. There are, however, language-specific computational devices, "heuristic algorithms" ("computational tricks"), which reduce its fundamental globality to local properties, facilitating its usability in practice. The ICP is one such language-specific computational device. Hence, if our ICP approach is on the right track, it constitutes evidence against the language design assumed by, among others, Collins (1997) that considerations of

computational complexity matter for fundamental aspects of language and no globality is allowed.

It might be possible to claim that the ICP is also compatible with the language design that considerations of computational complexity matter for fundamental aspects of language. Under this design of language, the ICP would either count as the defining property of UFFs or as a local economy condition in the sense of Collins (1997). We are reluctant to take this view, however, since we would lose an explanation of why UFFs are subject to the ICP. Recall that we are claiming that UFFs are subject to the ICP because the latter gives us an approximate solution to the computationally intractable problem induced by the BOC-driven interface condition on UFFs. Without having the ICP based on the interface condition on UFFs, there would be no reason why only UFFs, but not the other features, must be checked and deleted immediately.

To summarize, the ICP, being an independent principle, functions as a local "heuristic algorithm" ("computational trick") for the computationally intractable problem induced by the interface condition on UFFs. This explains why only UFFs, but not the other features, must be checked and deleted immediately when they become accessible to a computation.

2.5.2 Elimination of the Notion of LF-Convergence

Another theoretical consequence of the ICP is that it enables us to virtually eliminate the notion of LF-convergence. Recall that under Chomsky's (1993, 1994, 1995) framework, a derivation converges when its interface levels only consist of legitimate objects and thus satisfy BOCs. Otherwise, the derivation crashes. UFFs like strong features, Case

features, and the ϕ-features of verbs, are illegitimate objects at LF. If they are checked and deleted (made invisible at LF) during a derivation, the derivation converges at LF. If they remain unchecked and undeleted, the derivation crashes at LF. Hence, the notion of LF-convergence is needed.

The ICP requires, on the other hand, that UFFs should be checked and deleted (made invisible at LF) immediately when they become accessible to a computation. Otherwise, the derivation is canceled. Thus, it never happens that a derivation reaches LF with UFFs being left unchecked and undeleted. The ICP makes it mandatory that UFFs are checked and deleted (made invisible at LF) before a derivation reaches that interface level. The interface condition on UFFs is therefore always satisfied. In other words, a derivation which reaches LF is always convergent as far as UFFs are concerned. Hence, the ICP virtually eliminates the notion of LF-convergence regarding UFFs.

There are, however, other conditions which are assumed to apply at LF, i.e., the binding theory, the uniformity condition on chains, and the condition on the operator-variable construction, which consists of the bans against vacuous quantification and free variables.[20] One might say that we still need the notion of LF-convergence for these principles. It is not clear, however, whether violations of these principles would lead derivations to crash at LF. It might be possible to claim that the binding theory, the uniformity condition on chains, and the condition on the operator-variable construction purely count as principles of interpretation on the LF-interface level. They never play any role in making

[20]See Fukui (1993b) for the alternative view that the uniformity condition on chains should apply derivationally.

derivations (computations) converge or crash. Their violations would not make derivations crash, but only make their interpretations deviant.[21]

There is another principle where the notion of LF-convergence plays a crucial role, that is, the economy conditions like the Minimal Link Condition (MLC) and the principle of Procrastinate. Recall that Chomsky (1993, 1994, 1995) assumes that the economy conditions only compare convergent derivations. Among convergent derivations, the economy conditions choose the most "economical" one, blocking all the others.[22] Several attempts have been made to eliminate the economy conditions and find alternative treatments. Chomsky (1995) pursues the possibility of incorporating the MLC as part of the definition of Attract-F. Collins (1997) suggests that the principle of Procrastinate should be subsumed under the modified definitions of strong/weak features. As far as we can do away with the economy conditions, we do not need the notion of LF-convergence for the purpose of the economy conditions either.

If the above line of reasoning is correct, we can virtually eliminate the notion of LF-convergence through nullifying the effects of the interface condition on UFFs in terms of the ICP. I must admit, however, that the discussion of this subsection is only sketchy and much still remains to be done.

