Processing of stacked NPs in embedded sentences in Malayalam

REVATHI SURESH
St. Joseph's University, Bangalore

ABSTRACT

The paper examines the effect of stacked NPs in centre-embedded sentences in Malayalam and uses the experimental results to compare two theories of processing difficulty: Gibson (2000)'s Dependency Locality Theory (DLT) and Hale (2001)'s Surprisal Theory. Crucially, the study also looks at the definite vs. non-definite NP distinction in Malayalam in a stacked NP context. Non-definiteness in Malayalan is indicated by having the determiner *oru* before the noun; a non-definite NP in a centre-embedded sentence disrupts NP stacking. A self-paced reading task is run using single and double-embedded sentences in Malayalam with the embedded NP alternating between definite and non-definite conditions. The test is designed to determine if (i) stacked NPs in a centre-embedded sentence result in a processing difficulty (ii) having *oru* preceding an NP affects processing. The results showed that processing difficulty increases with the addition of each NP; NPs from the embedded clauses take longest to be read. When the embedded NP is preceded by *oru*, reaction time significantly drops at *oru* and the following NP, indicating that the determiner facilitated integration of the NP, disrupting NP stacking. The results were compared against the predictions of DLT and Surprisal models. We found that the anticipation based Surprisal account best accounted for the results for Malayalam.

1 Introduction

Sentence processing studies have established that Case is crucial in driving sentence parsing in head-final languages like Malayalam. Case indicates the relationship between a noun phrase and its role in relation to verbs. Malayalam being a head-final language uses Case-marking suffixes to express this relationship. Case affixes provide information that enable the parser to build structures that incorporate the Case-carrying NPs as well as predict incoming verbs and their argument structures (Konieczny, 2000; Konieczny & Döring, 2003). However, it is still not very clear as to whether their predictive capacity holds in embedded structures, where three or more noun phrases are stacked one after the other, and where the subject of the main clause is separated by the embedded clause (1).

(1)	Radha	Raju-vinooDə	[novel	vaayikk-aan]	paranj-u
	Radha.NOM	Raju-DAT	[novel.NOM	read-NF]	tell-PAST
	'Radha told Raju to read the novel'			[Centre-embedded]	

Contexts like (1) may either (i) overload working memory by having to store too many items or handle interference due to similar items or (ii) further increase the predictive capacity of the parser by providing more information about what items are yet to come. In this paper, I will also look at (i) stacked NP contexts in embedded sentences, and (ii) what happens when a determiner (*oru*) interrupts the stacking and it is followed by a non-definite NP. Experimental data from the study will be used to examine how Case-marking functions and determine if it still retains its predictive capabilities in these contexts. The broader aim of the study is to use the experimental findings to evaluate existing sentence processing models to identify some aspects or criteria that are crucial in accounting for processing data from SOV languages like Malayalam.

2 Embedded structures, definiteness and the processing models

This section introduces the two factors examined in the study, embedded structures and definitions, and crucially how they may help in determining a best fit model for Malayalam. The two approaches in consideration here are (i) memory-based models and (ii) anticipation-based models.

2.1 Processing of Embedded Structures in Malayalam

An embedded sentence contains an independent main clause and one or more dependent or subordinate clauses. A simple example would be (2), where the complement clause within square brackets is embedded on the right of the main clause.

(2) Radha said [$_{CP}$ that she wants to buy a new book].

Embedded sentences can be more complex when there is more than one level of embedding. Compare sentences (3a) and (3b), which have one and two levels of embedding respectively.

(3) a. *Single-embedded*

Radha found the ball [$_1$ that the boy who juggles] lost

b. *Double-embedded*

Radha found the ball [$_1$ that the boy who juggles [$_2$ bottles that have yellow stripes on it]] lost

The possibility of embedding technically allows for recursion with infinite levels of embedding which are grammatical but may be difficult to comprehend after a point (4).

(4) a. The rat [that the cat [that the dog bit] chased] died.

b. The rat [that the cat [that the [the man [....] dog bit] chased] died].

(Miller & Chomsky, 1963)

Malayalam, which has a relatively free word and clause order, allows left-, centre- or right-embedded sentences where the same clause can be embedded in all three positions (5a-c).

(5)	a.	[novel	vaayikk-aan]	Radha	Raju-vinooDə	paranj-u
		[novel.NOM	read-NF]	Radha.NOM	Raju-DAT	tell-PAST
		‘Radha told Raju to read the novel’			[Left-embedded]	
	b.	Radha	Raju-vinooDə	[novel	vaayikk-aan]	paranj-u
		Radha.NOM	Raju-DAT	[novel.NOM	read-NF]	tell-PAST
		‘Radha told Raju to read the novel’			[Centre-embedded]	
	c.	Radha	Raju-vinooDə	paranj-u	[novel	vaayikk-aan]
		Radha.NOM	Raju-DAT	tell-PAST	[novel.NOM	read-NF]
		‘Radha told Raju to read the novel’			[Right-embedded]	

The centre-embedded clause in (6a) results in three NPs placed next to each other. A double-embedded sentence of the same type results in four NPs stacked next to each other, making it difficult to establish clause boundaries (6b).

