



# **ESCUELA NACIONAL DE ANTROPOLOGÍA E HISTORIA**

## **PROCESADOR DE LENGUAS NATURALES**

TESIS QUE PARA OPTAR POR EL TÍTULO DE LICENCIADO EN LINGÜÍSTICA

PRESENTA:

**EDGAR URIEL DOMÍNGUEZ ESPINOZA**

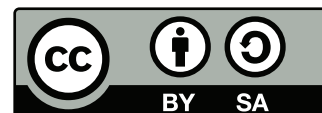
DIRECTOR DE TESIS:

**DR. CARLOS FRANCISCO MÉNDEZ CRUZ**

CIUDAD DE MÉXICO

2017

Este documento esta realizado bajo licencia Creative Commons “Reconocimiento-CompartirIgual 4.0 Internacional” .



Puede leer la licencia en <https://creativecommons.org/licenses/by-sa/4.0/deed.es>

## Resumen

Este trabajo de investigación analiza las representaciones formales de los fenómenos lingüísticos con el fin de encontrar una posible representación común a todos ellos.

Luego de una breve introducción, en el capítulo 2 se presentan los principios básicos necesarios para entender los capítulos posteriores. En el capítulo 3 se analizan los elementos formales más comunes en lingüística, también se hace un acercamiento entre las matemáticas y la lingüística.

El capítulo principal es el número 4, en él se detalla una propuesta de un lenguaje que pretende representar los métodos existentes de una forma única. Se detallan los elementos que conforman el lenguaje, las operaciones y estructuras fundamentales para su existencia y paso a paso se construyen ejemplos más complejos que permiten ver los alcances de dicho lenguaje.

El apéndice A contiene información sobre la utilidad de la lingüística fuera del campo estrictamente científico. Esta visión puede ser importante si se considera una de las premisas de este trabajo: «el intercambio de conocimiento».

El apéndice B contiene información sobre el primer borrador del lenguaje aquí presentado, útil para establecer evolución que hasta el momento ha tenido el lenguaje.

Finalmente, el trabajo impreso contiene la bibliografía completa de las obras citadas en esta tesis, una bibliografía más extensa (pues contiene todas las obras leídas) se encuentra junto a los archivos fuente de la obra, que pueden consultarse en <https://www.overleaf.com/read/mvybkmghrrbb>



# Índice general

<b>Lista de Siglas</b>	<b>IX</b>
<b>1 Introducción</b>	<b>1</b>
<b>2 Marco Teórico</b>	<b>5</b>
<b>3 Análisis de lenguajes formales</b>	<b>25</b>
<b>4 Propuesta</b>	<b>57</b>
<b>5 Conclusión</b>	<b>113</b>
<b>A La lingüística en la SIC</b>	<b>119</b>
<b>B Algunos de los ejemplos originales del desarrollo del lenguaje</b>	<b>129</b>
<b>Bibliografía</b>	<b>141</b>
<b>Índice alfabético</b>	<b>151</b>



# Índice de tablas

2.1. Definición de conectivos lógicos . . . . .	12
3.1. Datos del Kongo del sur . . . . .	40
3.2. Asociación de palabras con valor positivo y negativo . . . . .	47
3.3. Tabla de verdad $NS(O(x))$ . . . . .	54
4.1. Tabla de datos POS . . . . .	64
4.2. Tabla de datos EAGLES . . . . .	65
4.3. Ejemplos de listas . . . . .	66
4.4. Tabla notaciones de par de celda y lista . . . . .	67
4.5. Sucesión de Fibonacci . . . . .	86
4.6. Tabla de rasgos de la palabra <i>cabo</i> . . . . .	90





# Índice de figuras

2.1. Ejemplo de estructura arbórea . . . . .	14
2.2. Árbol de una estructura sintáctica . . . . .	16
3.1. Regla gramatical en diagrama . . . . .	32
3.2. Regla gramatical en diagrama 2 . . . . .	32
3.3. Estructura arbórea de una oración completa . . . . .	33
4.1. Diagrama de la arquitectura . . . . .	62
4.2. Diccionario . . . . .	62
4.3. Ejemplo de un operador . . . . .	67
4.4. Estructura arbórea de un operador . . . . .	68
4.5. Diagrama de la multiplicación . . . . .	68
4.6. Diagrama de la multiplicación y su argumento . . . . .	69
4.7. Modelo autosegmental-jerárquico . . . . .	92
4.8. Modelo autosegmental-jerárquico 2 . . . . .	92
4.9. Estructura sustitución wh . . . . .	107
4.10. Estructura movimiento-wh y movimiento-I . . . . .	109
B.1. Estructura arbórea de un sintagma, teoría X-barra . . . . .	130
B.2. Estructura arbórea de un sintagma, LanPro . . . . .	130
B.3. Primera parte de la figura B.2 . . . . .	131
B.4. Segunda parte de la figura B.2 . . . . .	131
B.5. Árbol de rasgos según el modelo autosegmental-jerárquico . . . . .	132



# Lista de Siglas

**AFI** Alfabeto Fonético Internacional

**EAGLES** Expert Advisory Group on Language Engineering Standards

**ENAH** Escuela Nacional de Antropología e Historia

**IN** Inductor Negativo

**LISP** LISt Processing

**PN** Palabra Negativa

**POS** Part of Speech

**SPE** The Sound Pattern of English

**TPN** Término de Polaridad Negativa

**G11N** Globalización

**I18N** Internacionalización

**L10N** Localización

**PLC** Controlador lógico programable

**R25A** Regionalización Lingüística

**SIC** Sociedad de la Información y el Conocimiento

**TIC** Tecnología de la Información y la Comunicación



---

Actualmente las ciencias interactúan unas con otras. Para algunas personas ya no es posible distinguir con toda claridad el límite entre el estudio de la física y la química. Si observamos más detalladamente, la astronomía, la física, la química, la biología, la geología y otras ciencias naturales tienen algo en común con diversas ciencias sociales como la administración, la demografía, la economía e incluso la antropología; todas usan comúnmente el lenguaje matemático.

Es gracias al lenguaje matemático, entre otras cosas, que la interdisciplinariedad se ha acelerado en los últimos años, haciendo posible que los científicos de todos los países, por medio de las matemáticas, sean capaces de tomar modelos de otras ciencias sin tener que ser expertos en todas las teorías de una determinada área del conocimiento.

La lingüística, por su parte, está dentro de un grupo de ciencias cuyo objeto de estudio no puede describirse enteramente por los medios matemáticos actuales. Por ello, ha tenido que buscar otros medios para explicar sus principios científicos, generalmente usa descripciones específicas para cada problema acompañadas de ejemplos pertinentes; sin embargo, para entender estas explica-

Diversidad de representaciones.

## 1. INTRODUCCIÓN

---

ciones es necesario tener un amplio conocimiento previo sobre esta ciencia. Para dar un ejemplo de lo anterior, en el terreno de la fonología, las primeras teorías basadas en rasgos distintivos<sup>1</sup> como en *The Sound Pattern of English* (SPE) plantean la representación de fonemas por medio de pequeñas matrices mientras que los procesos fonológicos tienen forma de relaciones funcionales entre dichas matrices. Lo anterior se asemeja visualmente a las descripciones simbólicas de las relaciones químicas; sin embargo, en estas teorías (también llamadas lineales) los rasgos distintivos no tienen una organización interna precisa. El problema es resuelto con los modelos jerárquicos, los cuales también cambian de forma drástica la representación de los fenómenos. Estos modelos tienen una representación gráfica arbórea que poco tiene que ver con las representaciones lineales. Finalmente teorías más recientes como la Teoría de la Optimidad cambian nuevamente la representación que ahora es tabular y no arbórea.

La lingüística y otras ciencias.

Por otra parte, hay muchos científicos que pretenden abordar los problemas del lenguaje. Entre otros, con mayor o menor éxito, destacan los sociólogos, psicólogos, matemáticos y computólogos. Ellos y otros científicos de diversas áreas se ven limitados al adentrarse en la lingüística por la diversidad de representaciones escritas que deben aprender para poder comprender los avances realizados en esta ciencia. Además, la comunicación de ideas con otras ciencias no es extraña, tan solo por dar un ejemplo podemos decir que la idea de valencia verbal está fuertemente relacionada con la valencia atómica de la cien-

---

<sup>1</sup>Se denominan rasgos distintivos los elementos fónicos mínimos susceptibles de oponer en una misma lengua dos enunciados de sentido diferente, cuyo significante es en todo lo demás idéntico. En español, por ejemplo, las palabras *pino* y *vino* se oponen por el rasgo de sonoridad, presente en la inicial de la segunda palabra y ausente en la inicial de la primera. Puede representarse el sistema de toda lengua mediante una matriz de rasgos distintivos, en la que los fonemas se definen por una elección positiva o negativa entre los dos términos de las diferentes oposiciones. Todos los rasgos distintivos utilizados por la lengua no intervienen necesariamente en la definición de todos los fonemas y pueden no presentar más que una función de rasgo pertinente. La matriz fonológica tiene en cuenta esta diferencia representando mediante un cero o un blanco la ausencia de elección, distintiva entre los dos términos de una oposición. Pero para conocer la realización de los fonemas en la pronunciación estándar de una lengua dada, se requiere una matriz fonética que represente igualmente los rasgos pertinentes.[Jakobson et al., 1969]

---

cia química[Tesniere et al., 2015]. Es bajo esta perspectiva cuando se pone de manifiesto la necesidad de un medio de comunicación científico y pertinente en lingüística que sea claro y facilite la comprensión rápida de los hechos lingüísticos para todos los científicos interesados ya sean parte la lingüística o de otras ciencias.

Numerosos lenguajes formales fuera de la lingüística han sido creados con los fines anteriormente mencionados, es el caso de las matemáticas y los lenguajes de programación. También podemos destacar las extensiones y adecuaciones que han tenido estos lenguajes a lo largo del tiempo, tal es el caso del calculo infinitesimal.

La idea de la posibilidad de definir un lenguaje formal para la lingüística que logre representar una lista, una matriz, un árbol o un desarrollo tabular por igual sería una forma de solucionar los problemas planteados anteriormente. Es necesario pensar en si es posible extender un lenguaje formal ya existente o, como el caso de la Química Orgánica, agregar nuevas representaciones escritas y gráficas que se conservan en todo el desarrollo de esta ciencia.

Un lenguaje formal para la lingüística.

Un nuevo lenguaje formal o una extensión de uno ya existente destinado a la lingüística debe cumplir ciertas características, por ejemplo:

- Ofrecer una representación formal de los fenómenos lingüísticos,
- Ser útil en investigaciones de los distintos niveles de análisis,
- Representar estructuras y procesos ofrecidos por cualquier teoría lingüística,
- Ser fácil de aprender para lingüistas y otros científicos interesados en el área,
- Ser fácilmente computable.

Así, en este trabajo de investigación se indagará sobre los problemas de for-

Objetivo y alcance.

## 1. INTRODUCCIÓN

---

malizar un lenguaje para la representación de fenómenos lingüísticos. Además, se examinará si es posible extender algún lenguaje ya existente y qué estructuras debe tener este posible lenguaje, así como su pertinencia en la ciencia lingüística. También, debe quedar claro que este trabajo no pretende contener una nueva teoría de análisis lingüístico, solo la representación unificada de las posturas ya utilizadas. En este trabajo se abordan, entonces, los siguientes puntos:

- Estudiar lenguajes formales existentes en la lingüística y en otras ciencias.
- Estudiar las necesidades de la lingüística en cuanto a la representación formal de fenómenos.
- Elaborar la propuesta de lenguaje.
- Representación de algunos ejemplos.



A lo largo de este capítulo se explicarán los conceptos básicos e indispensables para el claro entendimiento de este trabajo de investigación. Es necesario aclarar que la mayoría de estos conceptos no serán extraños para los iniciados en la ciencia lingüística, sin embargo, si pueden representar alguna dificultad para los menos expertos, es por eso que este capítulo tiene una importancia capital para la comprensión de este trabajo. Particularmente los conceptos presentados sobre el cálculo lambda y las expresiones simbólicas pueden resultar temas nuevos inclusive para algunos lingüistas.

También es necesario decir que este capítulo puede resultar para más de un lector un marco teórico bastante reducido; lo anterior es verdad, así como lo es también que a lo largo del presente trabajo de investigación se explican con el mayor detalle posible los conceptos teóricos adicionales que pudieran ser requeridos para su comprensión. Si al finalizar la lectura el lector necesitara alguna ampliación de los conceptos presentados, puede referirse a la bibliografía que se encuentra en la parte final del trabajo, en ella encontrará todos los textos que fueron leídos para la creación del presente escrito.

### 2.1. La ciencia lingüística

Si preguntamos por la definición más sencilla, todo profesor diría que la lingüística es «la ciencia que tiene como objeto de estudio el lenguaje y las lenguas»[Simone, 2001]. Para más de una persona, la definición anterior plantea una pregunta que es indispensable contestar antes de seguir cualquier investigación con relación a la lingüística: ¿Es la lingüística realmente una ciencia? Si comparamos la lingüística con ciencias consolidadas (física, biología, química), la primera crítica es por regla general que los métodos lingüísticos suelen ser bastante variables, sus técnicas de investigación son tan particulares que en algunas ocasiones son calificados como *controvertidos* incluso por los propios lingüistas.

Efectivamente, la lingüística carece de un aparato formal que le dé una capacidad de repetición y verificación experimental tan efectiva como en otras ciencias. Sin embargo, para juzgar a la lingüística correctamente es necesario tomar en cuenta algunas consideraciones importantes: la primera es que los fundamentos actuales de la lingüística provienen, tan solo por nombrar una fecha, del año de 1916 cuando fue publicado el *Curso de lingüística general* de Ferdinand de Saussure. Lo anterior pone en manifiesto que nos encontramos con una ciencia nueva con aproximadamente cien años de existencia. En segundo lugar, podemos decir que la lingüística, tal y como nos dice la definición mencionada al principio de este capítulo, tiene bien definido su objeto de estudio como lo haría la más formal y antigua de las ciencias. Sin embargo, a diferencia de las demás, el objeto de estudio de la lingüística es bastante esquivo. Mientras que la sangre siempre estará compuesta por los mismos elementos, una lengua es una mezcla muy particular de conocimiento e idiosincrasias que para llegar a su comprensión total no se puede ser tan concreto y sus métodos de estudio no podrían jamás tener el mismo rigor. Finalmente es necesario decir que el formalismo en lingüística si está bien definido y es tan preciso que para los estudiosos resulta

incluso obvio: las lenguas estudiadas son diseccionadas paso a paso según un determinado *nivel de análisis*.

### 2.1.1. Niveles de análisis lingüísticos

En lingüística la palabra *nivel* es fundamental debido a que permite establecer y ordenar de forma racional los fenómenos estudiados y darles un tratamiento adecuado, con los mismos conceptos y criterios, una vez establecido un determinado nivel.

La noción de nivel nos parece esencial en la determinación del procedimiento de análisis. Solo ella es adecuada para hacer justicia a la naturaleza articulada del lenguaje y al carácter discreto de sus elementos; ella sola puede permitirnos, en la complejidad de las formas, dar con la arquitectura singular de las partes del todo. El dominio en que la estudiaremos es el de la lengua como sistema orgánico de los signos lingüísticos. [Benveniste, 1971]

Cada nivel de análisis lingüístico tiene una unidad mínima de estudio, la cual es posible identificar en una unidad más elevada [Benveniste, 1971]. Este hecho permite decir que los niveles de análisis tienen una jerarquía que fue encontrada y definida gracias a un método distributivo:

Consiste en definir cada elemento por el conjunto de los alrededores en que se presenta, y por medio de una doble relación, relación del elemento con los demás elementos simultáneamente presentes en la misma porción del enunciado (relación sintagmática); relación del elemento con los demás elementos mutuamente sustituibles (relación paradigmática). [Benveniste, 1971]

Este método nos permitirá distinguir dos operaciones generales: sustitución para la relación paradigmática y segmentación para la relación sintagmática. Es

## 2. MARCO TEÓRICO

---

pertinente decir que para identificar el nivel con la jerarquía más baja la operación de sustitución será válida pero no lo sería la operación de segmentación.

Con esta explicación es claro que los análisis lingüísticos se identifican bajo una regla común y, en general, una regla regular que si bien escapa del común científico, ha favorecido la solución de numerosas preguntas sobre la naturaleza del lenguaje, los cambios que éste sufre y es totalmente pertinente. En la actualidad la lingüística distingue generalmente cuatro niveles de análisis lingüísticos [Bosque and Gutiérrez-Rexach, 2009]<sup>1</sup>:

**Fonología** Tiene como unidad mínima el fonema. En este nivel de análisis se estudia el sistema de sonidos de una lengua atendiendo los sonidos pertinentes y ordenados.

**Morfología** Tiene como unidad mínima el morfema. En este nivel se estudia la estructura interna de las palabras (bajo alguna definición de palabra), así como la formación de las mismas.

**Sintaxis** Tiene como unidad mínima el lexema. En este nivel se estudian las relaciones, estructura y ordenamiento de lexemas para formar estructuras mayores.

**Semántica** Tiene como unidad mínima la frase. En este nivel se estudia el significado lingüístico de las expresiones. Si bien el significado tiene pertinencia en niveles anteriores, la semántica se ocupa de explicar porqué una misma forma puede expresar significados variables en relación con el contexto.

Una de las propiedades más importantes de los niveles de análisis anteriores es que no funcionan de forma independiente, por lo que es posible, por ejemplo,

---

<sup>1</sup> Algunos autores han diferido en el número de niveles, por ejemplo en la *Nueva gramática de la lengua española: manual*, aquí mencionamos una clasificación bien aceptada y útil para esta tesis.

que un problema morfológico sea explicado con ayuda de la fonología (como la presencia de alomorfos) o un proceso sintáctico sea explicado por medios morfológicos (polisíntesis).

Finalmente es necesario decir que cada uno de estos niveles de análisis tiene sus propias herramientas conceptuales y criterios de validación que con el tiempo son cada vez más refinados y apuntan al avance científico de la lingüística.

## 2.2. Herramientas conceptuales de la lingüística

La lingüística ha usado a lo largo del tiempo diversas herramientas y estructuras provenientes de otras ciencias por medio de las matemáticas discretas.

Las *matemáticas discretas* son una rama de las matemáticas que se ocupan de los objetos que solo pueden asumirse como entes individuales. Los objetos discretos son a menudo asociados con la numeración, por lo tanto se consideran contablemente infinitos. Las matemáticas discretas abarca conocimientos como la estadística, la lógica proposicional, la teoría de grafos y la teoría de la computación donde se estudian temas que incluyen algoritmos, sus implementaciones y eficiencias[Renze and Weisstein, 2014, Hines and Montgomery, 2005].

Definición y ramas de las matemáticas discretas.

Durante los cuatro años de aprendizaje en la lingüística, se puede observar el uso recurrente de dos herramientas formales, las cuales serán descritas brevemente a continuación.

### 2.2.1. Lógica proposicional

La lógica se encarga de los métodos del razonamiento. Uno de sus propósitos es proporcionar las reglas para determinar la validez<sup>2</sup> de un razonamiento o argumento particular [Veerarajan, 2008]. Si fijamos una relación entre la lengua

---

<sup>2</sup>La validez consiste en estudiar las condiciones para que el pensamiento sea verdadero y las condiciones para que el pensamiento sea correcto.

y el pensamiento, la lingüística puede usar a la lógica para examinar la relación en términos estructurales.

### 2.2.1.1. Propositiones

Una proposición es un enunciado declarativo (o afirmación) que es verdadero o falso pero no ambos, es decir, tiene un valor de verdad. Por convención una proposición se denota con letras minúsculas cursivas, por ejemplo:  $p$ ,  $q$ ,  $r$ ...