---

[21]Among the conditions of the binding theory, Condition A might make reference to formal features and thus play a role in making derivations converge or crash. See, for example, Chomsky (1986b, 1993) for this view of Condition A of the binding theory, where reflexives and reciprocals are assumed to undergo movement at LF. See Battistella (1989), Cole, Hermon, and Sung (1990), Huang and Tang (1991), Katada (1991), Lebeaux (1983), and Pica (1987) for further arguments in favor of the LF-movement analysis of anaphors.

[22]See Ura (1995) for the different view that economy conditions compare not only convergent but also non-convergent derivations.

2.5.3 Reduction of Globality to Local Properties

The MP requires us to reformulate the D-structure and S-structure properties in the EST either as interface conditions or as derivational constraints. I have investigated the interface condition approach, arguing that it is conceptually problematic, though empirically adequate in capturing the D-structure and S-structure properties in the EST. This is because the interface condition approach has the "look-ahead" property and thus necessarily induces globality. Pursuing the derivational constraint approach, I have proposed the ICP, which is conceptually desirable in that it greatly contributes to reducing globality to local properties. It should be noted that we are arguing against globality not against interface conditions. Interface conditions do not induce any globality problems as far as they are local in nature. This is a good place to consider whether other interface conditions are local and therefore conceptually desirable.

Under our analysis, the remaining interface conditions are the binding theory, the uniformity condition on chains, and the condition on the operator-variable construction. It is conceivable that these interface conditions are all local in nature. Recall that global interface conditions like Chomsky's (1993) interface condition on UFFs require us to look at the interface levels in order to decide whether to apply an operation at an intermediate stage of a derivation. Global interface conditions, with this "look-ahead" property, necessarily induce computational intractability and thus raise conceptual problems. The binding theory, the uniformity condition on chains, and the condition on the operator-variable construction, on the other hand, can plausibly be interpreted as local

interface conditions which only make reference to LF. Since these interface conditions have nothing to do with formal features, they should never trigger the application of an operation during a computation given the minimalist assumption that only formal features are accessible to a computation.[23] It follows that these interface conditions may not trigger any operation during a computation in order to avoid their violations at the LF interface level. The LF-representations which violate any of these interface conditions are simply ruled out. Since these interface conditions as such formulated only make reference to a single stage in a derivation, i.e., the LF interface level, they are local in nature.[24]

As a result, we can largely dispense with globality in the theory of language. What remains global is the economy conditions which crucially rely on the notion of convergence. As mentioned in the previous subsection, however, the economy conditions may possibly be eliminated. As far as we can do away with the economy conditions, we can totally eliminate globality in the theory of language.

---

[23]As mentioned in note 21, Condition A of the binding theory might involve formal features.

[24]Tsai (1994) claims that the condition on the operator-variable construction triggers overt category movement. His analysis, however, is conceptually problematic. First, it has the "look-ahead" property and thus necessarily induces globality. Second, it is against the minimalist view that only formal features are accessible to a computation. His analysis would require a computation to make reference to other entities than formal features, since the condition on the operator-variable construction does not involve any formal features.

2.6 Adjuncts and the Immediate Checking Principle

Before closing this chapter, let us consider a consequence of the ICP for the theory of phrase structure. I will propose the Earliness Principle (EP) on Select, a local "heuristic algorithm" ("computational trick") for a global interface condition on a numeration N, arguing that the ICP coupled with the EP gives rise to an asymmetry concerning the composition of phrase structure. I will argue that while the terms required by UFFs are merged cyclically, those not required by UFFs are merged postcyclically. It then follows that arguments, which are required by UFFs, are merged cyclically. On the other hand, typical adjuncts, which are not required by any UFFs, are "hooked-up" to the skeletal argument structure postcyclically. It is shown that postcyclic merger of adjuncts gives us a derivational way of capturing the argument/adjunct distinction, which is compatible with the minimalist spirit and thus theoretically desirable. I will then investigate Lebeaux's (1988, 1991) theory of phrase structure, which also claims that the argument/adjunct distinction should be made derivationally. It is shown that under Lebeaux's theory, adjuncts are allowed to be merged postcyclically, which is in contrast to our theory where adjuncts are required to be merged postcyclically.