(6)	a.	Radha	[amma	paalkaaran-e	viLikkunna-atə]	keeTT-u
		Radha.NOM	mother.NOM	milkman-ACC	call-NZ	hear-PAST
		“Radha heard mother calling the milkman”			[Single-embedded]	

b.	Radha	Raju-vinooDə$_i$	[PRO$_i$	[amma	paalkkaaran-e
	Radha.NOM	Raju-DAT$_i$	PRO$_i$	mother.NOM	milkman-ACC
	viLikkunna-atə]	keeLkk-aan]	paranj-u		

call-NZ	hear-NF	tell-PAST

‘Radha told Raju to listen to mother calling the milkman’

[Double-embedded]

Embedded structures of various types, especially center-embedded structures, have been extensively used in sentence processing research to study memory limitations. The idea that sentence processing is constrained by memory limitations first appeared in discussions regarding the gap in the relationship between competence and performance (Chomsky, 1957). Yngve (1960) attributes this comprehension difficulty to the language faculty’s inability to handle more than three levels of embedding. Owing to their comprehension difficulty, it is possible to expect that these types of embedded sentences would be uncommon in languages. On the contrary, embedded structures are very common, especially in head-final languages (De Roeck et al., 1982; Vasishth, 2003). The results from studies on head-final languages have shown that Case is a major source of information for processing. A number of studies have shown that embedded structures from head-final languages with rich Case systems do not display retrieval-based effects (Nakatani & Gibson, 2010; Konieczny & Döring, 2003; Vasishth, 2003; Lewis & Vasishth, 2005). In addition to this, models that have parameters that take into account Case information have better explanations for processing of embedded sentences (Nakatani & Gibson, 2010; Konieczny & Döring, 2003).

2.2 Definiteness

In Malayalam, a non-definite NP is indicated by the preceding non-definite determiner *oru* (7).

(7) a.

Radha	Raju-vinooDə	kuTTi-ye	viLikk-aan	paranj-u
Radha.NOM	Raju-DAT	child-ACC	call-NF	tell-PAST

‘Radha told Raju to call the child’ [Definite]

b.

Radha	Raju-vinooDə	oru	kuTTi-ye	viLikk-aan	paranj-u
Radha.NOM	Raju-DAT	a	child-ACC	call-NF	tell-PAST

‘Radha told Raju to call a child’ [Non-Definite]

The presence of *oru* in a DP predicts an incoming NP. The determiner is the structural manifestation of definiteness, which is also a pragmatic factor. Definiteness can be studied either as a structural factor, where continuous stacking of NPs is disrupted with an *oru*, as a pragmatic factor, where definiteness can imply a difference between old and new information.

What are the implications of definiteness on sentence or phrase processing? Is there a difference between non definite and definite NPs? What motivates this difference, a structural factor or a pragmatic factor? Vasishth (2003) looked at processing of definiteness in Hindi, where the definiteness NP has an overt objective case marking *-ko*, and the nondefinite is indicated with a null marking. Vashishth (2002) found definite NP marked with *-ko* to be more difficult to process and thereby less acceptable than a bare indefinite NP in the same position. He attributes this processing difficulty to the position in which the NP occurred, the direct object position; Vasishth suggests that a definite NP in the subject or oblique position would not have been difficult to process.

(8)	a.	Sita	kitaab	paDh	rahii	hai
		Sita	book	read	CONT	is
		'Sita is reading a book'				
	b.	Sita	kitaab-ko	paDh	rahii	hai
		Sita	book-ACC	read	CONT	is
		'Sita is reading *a/ the book'				(Vasishth, 2003)

Vasishth (2003)**,** generalising from Aissen (2003), suggests that cross-linguistically Case marking indicates markedness and that Case marking becomes obligatory as the degree of markedness increases. He evaluates corpora to show that marked objects require Case marking and that indefinites occur more frequently in the direct object position than definite NPs. Non-definites occurring more frequently in direct object position concurs with Givón's hypothesis (1978) which says that direct object positions usually introduce new discourse referents[1]. A definite NP, which is 'old information' in this position is thus marked, and markedness has to be expressed using overt Case marking. Since a definite NP is not expected in the direct object position, their occurrence is non-canonical and unexpected, and thus increases the processing load.

Malayalam does not implement DOM to indicate definiteness as in (7); does it imply that there will be no processing cost associated with definite NPs in Malayalam? It is also possible that a non-definite NP introduces a new discourse referent forcing the parser to allocate memory resources to identify and encode this new referent. In contrast, a definite NP is presupposed and requires less memory load for processing. This hypothesis is independent of the position in which the non-definite NP occurs and predicts that a definite NP will be easier to comprehend. The possible outcomes regarding the effect of having non-definite NPs in an embedded sentence provide an environment where a determiner provides cues for an incoming item in the absence of Case-like features.

[1] A discourse referent is an entity that exists in a spatio-temporal location, which can be referred to in speech using a pronoun or NP if it is an object, or using tense on a verb if it is an event (Gibson, 2000). A new discourse referent is an entity that has not been presupposed or previously mentioned in a context; it is usually introduced using non-definite NPs.

2.3 Memory and anticipation-based processing models

Computational theories of language processing difficulty argue for either a memory or expectation-based approach. For the present study, we will examine one representative theory from each group: Dependency Locality Theory, or DLT (Gibson, 2000) and Surprisal theory (Hale, 2001; Levy 2008). This section provides a brief introduction to these approaches. A detailed account regarding their architecture and metrics is presented in Section 6.

DLT posits two separate components of a sentence's processing cost: storage and integration costs. Storage or memory cost depends on the number of syntactic heads required to complete the grammatical structure and seems to be independent of the amount of time that an incomplete dependency is held in memory (Gibson, 2000). Integration cost, on the other hand, is locality-based, i.e. the cost is based on the distance between the dependent and its head; this distance is based on the number of new intervening discourse referents (Gibson, 2000). DLT accounts for a wide range of sentence processing data from English (Gibson, 1998), Japanese (Babyonyshev & Gibson, 1999; Nakatani & Gibson, 2010) and Hindi (Vasishth, 2003; Agrawal, Agarwal, and Husain, 2017; Husain, Vasishth, & Srinivasan, 2015), to name a few.