Existen varias formas de clasificar proposiciones, aquí algunas de ellas:

1. Por su cantidad. Imagine el conjunto universal  $U$ , que contiene a los sujetos de un conjunto de proposiciones.
  - Particulares. El sujeto no agota el conjunto  $U$ .
  - Universales. El sujeto agota el término al que se refiere, es decir, el conjunto  $U$ .
2. Por su cualidad.
  - Afirmativas. Comunica un consentimiento de la cualidad atribuida al sujeto expresada en la proposición.
  - Negativas. Comunica una disconformidad de la cualidad atribuida al sujeto.
3. Por su valor veritativo.
  - Verdaderas. La proposición coincide con la realidad contextual en la que es expresada. Su valor de verdad es verdadero.
  - Falsas. La proposición discrepa de la realidad contextual en la que es expresada.
4. Por su complejidad.

- Primitivas. La proposición no contiene ningún conectivo lógico.
- Compuestas. La proposición contiene al menos un conectivo lógico. Su valor de verdad depende de como son combinados los correspondientes valores de las subproposiciones que la componen.

### 2.2.1.2. Conectivos lógicos y tablas de verdad

Como se dijo anteriormente, cada proposición puede tener un valor de verdad, falso o verdadero. También hay proposiciones compuestas, las cuales son resultado de unir dos o más proposiciones primitivas por medio de al menos un conectivo lógico, este proceso a su vez es recursivo, es decir, una proposición compuesta puede también estar conformada por otras proposiciones compuestas.

Un conectivo lógico es una función veritativa cuyo valor depende de los valores de sus operandos. Distinguimos tres conectivos lógicos bimembres y uno unimembre, todos definidos por medio de una tabla de verdad que muestra las relaciones entre los valores de verdad de las subproposiciones y de la proposición compuesta construida a partir de ellas[Veerarajan, 2008] (ver 2.1).

Definición de conectivo lógico.

Los conectivos lógicos son los siguientes:

**Conjunción** Sean dos proposiciones  $p, q$ . La conjunción se denota como  $p \wedge q$  donde su valor de verdad es  $V$  si ambas proposiciones son verdaderas. Su equivalente binario es la operación multiplicación.

**Disyunción** Sean dos proposiciones  $p, q$ . La disyunción se denota como  $p \vee q$  donde su valor de verdad es  $F$  si ambas proposiciones son falsas. Su equivalente binario es la operación suma.

**Condicional y bicondicional** Sean dos proposiciones  $p, q$ , también llamadas *hipótesis* y *consecuencia* respectivamente. La condición se denota como  $p \implies q$  donde su valor de verdad es  $F$  si la hipótesis  $p$  es  $V$  y la

## 2. MARCO TEÓRICO

---

consecuencia  $q$  es  $F$ . Si la condición funciona en los dos sentidos, es decir, si se cumple  $p \implies q$  y  $q \implies p$  entonces se llama bicondicional y se denota como  $p \iff q$ .

**Negación** Sea la proposición  $p$ . Su negación se denota como  $\neg p$  donde si  $p$  es  $V$  entonces  $\neg p$  es  $F$ . La negación no da como resultado una proposición compuesta, pues opera sobre una sola proposición primitiva (unimembre).

$p$	$q$	$p \wedge q$	$p \vee q$	$p \implies q$	$p \iff q$	$p$	$\neg p$
$V$	$V$	$V$	$V$	$V$	$V$	$V$	$F$
$V$	$F$	$F$	$V$	$F$	$F$	$F$	$V$
$F$	$V$	$F$	$V$	$V$	$F$		
$F$	$F$	$F$	$F$	$V$	$V$		

Tabla 2.1: Tablas de verdad para la definición de los conectivos lógicos.

### 2.2.1.3. Prioridades de conectivos

Cuando varios conectivos son parte de una proposición compuesta, se usan reglas para determinar qué conectivo se lleva a cabo primero. También se hace uso de paréntesis para evitar confusiones y se inicia preferencialmente de izquierda a derecha. La forma inequívoca de realizar procedimientos en el cálculo proposicional es la siguiente[Veerarajan, 2008]:

- Los términos entre paréntesis tienen prioridad. Si hay paréntesis embebidos, se resuelve primero el más interno.
- La negación tiene prioridad sobre todos los demás conectivos lógicos. De esta manera,  $\neg p \wedge q$  significa  $(\neg p) \wedge q$ , no  $\neg(p \wedge q)$ .
- La conjunción tiene prioridad sobre la disyunción. Así,  $p \wedge q \vee r$  significa  $(p \wedge q) \vee r$ , pero no  $p \wedge (q \vee r)$ .



- Los condicionales y bicondicionales tienen menor prioridad que otros conectivos. Entre ellos,  $\implies$  tiene prioridad sobre  $\iff$ .

### 2.2.1.4. Propiedades de los conectivos lógicos

Los conectivos lógicos, al igual que las operaciones matemáticas, tienen algunas propiedades que son de ayuda al momento de tratar con el cálculo de proposiciones. Por medio de estas propiedades se han descubierto numerosas equivalencias y leyes lógicas, por ejemplo las *leyes de Morgan*<sup>3</sup>.

**Asociatividad** Cuando una proposición compuesta tiene dos o más ocurrencias del mismo conectivo, el orden en que se resuelvan los conectivos no afectará el resultado siempre y cuando el orden de las proposiciones no cambie. Si el conectivo puede resolverse al asociar los dos primeros elementos a la derecha, eso quiere decir que es asociativo a la derecha, si es posible resolver el conectivo al asociar los dos primeros elementos a la izquierda, se dirá que es asociativo a la izquierda. En lógica, la conjunción y la disyunción cumplen con esta propiedad. Ejemplo:  $(p \vee q) \vee r \equiv p \vee (q \vee r)$ .

**Conmutatividad** También llamada permutación. Esta propiedad indica que el resultado de operar dos elementos no depende del orden. En lógica, la conjunción y la disyunción cumplen con esta propiedad. El bicondicional existe gracias a esta propiedad. Ejemplo:  $p \vee q \equiv q \vee p$ .

**Distribucionalidad** Se establece que un conectivo  $c_1$  (por ejemplo  $\wedge$ ) es distributivo respecto a otro conectivo  $c_2$  (por ejemplo  $\vee$ ) cuando aplicar  $c_2$  sobre las proposiciones  $q, s$  y operar  $c_1$  sobre el resultado junto con una nueva proposición  $p$ ; es igual que operar  $c_1$  sobre  $p, q$  y  $p, s$  para luego aplicar  $c_2$  con los resultados como operandos. Ejemplo:  $p \wedge (q \vee s) \equiv (p \wedge q) \vee (p \wedge s)$ , donde  $\vee$  es distributivo con respecto a  $\wedge$ .

---

<sup>3</sup>Estas leyes son: 1) La negación de la conjunción es la disyunción de las negaciones. 2) La negación de la disyunción es la conjunción de las negaciones [Renze and Weisstein, 2016].

### 2.2.2. Teoría de árboles

Un árbol es una estructura aplicada a una colección de elementos relacionados entre sí. En esta relación, dos elementos conectados directamente varían en importancia según una jerarquía. Un árbol se define de la siguiente manera:

Formalmente se define un árbol de tipo  $T$  como una estructura homogénea resultado de la concatenación de un elemento de tipo  $T$  con un número finito de árboles disjuntos, llamados subárboles. Una forma particular de árbol es el árbol vacío. [Cairó et al., 2006]

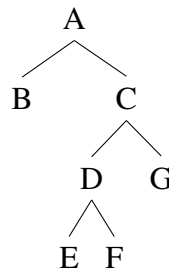


Figura 2.1: Ejemplo de estructura arbórea

Este tipo de estructura es caracterizado de la siguiente manera[Cairó et al., 2006]:

1. Todo árbol que no es vacío tiene un único nodo raíz.
2. Uno nodo  $X$  es descendiente directo de un nodo  $Y$ , si el nodo  $X$  es apuntado por el nodo  $Y$ . En este caso es común utilizar la expresión  **$X$  es hijo de  $Y$** .
3. Un nodo  $X$  es antecesor directo de un nodo  $Y$ , si el nodo  $X$  apunta al nodo  $Y$ . En este caso es común utilizar la expresión  **$X$  es padre de  $Y$** .
4. Se dice que todos los nodos que son descendientes directos –hijos– de un mismo nodo –padre– son hermanos.

5. Todo nodo que no tiene ramificaciones –hijos–, se conoce con el nombre de terminal u hoja.
6. Todo nodo que no es raíz ni terminal u hoja se conoce con el nombre de interior.
7. Grado es el número de descendientes directos de un determinado nodo.
8. Grado del árbol es el máximo grado de todos los nodos del árbol.
9. Nivel es el número de arcos que deben ser recorridos para llegar a un determinado nodo. Por definición la raíz tiene nivel 1.
10. Altura del árbol es el máximo número de niveles de todos los nodos del árbol.

De forma lineal, un árbol en general puede escribirse de tres formas diferentes, aunque es importante destacar que estas notaciones son usadas particularmente para árboles binarios<sup>4</sup>[Veerarajan, 2008]:

**Notación infija** Corresponde al recorrido inorden del árbol. El proceder consiste en recorrer el subárbol izquierdo, se visita la raíz y se recorre el subárbol derecho. La notación infija de la figura 2.1 ubicada en la página 14 es la siguiente: «B A E F D C G»

**Notación prefija** Corresponde al recorrido preorden del árbol. El proceder consiste en recorrer la raíz, luego visitar el subárbol izquierdo y luego el derecho en ese orden. La notación prefija de la figura 2.1 es la siguiente: «A B C D E F G»

---

<sup>4</sup>Un árbol binario es aquel en el que cada nodo puede tener dos o menos hijos, llamados hijo izquierdo e hijo derecho, según su ubicación con respecto al nodo padre. Estos árboles son los más socorridos para encontrar soluciones computacionales a una gran variedad de problemas[Cairó et al., 2006].

**Notación posfija** Corresponde al recorrido posorden del árbol. El proceder consiste en recorrer el subárbol izquierdo, luego el derecho y finalmente la raíz. La notación infija de la figura 2.1 es la siguiente: «B E F D G C A»

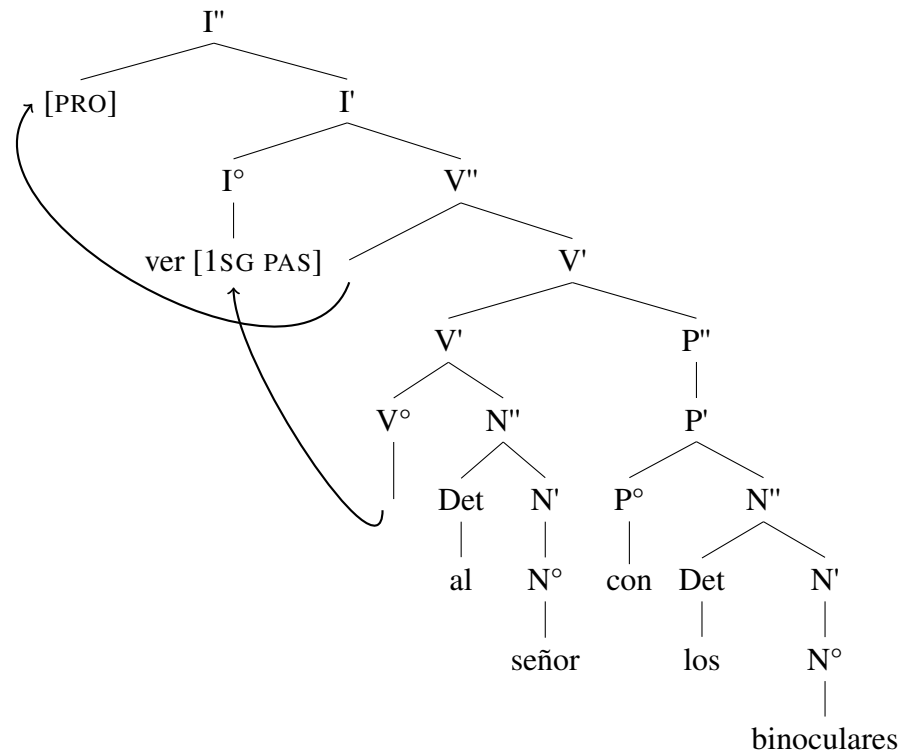


Figura 2.2: Ejemplo de estructura sintáctica representada mediante diagrama de árbol

Uso en la lingüística.

En lingüística, las estructuras arbóreas son utilizadas porque facilitan la visualización de las jerarquías de los diferentes elementos teóricos, es decir, reflejan los principios conforme a los que se construyen o pueden analizarse los elementos mayores a partir de sus constituyentes. La estructura que se produce será diferente según los principios a los que se adhiera el lingüista en su construcción[Jackendoff, 2011].

En la figura 2.2 (página 16) observamos la estructura de la oración «Vi al señor con los binoculares» y podemos hacer varias observaciones respecto a esta figura: *a)* representa la forma subyacente de la oración, también llamada estructura profunda en Chomsky (1965); *b)* gracias a la posición en el árbol, sabemos que *los binoculares* son el instrumento utilizado para ver y no una característica distintiva del *señor* al que se mira. En este segundo caso, la frase de la preposición se encontraría dentro de la *N'*; *c)* las flechas indican reglas de movimiento, en este caso la regla se llama *mover  $\alpha$*  que consiste en mover un elemento a otro lugar en la estructura[Jackendoff, 2011].

Finalmente, es necesario reiterar que las estructuras arbóreas son ampliamente utilizadas porque con ellas es posible representar jerarquías, reglas y restricciones que se observan en lingüística, aunque quizás su principal inconveniente es la dificultad de representar el orden de aplicación de las mismas.

## 2.3. Cálculo Lambda

A principios del siglo XX, antes de la invención de las computadoras, se pretendía precisar qué cosas podíamos adjetivar como *calculables*. Más allá de las operaciones aritméticas y sin poder omitir su importancia, se exploraba en procedimientos con pasos bien definidos: algoritmos. En ese momento, la herramienta más importante para diferenciar lo *calculable* de lo *no calculable* eran las matemáticas, pues antes de las computadoras lo más automático que se tenía a la mano eran las calculadoras.

El punto de partida de los primeros trabajos sobre este tema fueron las cuestiones fundamentales que el matemático alemán David Hilbert formuló en 1928, durante el transcurso de un congreso internacional[Castro Peña, 2016]:

1. ¿Son completas las matemáticas?, en el sentido de que pueda probarse o no cada aseveración matemática.

## 2. MARCO TEÓRICO

---

2. ¿Son las matemáticas consistentes?, en el sentido de que no pueda probarse simultáneamente una aseveración y su negación.
3. ¿Son las matemáticas *decidibles* (.SIC)? en el sentido de que exista un método definido que se pueda aplicar a cualquier aseveración matemática y que determine si dicha aseveración es cierta.

Teorema de la incompletitud de Gödel.

En respuesta a las preguntas de Hilbert, Kurt Gödel, filósofo y matemático, señaló en 1931 que[Castro Peña, 2016]:

Todo sistema de primer orden consistente que contenga los teoremas de la aritmética y cuyo conjunto de axiomas sea recursivo no es completo.[Feferman, 1995]

Dicho de otra manera, no es posible encontrar un algoritmo que determine la verdad o falsedad de cualquier proposición en un sistema formal, pues aunque se encuentre el algoritmo, éste no puede ser probado dentro del mismo sistema<sup>5</sup>. En ese momento aún se encontraba presente la idea de establecer un conjunto de axiomas no recursivos<sup>6</sup>, que por medio de un sistema deductivo pudiera resolver las preguntas de Hilbert, sin embargo, Gödel elimina también esta opción al decir[Castro Peña, 2016]:

Generalización del Teorema de la incompletitud de Gödel.

Ningún sistema deductivo que contenga los teoremas de la aritmética, y con los axiomas recursivamente numerables puede ser consistente y completo a la vez.[Feferman, 1995]

Trabajos posteriores, aportaron evidencias que reafirman las ideas de Gödel. Alonzo Church, por ejemplo, creó un lenguaje formal para investigar la defini-

---

<sup>5</sup>A esto se le conoce como *Entscheidungsproblem* o problema de la decisión. Consiste en encontrar un algoritmo general que decidiera si una fórmula del cálculo de primer orden es o no un teorema.

<sup>6</sup>Hasta el día de hoy no se conoce una forma de decidir si una *fórmula* es un axioma.

ción de función y sus aplicaciones dentro de cualquier sistema[Church, 1936b]. A este lenguaje se le conoce como *cálculo  $\lambda$* . [Arbiser, 2005]

Church distingue con el cálculo  $\lambda$  dos partes en la definición de una función: el nombre (miembro izquierdo), que identifica la función; y la regla (miembro derecho) que describe el significado de la función, tomaremos por ejemplo la función (2.1).

$$f(x) = x^2 + y \quad (2.1)$$

$$f(y) = x^2 + y \quad (2.2)$$

$$f(x, y) = x^2 + y \quad (2.3)$$

En el cálculo  $\lambda$  se afirma que el nombre de la función no tiene más relevancia que expresar la función de forma implícita, solo son relevantes las variables y reglas de la función y por eso (2.1), (2.2) y (2.3), aunque tienen una misma regla de correspondencia no son la misma función y representan diferentes espacios geométricos: (2.1) corresponde a una parábola, (2.2) corresponde a una línea recta y (2.3) corresponde a un plano.

De esta manera, el cálculo  $\lambda$  propone representar la función de forma *explícita* con un encabezado que inicia con la letra  $\lambda$  y termina con un punto (.), así las funciones (2.1), (2.2) y (2.3) tienen su correspondencia en  $\lambda$  en (2.4) (2.5) y (2.6) respectivamente:

$$\lambda x.x^2 + y \quad (2.4)$$

$$\lambda y.x^2 + y \quad (2.5)$$

$$\lambda x, y.x^2 + y \quad (2.6)$$

Aquí ya podemos observar algunos conceptos de empleo del cálculo  $\lambda$ : en (2.4), la variable  $x$  espera que se especifique su valor ya que está presente en el Variables ligadas y libres.

encabezado, por lo que se conoce como variable ligada al contexto, mientras que  $y$  no recibirá ningún valor concreto, por lo que se dice que es una variable libre del contexto. En (2.5) sucede el caso contrario, mientras que en (2.6) ambas son variables ligadas.

Para utilizar el cálculo  $\lambda$  se utilizan abstracciones del siguiente tipo:

$$\begin{aligned} f(x) &= x^2 + y \\ f(x) &= M \\ (\lambda x.M) \\ (\lambda x.M)N \end{aligned} \tag{2.7}$$

En la abstracción (2.7),  $M$  representa la regla de correspondencia respecto a la variable  $x$ , por su parte  $N$  representa el argumento o parámetro que recibirá la función. De esta forma se puede deducir que el cálculo  $\lambda$  es asociativo a la izquierda<sup>7</sup> (Ver sección 2.2.1.4). Supongamos el caso de que hay 3 expresiones  $\lambda (e_1, e_2, e_3)$ , de la siguiente manera:

$$\begin{aligned} (e_1 e_2 e_3) \\ (e_4 e_3) \\ (e_5) \end{aligned} \tag{2.8}$$

La forma de evaluar una expresión  $\lambda$  como la escrita en (2.8) es tomar  $e_1$  y darle como argumento la expresión  $e_2$ , debido a la asociatividad a la izquierda. Eso da como resultado una nueva expresión  $e_4$  que tomará como argumento  $e_3$ . Esta evaluación se conoce como reducción  $\beta$ , que corresponde a «tomar el parámetro formal  $x$  y sustituirlo por el parámetro real  $N$ » (Véase (2.7)) y se denota como  $Mx \xrightarrow{\beta} N$ . [Arbiser, 2005]

Reducción  $\beta$ .