2.6.1 Postcyclic Merger of Adjuncts

The ICP requires that UFFs should be checked immediately when they become accessible to a computation. In other words, when a UFF becomes accessible to a computation, we have to apply its checking operation before any other operations that manipulate terms. Hence, if we have two possible continuations at any stage of a derivation, one is

triggered by an UFF and the other is not, then the ICP requires that we should always choose the former option. It then follows that typical adjuncts, whose merger is not triggered by any UFF, are forced to be merged postcyclically.

Let us consider in detail why this generalization follows from the ICP, taking (57) as an example:

(57) Bill said that John saw Suzy after he met Mary.

We will consider how to construct the structure of (57) under the interpretation where the adjunct *after he met Mary* modifies the embedded clause. (57) can be divided into two parts; what we call the main structure, i.e., *Bill said that John saw Suzy*, and what we call the adjunct, i.e., *after he met Mary*.

Let us first look at how to construct the adjunct, i.e., *after he met Mary*. We first select the verb *meet*, whose lexical entry is as follows:

(58) $\text{meet}_{[[A]\ T,\ ACC]}$

Recall that since the ϕ-features of verbs are optional, *see* does not have any ϕ-features when selected. After selecting *see*, we add ϕ-features to it, resulting in the following hierarchical structure among its features:

(59) $\text{meet}_{[[A,\ 3SM]\ T,\ ACC,\ 3SF]}$

At this point of the derivation, the THEME, Accusative Case, and [3rd person, singular, female] features of *meet* are accessible to the computation. Since these features are uninterpretable, the ICP requires that they should be checked immediately by selecting *Mary* and combining *meet* with *Mary*. Note that *Mary* has [3rd person, singular, female] as its intrinsic feature. Its THEME and Accusative Case features, being optional, are added after it is selected:

(60) $[V^{max}\ \text{meet}_{[A,\ 3SM]}\ \text{Mary}_{[T,\ 3SF]}]$

The Accusative Case features of *meet* and *Mary*, being uninterpretable, are deleted when checked. While the THEME and ϕ-features of *Mary*, being interpretable, remain when checked, those of *meet*, being uninterpretable, are deleted.

At this stage of the derivation, the AGENT and [3rd person, singular, male] features of *meet* become accessible to the computation. These features, being uninterpretable, must be checked immediately by selecting *he* and combining *he* with *meet Mary* in accordance with the ICP. Note that *he* has the ϕ-features as its intrinsic property. Its AGENT feature, being optional, is added after it is selected:

(61) $[V^{max}\ he_{[A,\ 3SM]}\ [meet\ Mary_{[T,\ 3SF]}]]$

While the AGENT and [3rd person, singular, male] features of *he*, being interpretable, remain when checked, those of *meet*, being uninterpretable, are deleted.

When we come to the stage of the derivation where (61) is constructed, the next step is to select the finite T. Recall that the finite T has the following hierarchical structure of features:

(62) $T_{[[D,\ NOM]\ V]}$

Since the V-feature of T, being uninterpretable, is accessible to the computation at this stage, the ICP requires that it must be checked immediately by merger of T with V^{max}, as shown below:

(63) $[T^{max}\ T_{[D,\ NOM]}\ [V^{max}\ he_{[A,\ 3SM]}\ [meet\ Mary_{[T,\ 3SF]}]]]$

At this point, the Nominative Case feature and D-feature of T, being uninterpretable, become accessible to the computation. They are checked by the raising of *he* to the Spec of T^{max}, conforming to the ICP:

(64) [T^{max} $he_{[A, 3SM]}$ [T [V^{max} $he_{[A, 3SM]}$ [meet $Mary_{[T, 3SF]}$]]]]

Note that T can never be selected at the earlier stage of the derivation. If it were selected before V^{max} is constructed, its UFFs could not be checked immediately; this would lead to a violation of the ICP.

The next step must be to select the P *after*. The ICP requires that the selectional restriction feature of *after*, which states that it takes T^{max} as its complement, should be checked immediately by combining *after* with the T^{max} (64):

(65) [P^{max} after [T^{max} $he_{[A, 3SM]}$ [T [V^{max} $he_{[A, 3SM]}$ [meet $Mary_{[T, 3SF]}$]]]]]

Note again that the P *after* cannot be selected at the earlier stage of the derivation due to the ICP. As shown above, we can construct the adjunct *after he met Mary* through checking the UFFs of the selected lexical items, conforming to the ICP.