The Surprisal framework attempts to provide a unified account for ambiguity resolution as well as predictive parsing in the case of other syntactic complexities (Hale, 2001; Levy, 2008). When there is an unexpected input, there is a forced (and costly) discarding of a highly probable structure, indicating a higher Surprisal. The processing cost at a particular word is estimated as the effort involved in discarding the current analysis or triggering a reanalysis at that word (Hale, 2001). The Surprisal model has found cross-linguistic support from German (Levy & Keller, 2013), Hindi (Husain, Vasishth, and Srinivasan, 2014; Agrawal et al., 2017) and English (Staub, 2007).

3 Malayalam Data

The Malayalam sentence processing study was a self-paced reading task which had single and double centre-embedded Malayalam sentences with definite and non-definite NPs. The study was non-cumulative and had 32 target sentences and 32 non-target sentences (fillers). Tables 1 and 2 show the region-wise division in the self-paced reading task, for single and double embedded sentences with a definite NP.

Region	1	2	3	4		5	
Single embedded	Radha	amma	paalkaaran-e	viLikkunna	-atə	keeTT	-u
	NP1.NOM	NP2.NOM	NP3-ACC	VP2	-NZ	VP1	-PAST
‘Radha heard mother calling the milkman							

Table 1

Region-wise representation of a single-embedded sentence with a definite NP

Region	1	2	3	4	5		4		5	
Double	Radha	Raju-vinoodə	amma	paalkaaran-e	viLikkunna	-atə	keLkk	-aan	paranj	-u
embedded	NP1.NOM	NP2-DAT	NP3.NOM	NP4-ACC	VP3	-NZ	VP2	-NF	VP1	-PAST
	‘Radha told Raju to listen to mother calling the milkman’									

Table 2

Region-wise representation of a double centre-embedded sentence

Tables 3 and 4 show the region-wise division in the self-paced reading task, for single and double embedded sentences with a non-definite NP, preceded by a determiner *oru*.

Region	1	2	3	3	4		5	
Single	Radha	amma	oru	paalkaaran-e	viLikkunna	-atə	keeTT	-u
embedded	NP1.NOM	NP2.NOM	a.DET	NP3-ACC	VP2	-NZ	VP1	-PAST
	‘Radha heard mother calling a milkman’							

Table 3

Region-wise representation of a single-embedded sentence with a non-definite NP

Region	**1**	**2**	**3**	**4**	**5**	**6**		**7**		**8**	
Double	Radha	Raju-vinooDə	amma	oru	paalkaaran-e	viLikkunna	-atə	keLkk	-aan	paranj	-u
embedded	NP1.NOM	NP2-DAT	NP3.NOM	DET	NP4-ACC	V3	-NZ	V2	-NF	V1	-PAST
	“Radha told Raju to listen to mother calling a milkman”										

Table 4

Region-wise representation of a double-embedded sentence with a non-definite NP

The region-wise layout of a single-embedded sentence is ***NP1 [NP2 NP3*** *V2] V1* and that of a double embedded sentence is ***NP1 NP2 [NP3 NP4*** *V3 V2] V1* (Table 5). Note that in these embedded structures, the NPs are stacked at the beginning and verbs at the end. The region manipulated to examine the definiteness factor is NP3 for single and NP4 for double embedded sentences. These NPs are preceded by the determiner *oru*. The presence of the determiner disrupts NP stacking.

	Region 1	*Region 2*	*Region 3*	*Region 4*	*Region 5*	*Region 6*	*Region 7*
SE	NP1	NP2	NP3	V2	V1	---	---
DE	NP1	NP2	NP3	NP4	V3	V2	V1

Table 5

Layout and regions of analysis for single and double-embedded sentences

4 The experiment

The test sentences were presented using version 3.2.5 Kafkaesque Koffka of OpenSesame (Mathôt, Schreij, & Theeuwes, 2012) in a controlled environment. There were 64 experimental sentences: 32 target sentences and 32 non-target (fillers) sentences. The sentences were randomized on the OpenSesame software[2]. The self-paced sentences were presented one word at a time. Participants had to press the spacebar key to go to the successive word. The software recorded every key press response and the reaction time (time duration between the appearance of stimuli on the screen and the time of response). Examples (5a-b) illustrate the structure of test sentences used in the study. See (9) for example of a filler sentence.

(9) Cleft

[Ummar	aaN-ə]	Sheela-yooDə	kutti-ye	viLikk-aan	paranj-atə
Ummar.NOM	be-PRES	Sheela-DAT	child-ACC	call-NF	tell-NZ

‘It is Ummar who told Sheela to call the child’

The main experiment was preceded by a practice session consisting of ten trials. Each experimental trial consisted of a self-paced sentence which had 5-6 parts, and at the end of the sentence, a key press would lead to a comprehension question, which served as a distractor task (see example 10).

(10) a. Sentence

Radha	[amma	paalkaaran-e
Radha.NOM	mother.NOM	milkman-ACC

‘Radha heard mother calling the milkman’

b. Comprehension question

[2] The study reported here is one section of a larger study that looked at levels of embedding, position of embedding, effect of same Case on adjacent NPs and definiteness. The data and results reported here are associated with two factors: levels of embedding and definiteness.

Ar-aaNə	paalkaaran-e	viLich-a-tə
Who-is	milkman-ACC	call-PAST-NZ

‘Who called the milkman?’

c. Options

(a) Radha (b) amma

For the analysis, data from the self-paced task were used for region-wise analysis for the experimental conditions and results from the comprehension question were used for examining the accuracy of interpretation.