---

<sup>7</sup>A esta forma de reducción de una expresión  $\lambda$  se le llama *reducción en orden normal*. Existe también el *reducción en orden aplicativo*, el cual reduce primero la expresión más interna, igual que en las matemáticas.



Ahora suponga las siguientes abstracciones que cambian el orden de dos variables:

$$\begin{aligned}
 &(\lambda xy.yx) \\
 &\text{por distribución:} \\
 &(\lambda xy.yx) \equiv (\lambda x.(\lambda y.yx))
 \end{aligned}
 \tag{2.9}$$

Parece que (2.9) es una postura matemáticamente correcta, de hecho lo es; sin embargo, si evaluamos la expresión anterior con parámetros con el mismo nombre, es decir  $N_1 = y$  y  $N_2 = x$  veremos que la expresión es inconsistente y nos daría como resultado que  $yx \rightarrow_\beta xx$  lo cual es falso. Para solventar esta inconsistencia, se hace obligatorio realizar una conversión  $\alpha$  o renombramiento de variables: Conversión  $\alpha$ .

$$\begin{aligned}
 &(\lambda x.(\lambda y.yx))y \quad x \equiv_\alpha (\lambda s.(\lambda q.qs))y \quad x \\
 &(\lambda s.(\lambda q.qs))y \quad x \rightarrow_\beta (\lambda q.qy)x \\
 &(\lambda q.qy)x \rightarrow_\beta xy
 \end{aligned}
 \tag{2.10}$$

De esta manera, podemos observar por medio de (2.10) que la conversión  $\alpha$  nos dice que «en una serie de expresiones que aparecen en un contexto, todas las variables ligadas de tales expresiones se consideran diferentes a las libres», lo que es lo mismo que una variable no puede ser libre y ligada a la vez en un solo contexto.

Finalmente, es importante decir que el cálculo  $\lambda$  es equivalente a una *máquina de Turing*<sup>8</sup>, y es una herramienta muy utilizada por los lingüistas en la semántica formal[Partee et al., 1990], lo que pone en manifiesto la idea de que

---

<sup>8</sup>Turing expuso su tesis como un teorema demostrado, utilizó su concepto de máquina teórica y logró demostrar que existen funciones que no son *calculables* por un método definido y en concreto que el Entscheidungsproblem era uno de estos problemas: «La clase de las funciones que pueden ser calculadas mediante un método definido coincide con la clase de las funciones calculables mediante una Máquina de Turing»[Turing, 1936].

las lenguas pueden verse como un sistema formal. Originalmente el cálculo  $\lambda$  fue usado para mostrar qué problemas no eran *calculables*. En este sentido, tanto Alonzo Church como Alan Turing llegan a la conclusión de que «la clase de funciones que pueden ser calculadas mediante un algoritmo coincide con la clase de las funciones recursivas». La tesis Church-Turing es hoy en día un axioma de la teoría de la computación:

En los sistemas deductivos donde el conjunto de reglas de producción, y el conjunto de axiomas sea recursivamente numerable, una función (parcial) aritmética es calculable si y solo si es una función (parcial) recursiva.

### 2.4. Expresiones S

En 1956 John McCarthy (1927-2011) crea el concepto lógico-matemático de *Inteligencia Artificial*, y posteriormente en 1958 propone la utilización de expresiones simbólicas<sup>9</sup> (también llamadas expresiones S) para la creación de un lenguaje que representara una máquina de turing completa<sup>10</sup> este lenguaje sería la base del primer diseño del lenguaje formal LISP Processing (LISP)<sup>11</sup>. A partir de entonces el concepto de *Inteligencia Artificial* se comercializa y grandes avances científicos se dan a conocer, las computadoras hablan, juegan ajedrez, pero estarán siempre limitadas por ser máquinas, aunque sean más rápidas que el

---

<sup>9</sup>El documento original fue publicado hasta el año de 1960 bajo el nombre de *Recursive functions of symbolic expressions and their computation by machine (part 1)*.

<sup>10</sup>Para leer sobre la máquina de turing véase el capítulo 8 de Gurovich (2008).

<sup>11</sup>LISP es una familia de lenguajes caracterizados por usar fórmulas elaboradas con expresiones simbólicas. El nombre deriva de una de las actividades realizadas más comúnmente en LISP: el procesamiento de listas. En los lenguajes LISP se suele describir un problema con la declaración de un conjunto de reglas y propiedades a cumplir, sin embargo, también es posible describir procedimientos en forma de algoritmos. Estos lenguajes son muy socorridos en la actualidad para la comprobación automática de teoremas, la planificación y programación de sistemas de inteligencia artificial. Una introducción más detallada sobre LISP puede encontrarse en capítulo 1 de Seibel (2005).

cerebro humano nunca podrán mover el corazón, respirar, escribir, ver, oír, sentir y pensar al mismo tiempo; la computadora está destinada a la personalización no a la imitación.

A pesar de las limitaciones mencionadas, las expresiones S han perdurado y siguen siendo ampliamente utilizadas en las ciencias de la computación. Una expresión S se define de la siguiente forma:

Definición de expresión simbólica o expresión S.

S-expressions are data structures for representing complex data. They are either byte-strings (“octet-strings”) or lists of simpler S-expressions. [Rivest, 1997]<sup>12</sup>

Una expresión S es una forma adecuada para la expresión de estructuras de datos arbitrarias que resulten ser más complejas. Debido a su origen, son compuestas por cadenas de bytes o en base octal, tal como la definición anterior señala.

Las expresiones S, tomadas en forma independiente a su aplicación particular, tienen los siguientes propósitos[Rivest, 1997]:

- Generalidad. Pueden representar correctamente cualquier tipo de dato,
- Legibilidad. Son fáciles de examinar y entender en estructura,
- Economía. Son representaciones compactas,
- Transportación. Son fáciles de transportar como datos en un medio de comunicación electrónico,
- Flexibilidad. Son relativamente fáciles de modificar y extender a otras estructuras más complejas,

---

<sup>12</sup>Las expresiones S son estructuras de datos que sirven para representar datos complejos. Son cadenas de bytes (“cadenas octeto”) o listas de expresiones S más simples. Una cadena octeto es una secuencia finita de octetos de bits.

## 2. MARCO TEÓRICO

---

- Estandarización. Es posible establecer una forma canónica de representación de las expresiones S,
- Eficiencia. Su representación facilita su fácil procesamiento.

Un ejemplo de expresión S es el siguiente[Rivest, 1997]:

(1) (snicker “abc” (#03# |YWJl))

Esta expresión esta compuesta de los siguientes elementos:

- La cadena octeto *snicker*,
- La cadena octeto “*abc*”,
- Una lista de dos elementos:
  - La cadena hexadecimal #03#
  - La cadena base 64 |YWJl|, que es lo mismo que la cadena “*abc*”.

En forma general, el uso de expresiones S supone pocas pero importantes restricciones[Rivest, 1997]:

- No admite elementos vacíos,
- Una expresión S no puede tener como primer elemento una lista.

Los detalles, uso y pertinencia de las expresiones S serán abordados a lo largo del capítulo 4.

Entre las formas de representación del pensamiento, los lenguajes formales han sido hasta el día de hoy la principal forma de escritura científica. Estos lenguajes son tan recurrentes porque representan de forma sencilla los descubrimientos científicos a la vez que permiten el intercambio de conocimiento.

Aunque a primera vista estos lenguajes formales podrían parecer todos iguales, es posible distinguir ciertos paradigmas que obedecen a la forma en la que se observan los fenómenos de su respectiva ciencia. Es por eso, que una ciencia, dependiendo de su objeto de estudio, privilegiará una representación lógica mientras otra quizás privilegie una representación funcional. Para dar un ejemplo concreto, las ciencias que cuantifican grados de incertidumbre (algún tipo de probabilidad) gustan de lenguajes puramente algebraicos para estos fines, la incertidumbre parece tener siempre un valor concreto.

La lingüística por su parte, es una ciencia que puede ver un solo fenómeno desde varias perspectivas. Esto es debido a que muchos de los fenómenos lingüísticos pueden explicarse de formas diferentes, lo cual es útil ya que al ser una ciencia nueva, se toma la perspectiva más útil y más abarcadora dependiendo del

marco teórico de la investigación.

A continuación explicaremos algunas de las representaciones formales usadas en lingüística, si bien al juntarlas no forman un lenguaje formal, si es importante decir que cada una de estas representaciones es usada por ser útil y necesaria; aunque la lingüística no es una ciencia exacta, es tan formal como cualquier otra.

## 3.1. Lenguajes declarativos y lenguajes imperativos

Dentro de las variadas clasificaciones de lenguajes formales destaca la jerarquía de tipos de gramáticas de estructura de frase generadas por los lenguajes del mismo tipo, descrita por Noam Chomsky. En esta visión, una gramática  $G = \{V_n, V_t, S, R\}$ , es una estructura de conjuntos que consta de cuatro elementos[Veerarajan, 2008]:

1.  $V_n$  es un conjunto finito de símbolos no terminales de un vocabulario  $V$ , sustituibles por otros símbolos.
2.  $V_t$  es un conjunto finito de símbolos terminales de  $V$ , que no son sustituibles por otros símbolos.
3.  $S$  es un símbolo de inicio, que forma parte de  $V_n$  y desde el cual se empiezan a construir palabras.
4.  $R$  es un conjunto de reglas gramaticales de producción.

La definición anterior ha tenido gran repercusión en los estudios lingüísticos porque formaliza una buena parte de lo que los estudiosos describen cuando analizan las lenguas humanas. Para ejemplificarlo, tomaremos la oración usada como ejemplo tradicionalmente:

(2) Juan golpeó a Pedro en la cara.

La oración presente en (2) puede ser representada por medio de la siguiente gramática, cuyo símbolo inicial será  $O$  (en lugar de  $S$ ):

$$\begin{aligned}
 V_n &= \{SN, SV, SP, O, N, P, V, DET\} \\
 V_t &= \{Juan, golpeó, a, Pedro, en, la, cara\} \\
 R &= \{O \rightarrow SN \quad SV, \\
 &\quad SN \rightarrow (DET) \quad N, \\
 &\quad SV \rightarrow V_{trans} \quad SN \quad (SP), \\
 &\quad SP \rightarrow P \quad SN, \\
 &\quad N \rightarrow Juan, Pedro, cara, \\
 &\quad V \rightarrow golpeó, \\
 &\quad P \rightarrow a, en, \\
 &\quad DET \rightarrow la\}
 \end{aligned}$$

Cualitativamente, podemos decir que existe una regla sintagmática que reescribe un elemento de una categoría ( $O$ ) a otras más complejas ( $SN, SV, SP$ ) que finalmente están compuestas por categorías léxicas ( $DET, N, V, P$ ), las cuales se reescribirán como las palabras del conjunto del vocabulario terminal  $V_t$ . Entonces, una oración esta compuesta de un sintagma nominal y otro verbal; el sintagma nominal se integra por un determinante optativo seguido de un nombre; un sintagma verbal esta integrado por un verbo transitivo seguido de un sintagma nominal y un sintagma prepositivo se integra de una preposición seguida de un sintagma nominal [Arís, 2015].

La sintaxis de una lengua natural es muy compleja, por lo que escribir la serie completa de reglas sintácticas es una labor imposible. Las breves gramáticas que se escriben, son propias de la caracterización de un fenómeno particular. En el

caso de la gramática de (2), cada regla tiene una forma  $w_1 \rightarrow w_2$ , donde  $w_1$  es un símbolo único no terminal, por lo que se clasifica como una gramática tipo 2 o libre de contexto.

Es verdad que más adelante retomaremos estos conceptos y profundizaremos en el análisis de representación de este tipo de gramáticas, no obstante, en este momento es de nuestro interés centrarnos en una característica diferente de estas gramáticas formales: cada regla perteneciente al conjunto  $R$  declara una instrucción de desarrollo de una categoría.

En este sentido, es necesario destacar que los lenguajes formales representan gramáticas que son útiles para formar reglas compatibles con la capacidad de ejecución de una computadora, no directamente con la capacidad cognitiva del ser humano. A este tipo de lenguajes se les denomina *lenguajes imperativos*. En otras palabras, un lenguaje imperativo indica los pasos a seguir para obtener una estructura lingüística *superficial*.

La lingüística moderna basa sus más grandes avances en el principio de «describir la lengua sin normar sobre ella». Fructífero ha sido el resultado de seguir dicho principio y, como ya hemos señalado, también problemático al comparar el quehacer lingüístico actual con otras ciencias. La mayoría de las ciencias termina por declarar una serie de pasos e instrucciones que conducirán a la solución de un problema. Posteriormente, y muchas veces en base al sistema de «prueba-error», cada uno de los algoritmos es probado y elegido el que mejor cumple con las exigencias científicas. Aunque muchos de los paradigmas en este sentido son improbables en la realidad, se asumen ciertos. Este es el comportamiento de los lenguajes formales imperativos.

Dada esta visión, la lingüística trabaja de una forma diferente, muchas veces contraria. La lingüística siempre da como referencia pequeñas afirmaciones o restricciones visibles en un fenómeno, para posteriormente *declarar* las posibles condiciones que permiten llegar a la solución. Si buscáramos un tipo de lenguaje que fuera de la mano con esta forma de trabajo, llegaríamos a los *lenguajes*

Lenguaje imperativo.



### 3.1. Lenguajes declarativos y lenguajes imperativos

*declarativos*. Son denominados así los lenguajes que buscan describir las condiciones en las cuales se realiza un problema al tiempo que se indica que se desea obtener, pero no requiere especificar paso a paso cómo hacerlo. Lenguaje declarativo.

Un ejemplo de estas diferencias puede mostrarse a continuación usando un código de un lenguaje de programación puramente imperativo como Java, y otro puramente declarativo como lo es Haskell [Sjösten, 2012]:

```
1  -- Código declarativo en Haskell
2  fact n = product [1 .. n]
```

```
1  /* Código imperativo en Java */
2  class ClaseFact {
3  public static long fact(long n) {
4      return n == 0 ? 1 : n * fact(n - 1);
5  }
```

Como se puede apreciar, en el código Haskell, solo se declara la existencia de una función llamada «fact» y se dice qué es. En cambio, en Java, se describe bajo la sintaxis del lenguaje, el proceso que debe hacer el lector;<sup>1</sup> si el número cumple con la condición de ser igual a cero se regresará uno, y si es otro caso, cómo se hace la operación. Por supuesto, Haskell también podría describir un procedimiento como lo hace Java, mientras que una declaración de existencia es más complicada de implementar en Java.

```
1  -- Código imperativo en Haskell
2  fact n = if n < 2 then 1 else n * fact (n-1)
```

---

<sup>1</sup>De forma predeterminada, el lector es un intérprete de computadora. Otra forma, es la lectura del lenguaje con el fin de comprender un procedimiento, esto no requiere el uso de la computadora, es una actividad cognitiva de lectura que un individuo hace.

La lingüística debería incluir, dentro de sus representaciones formales, una manera de lenguaje declarativo, porque son pocas las ocasiones en las que el lingüista sabe cómo se lleva a cabo paso a paso un fenómeno, sobre todo porque en el lenguaje los procedimientos son cognitivos. En este sentido, aunque las gramáticas formales formuladas por Chomsky quieren explicar fenómenos de forma cognitiva [Arís, 2015], la elección de usar representaciones fuertemente imperativas es un error en el cual no se ha cavilado con mucha frecuencia.

## 3.2. Lenguajes Formales y sus representaciones gráficas

Al inicio de la sección 3.1 se habló de las gramáticas formales como un conjunto de reglas de generación. Es común que este tipo de gramáticas se escriban en notación matemática igual que en teoría de conjuntos. En este apartado veremos otras formas de representación que acompañan a las notaciones matemáticas con el objetivo de ser más accesible el conocimiento para aquellos que no son expertos en matemáticas. Estas formas de representación son las gráficas arbóreas y las pseudográficas anidadas con corchetes.

Para recordar lo visto anteriormente, ejemplificaremos con una nueva oración:

(3) Pedro fue golpeado (por Juan) en la cara.

La oración anterior es la forma pasiva del ejemplo (2), por lo que su gramática formal es muy similar, podemos decir que se adicionan tres reglas y se adiciona un termino final. Debe hacerse la distinción entre verbo en forma pasiva y el verbo transitivo en forma activa. Asimismo debe incluirse una segunda SP que permita recuperar el argumento que fue degradado a oblicuo. La gramática sería la siguiente:

$$\begin{aligned}
V_n &= \{SN, SV, SP, O, N, P, V, DET\} \\
V_t &= \{Juan, golpeó, golpeado, a, por, Pedro, en, la, cara\} \\
R &= \{O \rightarrow SN \quad SV, \\
&\quad SN \rightarrow (DET) \quad N, \\
&\quad SV \rightarrow V_{trans} \quad SN \quad (SP), \\
&\quad SV \rightarrow V_{pas} \quad (SP), \\
&\quad V_{pas} \rightarrow AUX \quad V \quad (SP), \\
&\quad SP \rightarrow P \quad SN, \\
&\quad N \rightarrow Juan, Pedro, cara, \\
&\quad AUX \rightarrow fue, \\
&\quad V \rightarrow golpeó, golpeado, \\
&\quad P \rightarrow a, en, por, \\
&\quad DET \rightarrow la\}
\end{aligned}$$

### 3.2.1. Gráficas arbóreas

Tanto la gramática presentada en la sección 3.1 como la anterior, podrían ser difíciles de leer, es por esta razón que los lingüistas no han adoptado la notación antes utilizada para explicar sus gramáticas. Consecuentemente se han requerido notaciones alternativas, de las cuales la más popular es una estructura arbórea.

Para formar una estructura arbórea partiendo de la gramática se debe tener una sola regla: Cada ramificación del árbol corresponde a una regla de reescritura. Entonces, la primera regla de las gramáticas presentadas anteriormente se representa como en la figura 3.1.

Debemos hacer notar que bajo esta equivalencia, la gráfica arbórea no necesariamente es una gráfica con ramificación binaria, esto hace diferencia entre una representación y una teoría específica como puede ser la teoría X-barra. De

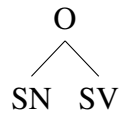


Figura 3.1: Representación arbórea de una regla gramatical

hecho, el número de ramificaciones esta ligado al número de derivaciones presentes en la regla, tal y como sucede en la figura 3.2 donde la expansión completa incluye tres elementos.

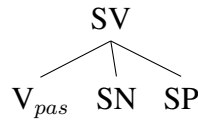


Figura 3.2: Representación arbórea de la regla gramatical no binaria:  $SV \rightarrow V_{pas} SN SP$

Lo anterior es todo lo necesario para saber construir el árbol correspondiente a (3), presente en la figura 3.3

Las ventajas de este tipo de representaciones son numerosas, entre ellas destacan[Arís, 2015]:

- Pueden representar ambigüedades sintácticas.
- Muestran la jerarquía y orden de palabras.
- Expresan jerarquías en la relación de dominio.
- Mantienen a la vista toda la estructura oracional y las relaciones sintácticas establecidas entre las categorías gramaticales.