Let us next look at how to construct the main structure, i.e., *Bill said that John saw Suzy*. In its derivation, we come to the stage where structure (66) is constructed. Although we do not look at how to construct this structure in detail, we can construct it through checking the UFFs of the selected items, conforming to the ICP:

(66) [T^{max} $John_{[A, 3SM]}$] [T [V^{max} $John_{[A, 3SM]}$ [see $Suzy_{[T, 3SF]}$]]]]

Let us suppose that the adjunct clause (65), i.e., *after he met Mary*, should be adjoined to the T^{max} (66) for its proper interpretation at LF. Then, there are two possible continuations at this stage of the derivation: (i) Selection of C and merger of C with the T^{max} (66), or (ii) Merger of the adjunct clause *after he met Mary* with the T^{max} (66). I propose the

Earliness Principle (EP) on Select (67) and argue that it is selection of C that applies at this stage of derivation:[25]

(67) Earliness Principle (EP) on Select

Lexical items must be selected from a numeration (N) as early as possible.

The EP requires that lexical items should be selected from an N at the earliest possible stage of a derivation unless it would lead a derivation to be canceled due to a violation of the ICP.

In the present case, since merger of the adjunct clause with the main structure is not triggered by any UFF and thus not required by the ICP to be applied immediately, selection of C should apply because of the EP. When C is selected, the ICP requires that the next step should be to combine C with T^{max} in order to check the selectional restriction feature of C, as shown below:[26]

---

[25]Note that the EP on Select differs from Pesetsky's (1989) earliness condition, which states that filters should be satisfied as early as possible on the hierarchy of levels S-structure > LF. Pesetsky's earliness condition essentially requires that overt operations should be preferred over covert operations. The EP on Select, on the other hand, makes a decision among choices at each step of a derivation within the overt component.

[26]One might wonder what ensures that we select C, but not some other element, at this stage. Note that all selected lexical items are immediately subject to Merge except the first selected lexical item in a derivation. Ignoring the first selected item, therefore, we have a condition which states that whenever Select takes place, it must be immediately followed by Merge. It then follows that we are forced to select C at this stage. This is because if we selected any other lexical item than C, the selected lexical item could not be immediately merged with any structure. Hence, no other lexical item than C may be selected at this stage of the derivation.

One might argue that the EP is not local and its corresponding computational problem is computationally intractable, since we have to look at the next stage of a derivation to decide which lexical item is to be selected. We have to look ahead to see whether the selected item may be immediately merged with any structure. I argue, however, that it does not induce any computational intractability. Its corresponding problem has a solution which needs only a polynomial amount of work. Suppose that the EP requires us to select a lexical item from an N at a stage Σ of a derivation. Suppose further that there are n lexical items in N whose index is not reduced to zero at Σ. In order to decide which lexical item is to be selected, n lexical items each must be inspected to see whether it may be immediately merged with any structure if selected.

(68) [$_{C}$max that [$_{T}$max John$_{[A, 3SM]}$ [T [$_{V}$max John$_{[A, 3SM]}$ [see Suzy$_{[T, 3SF]}$]]]]]

The derivation proceeds further, checking the UFFs of the selected items, resulting in the following structure:

(69) [$_{T}$max Bill$_{[A, 3SM]}$ [T [$_{V}$max Bill$_{[A, 3SM]}$ [say [$_{C}$max that [$_{T}$max John$_{[A, 3SM]}$ [T [$_{V}$maxJohn$_{[A, 3SM]}$ [see Suzy$_{[T, 3SF]}$]]]]]]]]]

It is important to point out that until this final stage of the derivation, the adjunct (65) has not been allowed to be merged with the main structure due to the ICP and the EP. At this final stage of derivation, we can combine the adjunct (65) with the main structure (69) by adjoining the former to the embedded T^{max} of the latter.

Like the ICP, the EP also receives strong conceptual support. Recall that a numeration N consists of lexical items and their index. Chomsky (1993, 1994, 1995) proposes a condition on N, which states that if the index is not reduced to zero at the interface levels, the derivation crashes. As argued by Collins (1997), however, this condition is global in nature, since it requires us to look at the interface levels in order to decide whether to apply Select at a certain stage of a derivation. Its corresponding optimization problem is therefore computationally intractable. It would then follow that this part of the language is unusable in practice, contrary to fact. Hence, there should exist a local "heuristic algorithm" ("computational trick") which brings about an

---

If the review of one lexical item requires one step of work, this problem requires *n* steps of work. Since this problem has a solution which only requires a polynomial amount of work, it belongs to Class P. Hence, it is computationally tractable.

approximate solution to this computationally intractable optimization problem. I argue that the EP serves for this purpose.