The reaction time data was trimmed to minimize the effect of outliers using the interquartile method (Jones, 2019). Statistical analysis of data was run on R and used log transformed values of reaction times (ms). The analysis involved determining areas of interests for each condition and doing contrast analysis using ANOVAs.

Participants: 26 native speakers of Malayalam within the age range of 30 - 85 years answered the questionnaire. The subjects could speak and read Malayalam without any difficulty. They used Malayalam in informal contexts, at home and with friends, and therefore had a sound knowledge of the language.

5 Results

The results sections contains results for (i) stacked NPs in centre-embedded sentences and (ii) non-definite vs. definite NP comparison in a stacked NP context.

5.1 NP Stacking

Table 6 given below, presents the mean latencies for each region for single and double embedded sentence types. Figures 1a-b graphically illustrated the data from Table 6.

	Region 1	*Region 2*	*Region 3*	*Region 4*	*Region 5*	*Region 6*	*Region 7*
SE	817.88	943	1156	---	---	845.4	864
DE	888.85	1026.8	1211.98	1133.99	875	815.5	890

Table 6

Mean RTs for each region of single and double-embedded sentences

Note: SE stands for single-embedded and DE for double-embedded. Mean RTs (ms) are given for each region in single and double-embedded sentences; the shaded regions form the embedded part. Regions 4 and 5 for single-embedded sentences are left empty to visually align its embedded verbs with that of double-embedded sentences; regions 5 and 6 are regions 4 and 5 respectively in single-embedded sentences.

Values from Table 4.4 and Figure 4.1 clearly show that there was an increase in processing cost starting from the first NP till the final NP followed by a decrease towards the end at the verbs. For single embedded sentences (see Figure 4.1 (a)), the processing time was shortest at NP1 and longest at NP3, then shorter at V2 and V1. For the double-embedded sentence (Figure 4.1 (b)), the processing time increased starting from NP1 and is the longest at NP3, and fell slightly at NP4. There was also no remarkable increase in reaction time between the embedded verbs V3 and V2, and the main verb V1.

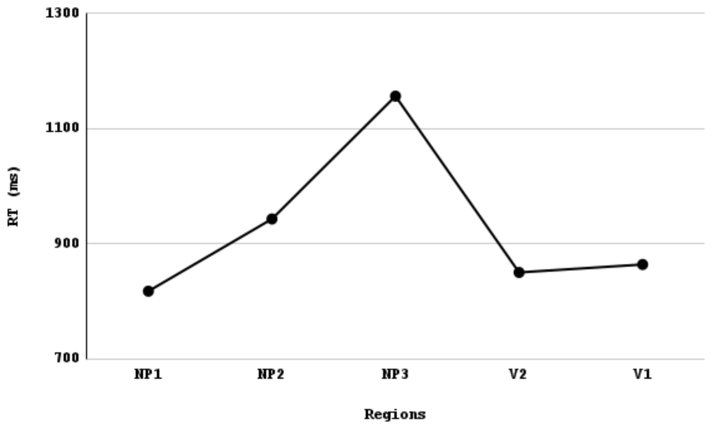

Figure 1 (a) Mean RTs (ms) for single-embedded sentences

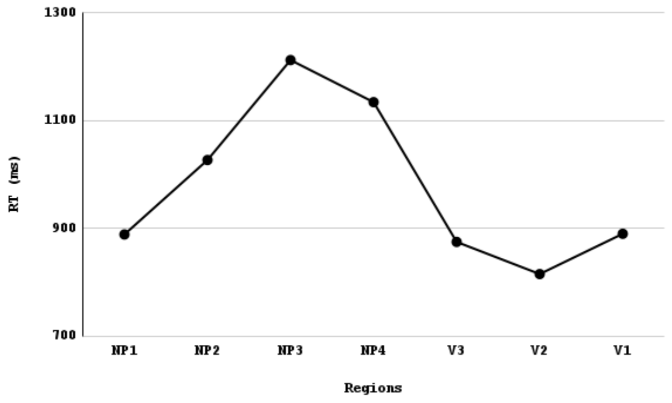

Figure 1 (b) Mean RTs (ms) for double-embedded sentences

The above-given data shows that the observed trend for embedded sentences was that the RT increased as more NPs were added and decreased as the corresponding verbs were encountered. As the data shows, the contributing factor towards the increase of processing time was the number of NPs at the beginning of the embedded clause.

A one-way ANOVA was run to check whether the number of NP in the stacking affected the reading time for each NP. The analysis compared the RTs for NP1, NP2 and NP3 for single embedded and NP1, NP2, NP3 and NP4 for double embedded sentences. Table 7 presents the mean reading times (in milliseconds) for the NPs along with the results of the analysis for comparison of the reading time for NPs in a sentence.

	NP1	*NP2*	*NP3*	*NP4*	*F-value*	*p-value*
SE	817.46[c]	943.17[c]	1156.33[a,b]	---	8.6	P< 0.0001
DE	888.84[c,d]	1027.59	1211.13[a]	1132.88[a]	6.78	P< 0.0001

Table 7

Comparison of RTs (ms) across stacked NPs for single and double-embedded sentences

The results showed a significant difference between the reaction times of the stacked NPs for both single- (F (2, 44) = 8.6, p<0.0001) and double-embedded sentences (F(3, 43) = 6.78, p < 0.0001). The reading time increased with each additional NP in both types of sentences. The first NPs had the shortest reading times (single: 817.46 ms; double: 888.84 ms), and the innermost NPs had longer RTs, at least by 250-300 ms. Post hoc Tukey tests showed that in single-embedded sentences, RTs were significantly longer for NP3 when compared to NP1 (t = 4.5, p < 0.0001) and NP2 (t= 2.6, p < 0.0001). However, there was no difference between the time taken to read NP1 and NP2. Significant differences were found between NP1 and NP3 (t = 4.6, p < 0.0001) and NP4 (t = 3.75, p < 0.0005) in double-embedded structure. There was no significant difference between NP3 and NP4.