Pese a esto, algunas de estas ventajas pueden no ser suficientes o ser inconvenientes en algunos casos: en estos árboles, no es posible representar un orden específico de los cambios en las estructuras si es que los hubiera. Al expresar

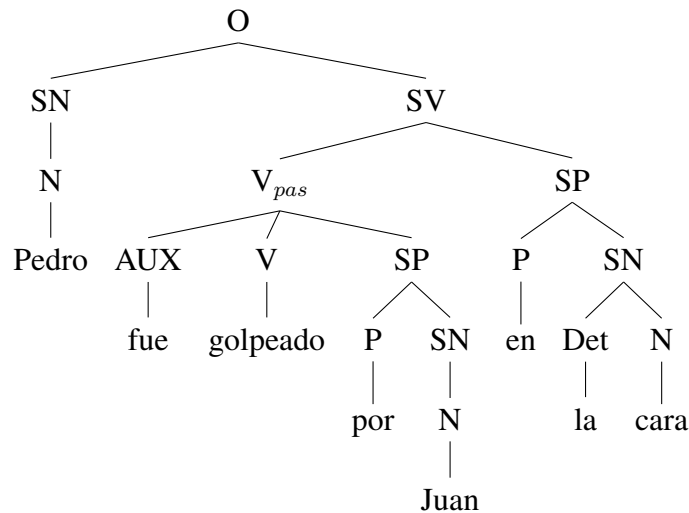


Figura 3.3: Estructura completa de la oración pasiva del ejemplo (3)

con claridad las estructuras, se tiene como contrapunto el amplio espacio que estas representaciones pueden ocupar, característica que pasa también a ser una dificultad relativa en la creación de textos impresos. Además puede resultar de poco interés recuperar el conocimiento a nivel formal por medio de estas estructuras, puesto que para esos motivos se prefiere comúnmente la representación con cajas o corchetes.

### 3.2.2. Corchetes rotulados

Las gráficas arbóreas son equivalentes a los corchetes rotulados. Estos corchetes son usados comúnmente cuando un autor pretende hacer énfasis en la descripción y no en el procedimiento teórico. La idea es que cada regla de transformación se encierre entre corchetes. Se escribe el primer elemento de la regla como primer elemento en la lista que se encierra entre corchetes y además (opativamente) se coloca una tipografía de subíndice. Los demás elementos se escriben con la tipografía estándar. De esta forma, el árbol de la figura 3.1 es

equivalente a (4)

(4) [<sub>O</sub> SN SV]

Así, podríamos hacer la equivalencia de (3) y la figura 3.3 con cierta facilidad, veríamos como en (5) se incluyen exactamente los mismos elementos pero la anidación de corchetes hace difícil su lectura en estructuras que podrían considerarse no tan básicas. Adicionalmente, la jerarquía que era totalmente visible en la figura 3.3, ya no lo es apreciable en (5).

(5) [<sub>O</sub> [<sub>SN</sub> [<sub>N</sub> Pedro]] [<sub>SV</sub> [<sub>V<sub>pas</sub></sub> [<sub>AUX</sub> fue] [<sub>V</sub> golpeado] [<sub>SP</sub> [<sub>P</sub> por] [<sub>SN</sub> [<sub>N</sub> Juan ]]]] [<sub>SP</sub> [<sub>P</sub> en ] [<sub>SN</sub> [<sub>Det</sub> la ] [<sub>N</sub> cara ]]]]]

De cualquier manera, la supuesta ventaja de la representación gráfica, se desvanece al transcribirla en un documento. En seguida se muestra el código con el que se generó la figura 3.3 para este documento.

```

1  \Tree [.O
2    [.SN [.N Pedro ] ]
3    [.SV
4      [.V$_{pas}$
5        [.AUX fue ]
6        [.V golpeado ]
7        [.SP
8          [.P por ]
9          [.SN [.N Juan ] ] ] ]
10   [.SP
11     [.P en ]
12     [.SN
13       [.Det la ]
14       [.N cara ] ] ] ] ]

```

Para mantener un mejor orden y entendimiento sobre la figura, se usó una tabulación y un salto de línea que no es correspondiente a cada nivel del árbol pero bien se puede hacer esa referencia que soluciona totalmente el problema de la visualización de la jerarquía y el orden de palabras. Esto es una solución implementada en varios lenguajes formales de la computación y según Osvaldo Cairó (2006), existen cinco formas de representar una estructura de árbol:

- Diagramas de Venn o cajas.
- Anidación de paréntesis o corchetes.
- Notación decimal de Dewey.
- Notación indentada.
- Grafo.

La solución aquí expuesta brevemente es una combinación entre la anidación de paréntesis y la notación indentada. Es decir, el uso de estructuras arbóreas es fundamental en muchas ciencias y, en este caso particular, aporta mucho más a la teoría lingüística que cualquier otra estructura de datos, sin embargo, las formas de representación son por sí mismas insuficientes, cualquier lenguaje elaborado especialmente para la lingüística debe hacer accesible una representación abreviada (como lo son los corchetes o las representaciones de conjuntos vistas anteriormente) que conserve el orden de una representación gráfica (como lo son los árboles) que a su vez sea obtenida a partir de la representación abreviada, la cuál es la representación origen.

#### 3.2.3. Matrices

Otra de las representaciones formales usadas por la lingüística más comúnmente son las matrices. Las matrices son por naturaleza una representación declarativa del conocimiento por que mencionan características que en conjunto

dan forma a una entidad. Al parecer fueron utilizadas en lingüística por primera vez por Martin Kay, investigador de *Xerox*, en 1979.

Las matrices fueron usadas en teorías formales *constraint-based* que se identifican por estudiar la estructura superficial a nivel gramatical y dan cuenta de restricciones. Posteriormente, estas teorías formales serían muy utilizadas en fonología, semántica y pragmática; donde tomarían forma de representaciones *feature-based* o basadas en rasgos donde son parte de las formulas de las gramáticas dependientes del contexto[Arís, 2015, Ortiz, 2000].

Actualmente, las matrices suelen usarse para expresar dependencias o relaciones entre elementos gramaticales. La definición informal de estas estructuras es la siguiente:

Una matriz de rasgos es un conjunto de pares atributo valor. Un atributo se extrae de un conjunto que representa el rango de propiedades lingüísticas. Un valor puede extraerse de un conjunto que representa el dominio de propiedades lingüísticas. Un valor puede estar formado por una matriz de rasgos[Arís, 2015].

$$\begin{bmatrix} \text{alto} & + \\ \text{retraído} & - \\ \text{redondeado} & + \end{bmatrix} \quad (3.1)$$

$$\begin{bmatrix} \text{alto} & + \\ \text{retraído} & + \\ \text{redondeado} & + \end{bmatrix} \quad (3.2)$$

$$\begin{bmatrix} \text{LEX} & \text{gatos} \\ \text{CAT} & \text{N} \\ \text{CONC} & \begin{bmatrix} \text{GEN} & \text{MASC} \\ \text{NUM} & \text{PLU} \end{bmatrix} \end{bmatrix} \quad (3.3)$$



La matriz (3.1) representa a la vocal [u] del turco, mientras que (3.2) representa a la vocal [ü] de la misma lengua. Ambas son matrices con valores simples con los que pueden caracterizarse ambas vocales, lo cual es necesario debido a que en turco «la vocal del sufijo de plural concuerda con la vocal de la raíz en cuanto al rasgo *retraído*»[Contreras and Lleó, 1982].

Por su parte, la matriz (3.3) representa un elemento léxico *gatos*, que tiene una categoría gramatical CAT y una concordancia CONC que a su vez esta compuesta por dos elementos: género GEN y número NUM.

Adicionalmente, es necesario decir que las operaciones básicas que se efectúan sobre estas estructuras son la unificación y la herencia. La herencia permite representar la naturaleza jerárquica de un elemento, de esta manera la información compartida por varios elementos será expresada una sola vez. Por otro lado, la unificación, es un mecanismo básico que permite integrar las especificaciones locales con las heredadas[Ortiz, 2000].

Por ejemplo, según sus rasgos fonológicos, sabemos que el conjunto de sonidos [t, d, s, z] poseen tres rasgos pertinentes comunes: son consonánticos, no resonantes y corales, por lo que podemos agruparlos como sonidos corales, cuya matriz de rasgos sería representada en (3.4)[Riggle, 2012].

$$\begin{bmatrix} \text{cons} & + \\ \text{son} & + \\ \text{cor} & + \end{bmatrix} \quad (3.4)$$

Una vez definida esa clasificación, también podemos decir, por ejemplo, que la diferencia entre [t, d] y [s, z] radica en dos rasgos, los sonidos del segundo grupo tienen la característica de ser continuos y estridentes, por lo que podemos clasificar estos grupos como oclusivo (3.6) y fricativo (3.5) respectivamente [Riggle, 2012]. Ambos grupos, heredan los rasgos de la clasificación superior, es decir, de los sonidos corales, por lo que ya no es necesario repetir su escritura.

$$\begin{bmatrix} \text{cont} & + \\ \text{stri} & + \end{bmatrix} \quad (3.5)$$

$$\begin{bmatrix} \text{cont} & - \\ \text{stri} & - \end{bmatrix} \quad (3.6)$$

Dicho de otro modo, la operación de obtener la matriz padre (3.4) a partir de los sonidos [t, d, s, z] se llama unificación. La operación de extender los rasgos de las matrices hijas (3.5) y (3.6) por medio de la matriz padre (3.4) se llama herencia. Dichas operaciones son mecanismos muy comunes en los lenguajes formales para poder reutilizar y extender sus procedimientos escritos, y se han refinado formas de representación mucho más compactas que la representación gráfica de matrices.

Las ventajas principales del uso de matrices en modo gráfico son [Arís, 2015, Zajac, 1992]:

- Permiten concentrar gráficamente la información.
- Son claras, expresivas y rigurosas en su contenido y evolución.
- Pueden usarse en varios niveles de análisis.
- Los elementos pueden estar jerarquizados o en paralelo.
- Son consistentes al momento de establecer procesos. Un proceso que reciba una matriz tendrá como resultado una nueva matriz, esto se llama propiedad de cerradura.

#### 3.2.4. Ecuaciones

Una última representación gráfica-formal usada con gran frecuencia en lingüística es lo que llamaremos en esta tesis como *ecuación simbólica*.

Una ecuación simbólica es una representación que permite ver los elementos involucrados en un fenómeno lingüístico que, al integrarse, dan lugar a un proceso que derivará en una estructura final visible en la lengua.

Lo más destacable desde el punto de vista de este trabajo es que estas ecuaciones hacen hincapié en los elementos que originan el fenómeno lingüístico, y es posible establecer un orden secuencial de éstas si es necesario. En forma general, una ecuación simbólica puede representar la gran mayoría de las transformaciones resultado de un fenómeno lingüístico, pero destaca su utilización en las gramáticas dependientes del contexto cuyo uso más frecuente son las reglas fonológicas. Una regla fonológica tiene la siguiente forma general:

$$A \rightarrow B/\alpha\_ \beta \quad (3.7)$$

Lo cual en una abstracción matemática sería:

$$\alpha A \beta \rightarrow \alpha B \beta \quad (3.8)$$

Como podemos ver, la única diferencia entre (3.8) y (3.7) es que (3.7) aprovecha la propiedad distributiva para hacer una lectura más sencilla y más efectiva para propósitos lingüísticos.

Este tipo de reglas fueron introducidas por Chomsky y Halle en 1968 como una forma de caracterizar procesos cognitivos visibles en la fonología de la lengua:

Los procesos que describen las reglas fonológicas son considerados por Chomsky como procesos cognitivos. Efectivamente, Chomsky considera que estas operaciones se producen en el cerebro del hablante competente. La hipótesis de Chomsky es que el cerebro tendrá almacenados los fonemas y las reglas fonológicas y los alófonos se producirán por medio de la aplicación de las reglas en determinados contextos muy bien especificados [Arís, 2015]

### 3. ANÁLISIS DE LENGUAJES FORMALES

---

Las ecuaciones han sufrido cambios y se ha refinado su escritura, a continuación presentamos un ejemplo sencillo: En la lengua Kongo del sur<sup>2</sup> los sonidos [t, s, z] tienen alófonos que son respectivamente: [t̪, ʃ, ʒ].

Kongo del sur	Español
Tobola	perforar
t̪ina	cortar
kesoka	ser cortado
nkofi	león
zenga	cortar
ʒima	extender
kasu	demacración
t̪iba	plátano
nselele	termita
lolongi	lavar
zevo	entonces
aʒimola	limosnas
nzwetu	nuestra casa
kunezulu	al cielo
tanu	cinco

Tabla 3.1: Tabla con datos del Kongo del sur para la observación de los sonidos [t, s, z, t̪, ʃ, ʒ].

Los sonidos [t̪, ʃ, ʒ] solo aparecen antes de la vocal [i], por lo que podemos elaborar la ecuación (3.9).

$$\begin{bmatrix} t \\ s \\ z \end{bmatrix} \rightarrow \begin{bmatrix} t̪ \\ ʃ \\ ʒ \end{bmatrix} / - [i] \quad (3.9)$$

---

<sup>2</sup>Este ejemplo es tomado de notas de clase de fonología, en esas notas se identificó la lengua como Kongo del sur. Al realizar una búsqueda para validar el ejemplo, parece ser que se trata de la lengua kikongo[Bentley, 1887].

Como puede verse, esta ecuación contiene cada uno de los elementos que cambian y el contexto. Los estudios lingüísticos han refinado esta representación ya que podemos imaginar la existencia de ecuaciones donde será necesario el uso de matrices realmente grandes debido a sus elementos de entrada/salida o bien sus contextos. Lo anterior ha motivado la optimización de los elementos de la ecuación<sup>3</sup>, como se ve en (3.10).

$$\begin{bmatrix} +\text{cons} \\ -\text{reson} \\ +\text{ant} \end{bmatrix} \rightarrow \begin{bmatrix} -\text{ant} \end{bmatrix} / - \begin{bmatrix} V_{C. \text{ Ant.}} \end{bmatrix} \quad (3.10)$$

Como se puede ver, las ecuaciones simbólicas son una gran herramienta para representar cómo un elemento lingüístico es afectado por un contexto. Además los elementos que forman las reglas pueden ser de cualquier naturaleza, aunque los más comunes sean matrices como las vistas en la sección 3.2.3.

No debemos pasar por alto, que estas ecuaciones son de naturaleza procedimental y cuyo origen es matemático<sup>4</sup>, por lo que es muy fácil su reescritura en otros lenguajes formales.

### 3.3. Las matemáticas como forma de representación lingüística

Para discutir este apartado hay que partir de un hecho: el hombre no puede enunciar leyes universales infalibles. Las matemáticas son una ciencia formal

---

<sup>3</sup>Nótese que la ecuación en sí no es optimizada pues conserva su estructura, son las matrices las que se optimizan por medio de las operaciones de herencia y unificación explicadas en la sección 3.2.3.

<sup>4</sup>En este trabajo deducimos un origen matemático debido a la fuerte relación presentada con la representación de reglas de los lenguajes formales. A pesar de ello, también podríamos sugerir una influencia de la ciencia química, donde se han usado ecuaciones muy similares y con los mismos propósitos desde el siglo XVII[Crosland, 1959].

que basada en axiomas permite llegar a conclusiones que pretenden ser verdaderas. La razón principal de que las matemáticas sean hoy en día tan importantes universalmente es que permiten modelar los eventos del mundo natural; estos modelos cuando son exactos permiten predecir los hechos de la naturaleza y muchas veces, cuando se justifican las condiciones necesarias para que el modelo sea verdadero, este puede ser referido como *ley*.

Sin embargo, aun dentro de las matemáticas es posible modelar un evento de diferentes maneras, y estos modelos pueden ser más o menos complejos dependiendo del *objetivo científico* que se persiga, así pueden guardar relaciones entre sí sin llegar a ser idénticos unos con otros.

Cada cierto tiempo, las ciencias que usan las matemáticas se ven en la necesidad de expandir los alcances de las mismas para que cumplan con las necesidades de modelado y representación que requieren. Tan es así que hoy en día se reconocen sesenta y tres clasificaciones primarias en las matemáticas según su tema de empleo[(MR) and (Zbl), 2010]. Estas clasificaciones van desde clasificaciones algebraicas hasta otras más especializadas como geofísica o astronomía. Cada clasificación tiene a su vez, otras subcategorías, algunas de las cuales aparecen en varias de las clasificaciones primarias.

La expansión de los métodos matemáticos son un proceso lento que requiere, al menos, que muchos científicos intercambien sus ideas para llegar a la creación de una nueva rama de las matemáticas, tal fue el caso del cálculo infinitesimal (creado por Leibniz y Newton) y la teoría de la computación (Church y Turing).

Los métodos formales de la lingüística.

En lingüística, los métodos formales son basados en matemáticas y se pueden clasificar a grandes rasgos en: gramáticas formales, matrices, álgebra, teoría de conjuntos, lógica, funciones y sistemas deductivos [Arís, 2015]. Todos estos métodos han ayudado a la lingüística principalmente desde la segunda mitad del siglo XX, a pesar de ello, podemos observar dos problemas concretos: la diversidad de representaciones y una *escisión* entre las matemáticas y la lingüística.

El primer problema, la diversidad de representaciones, puede verse al compa-

### 3.3. Las matemáticas como forma de representación lingüística

---

rar a la lingüística con otras ciencias. En física, por ejemplo, el lenguaje matemático es constante, se ha usado continuamente desde el siglo XVIII y fue Leonhard Euler un personaje importante para que esto sucediera[Dunham, 1999]. Desde entonces, las representaciones matemáticas en física son un continuo, los símbolos de representación no cambian y si es necesaria una representación nueva para un elemento, siempre se sigue el mismo principio: una notación reducida, donde cada símbolo codifica gran cantidad de información y cuyo uso es lo más riguroso posible.

En lingüística sucede algo un poco diferente. Se trata de tomar un modelo formal que pueda ser utilizado en la descripción de un fenómeno lingüístico. La elección del modelo se hace por similitud, lo cual trae como consecuencia directa tener que explicar textualmente la notación, además se pretende su utilización en numerosos niveles de análisis. Cuando esta compatibilidad no es posible, se hacen adaptaciones mínimas que se alejan con grandes reservas del modelo original, de tal manera que se elimina el rigor presente en la física y la representación se convierte en un mal necesario para lograr una formalización del fenómeno. Es así como el continuo matemático en lingüística no es una amalgama, sino una ensalada.

Elección de modelos formales en lingüística.

El segundo problema, la *escisión* entre las matemáticas y la lingüística, es igualmente visible. Como ya se dijo, la notación matemática es de uso riguroso, no admite por lo general gradientes, la lingüística no solo los admite, sino que los requiere.

Por un lado, la lingüística usa las matemáticas para validarse, tal es el caso del uso de la estadística en áreas como la sociolingüística. Solias Arís lo explica de la siguiente manera:

La sociolingüística utiliza las herramientas estadísticas como recurso matemático que le ayuda a delimitar un determinado uso social y una determinada tendencia de uso. Con los resultados de un análisis cuantitativo y cualitativo del uso social de una lengua no solo se

pueden conocer mejor las variedades de esa lengua y sus relaciones, sino que también se pueden hipotetizar tendencias de cambio lingüístico, determinadas por esa variabilidad lingüística introducida por nuevos usos lingüísticos.[Arís, 2015]

Hasta aquí todo parecería adecuado, sin embargo, es necesario pensar que la descripción de un fenómeno varía de forma considerable de una lengua a otra. Esto también pasa dentro de las matemáticas: Sin pretender explicar a detalle, existen en matemáticas tres transformadas (Laplace, Fourier y Z) usadas para modelar un fenómeno que existe en el dominio del tiempo, pasándolo al dominio de la frecuencia. Cada una de las transformadas tiene casos particulares de especial utilidad, e incluso se encontrará que las tres transformadas tienen el mismo resultado para algún valor de la frecuencia, aun así, no son lo mismo.

Esta misma situación es la norma en las lenguas naturales, cada lengua humana es un sistema que toma la realidad y la representa de una manera particular. Es por eso que las lenguas no son iguales así los sistemas coincidan en algunos valores. En ese sentido, las matemáticas y las lenguas naturales tienen algo en común.

Diferencia entre la lógica proposicional y las lenguas naturales.