Let us first consider the local property of the EP. Note first that the EP is overridden by the ICP. Recall that the EP requires that we should exhaust an N as early as possible unless it would lead a derivation to be canceled. If the ICP requires some other operation to apply at a certain stage of a derivation in order to check a UFF, the selection of a lexical item cannot be applied at that stage. This is because if we applied the select operation, the derivation would be canceled due to a violation of the ICP. It then follows from the EP together with the ICP that at each step of a derivation, a lexical item should be selected from an N unless there exists any UFF accessible to a computation. To be specific, suppose that we come to a stage Σ of a derivation where we have an option of applying Select. If there is a UFF which is accessible to the computation at Σ, we should not apply Select but rather apply an operation to check the UFF. If there is no UFF which is accessible to the computation at Σ, we should apply Select. Hence, the EP is local in that when we come to Σ where we have an option of applying Select, it can determine whether to apply Select only on the basis of information available at Σ.

The EP also brings about an approximate solution to the computationally intractable optimization problem induced by the interface condition on N. In most cases, the EP gives us the same answer to a computational problem as the interface condition on N. In other words, the EP only requires us to apply Select when the interface condition on N would also require us to do so. For example, suppose that we have constructed T^{max}. Here, we have an option of selecting C from

the N. The EP requires us to apply that operation, and so would the interface condition on N. Hence, in such cases, the EP brings about a perfect solution to the computationally intractable problem induced by the interface condition on N. There are, however, cases where the EP does not give us the same answer to a computational problem with the interface condition on N. For example, suppose that as in the derivation of (57) discussed above, we come to a stage Σ where there are two independent syntactic objects, T^{max} and an adjunct which modifies the T^{max}. We have two options at Σ. We can select C and merge it with T^{max}. Alternatively, we can merge the adjunct with T^{max}. The interface condition on N allows us to apply either of these operations at Σ as long as C is eventually selected before the derivation reaches the interface levels. The EP, on the other hand, only allows us to select C at Σ, excluding merger of the adjunct with T^{max}. In such cases, the EP does not give us the same answer to a computational problem with the interface condition on N. Hence, the solution which the EP brings about to the computationally intractable problem induced by the interface condition on N is only approximate.

In conclusion, the EP on Select receives conceptual support from the fact that it serves as a local "heuristic algorithm" ("computational trick") which gives us an approximate solution to the computationally intractable problem induced by the condition on N. It should also be noted that if we conform to the EP during a derivation, it never happens that the derivation reaches the interface levels with any lexical items being left unselected from N. The condition on N is therefore always satisfied and virtually plays no role once we assume the EP.

To summarize, if we conform to the ICP and the EP during derivations, adjuncts, whose merger is not triggered by any UFF, are required to be merged postcyclically.

2.6.2 A Derivational Notion of Adjuncts

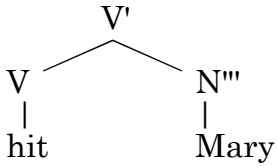
It has been assumed in the pre-minimalist period (see, among others, Chomsky (1972) and Jackendoff (1977)) that the argument/adjunct distinction is made representationally. Given the X-bar theory, while arguments are attached under X'-level, adjuncts are attached under higher-bar levels. Let us consider the following:

(70) a. John hit Mary

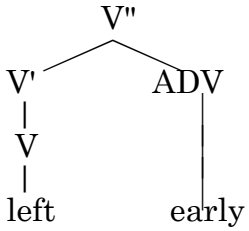
b. John left early

While *Mary* in (70a) is the argument of the verb *hit*, *early* in (70b) is an adjunct. Under Jackendoff's (1977) X-bar theory where the uniform three-level hypothesis is adopted, for instance, (70a) and (70b) are represented as in (71a) and (71b), respectively, with the irrelevant parts being ignored:

(71) a.



b.



While *Mary* in (70a), being an argument, is attached under V', *early* in (70b), being a restrictive modifier and thus an adjunct, is attached under V".