The results show that for single-embedded sentences, NPs arranged according to RT values give the order: *NP1 < NP2 < NP3*. And, for double embedded sentences, the arrangement gives the order: *NP1 < NP2 < NP3, NP4*.

5.2 Definite and Non-definite NP

We compared (a) whether non-definite (and definite) NPs were read in the same time in single and double-embedded sentences (comparison across a number of embeddings), and (b) whether the non-definite NPs were read in the same time as the definite NPs in embedded sentences (comparison of definite and non-definite status). The region-wise layout for non-definite NPs and definite NP in the embedded sentences are represented in

Table 8[3]. The shaded regions were the points of interest. We also looked at the spillover regions (V2 in single embedded sentences and V3 in double embedded sentences). Regions other than the target regions were not analysed.

	Region 1	*Region 2*	*Region 3*	*Region 4*	*Region 5*	*Region 6*	*Region 7*	*Region 8*
NON-DEFINITE								
SE	NP1	NP2	oru	NP3	V2	V1	__	__
DE	NP1	NP2	NP3	oru	NP4	V3	V2	V1
DEFINITE								
SE	NP1	NP2	NP3	V2	V1	__	__	__
DE	NP1	NP2	NP3	NP4	V3	V2	V1	__

Table 8

Layout and regions of analysis for single and double-embedded sentences with definite and non-definite NPs

	Region 1	*Region 2*	*Region 3*	*Region 4*	*Region 5*	*Region 6*	*Region 7*	*Region 8*
SE	883.6	980	913.5	816.4	812.6	873		
DE	793	1093	1288.7	911.05	806.9	803.9	776	806

Table 9

Region-wise mean RTs of single and double-embedded sentences with a non-definite NP

Table 9 presents the comparisons of RT at the three regions: determiner, NP and the following verb across single- and double-embedded sentences; the values showed that '*oru*' which took (approximately 900 ms) helped in anticipating an NP and therefore brought down the reading time of the consequent NP by 100-125 ms. However, the number of embedded clauses made little difference to the reading time of '*oru*' or the NP. The differences were not statistically significant, which is evident from Figures 2a-b given below. The V2 and V3 comparison (spillover regions) did not show a dramatic

[3] Initially, the study intended to study definiteness as a semantic factor, for which the determiner and following the NP would form a single segment for analysis. However, in the region-wise segmentation, the sentences were segmented at the level of word, and therefore the non-definite NP *oru paalkkaran* were presented as two segments rather than at a phrasal level. The intention was also to check whether *oru* allows the parser to anticipate an NP. However, at the time of analysis, a direct comparison of DP was not possible, since the determiner and the NP formed separate regions. Thus a comparison was made between the two NPs (one which had a preceding *oru* and the other which had a null determiner).

change in reading time in either of the embedded contexts. Therefore, the number of embeddings did not seem to affect the processing of the embedded object with respect to definiteness.

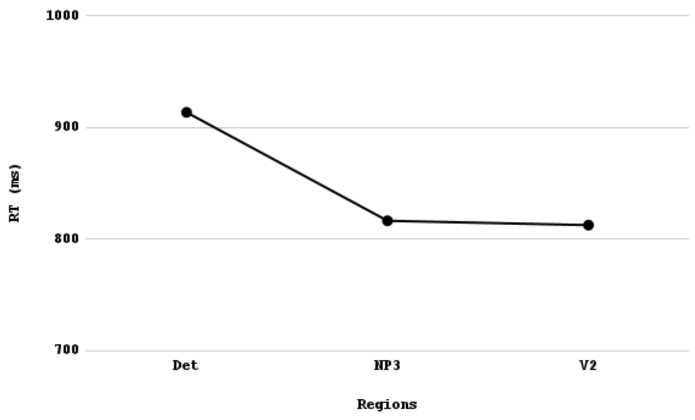

Figure 2 (a) Mean RTs (ms) for *oru*-NP3-V2 sequence from single-embedded sentences

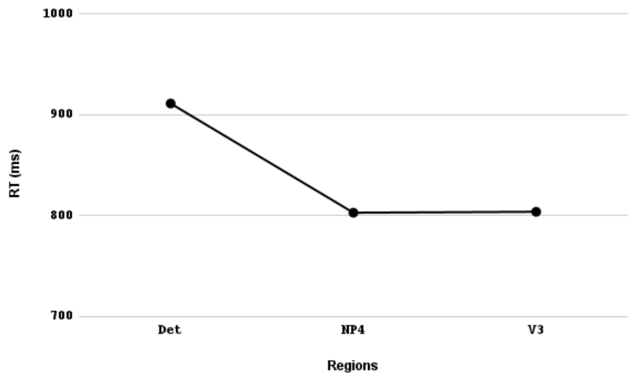

Figure 2 (b) Mean RTs (ms) for *oru*-NP4-V3 sequence from double-embedded sentences

To check whether the definiteness status affected the processing of the NPs, a comparison was made between the RT of NPs (when they were (not)/preceded by *oru*). For the comparison, we looked at single and double-embedded sentences individually.