La lógica proposicional, como rama de las matemáticas, es ampliamente usada en lingüística debido a su capacidad de tomar las lenguas humanas como un objeto de la realidad y representarlo para explicar así su comportamiento. Esto no significa que exista una relación unívoca entre los elementos de la lógica y las lenguas naturales, de hecho parece que no existe esa relación. La lógica y una lengua natural tratan de dar cuenta de la misma realidad y por lo tanto pueden estar estrechamente relacionados y, sin embargo, igual que las transformadas integrales, no son lo mismo a nivel de sus elementos. Este es el motivo de que el conectivo lógico de negación ( $\neg$ ) y la negación en español son diferentes, lo mismo sucede entre la conjunción lógica ( $\wedge$ ) y la conjunción básica del español (y).



Resumiendo lo anterior, cuando las matemáticas y las lenguas naturales son usadas para describir una realidad, resultan muy parecidas, antes bien, si se usa una para describir a la otra, es necesario considerar que habrá igualdades pero en realidad se deben considerar reglas de correspondencia complejas que permitan pasar de las matemáticas a las lenguas naturales y viceversa. Esta última idea nos puede llevar a considerar la existencia de una rama de las matemáticas dedicada exclusivamente a describir las lenguas naturales.

Esta rama de las matemáticas aun no establecida deberá contener un conjunto de reglas de transformación que permitan la equivalencia entre lenguaje matemático y lengua natural. Con las matemáticas actuales, se busca la prescripción sobre el objeto de estudio de una ciencia dentro de una realidad, la lingüística busca la descripción de los fenómenos presentes en su objeto de estudio. Efectivamente esas dos ideas científicas son incompatibles entre sí, en el estudio lingüístico se ha entendido en mucho menos tiempo y con mucha más precisión que no es posible enunciar leyes universales infalibles.

Los problemas que implica esta idea rebasan el tema considerado en este trabajo de investigación, aunque sí es de nuestra consideración mostrar si es posible encontrar estas reglas, aunque sea con un ejemplo muy específico que por supuesto se supone cierto sin descartar la posibilidad de error debido a su acotación.

#### **3.3.1. Obtención de regla de correspondencia entre la negación lógica y la concordancia negativa en el español**

En lengua española existe un fenómeno en el que parece, a primera vista, que hay en algunas frases la doble negación, tal y como sucede en (6).

(6) No vino nadie

Dentro de la lógica proposicional, la oración anterior se denota como una negación que opera sobre otra negación formando una proposición positiva. En español este fenómeno se explica con una concordancia gramatical, donde la palabra *nadie* esta concordando con la negación inicial. Esto es cierto lingüísticamente, es por eso que la negación lógica se considera diferente a la negación en español en su realización.

Sin embargo, supongamos que deseamos encontrar una regla de correspondencia entre estas dos negaciones que nos permita *traducir* la negación lógica y la negación del español.

#### 3.3.1.1. Sobre las palabras negativas

En el texto de Camus Bergareche titulado *Negación doble y negación simple en español moderno* se habla de *palabras negativas* (PNs) y explica los tratamientos previos que se han dado a palabras como *nada*, *nadie*, *nunca*, *jamás* y *tampoco*. Se explica que algunas de las palabras tienen históricamente un significado positivo y que actualmente en español su significado en uso preverbal es diferente al uso postverbal.

En el propósito de corroborar lo observado por Camus Bergareche y saber cual es la asociación general que los hablantes le dan a estas PNs se le pidió a un grupo de personas<sup>5</sup> que colocaran las palabras que se les mencionara en una de las dos categorías: las palabras que usan para decir *si* y las palabras que usan para decir *no*.

Es notable destacar que las palabras *algo* o *nadie* les causan dudas a los que ubican a la primera en ambas categorías pero se inclinan a ponerla en la columna *si*; mientras que la segunda palabra es ubicada en muchas ocasiones

---

<sup>5</sup>Este análisis se realizó con tres informantes, dos de ellos alumnos de la Escuela Nacional de Antropología e Historia (ENAH), ninguno estudiante de lingüística. El tercer informante tiene un nivel de estudios inferior, se trata de un adulto con secundaria terminada. Los tres han vivido más de 10 años en la Ciudad de México y solo el último vivió anteriormente fuera de esta zona metropolitana.

### 3.3. Las matemáticas como forma de representación lingüística

Si	No
Alguien	Nadie
Algo	Nada
Alguno	Ninguno
	Nunca
	Jamás
	Tampoco

Tabla 3.2: Tabla que contiene la asociación general que los hablantes tienen entre palabras positivas o negativas.

como ofensiva y no puramente negativa; esto no sucede con *nada*, el motivo es que usan *nadie* para restarle importancia a una persona.

Así es como, por ejemplo, (7a) que documenta Camus Bergareche es para los entrevistados una oración semigramatical porque en ese contexto<sup>6</sup> (7b) es más natural.

(7) a. ?? Dudo que tenga nada

b. Dudo que tenga algo

#### 3.3.1.2. La concordancia negativa

La concordancia negativa es la explicación gramatical a la presencia de lo que a primera vista es una negación doble en una oración, según [Jespersen, 1917] la estructura característica de estas oraciones sería:

(8) [<sub>n</sub>... \* (neg) ... V ... PN ...]

Podemos ver que la oración (7a) presentada por Camus Bergareche como un primer caso de oraciones de este tipo podría no serlo. El verbo *dudar* es

<sup>6</sup>El contexto planteado fue el siguiente: «Tú y un amigo están viendo a la distancia a un tercer amigo mutuo preocupado; entonces dice el amigo que está contigo “Dudo que tenga nada”».

introducido como un Inductor Negativo (IN)<sup>7</sup> citado por Bosque y el indefinido *nada* es un Término de Polaridad Negativa (TPN)<sup>8</sup>. Sin embargo este IN no exige la concordancia, pues como podemos ver, los ejemplos de (9) funcionan en el mismo contexto<sup>9</sup>, la única diferencia es que (9c) y (9d) muestran más seguridad para nuestros entrevistados, y las palabras postverbiales no son TPN<sup>10</sup>.

- (9) a. Dudo que valga gran cosa
- b. Dudo que valga mucho
- c. No vale gran cosa
- d. No vale mucho

Ante esta discrepancia, resulta interesante verificar los contextos de los que Camus Bergareche dice:

En estos contextos no existe una negación que provoque la palabra negativa por lo que difícilmente podemos hablar de negación doble. Es mas, ni siquiera, podemos hablar de negación ya que la interpretación de los indefinidos de (10) no puede ser negativa. Notemos que en todos los casos podemos parafrasearlos por un indefinido positivo: *algo, alguien, cualquiera*. [Bergareche, 1992]

- (10) a. (Comparación) Pedro es más alto que *nadie*
- b. \* (Ordinales) Es la última vez que te digo *nada*

---

<sup>7</sup>Un IN es un elemento capaz de inducir la aparición de un TPN

<sup>8</sup>Los TPNs son aquellas construcciones cuyo funcionamiento está condicionado por la presencia previa de una negación.[Bergareche, 1992]

<sup>9</sup>El contexto planteado fue el siguiente: «Tú y un amigo encuentran un objeto tirado en la calle, el quiere decirte que es un objeto sin valor».

<sup>10</sup>Aunque al parecer Bosque si propone *gran cosa* como un TPN.

- c. ?? (Factivos) Me indigna que venga *nadie*
- d. \* (Interrogación) ¿Cuándo me has regalado *nada*?
- e. ?? (Cuantificadores) Poca gente entendería *nada* de lo que dices

- (11) a. Es la última vez que te digo algo
- b. Me indigna que venga alguien
- c. ¿Cuándo me has regalado algo?

Los datos obtenidos al preguntar sobre las oraciones de (10) nos dejaron ver que (10a) es negativo porque se está diciendo que «no hay otra persona más alta que Pedro»; (10b) y (10d) son agramaticales ya que las expresiones que los entrevistados identifican como gramaticales son (11a) y (11c); (10c) es semigramatical ya que es (11b) la oración identificada como correcta. Aún así en caso de que fuera usada significaría que el individuo «estaba esperando a unas personas pero no llegaron». Finalmente (10e) es un caso similar al anterior donde se cambiaría *nada* por *algo* y el emisor estaría diciendo que «no entiende lo que dices».

Es por esto que la afirmación hecha de Camus Bergareche no concuerda con estos resultados. Sin embargo aquí podemos ver una explicación de las dudas de los hablantes presentadas en la sección 3.3.1.1 sobre el carácter positivo de palabras como *algo* y comprobamos que una oración como las presentadas en (10a) y (11) tienen una semántica negativa, es decir una negación semántica simple respaldada sintácticamente por un TPN positivo.

La negación en las lenguas naturales es algo más que un operador sintáctico<sup>11</sup> como sucede en la lógica proposicional, es por eso que las oraciones de (12)

---

<sup>11</sup>Con esta expresión no se pretende negar el reflejo de la negación semántica de las lenguas naturales en la sintaxis, solamente dejó ver que la correspondencia entre la sintaxis y la semántica no es unívoca.

tienen a nivel sintáctico una concordancia que simplifica los valores negativos semánticos.

- (12) a. No vinieron algunos hombres  
b. No vi a algunos amigos  
c. No vino ningún hombre  
d. No vi a ningún amigo

El paradigma presente en (12) es analizado por Camus Bergareche en términos de cuantificadores lógicos debido a que:

Postular entonces esta concordancia en sentido amplio [...] se trata en definitiva, de determinar qué valor cuantificador aportan las palabras que venimos estudiando.[Bergareche, 1992]

De esta manera explica que (12a) y (12b) refieren a una negación de predicado que no afecta al cuantificador y en (12c) y (12d) el cuantificador es el negado. A continuación se tomará esta perspectiva para mostrar un punto de vista diferente.

Desde el punto de vista de la lógica (12a) y (12b) se representarían de la siguiente manera:

$$(12a) = (\exists x) \neg P(x)$$

$$(12b) = (\exists y) \neg P(y)$$

Ahora por equivalencia de cuantificadores podemos obtener:

$$(\exists x) \neg P(x) \equiv \neg(\forall x) P(x)$$

$$(\exists y) \neg Q(y) \equiv \neg(\forall y) P(y)$$

### 3.3. Las matemáticas como forma de representación lingüística

---

Lo anterior significaría que podemos negar el cuantificador o el predicado. Podemos confirmar que esto es cierto:

(13) a. No todos los hombres vinieron

b. Hay unos hombres que no vinieron

(14) a. No a todos mis amigos vi

b. Hay (al menos) un amigo que no vi

Todas las oraciones de (13) y (14) son gramaticales, confirmamos que efectivamente estas oraciones pueden verse de forma lógica de dos maneras: con una negación en el cuantificador o con una negación en el predicado. Si se aplica la misma equivalencia a (12c) y (12d):

$$(12c) = \neg(\exists x)P(x)$$

$$(12d) = \neg(\exists y)P(y)$$

$$\neg(\exists x)P(x) \equiv (\forall x)\neg P(x)$$

$$\neg(\exists y)Q(y) \equiv (\forall y)\neg P(y)$$

(15) a. Todos los hombres no vinieron

b. No hay un hombre que haya venido

(16) a. A todos mis amigos no los he visto

b. No hay un amigo al que haya visto

Al preguntar por las oraciones anteriores descubrimos que (16a) era identificada por los hablantes como una oración cuyos elementos no estaban ordenados de la forma menos marcada, por lo que «No he visto a todos mis amigos» es la forma menos marcada, sin embargo no hubo problema de interpretación por parte de los entrevistados.

Con los datos anteriores es posible decir que cada elemento que forma parte de la concordancia negativa esta asociado a una proposición. Por lo tanto una oración con concordancia negativa estaría representada de la siguiente manera:

$$\neg P(x) \wedge Q(x)$$

O bien:

$$P(x) \wedge \neg Q(x)$$

Sin embargo solo una se interpreta por lo que:

$$NS(O(x)) = (\neg P(x) \wedge Q(x)) \vee (P(x) \wedge \neg Q(x)) \quad (3.11)$$

La expresión (3.11) es una aproximación de la Negación Semántica en oraciones con concordancia negativa en español. Debemos hacer énfasis en que aunque los significados semánticos son potencialmente posibles en la lengua, no siempre son vistos como posiblemente sucedería con un verbo impersonal en una oración con concordancia negativa.

Ahora hay que probar el modelo presente en (3.11). Sea un contexto que contiene una oración con concordancia negativa:

(17) A: Ayer estaba esperando a Juan y a María

B: ¿Quién vino?

A: No vino nadie.



### 3.3. Las matemáticas como forma de representación lingüística

---

La conversación de (17) fue verificada con tres entrevistados, ninguno identificó anomalías, así que procedimos a cambiar la tercera oración de la siguiente manera:

- (18) a. No vino nadie.  
b. Ni Juan ni María vinieron  
c. Juan y María no vinieron

Los entrevistados coincidieron que (18a), (18b) y (18c) son equivalentes en este contexto, por lo que procedemos a su representación proposicional:

$$j = \text{Juan}$$

$$m = \text{Maria}$$

$$P(a, b) = a \wedge b$$

$$V(x) = \{x: x \text{ vino}\}$$

Comprobaremos que:

$$(\neg P(a, b) \wedge V(x)) \vee (P(a, b) \wedge \neg V(x))$$

Es importante mencionar:

$$\neg P(a, b) \equiv \neg(a \wedge b) \equiv (\neg a \wedge \neg b)$$

$$P(j, m) = \text{Juan y Maria}$$

$$\neg P(j, m) = \text{Ni Juan Ni Maria}$$

Gracias a la tabla de verdad podemos observar que cuando las dos funciones son positivas, la negación general resulta falsa; cuando una de las dos funciones es negativa entonces la negación general resulta verdadera, tal y como se

### 3. ANÁLISIS DE LENGUAJES FORMALES

$P(j, m)$	$V(x)$	$\neg P(j, m)$	$\neg P(j, m) \wedge V(x)$	$\neg V(x)$	$P(j, m) \wedge \neg V(x)$	$(\neg P(j, m) \wedge V(x)) \vee (P(j, m) \wedge \neg V(x))$
$T$	$T$	$F$	$F$	$F$	$F$	$F$
$T$	$F$	$F$	$F$	$T$	$T$	$T$
$F$	$T$	$T$	$T$	$F$	$F$	$T$
$F$	$F$	$T$	$F$	$T$	$F$	$F$

Tabla 3.3: Tabla de verdad de la expresión (3.11)  $NS(O(x))$

esperaría ya que confirma que el hablante interpreta solo una negación a la vez. Finalmente si interpretara las negaciones al mismo tiempo la negación general sería falsa nuevamente tal y como resulta cuando aplicamos la doble negación lógica, es decir:

$$\neg(\neg p) \equiv p$$

De esta forma, parece que el modelo (3.11) es correcto y responde a una regla de correspondencia particular entre la negación lógica y la negación con concordancia negativa en español. Si esto es verdad, podríamos haber encontrado una región de convergencia más grande entre estas dos negaciones, región que ya había sido descrita de forma cualitativa:

En español la concordancia negativa, que se manifiesta a través de palabras que son también indefinidos existenciales, ha de ser obligatoria en contextos en que interviene la negación e indica inequívocamente la negación interna de cuantificador. [Bergareche, 1992]

El modelo encontrado también abre la puerta a nuevas preguntas:

- ¿Cuál es la negación a la que el hablante da preferencia de interpretación?
- ¿Serán los verbos impersonales una excepción a este modelo?
- ¿Es la concordancia negativa una operación básica de la lengua o bien una operación de optimización de la misma?

### 3.3. Las matemáticas como forma de representación lingüística

---

Podemos inicialmente pensar que la preferencia de interpretación es la negación interna del cuantificador; que, como se dijo anteriormente, los significados semánticos son potencialmente posibles en la lengua aunque no siempre observables, esto además justificaría la presencia de una concordancia negativa en unos pocos dialectos del inglés debido a que en inglés no puede incluirse una negación en cualquier parte de la oración.

Todas estas preguntas pueden solucionarse realmente ampliando el corpus y ajustando el modelo por uno más avanzado, un modelo que tome describa la regla de correspondencia de negaciones de forma global. De momento se ha favorecido el argumento presentado en la sección 3.3: si bien es posible explicar la semántica en las lenguas naturales por medio de las matemáticas discretas, las operaciones presentes en ambas representaciones no son idénticas, es algo que debería darse por sentado.



---

Como se planteó en el capítulo anterior, existen elementos formales que permiten la representación del objeto de estudio de la lingüística. Estos elementos son usados ampliamente en situaciones específicas, como las planteadas anteriormente. Sería un error tratar de sustituirlos en su uso, sin embargo si es posible proponer otra forma de representarlos. El objetivo de esta propuesta es un aporte en la claridad y sencillez de su representación, además de ampliar los paradigmas de uso de una posible representación común.

En este momento, es necesario considerar que esta propuesta es una *representación* y no una *codificación*. Desde el punto de vista lingüístico, la diferencia es significativa: en la codificación encontramos elementos y relaciones predeterminados; en la representación encontramos elementos y relaciones no predeterminados. Como se verá durante el presente capítulo, el lenguaje propuesto no es un sistema X adecuado para los fenómenos lingüísticos desde el punto de vista de la teoría T, sino un sistema con un proceso de construcción y reglas de producción. [Ferreiro, 1998]

Al tener como propósito ser una representación de los fenómenos lingüísti-

cos, nuestro lenguaje puede poseer propiedades de los elementos formales vistos en el capítulo anterior y excluye otras. Tampoco se encuentra limitado a un sistema alternativo exclusivo para escribir unidades lingüísticas como fonemas o morfemas, por lo que no puede considerarse como un método de transcripción o cambio de sustancia, es por eso que elegimos llamarlo un lenguaje de representación, que implica la reconfiguración de los elementos formales originales para reconstruirlos con las reglas del lenguaje propuesto. [Zamudio, 2010]

A lo largo de todo este capítulo se explicará paso a paso la estructura léxica, sintáctica y semántica de un lenguaje formal que permita la descripción de una lengua y su posterior procesamiento. Todo este capítulo se basa en la lectura de *Recursive Functions of Symbolic Expressions and their Computation by Machine, Part I* y apoyada por numerosos libros sobre diversos dialectos de la familia de lenguajes formales LISP, dichos libros son mencionados en la bibliografía.

En la primera sección se describirán y ejemplificarán los elementos que componen el lenguaje, sus combinaciones sintácticas (que aquí denominaremos Arquitectura). Después se listarán los elementos predeterminados que restringen esas combinaciones para luego explicar la definición de variables y finalmente se darán ejemplos de uso del lenguaje.

### 4.1. Unidades fundamentales

Las expresiones usadas por este lenguaje son expresiones S (Ver sección 2.4) compuestas por objetos llamados átomos. Cualquier expresión S puede ser válida en dos diferentes contextos: como elemento de lectura y descripción, tal como pretende mostrar esta tesis, y como elemento de procesamiento automático del lenguaje.

En el contexto de lectura y descripción, el lenguaje requiere un ciclo iterativo constante que consiste en leer y evaluar la expresión tantas veces como sea necesario. La evaluación consiste en lo siguiente:

Definición de evaluación.

- La expresión debe ser traducida a sus términos estructurales mínimos. Esta traducción es un proceso mental que el lector hace y de su correcto seguimiento dependerá la reconstrucción del conocimiento científico.
- La interpretación semántica de la expresión en términos de la estructura mínima obtenida en el punto anterior. Esta interpretación por parte del lector le permitiría tener acceso al conocimiento lingüístico expresado por el investigador en la expresión.

La evaluación es un proceso en el que interviene en todo momento el significado. Al ser un lenguaje formal, el nivel es descriptivo donde la información es codificada en un predicado lógico (Ver sección 2.2.1). Como resultado del proceso de evaluación, se obtiene un determinado valor.