Such a representational argument/adjunct distinction, however, is no longer available in the MP. Recall that the MP requires that phrase structures should be "bare." Crucially, neither non-branching nodes like V' in (71b) nor bar-levels in the sense of the X-bar theory are allowed any more. Hence, we need an alternative way of making the argument/adjunct distinction which is compatible with the minimalist spirit.

The ICP coupled with the EP gives us a minimalist way of capturing the argument/adjunct distinction. Our theory claims that arguments are merged cyclically whereas adjuncts are merged postcyclically. Arguments and adjuncts are therefore distinguished by means of derivational terms instead of representational terms. Our derivational notion of adjuncts is theoretically desirable. This is because it does not rely on the notion of non-branching nodes or bar-levels and thus can be accommodated under the bare phrase structure in accordance with the minimalist spirit. I will argue in the following chapters that apart from this conceptual support, there is strong empirical support in favor of this derivational notion of adjuncts.

2.6.3 Lebeaux's (1988, 1991) Analysis

Lebeaux (1988, 1991) also argues for a derivational distinction between arguments and adjuncts. He proposes a heterogeneous licensing of phrase structures within the framework of the EST, arguing that elements in adjunct-of relations are licensed in a different way from those in argument-of relations. Following Chomsky (1981), Lebeaux assumes that the Projection Principle holds at all levels of representation. Then, elements in argument-of relations must be present at all levels of

representation, crucially at D-structure. Elements in adjunct-of relations, not being subject to the Projection Principle, may not be present at D-structure. They may be added by the operation called Adjoin-α in the course of derivations.

Our ICP analysis and Lebeaux's Adjoin-α analysis agree on the view that there is an argument/adjunct asymmetry with respect to merger. They differ, however, as to obligatoriness/optionality of postcyclic merger of adjuncts. Our theory claims that adjuncts must be merged after argument-of relations are established. Lebeaux's theory, on the other hand, claims that adjuncts may be merged after argument-of relations are established. In the following chapters, I will argue that our theory should be preferred over Lebeaux's theory, presenting various empirical facts which only follow from our theory but not from Lebeaux's theory.[27]

2.6.4 Strict Cyclicity and Postcyclic Merger of Adjuncts

It has been proposed by, among others, Chomsky (1993), Collins (1997), Kitahara (1995, 1997), and Watanabe (1995) that merger should be subject to strict cyclicity. Since our analysis allows countercyclic merger of adjuncts, this is a good place to consider empirical evidence which has been presented in support of cyclic application of merger. It is

---

[27]Fukui (1986) also argues for the argument/adjunct asymmetry with respect to structure building within the EST framework. As a consequence of his theory of phrase structure, he claims that non-thematic elements appear after every thematic element has been introduced within a clause. Our analysis agrees with Fukui's in claiming that non-thematic elements are forced to be introduced into derivations after thematic elements. In this respect, we can say that our analysis counts as a minimalist extension of Fukui's analysis. Our analysis, however, differs from Fukui's in that the latter only regulates the order of merger between arguments and adjuncts within a clause. In complex sentences, Fukui's analysis allows the embedded adjuncts to be merged earlier than the matrix arguments whereas our analysis does not.

shown that empirical arguments for cyclic merger are only based on merger of arguments but not on merger of adjuncts. Hence, they can be accommodated under our analysis, where only adjuncts, but not arguments, are allowed to be merged countercyclically. I will also argue that cyclic merger of arguments straightforwardly follows from the ICP coupled with the EP without recourse to any extra device.

Let us consider empirical arguments which have been presented in favor of cyclic merger. First, as pointed out by Collins (1997), under the AGRo or small *v* theory of clausal structure, if merge were allowed to apply countercyclically, then examples like (72) would be acceptable:

(72) *Mary John hit
'John hit Mary'

Suppose that *Mary*, which has a Nominative Case feature, is generated in the complement position of the verb *hit*, yielding *hit Mary*. Suppose further that before *John* is inserted in the Spec of VP, *hit Mary* is merged with AGRo or *v* and the resultant structure is merged with T. Then, since *John* is not present, the raising of *Mary* to the Spec of TP does not violate the MLC. After the raising of *Mary* to the Spec of TP, *John*, which has an Accusative Case feature, is inserted in the Spec of VP countercyclically. Its Accusative Case feature moves to AGRo or *v* to be checked off in the covert component. Hence, if we allowed countercyclic merger, examples like (72) would be acceptable, contrary to fact.