Figures 3 (a) and (b) show that determiner *oru* lowered the processing cost for the non-definite NP when compared to the definite sentences. The analysis attempts to determine if the determiner *oru* significantly facilitated the addition of NP by predicting it; if it allowed the parser to anticipate and allot memory resources for storage of the incoming NPs.

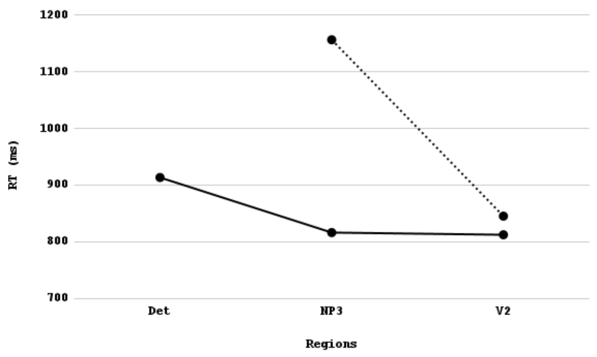

Figure 3 (a): Comparison of RTs for non-definite and definite NPs: ***oru*-NP3-V2** (non-definite; solid) and **NP3-V2** (definite; dotted) sequence from SE sentences

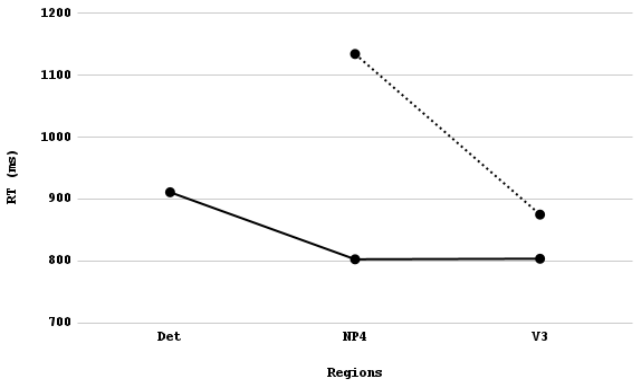

Figure 3 (b): Comparison of RTs for non-definite and definite NPs: ***oru*-NP4-V3** (non-definite; solid) and **NP4-V3** (definite; dotted) sequence from DE sentences

	Non-definite (ms)	***Definite (ms)***	***F-value***	***p-value***
Single-embedded	816.4	1156.05	13.8	p<0.001
Double-embedded	806.9	1133.9	31.23	p<0.001

Table 10

Effect of definiteness in sentences with definite and non-definite NPs

Table 10 shows that there was a significant difference between the reading times at definite and non-definite NPs for single (F (1,45) = 13.8, $p < 0.001$) and double-embedded (F (1,45) = 31.23, $p < 0.001$) sentences. These results indicate the determiner *oru* facilitates the processing of the following (non-finite) NP. The final NPs (NP3/NP4) were easier to process because the determiner *oru* helps the parser to anticipate the incoming verb. There was also no significant difference between non-definite and definite conditions at the innermost verb; the definiteness of the NP did not affect the processing of the following verb (V3/V2). The results showed that the determiner *oru* that precedes a definite NP helps the parser in predicting the incoming NP, making the storage easier.

6 Discussion

To summarise the results from the previous section, processing difficulty increases with the addition of each NP in a centre-embedded sentence. If arranged according to RT measurement, we can come up with the following NP sequences: *NP1<NP2<NP3* for single-embedded sentences and *NP1<NP2<NP3,NP4* for double embedded sentences. In this case, all NPs are definite. When the embedded clause has a non-definite NP, (*oru palkkaaran*), the reading times significantly decrease at the determiner and the following NP. The determiner thus facilitates the faster integration of the NP by creating an anticipation for it. Following sections will look at how the two introduced earlier: Dependency Locality Theory, or DLT (Gibson, 2000) and Surprisal theory (Hale, 2001; Levy 2008), will account for the results.

6.1 Dependency Locality Theory (Gibson 2000)

The locality metric of the Dependency Locality Theory (Gibson, 2000) predicts that the processing of a verb should be easier if its arguments are closer. DLT's version of locality metric counts new discourse referents to measure distance. The integration component of DLT counts the number of intervening discourse referents between the item being integrated and the structure into which it is being integrated. Dependency Locality Theory, also assigns a separate storage cost for the memory load incurred by keeping unresolved dependencies or predictions active. Storage cost is a measure of the number of heads required to complete a structure that is currently being constructed. We will first look at how DLT's metrics can be used to estimate the processing cost for nouns and verbs in embedded sentences in Malayalam.

Integration cost: In DLT, nouns and verbs are discourse referents, pronouns and adjectives are not. Each new discourse referent gets 1 unit of integration cost. The Dependency Locality measures the distance from the item being integrated to its dependent, which has already been stored in the working memory. Table 11 demonstrates how integration cost is calculated for a single-embedded sentence in Malayalam[4].

(11) Radha [amma paalkaaran-e viLikkunna-atə] keeTT-u

[4] The calculation of Integration Cost here is very basic and not explored further here. DLT model is evaluated on the basis of the experimental data regarding the processing of stacked NPs. This requires a detailed look at the Storage component of DLT, which is provided in the following segments.

Radha.NOM	amma.NOM	milkman-ACC	call-NZ	hear-PAST

"Radha heard mother calling the milkman"

Regions	1	2	3	4		5	
Category	NP1.NOM	NP2.NOM	NP3.ACC	V2	-NZ	V1	-PAST
New Discourse Referents	1	1	1	1	0	1	0
Distance of Integration	0	0	0	2	0	4	0
Integration cost	1	1	1	1.5		2.5	

Table 11

Integration cost for a Malayalam single-embedded sentences

Note: For estimating the integration cost (ans storage cost in subsequent tables), I have split the verb into stem and tense inflection (V2, -NZ) to capture the parsing capabilities based on inflectional morphology of the verb following the method adopted in (Gibson & Nakatani, 2010). The total integration cost (or storage cost) at a word is then the average of the storage cost of the stem (V2) and the inflection (-NF).