En el contexto de procesamiento automático del lenguaje, se necesitará un interprete que traducirá la expresión a expresiones  $S$  más sencillas. Para esto, primero verificará que cada elemento de la expresión puede pertenecer al lenguaje (un proceso de verificación de léxico), para luego definir la sintaxis que determinará la validez de la expresión que es finalmente interpretada y que siempre regresa el valor correspondiente.

Finalmente, es necesario decir que el interprete trabajará como un operador que reciba como argumento la expresión  $S$ ; es por esa razón que toda expresión del lenguaje está entre paréntesis.

### **4.1.1. Elementos de Descripción**

#### **4.1.1.1. Átomos**

Un átomo es una combinación de los símbolos de consonantes y vocales del Alfabeto Fonético Internacional (AFI) en conjunto con los símbolos suprasegmentales y diacríticos de los mismos. Los números también son átomos y la combinación de números y letras, pero esta combinación no puede iniciar con

#### 4. PROPUESTA

---

un dígito. Tampoco son átomos aquellas combinaciones que comienzan con símbolos ajenos al AFI.

Son ejemplos de átomos:

- (19) a. L5
- b. 2343
- c. hola
- d. kə'βΔhk<sup>h</sup>
- e. t̂z<sup>y</sup>fs

Son ejemplos de no átomos:

- (20) a. 54dg  
Inicia con un dígito
- b. #kasas  
Inicia con un modificador
- c. ola mundo  
Existe un espacio en medio

Al usar el lenguaje, distinguiremos varios tipos de átomos dependiendo del nivel de análisis lingüístico y el fenómeno que nos ocupe en ese momento, así tendremos átomos fonémicos, silábicos, morféminos, léxicos, etc. Para realizar un análisis desde el punto de vista cuantitativo, como puede suceder en el terreno fonético, se usan los átomos numéricos. Sin embargo, siempre tendremos a nuestra disposición algunos tipos de átomos [Arroyo Méndez and Marchesi, 2005]:

**Simbólico:** Puede ser simplemente llamado símbolo. Cuya característica principal es que nunca existirán dos símbolos iguales. Los símbolos sirven para



asociar valores y tienen un carácter estrictamente funcional para el sistema descriptivo de la lengua.

**Cadenas:** En caso de que se requiera en un átomo un elemento que no lo sea, por ejemplo que tenga un espacio en blanco, éste puede introducirse como cadena. Las cadenas siempre serán escritas entre comillas dobles (“ ”).

***t y nil:*** Estos dos son átomos simbólicos estándar. Sirven para distinguir el final de una lista, como resultado de los operadores lógicos y en la definición de operadores, entre otras cosas porque tienen la característica de *siempre evaluarse a si mismos*.

### 4.1.2. Arquitectura de Información

La arquitectura básica del lenguaje se lleva a cabo en dos secciones diferentes. La primera es una estructura de datos y la otra un área de trabajo donde se llevan a cabo la mayoría de las relaciones entre los elementos que son de interés lingüístico. Las estructuras de datos que utiliza el lenguaje son dos: el diccionario y los pares de celda. En cuanto al área de trabajo, ésta contiene los pares de celda. El diccionario estaría fuera del área de trabajo. La figura 4.1 muestra un ejemplo de como se vería esta arquitectura.

En realidad, conocer la arquitectura del lenguaje podría no ser indispensable para escribir en él, sin embargo, cuando el objetivo es recuperar en su totalidad el conocimiento lingüístico o escribir un interprete para este lenguaje, si resulta fundamental entender esta organización arquitectónica.

A continuación explicaremos esta arquitectura y sus elementos, posteriormente se explicará con detalle el uso y extensión de la misma.

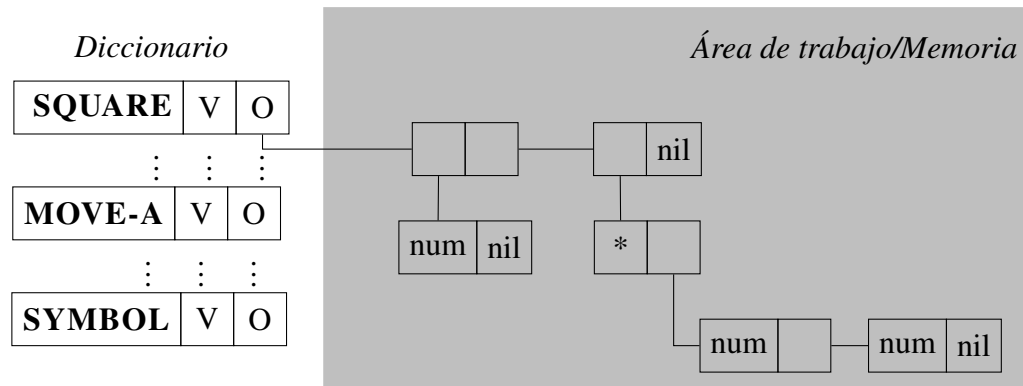


Figura 4.1: Diagrama de la arquitectura. El diccionario a la izquierda y el área de trabajo a la derecha.

#### 4.1.2.1. Diccionario

Existen solo dos estructuras de datos necesarias y suficientes para el funcionamiento del lenguaje. La primera estructura es también una de las partes de mayor extensión de la arquitectura y puede considerarse como un arreglo o una lista de símbolos, desde ahora denominaremos *diccionario* a esta estructura, en la cual cada símbolo tiene asociados dos apuntadores *V* y *O* (Figura 4.2).

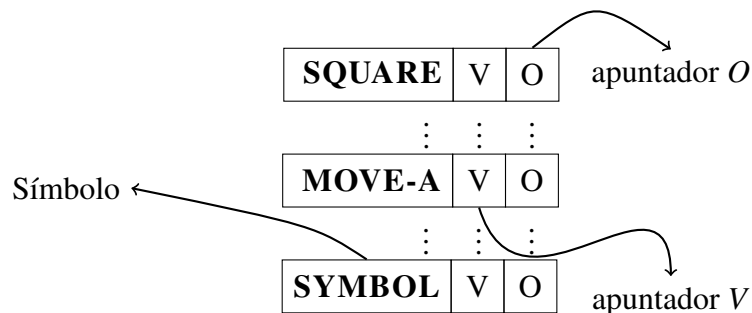


Figura 4.2: Diagrama del diccionario obtenido de la Figura 4.1.

Definición de  
apuntador.

Un apuntador es un elemento abstracto que permite distinguir el rol que tendrá el símbolo en la organización de la información. El primer apuntador *V* aso-

cia un valor al símbolo y el segundo apuntador  $O$  asocia una operación al símbolo. En ambos casos, las asociaciones se hacen a los átomos que se encuentran en el área de trabajo, lugar donde se encuentra y organiza toda la información conocida.

Como ya fue mencionado, los símbolos son átomos con carácter funcional. Funciones de un símbolo. Un símbolo tendrá tres funciones diferentes dentro del lenguaje:

1. Evalúa al valor que tiene el apuntador  $V$ ,
2. Si es el primer elemento de una lista, evalúa el valor del apuntador  $O$ ,
3. Si el símbolo es el primer argumento de la operación **SETQ** (Ver sección 4.2) el símbolo es usado pero no se evalúa ningún apuntador.

#### 4.1.2.2. Par de celda

Un par de celda es la segunda estructura de datos necesaria en nuestro lenguaje, de ella derivan todas las operaciones y es la forma básica de organización de información de la lengua que se busca describir. Un par de celda es, como su nombre indica, un par de datos, definido de la siguiente manera:

(fst . snd)

Donde fst funcionará como discriminante para una búsqueda eficiente y snd es la información asociada al discriminante. Para ejemplificar esta estructura pensemos en clasificar un grupo de elementos léxicos según su Part of Speech (POS):

La tabla, fila por fila, se representaría con pares de celda de la siguiente manera:

(a . preposición)	(pésimo . adjetivo)
(tesis . nombre)	(cantábamos . verbo)
(mí . pronombre)	(primeras . adjetivo)

#### 4. PROPUESTA

---

Elemento léxico	POS
a	preposición
pésimo	adjetivo
tesis	nombre
cantábamos	verbo
mí	pronombre
primeras	adjetivo
cortapapeles	nombre
Pedro	nombre
ser	verbo
que	pronombre

Tabla 4.1: Tabla con elementos léxicos clasificados según su POS [Universitat Politècnica de Catalunya, 2016].

(cortapapeles . nombre)      (Pedro . nombre)  
(ser . verbo)                      (que . pronombre)

Como puede verse, el elemento léxico es el fst y su clasificación de POS es el snd.

Debe ponerse especial atención al espacio que hay entre los elementos y el punto, esta notación permite diferenciar un par de celda con la notación lingüística estándar de la sílaba que algún análisis podría requerir. A pesar de que un par de celda por si solo sirve para mantener orden entre pares de datos simples, lo más frecuente es anidar un par con otro para construir estructuras más complejas como árboles. Los pares se anidan de la siguiente manera:

(a . (b . c))

Debe distinguirse que la anidación de pares sucede en el lugar del snd debido a que lo que aumenta es la información asociada al discriminante, esto no significa que no pueda existir un discriminante complejo, es decir, compuesto de otros pares de celda anidados.

Agreguemos más información a la tabla 4.1, ahora con el elemento léxico, el POS y su etiqueta del Expert Advisory Group on Language Engineering Standards (EAGLES) correspondiente, tal como se muestra la tabla 4.2.

Elemento léxico	POS	EAGLES
a	preposición	SPS00
pésimo	adjetivo	AQSMP0
tesis	nombre	NCFN000
cantábamos	verbo	VMII1P0
mí	pronombre	PP1CSO00
primeras	adjetivo	AO0FP0
cortapapeles	nombre	NCMN000
Pedro	nombre	NP000P0
ser	verbo	VSN0000
que	pronombre	PR0CN000

Tabla 4.2: Tabla con elementos léxicos clasificados según su POS y su etiqueta EAGLES [Universitat Politècnica de Catalunya, 2016].

De esta forma, obtendríamos los elementos de la lista concatenados de la siguiente manera:

(a . (preposición . SPS00))	(pésimo . (adjetivo . AQSMP0))
(tesis . (nombre . NCFN000))	(cantábamos . (verbo . VMII1P0))
(mí . (pronombre . PP1CSO00))	(primeras . (adjetivo . AO0FP0))
(cortapapeles . (nombre . NCMN000))	(Pedro . (nombre . NP000P0))
(ser . (verbo . VSN0000))	(que . (pronombre . PR0CN000))

Podemos considerar poco práctico terminar una anidación con un valor diferente a nil. El símbolo nil puede ayudar a identificar el final de la concatenación de los datos y resulta importante para la creación de estructuras más complejas. Si el elemento snd del último par de celda anidado es nil entonces la estructura puede representarse en forma de lista, es decir:

$$(a . (b . (c . nil))) = (a b c)$$

#### 4.1.2.3. Listas

La lista es un tipo de dato compuesto y, a diferencia de los átomos, no es fundamental. Una lista puede ser una secuencia de átomos separados por un espacio y encerrados entre paréntesis, incluyendo la posibilidad de que una lista contenga una sublista que cumpla con las mismas características.

Definimos como *término* de una lista a un elemento de la misma, ya sea o no un átomo, por lo que una lista se define como:

$$(\text{término}_1 \text{ término}_2 \dots \text{término}_k)$$

donde  $k$  es el número de elementos de la lista

La tabla 4.3 son ejemplos de listas, se especifican los términos uno a uno. En el ejemplo de lista con dos términos se puede ver que esos términos son a su vez listas de dos términos, en este segundo nivel los términos si son átomos.

Lista	Número de términos	Términos
(esto es una lista)	4	esto, es, una, lista
((esto es) (una lista))	2	(esto es), (una lista)
(lista)	1	lista

Tabla 4.3: Ejemplos de listas

Si volvemos a la tabla 4.2 podemos ver que cada fila podría considerarse como una lista de elementos y la correspondencia sería como en la tabla 4.4.

## 4.2. Operadores

Un operador es un símbolo que en el diccionario tiene el apuntador  $O$  asociado a una estructura de pares de celda. La figura 4.3 muestra un ejemplo básico de un operador.

Forma en pares de celda	Forma de lista
(a . (preposición . (SPS00 . nil)))	(a preposición SPS00)
(pésimo . (adjetivo . (AQSMP0 . nil)))	(pésimo adjetivo AQSMP0)
(tesis . (nombre . (NCFN000 . nil)))	(tesis nombre NCFN000)
(cantábamos . (verbo . (VMII1P0 . nil)))	(cantábamos verbo VMII1P0)
(mí . (pronombre . (PP1CSO00 . nil)))	(mí pronombre PP1CSO00)
(primeras . (adjetivo . (AO0FP0 . nil)))	(primeras adjetivo AO0FP0)
(cortapapeles . (nombre . (NCMN000 . nil)))	(cortapapeles nombre NCMN000)
(Pedro . (nombre . (NP000P0 . nil)))	(Pedro nombre NP000P0)
(ser . (verbo . (VSN0000 . nil)))	(ser verbo VSN0000)
(que . (pronombre . (PR0CN000 . nil)))	(que pronombre PR0CN000)

Tabla 4.4: Tabla que muestra la correspondencia entre la notación de par de celda y la notación tipo lista[Universitat Politècnica de Catalunya, 2016].

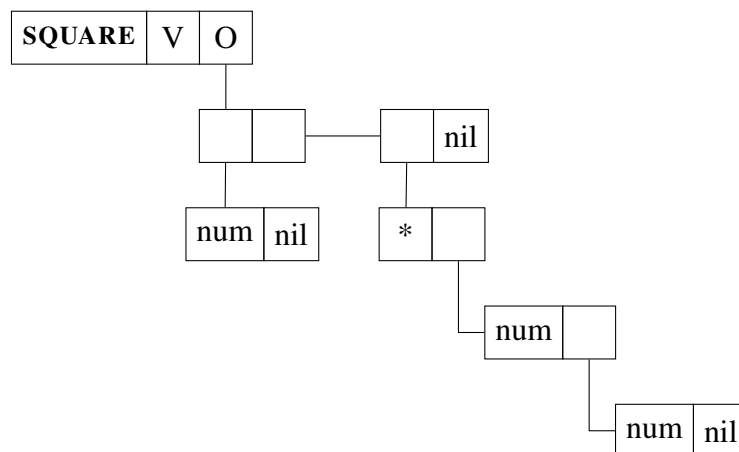


Figura 4.3: Ejemplo de un operador dibujado en el diagrama de la arquitectura

El operador **SQUARE** de la figura 4.3 recibe como argumento un número y lo multiplica por sí mismo. Matemáticamente, es la representación de la segunda potencia de *num*, comúnmente escrito como  $num^2$ . La figura 4.4 muestra la estructura arbórea de la operación aritmética que es representada por **SQUARE**. Este tipo de estructuras arbóreas pueden ser asociadas fácilmente al lenguaje y se recorren en preorden (como en gran parte de las teorías lingüísticas que

#### 4. PROPUESTA

---

se apoyan en árboles) por lo tanto, la notación de escritura es prefija (Ver sección 2.2.2).

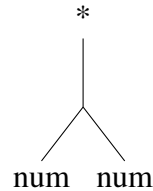


Figura 4.4: Estructura arbórea utilizada en la figura 4.3

Debido a su escritura prefija, el árbol de la figura 4.4 se escribiría en notación de pares de puntos de la siguiente manera:

```
1 | (* . (num . num) )
```

Para abreviar la notación, podemos modificar esos pares de celda para que puedan ser representados en forma de lista, de este modo obtenemos el diagrama de la figura 4.5 y que se escribe:

```
1 | (* num num)
```

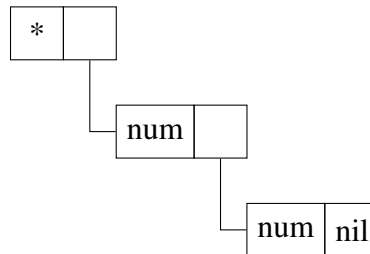


Figura 4.5: Diagrama de la multiplicación de dos números en forma de lista

Hasta este momento distinguimos que podemos multiplicar dos números *num*, lo que podemos usar asumiendo que ambos números tienen un mismo valor o bien, dos valores diferentes:



```

1  (* 3 2)
2  6
3  (* 2 2)
4  4

```

Sin embargo, en este ejemplo deseamos hacer  $num^2$  por lo que es necesario que *num* pueda tomar únicamente un valor y no dos valores cualquiera. Con ese propósito debemos crear una lista de argumentos que contiene un solo término: *num*. Dicha lista estará asociada nuevamente a nuestra primera lista, tal y como sucede en la figura 4.6.

```

1  ((num) . ((* num num) . nil))

```

Si explicamos con un poco más de detalle el significado a nivel de nuestro lenguaje, la representación anterior significa que al recibir el discriminante (*num*) la información que se obtiene es la de la multiplicación de ese número por sí mismo.

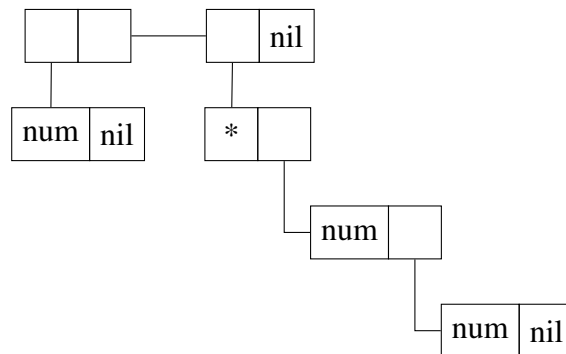


Figura 4.6: Diagrama de la multiplicación y su lista de argumentos, unidos en una sola lista

Finalmente, si asociamos esa última lista al símbolo **SQUARE** por medio del apuntador *O*, obtendremos nuestra figura 4.3. Escrito en el lenguaje, la definición del operador es la siguiente:

## 4. PROPUESTA

---

```
1 (DOP SQUARE (num)
2   "SQUARE recibe un numero y lo multiplica por si mismo"
3   (* num num) )
```

Un ejemplo de uso es el siguiente:

```
1 (SQUARE 5)
2 25
```

Hay varios detalles que deben mencionarse a pesar de la sencillez del ejemplo, el primer detalle es el uso del operador **DOP** para definir **SQUARE**. Este primer operador es parte nativa del lenguaje, que sirve para asociar un elemento al apuntador *O* de un símbolo.

En la línea 2 de la definición del operador tenemos un comentario que describe el operador que estamos escribiendo, este comentario es optativo pero útil porque hará que la descripción lingüística sea auto-documentada. Es decir, para entender lo escrito en el lenguaje solo basta con conocer la estructura básica de este documento porque cualquier particularidad lingüística o de cualquier otra naturaleza, sin importar la complejidad de la misma, puede ser documentada en el lugar preciso.

Para cerrar los detalles sobre esta función, cabe mencionar que existen operadores fundamentales para el funcionamiento del lenguaje, estos símbolos se pueden describir como *palabras reservadas* escritas en mayúsculas y siempre al inicio de una lista. A continuación se presenta la lista de ellos.

### 4.2.1. Operadores fundamentales

En esta sección definiremos una lista de operadores básicos que sirven para construir operadores más complejos que facilitan la estandarización de las descripciones[Burgos Cárdenas, 1997].

### **Operador: SETQ**

Número de argumentos: 2

Argumentos: Un símbolo y un término que será valor del símbolo. Puede recibir más de un par de argumentos.

Resultado: El valor del último argumento

Ejemplo 1: (SETQ a 3) → 3

Ejemplo 2: (SETQ a 3 b 0 c 5) → 5

Notas adicionales: El primer argumento es un símbolo que no es evaluado. El segundo argumento es un valor. SETQ realiza la asociación entre el símbolo y el valor por medio del apuntador V. Es posible que se reciba más de un par de argumentos, como en el ejemplo 2; en este ejemplo a=3 b=0 y c=5.