Second, the relativized minimality effects would not be explained by the MLC if countercyclic merger were allowed. The MLC is intended to subsume the relativized minimality effects, i.e., superraising (73a), the Wh-island Constraint (73b), and the Head Movement Constraint (73c):

(73) a. ***John** seems that it is certain ***t*** to be here

b. ***how**$_{\mathbf{j}}$ did John wonder **what**$_{\mathbf{i}}$ Mary fixed $\boldsymbol{t_i}$ $\boldsymbol{t_j}$

c. ***fix** John can ***t*** the car

If countercyclic merger were allowed, we would lose an MLC account of the relativized minimality effects. In (73a), it would be possible to raise *John* directly to the matrix Spec of TP and later insert *it* in the embedded Spec of TP countercyclically. In (73b), we could raise *how* directly to the matrix Spec of CP. Then, we could countercyclically insert the embedded C, which would attract *what*. In (73c), it would be possible to raise *fix* directly to C and then insert *can* countercyclically. Hence, if we allowed countercyclic merger, examples like (73) would be generated without violating the MLC.

Third, if we allowed countercyclic merger, examples like (74) could be generated without violating the Subject Condition:

(74) *who$_{\mathrm{i}}$ was [pictures of $\boldsymbol{t_i}$]$_{\mathrm{j}}$ taken t_{j} by Bill

If countercyclic merger were allowed, it would be possible to move *who* to the Spec of CP first and then move the remaining NP *pictures of t* to the Spec of TP countercyclically. In this derivation, there would be no violation of the Subject Condition. Examples like (74) would be acceptable, contrary to fact.

It should be noted, however, that these arguments only provide empirical support for cyclic merger of arguments but not cyclic merger of adjuncts. No empirical arguments have ever been presented in favor of cyclic merger of adjuncts. Recall that under our theory, arguments are required to be merged cyclically, though adjuncts, which are required to be merged postcyclically, may be merged countercyclically. The empirical arguments which have been presented in favor of strict cyclicity

therefore are compatible with our theory of phrase structure. Since our theory of phrase structure ensures cyclic merger of arguments, it can subsume the principles or conditions which are intended to derive strict cyclicity as far as the overt component is concerned.[28]

2.7 Concluding Remarks

To recapitulate this chapter, I have first introduced Chomsky's (1993) interface condition approach to the elimination of D-structure and S-structure. It was shown that although Chomsky's interface condition approach contributes to the elimination of these two theory-internal linguistic levels and thus counts as a step toward the goal of the MP, it raises a serious conceptual difficulty, i.e., it sneaks in an element of globality into the theory of language. In order to solve this conceptual problem, I have developed Chomsky's (1995) derivational interpretation of strong features, proposing the ICP on UFFs. The ICP states that UFFs must be checked immediately when they become accessible to a computation. It was shown that the ICP approach enables us to capture the D-structure and S-structure properties in the EST in a local fashion without inducing any globality. Finally, I have discussed the consequence of the ICP regarding the composition of phrase structure. It was shown that the ICP coupled with the EP on Select gives rise to the asymmetry with the composition of phrase structure. I have also shown that the ICP and the EP receive strong conceptual support. I have argued that they are "heuristic algorithms" ("computational tricks") which

---

[28]Bures (1992), Jonas and Bobalijk (1993), Branigan and Collins (1993), and Watanabe (1995) present arguments for the view that cyclicity is relevant not only in the overt component but also in the covert component. If they are correct, the ICP coupled with the EP cannot completely subsume strict cyclicity.

give us approximate solutions to the computationally intractable optimization problems induced by the interface conditions on UFFs and Ns. It was shown that our analysis supports the language design that language is fundamentally global and its corresponding optimization problem belongs to Class NP. For its usability in practice, there are language-specific computational devices like the ICP on UFFs and the EP on Select which reduce its fundamental globality to local properties.

In the rest of this thesis, I will argue that apart from this conceptual advantage, our theory of phrase structure also receives strong empirical support from a wide range of facts pertaining to movement constraints, scrambling in Japanese, the distribution of *wh*-elements in-situ, and reconstruction effects.