Storage Cost: Storage cost is the count of the number of heads (nouns or verbs) minimally required to complete the current structure (Gibson, 2000). This predictive feature of the storage component can be evaluated on the basis of whether it can account for (i) the processing difficulty at nouns (NP stacking and Similarity-based interference), (ii) lower processing cost at non-definite NPs and (iii) increased processing difficulty at the final verb when it follows a nominalised verb.

Tables 12 shows the storage cost for single-embedded sentences given in (12). The values show that the storage component of the models predicts that the memory load increases as the NPs are added to the sentence and decreases as verbs are encountered. As NPs are added, they have to stay in the memory till their head (verb) is encountered and integrated with the subject. As they are being integrated, memory load is taken off. This is not explicitly treated as a distance problem but having a separate storage cost component is the consequence of defining locality as a linear-order metric where processing requires that these predictions stay active over a particular distance, until their retrieval is triggered.

(12)	a.	Radha	[amma	paalkaaran-e	viLikkunna-atə]	keeTT-u
		NP1.NOM	NP2.NOM	NP3-ACC	VP2-NZ	VP1-PAST

"Radha heard mother calling the milkman"

Regions	1	2	3	4		5	
Categories	NP1.NOM	NP2.NOM	NP3.ACC	V2	-NZ	V1	-PAST
Predicted heads	V1	V1, V2	V1, V2	V1	V1	0	0
Memory cost	1	2	2	1	1	0	0
Total cost	1	2	2	1		0	

Table 12

Storage cost for Malayalam single-embedded sentences

For the single-embedded sentence, the storage cost is 1 MU at region 1 because it needs only 1 finite verb (V3) to complete the sentence. As another nominated NP is encountered, the parser predicts an embedded verb V2 and a finite final verb V1; the storage cost is revised to be 2 MUs. Similarly, as more NPs are encountered, the parser predicts more verbs depending on the Case information it receives from the NPs. As verbs are encountered the storage cost reduces because memory is freed up as dependencies are resolved; in addition, no new items are predicted at these verbs.

The storage component of DLT rightly predicts the increase in RT due to repeated encounter and storage of nouns. However, it does not account for how the parser is able to anticipate a non-definite NP as there is no mechanism for accounting for how predictions reduce processing cost. Table 13 presents the results against storage cost predictions.

Storage results	***Level of embedding***	***DLT's prediction***	***Data results***	***Remarks***
NP stacking	Single	NP1 < NP2, NP3	NP1 < NP2 < NP3	Confirmed
	Double	NP1<NP2,NP3,NP4	NP1< NP2< NP3,NP4	Confirmed
Definiteness	Single & double	$NP_{ND} = NP_{D}$	$NP_{ND} < NP_{D}$	Not confirmed

Table 13

Evaluation of the storage component of DLT

6.2 Surprisal Model

The Surprisal model (Hale, 2001; Levy, 2008) is an expectation-based probabilistic model which claims that reading times are lower at the head when there is more preverbal (intervening) material in the clause. In addition to this, the Surprisal model also claims that the intervening items help accurately predict the final verb's identity and location, and that expectation for a verb becomes stronger as the length of the sentence increases.

The models measure processing difficulty with a Surprisal cost. Surprisal cost is the processing difficulty at any point, taken as the effort or cognitive load involved in disconfirming a highly probable structure at that point. When the parser encounters a nominative NP, a verb that governs it is predicted; this would be the simplest and possibly the most predicted structure. If the next input is another NP, the parser needs to discard the prediction and posit either (i) a transitive verb if the new NP is dative or accusative marked or (ii) an embedded clause with a non-finite verb if the NP is nominative.

The three steps in (13) illustrate how expectation based parsing works for a single embedded sentence (12). At each step, the parser encounters a new NP: in (12a) with NP1 (Radha), a VP (*urangi*) is predicted; in (13b) with NP1 (Radha) and NP2 (*amma*), a nominalized VP2 and the main verb (VP2) is predicted; in (13c) with NP1, NP2 and NP3 (*paalkkaaran-e*), a non-finite/nominalized VP2 and the main verb (VP2) is predicted. Therefore, in (13a) the predictions are not met, leading to a Surprisal cost; in (13b) too, there is a Surprisal cost. However, in (13c), there is no or marginal Surprisal cost expected since the prediction of the embedded clause given in (12) is successfully met.

(13) a. **Radha**

Radha	urang-i
Radha.NOM	sleep-PAST

'Radha slept'

b. **Radha amma**

Radha	[amma	urangunna-atə]	kaND-u
Radha.NOM	mother.NOM	sleep-NZ	see-PAST

'Radha saw mother sleeping'

c. **Radha amma paalkaaran-e**

Radha	[amma	paalkaaran-e	viLikkunna-atə]	keeTT-u
Radha.NOM	mother.NOM	milkman-ACC	call-NZ	hear-PAST

"Radha heard mother calling the milkman"

Example (12) shows how NP stacking, despite temporarily resulting in processing load, ultimately provides information that allows easier integration of the verbs. In case of definite NPs, the Surprisal model predicts that when the parser encounters the determiner *oru*, it anticipates a noun, which was seen in the Malayalam results, where the NP showed a faster reading time when compared to a definite NP (where the definiteness is null). The determiner essentially functions as a highly reliable expectation for an incoming NP. Table 5.10 presents the results against predictions of the Surprisal model.