### **Operador: QUOTE**

Número de argumentos: 1

Argumentos: Un término cualquiera

Resultado: El argumento

Ejemplo: (QUOTE (a c)) → (a c)

Notas adicionales: El operador QUOTE regresa el argumento sin evaluarlo. Esto es útil por ejemplo, cuando se quiere trabajar con un símbolo cuyo valor es constante y no se quiere modificar.

### **Operador: FST**

Número de argumentos: 1

#### 4. PROPUESTA

---

Argumentos: Un par de celda o una lista no vacía

Resultado: El primer término de la lista o el primer elemento del par

Ejemplo: (FST (QUOTE (a c))) → a

##### **Operador: SND**

Número de argumentos: 1

Argumentos: Un par de celda o una lista no vacía

Resultado: El resto de la lista después de borrar el primer término o el segundo elemento del par

Ejemplo: (SND (QUOTE (a c))) → c

##### **Operador: PAIR**

Número de argumentos: 2

Argumentos: Dos términos cualquiera

Resultado: Crea un par de celda tal que su fst es el primer argumento y su snd es el segundo argumento

Ejemplo 1: (PAIR (QUOTE a) (QUOTE c)) → (a . c)

Ejemplo 2: (PAIR (QUOTE a) (QUOTE (b c))) → (a b c)

##### **Operador: ATOM**

Número de argumentos: 1

Argumentos: Un término cualquiera

Resultado: Retorna *t* si el argumento es un átomo; *nil* en otro caso

Ejemplo: (ATOM (QUOTE (a c))) → *nil*

**Operador: EQ**

Número de argumentos: 2

Argumentos: Dos átomos

Resultado: Retorna *t* si los átomos son iguales; *nil* en otro caso

Ejemplo: (EQ (QUOTE a) (QUOTE a)) → *t*

**Operador: STRING-EQ**

Número de argumentos: 2

Argumentos: Dos átomos cuyo contenido sea una cadena de caracteres

Resultado: Retorna *t* si los átomos son iguales; *nil* en otro caso

Ejemplo: (STRING-EQ "hola" "mundo") → *nil*

Notas adicionales: Este operador es un caso particular de **EQ**, su existencia es conveniente porque en lingüística se trabaja con cadenas de caracteres.

**Operador: SUBSTRING**

Número de argumentos: 3

Argumentos: Un átomo cuyo contenido sea una cadena de caracteres, dos dígitos que indican el carácter de inicio y el carácter fin de la subcadena

Resultado: Retorna una nueva cadena cuyo contenido es una subcadena de la original

Ejemplo: (SUBSTRING "hola mundo" 2 8) → "la mun"

#### 4. PROPUESTA

---

Notas adicionales: Las cadenas son el único elemento del lenguaje que es tratado como un arreglo (de estructura de datos), por lo que los elementos de una cadena de caracteres inician su numeración en 0. El primer número del operador es inclusivo y el segundo es exclusivo, véase el ejemplo. Si el segundo número es negativo, se interpreta que llegará al final de la cadena original.

##### **Operador: NULL**

Número de argumentos: 1

Argumentos: Un término cualquiera

Resultado: Retorna  $t$  si el argumento es una lista vacía; *nil* en otro caso

Ejemplo: (NULL (QUOTE ()))  $\rightarrow t$

##### **Operador: DOP**

Número de argumentos: 3

Argumentos: Un símbolo, una lista de argumentos y el cuerpo de la función

Resultado: El símbolo

Ejemplo: Ver sección 4.2

Notas adicionales: **DOP** asocia un valor al apuntador  $O$  de un símbolo. El símbolo será el nombre del nuevo operador, la lista serán los parámetros necesarios para que el operador cumpla su propósito. Como el objetivo de **DOP** es definir un nuevo operador, sus elementos no se evalúan y por lo tanto no se requiere el uso de **QUOTE**.

Análogamente, es necesario decir que están disponibles los operadores lógico-matemáticos que resultan útiles en todos los lenguajes formales y permitirán

ampliar el uso de nuestro lenguaje formal, éstos son escritos con sus signos comunes en el caso de los matemáticos y con la notación que hasta ahora se ha manejado para los operadores lógicos, mientras todos conservan la notación prefija:

- Adición:  $+$ ,  $-$
- Multiplicación:  $*$ ,  $/$
- Comparativos:  $<$ ,  $>$ ,  $<=$ ,  $>=$ ,  $=$  (equivalente a **EQ** especializado a números)
- Lógicos: **AND**, **OR**, **NOT**...

En realidad, es posible declarar cualquier operador que sea considerado *axiomático* para la descripción lingüística, de tal forma que es posible extender justificadamente el lenguaje para que se adapte a los requerimientos teóricos del lingüista.

### 4.2.2. Operadores Complejos

Los siguientes operadores, además de ser fundamentales para el lenguaje, debido a sus numerosos casos de uso, deben explicarse con más detalle.

#### 4.2.2.1. Operador **COND**

El operador **COND** es la forma más general de realizar una determinada acción según una condición en un determinado contexto.

Cada elemento de **COND** consiste en una lista de dos elementos que forman un predicado: el primero, una condición que será puesta a prueba y el segundo, una acción a realizar en caso de que la condición se cumpla. Dicho de otra manera, son dos proposiciones conectadas por un conectivo lógico condicional  $\implies$  como el descrito en la sección 2.2.1.

## 4. PROPUESTA

---

Si la primera condición es evaluada como cierta, entonces la acción correspondiente se realiza y el condicional regresa el valor resultante de realizar la acción, el resto de los predicados son omitidos. Si la primera condición es evaluada como falsa, entonces la acción correspondiente se omite y se evalúa la condición del segundo predicado, así sucesivamente. Si ninguna de las condiciones se cumple, el operador **COND** regresa nil.

```
(COND
  (condición1 acción1)
  (condición2 acción2)
  ⋮
  (condiciónn acciónn))
```

El siguiente ejemplo muestra el uso de **COND** comparando números, si el número es menor que cero regresa -1, si es igual a cero regresa 0 y si es mayor que cero regresa 1.

```
1 (COND ((< number 0) -1)
2       ((= number 0) 0)
3       ((> number 0) 1))
```

Un ejemplo más concreto sería suponer que tenemos una lista con los elementos de una frase nominal y cada elemento es una cadena de caracteres.

```
1 (SETQ FN (QUOTE("La" . ("vaca" . ("blanca" . nil)))))
2 (DOP POS-FN (word)
3 (COND ((STRING-EQ word "La") "Determinante")
4       ((STRING-EQ word "vaca") "Nombre")
5       ((STRING-EQ word "blanca") "Adjetivo")))
```

En el código anterior definimos la lista y un operador que toma la palabra y devuelve su respectiva POS. Es un operador bastante limitado ya que igual con



una cadena predeterminada, pero es un buen ejemplo para ver como extraer un elemento específico de una lista. El siguiente código obtiene cada uno de los elementos de la lista mediante los operadores vistos en la sección 4.2.1.

```

1  (FST FN)
2  "La"
3  (FST (SND FN))
4  "vaca"
5  (FST (SND (SND FN)))
6  "blanca"

```

De esta manera, el operador **POS-FN** puede recibir los argumentos de la siguiente manera:

```

1  (POS-FN (FST FN))
2  "Determinante"
3  (POS-FN (FST (SND FN)))
4  "Nombre"
5  (POS-FN (FST (SND (SND FN))))
6  "Adjetivo"

```

Por otra parte, si es necesario que se efectué una acción si se cumplen una serie de condiciones, es posible conectar cada acción mediante un operador lógico **AND**; sin embargo, al tratarse de una secuencia de condiciones se recomienda usar **COND\*** que evalúa una lista de condiciones para poder ejecutar una acción, en caso de que no se cumpla una condición regresará nil.

```

(COND*
  ((condición1)
   (condición2)
   :
   (condiciónn))
  acción)

```

La ventaja de usar **COND\*** en lugar de **COND** es simplemente la facilidad de

lectura, pues la primera conserva cada condición claramente separada por lo que es fácil leer la lista de condiciones a cumplir.

### 4.2.2.2. Operador LAMBDA

El operador **LAMBDA** es la forma en la que nuestro lenguaje usa las expresiones  $\lambda$  descritas por Alonzo Church en 1936 y brevemente explicadas en la sección 2.3.

La formación de un operador **LAMBDA** es la siguiente:

$$(\mathbf{LAMBDA} \ (p_1 \ p_2 \ \dots \ p_n) \\ \text{acciones})$$

Como puede verse, este operador tiene la misma estructura que **DOP**, pero con el nombre del nuevo operador elidido, de esta manera, el operador **LAMBDA** permite definir un nuevo operador sin asociarlo a un símbolo.

Resulta necesario listar algunas particularidades que podrían no ser evidentes a simple vista: primero, al ser una función anónima, un operador **LAMBDA** para ser útil como los demás operadores de nuestro lenguaje debe estar donde se espera que se encuentre un símbolo que quiere ser usado como un operador, es decir, al inicio de una expresión y los valores siguientes serán la lista particular de sus argumentos:

$$((\mathbf{LAMBDA} \ \dots) \text{argumentos})$$

De no ser así, el operador cumpliría con su misión de describir, pero no devolvería un valor útil si es evaluado. El siguiente ejemplo muestra lo que sucedería en la evaluación si se usara este operador para describir el operador **SQUARE** de la sección 4.2:

```
1  (LAMBDA (num)
2    (* num num))
3  (LAMBDA (num) (* num num))
```

```

4  ( (LAMBDA (num)
5    (* num num)) 3)
6  9

```

En las líneas 1 y 2, vemos una definición correcta de una expresión  $\lambda$ . Al ser evaluada en la línea 3, se regresa la misma expresión. Las líneas 4 y 5, coloca a todo el operador **LAMBDA** en la posición de un símbolo que utiliza su apuntador *O* y el siguiente valor de la lista será evaluado como argumento del operador, esto simula bastante bien la asociatividad a la izquierda que posee el cálculo  $\lambda$ .

La segunda particularidad refiere a que en el cálculo  $\lambda$ , cada expresión evalúa una variable a la vez, por lo que en un lenguaje como el nuestro obliga a operadores monoargumentales, la concatenación de pares de celda mostrada en la sección 4.2 demuestra que cada operador definido es fácilmente traducido a su forma monoargumental, tal como lo hace el cálculo  $\lambda$  por lo que no hay ningún error en la sintaxis del lenguaje, la forma del operador **LAMBDA** es solo una forma implícita del proceso de pasar una función  $f(x, y)$  a su forma monoargumental  $\lambda x. \lambda y.$  Ejemplificando nuevamente con el operador **SQUARE**, lo definiremos en forma explícitamente monoargumental:

```

1  (DOP SQUARE (y)
2    ( (LAMBDA (x)
3      (* y x) ) y) )

```

Se puede ver aquí que la función **LAMBDA** solo recibe un argumento  $x$ , a pesar de que exige recibir un par de argumentos  $x, y$ , por supuesto, no estamos haciendo una conversión  $\alpha$  porque no hay una segunda expresión  $\lambda$  que haga necesario el renombramiento.

#### 4.2.2.3. Listas de propiedades

En lingüística será muy útil poder estructurar datos de forma mucho más directa si tenemos propiedades y cada una de ellas contiene un valor. Esto es

## 4. PROPUESTA

---

necesario porque muchas de las descripciones son hechas directamente de esta forma.

Una lista de propiedades es una lista especial que tiene la siguiente forma:

$$(:propiedad_1 \text{ valor}_1 \dots :propiedad_n \text{ valor}_n)$$

De esta manera, es posible declarar una lista de propiedades para la concordancia del elemento léxico *gatos*, ejemplo visto en la sección 3.2.2.

```
1 (SETQ gatos (QUOTE (:gen masc
2                      :num sing)))
```

Se utiliza el operador **GETP** para obtener el valor específico de una propiedad. y el operador **PUTP** para agregar una propiedad nueva a la lista o bien, modificar una propiedad ya existente; sin embargo, es necesario tomar en cuenta que **PUTP** es un operador que destruye la lista de forma interna, por lo que cada vez que se modifica la lista debe declararse de nuevo.

```
1 (GETP gatos :num)
2 sing
3 (SETQ gatos (PUTP gatos (QUOTE (:num)) (QUOTE (plu))))
4 (:gen masc :num plu)
5 (SETQ gatos (PUTP gatos (QUOTE (:cat)) (QUOTE (N))))
6 (:gen masc :num plu :cat N)
```

En el ejemplo anterior, en la línea uno verificamos el valor de la propiedad *num*, al saber que es incorrecta, se reescribe la lista *gatos* con la información correcta. Finalmente en las líneas cinco y seis, se agrega una propiedad *cat* a la lista *gatos*.

Para terminar esta sección, hay que mencionar que también es posible eliminar propiedades de una lista, por medio del operador **REMP** que se usa igual que **GETP** y devuelve *t* si la propiedad fue borrada correctamente.

```
1 (REMP gatos :cat)
2 t
```

## 4.3. Variables

Una variable en nuestro lenguaje es un símbolo que tiene un valor asignado al apuntador  $V$  [Tutorials Point (I) Pvt. Ltd., 2015]. Los lenguajes formales comúnmente asignan diferentes tipos de variables, según el conjunto de valores que son capaces de manejar, por ejemplo: enteros, flotantes, caracteres, cadenas, entre muchos otros. En el lenguaje que se define en esta tesis, esas clasificaciones no son necesarias pues toda relación de información es desde el inicio intencionada, explícita y siempre es evaluada.

Gracias a eso, es posible asociar directamente un valor al apuntador  $V$  de un símbolo:

```
1 (SETQ tigre "felino")
2 "felino"
```

El ejemplo anterior muestra cómo el símbolo *tigre* es asociado a un átomo cadena con valor “*felino*” mediante el operador **SETQ**.

Hay que hacer notar que en nuestro lenguaje cada elemento siempre es evaluado de forma predeterminada y el operador **SETQ** únicamente identifica que el primero *no debe ser evaluado*; en cambio el segundo elemento si se trata de ser evaluado, por lo que expresiones como la siguiente son incorrectas:

```
1 (SETQ tigre felino)
```

La diferencia es que *felino* no es un átomo a priori<sup>1</sup>, por lo que es necesario

---

<sup>1</sup>Es necesario hacer notar que “*felino*” (con comillas) solo puede ser un átomo tipo cadena, mientras que *felino* (sin comillas), puede interpretarse como un átomo o como un símbolo.

#### 4. PROPUESTA

---

que se evalúe, en caso de no tener ningún valor, la expresión sería incorrecta. Hay dos posibles soluciones. La primera solución es asignar un valor a *felino*, lo cual tiene como consecuencia que *tigre* tenga el mismo valor que *felino*. También es importante señalar que los símbolos no permanecen relacionados por lo que si el valor de *felino* cambia posteriormente, no cambiará el valor de *tigre*.

En el siguiente ejemplo se muestra esta situación. En las líneas 1 y 3 se asigna el valor "mamífero" a *felino* y a *tigre*. Luego, a pesar de que se asigna el valor de "gato" a *felino* (línea 9), *tigre* mantiene el valor de "mamífero" (líneas 13 y 14).

```
1 (SETQ felino "mamífero")
2 "mamífero"
3 (SETQ tigre felino)
4 "mamífero"
5 felino
6 "mamífero"
7 tigre
8 "mamífero"
9 (SETQ felino "gato")
10 "gato"
11 felino
12 "gato"
13 tigre
14 "mamífero"
```

Otra solución es evitar que *tigre* sea evaluado. Eso se logra con el operador **QUOTE**, tal como indica el siguiente ejemplo:

```
1 (SETQ tigre (QUOTE felino))
2 felino
```

Esto tiene varias implicaciones, la primera es notar que *felino* es ahora un átomo diferente a "felino"; el segundo es de tipo cadena, mientras que el primero es simplemente *felino*. Si posteriormente se asignara un valor a un símbolo *felino*, el valor de *tigre* no se vería afectado, pues el valor de *tigre* es un átomo y no el símbolo.

Una última observación en este tema tiene que ver con el símbolo y sus apuntadores en general. Si bien se han presentado por separado los apuntadores *O* y *V*, debemos hacer hincapié en que ambos pueden estar en uso para un símbolo, y su evaluación depende del contexto, tal y como se dijo en la sección 4.1.2.1.

### 4.3.1. Variables locales

Como hemos visto, cada símbolo que asocie su apuntador *V* a un valor puede ser llamado variable, y estará disponible para su uso en cualquier parte de la descripción lingüística en la que se esté trabajando. Sin embargo, quizá sea necesario modificar una variable solamente por un momento, para describir un caso particular o para su uso en una función específica, debido a esto proporcionamos el operador **LET**.

#### 4.3.1.1. Operador LET

El operador **LET** es usado para asociar el apuntador *V* de un símbolo a un valor temporal incluso cuando el símbolo ya este definido como variable.

```
(LET ((símbolo valor)
      (símbolo valor)
      :
      (símbolo valor))
  acciones)
```

Para entender su situación de uso, considere el siguiente contexto: ha definido una lengua con un fonema /ɽ/ con tres realizaciones (alófonos): [l], [ɽ] y [r]. Bajo este supuesto, describiremos al fonema con las características (modo y punto de articulación) de [ɽ]. Como podemos ver tendríamos un símbolo que corresponde al fonema /ɽ/, con un valor particular y definido para toda nuestra descripción. Si por alguna razón, fuese necesario definir un operador que trabaje

#### 4. PROPUESTA

---

con el sonido [ɾ] (que podría definirse por medio de sus características acústicas), se tendría que distinguir esta ambigüedad entre símbolos, por lo que usaríamos el operador **LET** de la siguiente manera:

```
1 (SETQ ɾ (QUOTE(fonema)))
2 (LET ((ɾ (QUOTE(sonido))))
3     (aux ɾ))
4     (PAIR ɾ aux))
```

En este ejemplo definimos normalmente al símbolo  $\tau$  como una variable<sup>2</sup>, después dentro del operador **LET** vemos primero una lista formada por dos sublistas del tipo «(símbolo valor)». La primera sublista, escrita en la línea 2, define nuevamente el símbolo  $\tau$  con un valor distinto al usado en **SETQ** (línea 1, una realización del fonema), la segunda sublista, línea 3, define un símbolo auxiliar que nos sirve únicamente para completar la explicación de **LET**. Si evaluamos el ejemplo, veremos que dentro del operador **LET** podemos crear un símbolo o variable local cuyo valor solo está presente dentro del operador. También podemos ver que al asignar un valor al símbolo auxiliar *aux* su valor es tomado del símbolo o variable global (el fonema) y no del símbolo local (la realización), por lo tanto, el operador **LET** *no es secuencial* [Chassell, 2004]. Lo anterior se muestra de la siguiente manera:

```
1 (SETQ ɾ (QUOTE(fonema)))
2 fonema
3 (LET ((ɾ (QUOTE(sonido))))
4     (aux ɾ))
5 (PAIR ɾ aux))
6 (sonido . fonema)
```

---

<sup>2</sup>Se pretende conservar la sencillez máxima del ejemplo, por lo que en lugar de caracterizar en realidad los símbolos (punto y modo de articulación) se colocan valores representativos (fonema, sonido), para ver ejemplos reales pase a la sección 4.5.



Si es necesaria la secuencialidad que explicamos anteriormente, es posible usar el operador **LET\***, de esta manera obtendríamos lo siguiente:

```

1  (SETQ ɿ (QUOTE(fonema)))
2  fonema
3  (LET* ((ɿ (QUOTE(sonido)))
4        (aux ɿ))
5    (PAIR ɿ aux))
6    (sonido . sonido)

```

## 4.4. Recursividad

Una entidad recursiva es aquella que aparece en la definición de sí misma, Definición recursividad. así como la que se llama a sí misma. Un ejemplo cercano a la lingüística es el caso de los árboles (sección 2.2.2): un árbol es resultado de la concatenación de estructuras arbóreas[Cairó et al., 2006].