Position of Embedding	***Level of embedding***	***Expectation-based prediction***	***Data results***	***Remarks***
Storage				
NP stacking:	Single	NP1 < NP2 <NP3	NP1 < NP2 < NP3	Confirmed
	Double	NP1 < NP2 < NP3 < NP4	NP1< NP2 < NP3,NP4	Confirmed
Definiteness	Single & double	$NP_{ND} < NP_D$	$NP_{ND} < NP_D$	Confirmed

Table 14

Evaluation of the expectation-based Surprisal model

7 Conclusion

In the present study, we focused on the processing of nouns, stacked NPs specifically, in centre-embedded sentences. The intention was to test if stacked NPs incur a processing load. This prediction was proven, as reaction times increased with each NP. The result matched predictions of DLT, according to which NPs are items in an unresolved dependency. It predicts the incoming head which would resolve the expectation, but at a cost; the cost increases with the distance (and time) between the NP and its verb. Surprisal theory accounts for the difficulty associated with NP stacking as discarding or inhibiting inaccurate predictions regarding what is to come. The predictions made by the two models do not seem to be very different. It is also important to note that Case information does not facilitate the storage of NPs; or rather, Case information is not fully accessible to the parser. When it comes to definiteness, DLT and Surprisal make different predictions. While Surprisal theory can correctly predict how the determiner *oru* creates an anticipation for an NP. DLT does have a prediction-based Storage component, but it does not explain how that reduces processing load.

At one level, the processing difficulty at NPs seems like it is arising out of delayed resolution or inaccurate predictions. Another way to think of it would be in terms of similarity-based storage interferences. It is difficult to efficiently differentiate between and allocate memory resources to items of a similar type, i.e., NPs; this results in a storage-related cost. However, when a different kind of item is encountered, *oru*, the storage is easier, so is processing of the item predicted by *oru*. This analysis also directs one to look at more activation-based accounts (Lewis and Vasishth, 2005) for further research.

References

Agrawal, Arpit, Agarwal, Sumeet and Husain, Samar. (2017). Role of expectation and working memory constraints in Hindi comprehension: An eyetracking corpus analysis. Journal of Eye Movement Research, 10(2), 1-15. https://doi.org/10.16910/jemr.10.2.4

Aissen, Judith. (2003). Differential object marking: Iconicity vs. economy. Natural Language & Linguistic Theory, 21(3), 435–483. https://doi.org/10.1023/A:1024109008573

Chomsky, Noam. (1957). Logical structures in language. Journal of the American Society for Information Science, 8(4), 284.

Gibson, Edward. (2000). The dependency locality theory: A distance-based theory of linguistic complexity. In A. Marantz, Y. Miyashita, & W. O'Neil (Eds.), *Image, language, brain: Papers from the First Mind Articulation Project Symposium* (pp. 95-126). Cambridge, MA: MIT Press. Retrieved July 12, 2021, from https://www2.bcs.rochester.edu/sites/fjaeger/teaching/LabSyntax2006/readings/Gibson_2000.pdf

Givón, Talmy. (1978). Definiteness and referentiality. In J.H. Greenberg, C.A. Ferguson, & E.A. Moravcsik (Eds.), Universals of human language, Volume 4 (Syntax) (pp. 291–330). Cambridge, MA: Stanford University Press, Stanford, California.

Hale, John. (2001). A probabilistic Earley parser as a psycholinguistic model. In Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics. https://doi.org/10.3115/1073336.1073357

Husain, Samar, Vasishth, Shravan and Srinivasan, Narayanan. (2014). Strong expectations cancel locality effects: Evidence from Hindi. PLoS ONE, 9(7), e100986. https://doi.org/10.1371/journal.pone.0100986

Konieczny, Lars. (2000). Locality and parsing complexity. *Journal of Psycholinguistic Research*, *29*(6), 627–645. https://doi.org/10.1023/A:1026528912821

Konieczny, Lars and Döring, Philipp. (2003). Anticipation of clause-final heads: Evidence from eye-tracking and SRNs. In *Proceedings of iccs/ascs* (pp. 13-17).

Levy, Roger P. (2008). Expectation-based syntactic comprehension. Cognition, 106(3), 1126–1177. https://doi.org/10.1016/j.cognition.2007.05.006

Levy, Roger P. and Keller, Frank. (2013). Expectation and locality effects in German verb-final structures. Journal of Memory and Language, 68(2), 199–222. https://doi.org/10.1016/j.jml.2012.02.005

Lewis, Roger L. and Vasishth, Shravan. (2005). An activation-based model of sentence processing as skilled memory retrieval. *Cognitive Science*, *29*(3), 375–419. https://doi.org/10.1207/s15516709cog0000_25

Nakatani, Kentaro and Gibson, Edward. (2010). An on-line study of Japanese nesting complexity. Cognitive Science, 34(1), 94–112. https://doi.org/10.1111/j.1551-6709.2009.01067.x

Staub, Adrian. (2007). The parser doesn't ignore intransitivity, after all. Journal of Experimental Psychology: Learning, Memory, and Cognition, 33(3), 550–569. https://doi.org/10.1037/0278-7393.33.3.550

Vasishth, Shravan. (2003). Working memory in sentence comprehension: Processing Hindi center embeddings. (Doctoral dissertation), The Ohio State University.

Yngve, Victor H. (1960). A model and an hypothesis for language structure. *Proceedings of the American philosophical society*, *104*(5), 444-466. https://dspace.mit.edu/handle/1721.1/4453