De esta forma los componentes sintácticos de la gramática transformacional consisten en dos partes: un caso base que sirve como condición para terminar la recursividad y un conjunto de reglas transformacionales que se aplicarán hasta que se cumpla el caso base[Partee et al., 1990].

Para explicar sencillamente como se aplica la recursividad en nuestro lenguaje, recurriremos al ejemplo matemático básico de la sucesión de Fibonacci<sup>3</sup>:

<sup>3</sup>La sucesión comienza con los números 0 y 1, y a partir de estos, «cada término es la suma de los dos anteriores», es la relación de recurrencia que la define. A los elementos de esta sucesión se les llama números de Fibonacci. Esta sucesión fue descrita en Europa por Leonardo de Pisa, matemático italiano del siglo XIII también conocido como Fibonacci. Tiene numerosas aplicaciones en ciencias de la computación, matemáticas y teoría de juegos. También aparece en configuraciones biológicas, como por ejemplo en las ramas de los árboles, en la disposición de las hojas en el tallo, en las flores de alcachofas y girasoles, en las inflorescencias del brécol romano y en la configuración de las piñas de las coníferas.[Wikipedia, 2016]

$$fib(n) = \begin{cases} n & \text{si } n = 0 \\ & \text{ó } n = 1 \\ fib(n-1) + fib(n-2) & \text{si } n > 1 \end{cases} \quad \forall n \geq 0 \quad (4.1)$$

$n$	$fib(n)$	$n$	$fib(n)$
0	0	5	5
1	1	6	8
2	1	7	13
3	2	8	21
4	3	9	34

Tabla 4.5: Tabla con los primeros diez valores de la sucesión de Fibonacci.

Como es posible observar, en la ecuación 4.1 la sucesión de Fibonacci tiene dos casos base: cuando  $n = 0$  y cuando  $n = 1$ . El caso recursivo se da cuando  $n > 1$ . En nuestro lenguaje, la ecuación 4.1 se vería de la siguiente manera:

```
1  (DOP FIBONACCI (n)
2    "Versión recursiva de la sucesión de Fibonacci"
3    (COND ((= n 0) n)
4           ((= n 1) n)
5           ((> n 1) (+ (FIBONACCI(- n 1)) (FIBONACCI(- n 2)))
                     )))
```

Como es posible observar, dentro del operador **FIBONACCI** se manda a llamar el mismo operador en dos ocasiones, tal y como la ecuación 4.1 indica. A pesar de la solución anterior, puede ser que en problemas más complejos seguir este tipo de solución sea impráctico e ineficiente. Es por ello que los lenguajes formales pueden crear soluciones iterativas a este tipo de problemas. Una iteración es repetir un proceso hasta alcanzar una meta. Para realizar procesos

iterativos, nuestro lenguaje ofrece el operador **LOOP**, definido de la siguiente manera:

(**LOOP** expresión S)

La definición anterior resultaría en hacer un proceso iterativo sin fin, para evitar eso, usamos otro operador adicional: **RETURN**, el cual nos permite interrumpir el proceso y opcionalmente puede tener un argumento que correspondería al resultado de **LOOP**.

La sucesión de Fibonacci escrita de forma iterativa se escribiría de la siguiente forma en nuestro lenguaje:

```

1  (DOP FIBONACCI (n)
2    "Versión iterativa de la sucesión de Fibonacci"
3    (COND ((= n 0) (SETQ fibo n))
4          ((= n 1) (SETQ fibo n))
5          (t (SETQ fiba 0 fibb 1 i 2)
6            (LOOP
7              (COND ((> i n) (RETURN fibo))
8                  (t (SETQ fibo (+ fibb fiba) fiba fibb
                      fibb fibo i (+ i 1))))))))

```

Donde *fiba* y *fibb* son dos números consecutivos de la sucesión (0 y 1 respectivamente), el siguiente par de la sucesión es *fibb* y *fiba + fibb*, de esta manera se divide un algoritmo donde sólo se requieren dos números consecutivos de la sucesión de Fibonacci en cada paso, mientras *i* es el contador que nos indica cuando alcanzamos la meta de la iteración. Este método es el que usaríamos normalmente para hacer el cálculo a lápiz y papel [Wikipedia, 2016].

De esta forma, es posible establecer en nuestro lenguaje cualquier caso de recursividad sin tener que elegir entre gran cantidad de ciclos iterativos (por ejemplo: *while*, *for*, *repeat*) como lo hacen la mayoría de los lenguajes formales.

Finalmente, podemos decir que cada uno de los grandes ciclos iterativos son casos particulares del operador **LOOP**.

### 4.4.1. La recursividad en la lingüística

La lingüística moderna se ha empeñado en encontrar *universales del lenguaje*. Uno de estos universales propuestos es precisamente la recursividad sintáctica. Por medio de esta característica es posible enunciar «Luis duerme», y es posteriormente reenunciarla para introducir una frase relativa para obtener «Luis, que ha bebido té, duerme». A su vez, en este enunciado más complejo podemos repetir el proceso y decir «Luis, que ha bebido el té que le regalaste, duerme». El proceso de introducción de una frase relativa en español es formalizado por los lingüistas como en (4.2)[Simone, 2001].

$$N \rightarrow N + F_{relativa} \quad (4.2)$$

La recursividad sintáctica se ha supuesto universal porque es extremadamente productiva en las lenguas naturales, y va a favor de la economía del lenguaje o regla del menor esfuerzo. Sin embargo, como ya se señaló en la sección 3.3: *no es posible enunciar leyes universales infalibles*; y al parecer al menos una lengua natural en el mundo rompe el universal.

En *Cultural Constraints on Grammar and Cognition in Pirahã: Another Look at the Design Features of Human Language* Daniel L. Everett presenta evidencia de que el Pirahã no presenta recursividad sintáctica. Lo anterior cuestionaría la pertinencia del lenguaje formal aquí expuesto debido a que uno de los puntos fuertes de esta propuesta se basa en la recursividad en general. Contrario a lo que se podría suponer, el Pirahã no sufre por la ausencia de recursividad<sup>4</sup> sino que la sustituye por otro mecanismo: la yuxtaposición.

---

<sup>4</sup>En el texto original Everett estudia particularmente la subordinación o *embedding*.

A través de la yuxtaposición, los hablantes de Pirahã pueden expresarse con la misma facilidad, además, la yuxtaposición en el Pirahã podría ser recursiva. Una regla (como la de la yuxtaposición del Pirahã) que se pueda aplicar al resultado de una aplicación anterior de sí misma se denomina *recurrente*; y *recursividad* se llama al fenómeno por el cual, en un código, pueden existir reglas recurrentes[Simone, 2001]. Con lo anterior decimos que la recursividad sintáctica no limita la recursividad de las lenguas naturales, pues éstas siempre implementarán los mecanismos necesarios para realizar su función.

Moreover, they tend not to point with individual fingers, at least when talking to me. Commonly, if they use any part of their arms for pointing, they tend to extend a flat hand turned sideways or an open palm facing up or down. More often, they point, as is common around the world, with their lower lip or jaw or a motion of the head.[Everett et al., 2005]<sup>5</sup>

No existen lenguas más difíciles que otras, descubrir los mecanismos de expresión de una lengua es la labor del lingüista, por lo que se considera pertinente, en este sentido, la propuesta de lenguaje formal aquí desarrollada. La existencia de universales, las relaciones entre lengua y cultura, si bien son de importancia e interés, escapan al propósito de este trabajo.

## 4.5. Ejemplos de uso del lenguaje

En este apartado presentaremos algunos ejemplos que ilustran cómo es posible usar el lenguaje descrito en este capítulo. Estos ejemplos no agotan las

---

<sup>5</sup>Además, tienden a no apuntar con dedos individuales, al menos al hablar conmigo. Comúnmente, si usan cualquier parte de sus brazos para apuntar, tienden a extender una mano plana vuelta lateralmente o una palma abierta hacia arriba o hacia abajo. Más a menudo, señalan, como es común en todo el mundo, con su labio inferior o mandíbula o un movimiento de la cabeza.

posibilidades del lenguaje, asimismo, no son usados todos los elementos principales que se han descrito a lo largo del presente capítulo. Ahora bien, la ausencia de algunos elementos no significa que no sean elementos fundamentales y de potencial uso en muchas otras posibles situaciones de descripción lingüística. Todos los ejemplos mostrados aquí han sido obtenidos de artículos o libros de lingüística, lo único que se hace en esta sección es mostrar la representación de dichos ejemplos en el lenguaje propuesto. Consideramos que representar estos fenómenos nos permite validar la propuesta de esta tesis.

### 4.5.1. Fonología

Como se dijo en la sección 2.1.1, la fonología tiene como unidad mínima el fonema. A su vez, el fonema puede ser caracterizado por medio de rasgos distintivos. En la caracterización de fonemas es comúnmente usada algún tipo de tabla o matriz (Ver sección 3.2.3.), tal es el caso de la palabra *cabo*, cuya representación fonética<sup>6</sup> básica se presenta en la tabla 4.6.

De igual manera que las matrices, los sonidos pueden ser caracterizados en nuestro lenguaje como una lista, de tal forma que a cada sonido le corresponda como valor una lista de rasgos distintivos, es decir, cada columna de la tabla es una lista, como en el código 4.1 donde se crean cuatro listas, una por cada sonido.

Representación de unidades lingüísticas.

```
1 | (SETQ k (QUOTE((+consonántico -silábico -continuo -nasal -
   | coronal +retraído -redondeado -acentuado))))
2 | (SETQ a (QUOTE((-consonántico +silábico +continuo -nasal -
   | coronal +retraído -redondeado +acentuado))))
```

<sup>6</sup>Tal y como se dice en la obra *Aproximación a la fonología generativa: Principios teóricos y problemas*, la diferencia entre una representación fonética y la fonológica consiste en que la segunda excluye todo contraste que esté sujeto a una regla. En este caso, hay una vocal acentuada y otra inacentuada, además hay una regla que condiciona la aparición de la fricativa sonora [β] y no de la oclusiva sonora [b].

	k	á	β	o
consonántico	+	-	+	-
silábico	-	+	-	+
continuo	-	+	+	+
nasal	-	-	-	-
coronal	-	-	-	-
retraído	+	+	-	+
redondeado	-	-	-	+
acentuado	-	+	-	-

Tabla 4.6: Tabla que muestra los rasgos distintivos de los sonidos de la palabra *cabo*[Contreras and Lleó, 1982].

```

3  (SETQ β (QUOTE((+consonántico -silábico +continuo -nasal -
   coronal -retraído -redondeado -acentuado))))
4  (SETQ o (QUOTE((-consonántico +silábico +continuo -nasal -
   coronal +retraído +redondeado -acentuado))))

```

Código 4.1: Representación en forma de lista de rasgos distintivos de los sonidos de la palabra *cabo*.

También puede ser una buena idea recurrir a listas que correspondan directamente a una matriz, esto es posible lograrlo por medio de pares de puntos. Hay que hacer notar que en lugar de usar los signos de (+, -) hacemos uso de los símbolos t y nil, pues son más adecuados para nuestro lenguaje y cumplen con la misma función.

```

1  (SETQ k (QUOTE((cons . t)
2                (sil . nil)
3                (cont . nil)
4                (nas . nil)
5                (cor . nil)
6                (ret . t)

```

#### 4. PROPUESTA

```
7 | (red . nil)
8 | (acen . nil))))
```

Código 4.2: Representación de rasgos distintivos del sonido [k] con una lista de pares de puntos.

A pesar de esto, las representaciones anteriores son muy básicas y no requieren un orden predeterminado. Las teorías lingüísticas son más exigentes en la actualidad, por ejemplo, el modelo autosegmental-jerárquico ha usado una estructura arbórea no binaria para explicar diversos fenómenos de la lengua y ordenar los diversos rasgos distintivos. De hecho este modelo es una representación que está compuesta por un conjunto de diversas secuencias articulatorias simultáneas (como la lengua, los labios o la laringe) de segmentos interrelacionados o asociados [Goldsmith, 1999]. La figura 4.7 representa la estructura antes mencionada, la cual podemos convertirla en un esquema como los mostrados en la sección 4.2, que a su vez se muestra en la figura 4.8. Debemos hacer notar que ésta es la representación gráfica que internamente seguiría el lenguaje propuesto en este capítulo.

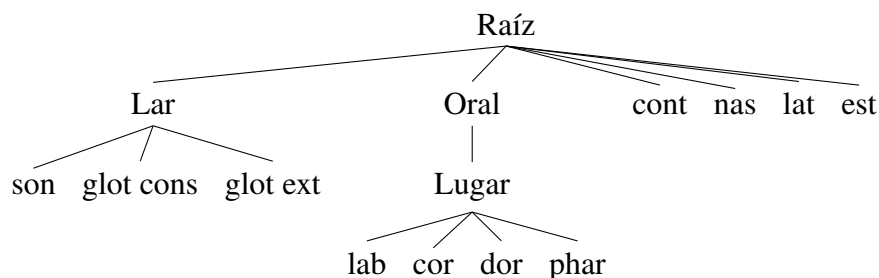
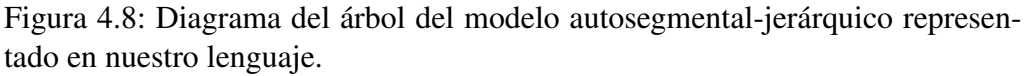


Figura 4.7: Árbol de rasgos según el modelo autosegmental-jerárquico.

Equivalencia de representaciones.

Una vez que vemos de forma gráfica la equivalencia entre las figuras 4.7 y 4.8 podemos también comprender con facilidad la representación formal de dicho árbol. Como es posible observar, la representación podría ser concatenada en una





```
(Raiz
  (Lar
    (son g.cons g.ext))
  (Oral
    (Lugar
      (lab cor dor phar)))
cont
nas
lat
est)
```

Por medio del código 4.3 podemos ya definir de forma clara cada fonema por medio del modelo autosegmental-jerárquico que representa la figura 4.7, sin em-

## 4. PROPUESTA

---

bargo, si es necesario, aún ahora tenemos alternativas, una de ellas, es ver cada rasgo como una propiedad dentro de nuestro nuevo lenguaje. Esto trae consigo varias ventajas, la más importante, poder incorporar fácilmente los elementos tonológicos que sean necesarios, tal y como se menciona en *An Overview of Autosegmental Phonology*. Por ejemplo, en la palabra inglesa *pin*, la vocal tiene dichos elementos[Goldsmith, 1999].

```
1 (SETQ i (QUOTE((:sil t
2   :nas nil
3   :lab nil
4   :cor nil
5   :pitch ((:high t :low nil)
6           (:high nil :low t))))))
```

Código 4.4: Rasgos distintivos de la vocal [i] por medio de listas de propiedades.

La representación anterior responde a que se toma en cuenta la actividad de la laringe al momento de la emisión. La laringe es responsable del tono, en este caso dicho tono es rápidamente descendente (esto tomando en cuenta la emisión aislada de la palabra). El tono descendente se representa como la secuencia de un todo alto y un tono bajo.

Como podemos ver, podemos agregar tantos niveles sean necesarios para la representación y tantas propiedades sean necesarias, ordenadas con claridad, sin recurrir a líneas de asociación, las cuales aun serían visibles si tratáramos de dibujar el diagrama correspondiente a cada código.

### 4.5.1.1. Lenguas no sonoras

Es necesario establecer al menos por una vez la pertinencia del lenguaje propuesto en lenguas no sonoras. Para este propósito tomaremos como ejemplo las lenguas de señas.

La primera impresión sería que en lenguas de señas no es posible hacer un análisis fonológico. Sin embargo, esa postura no es correcta, en principio porque más de una definición de fonema favorece la posibilidad de este tipo de análisis:

Los FONEMAS son unidades abstractas compuestas de elementos co-existent denominados RASGOS DISTINTIVOS.

[Real Academia Española and Asociación de Academias de la Lengua Española, 2010]

Como podemos leer, los fonemas son unidades abstractas mas no sonoras y si bien es sabido que los rasgos distintivos refieren a sonoridad, punto y modo de articulación, nada impide que el significado de una lengua se diferencie por otro tipo de gestos.

Si bien no es nuestro objetivo describir las gramáticas de las lenguas de señas<sup>7</sup> si es necesario decir que estas lenguas son lenguas naturales y como tales presentan parámetros dentro de una señal que en conjunto distinguen significado [Saldías Uribe, 2015, García, 2002], tales parámetros son:

**TAB:** Lugar donde se realiza la señal.

**DEZ:** La configuración de las manos.

**SIG:** El movimiento de las manos para realizar la señal.

La combinación de estos parámetros forman cada una de las señas posibles por lo que «el subconjunto de elementos contenido en cada una de estas categorías de localización, configuración manual y movimiento serían el equivalente de los criterios utilizados para llevar a cabo el inventario fonológico de las lenguas audio-vocales»[Saldías Uribe, 2015].

Tal y como lo hace la fonología tradicional, cuando se toma en cuenta un fonema de una lengua de señas:

---

<sup>7</sup>Para leer detalles sobre las lenguas de señas se recomienda a William Stokoe[Armstrong et al., 1995].

Se considera una mano débil y una fuerte. La mano derecha sería el articulador principal o activo (en el caso de las personas diestras); así como la lengua lo es en la lengua oral. Entonces se hablaría de rasgos de clase mayor, contorno, plano, calidad y lugar del movimiento. La organización del armazón articulatorio estaría dada en aspectos como: parte de la mano, proximidad, relación espacial, localización, parte de la mano, plano. A su vez los articuladores pasivos podrían relacionarse a las veinte mayores posiciones hacia donde la mano derecha se acercaría: detrás de la cabeza, parte alta de la cabeza, frente, sien, nariz, mejilla, oreja, boca, labio, mandíbula, barbilla, cuello, hombro, pecho, tronco, brazo levantado, antebrazo, abdomen y pierna. Esto configuraría puntos de contacto.

A su vez la dirección del movimiento se realizaría en la “ventana” donde hacemos las señas. Así esta ventana sería un plano de  $180^\circ$ . Asimismo, consideremos este plano como las cavidades que se usan en la lengua oral. De acuerdo con el lugar hacia donde se orienten las manos se le asignará un significado determinado. [García, 2002]

Como se puede observar, no existe evidencia para pensar que las lenguas naturales no sonoras funcionen bajo niveles de análisis diferentes a las lenguas orales. El modo de descripción es distinto pero para los propósitos del lenguaje aquí propuesto los principios sintácticos son los mismos. Aún más, nuestro lenguaje tiene la ventaja de no necesitar adaptaciones forzadas para ser efectivo en estos contextos complicados, por ejemplo, es fácil representar el plano de  $180^\circ$  por medio de un par de puntos en coordenadas polares<sup>8</sup>, otras representaciones

---

<sup>8</sup>En el sistema de referencia polar se toma: un punto  $O$  del plano, al que se llama origen o polo; y una recta dirigida (o rayo, o segmento  $\overline{OL}$ ) que pasa por  $O$ , llamada eje polar (equivalente al eje  $x$  del sistema cartesiano). Con este sistema de referencia y una unidad de medida métrica (para poder asignar distancias entre cada par de puntos del plano), todo punto  $P$  del plano corresponde a un par ordenado  $(r, \theta)$  donde  $r$  es la distancia de  $P$  al origen y  $\theta$  es el

usadas en lingüística simplemente no pueden o necesitan mezclar y adaptar sus representaciones.

### 4.5.2. Morfología

Uno de los conceptos más difíciles dentro de la lingüística es el de *palabra*. Muchas veces este concepto no resulta útil para un análisis y es sustituido por otro mucho más específico como el de *lexema*. El cambio se exige porque una unidad básica que expresa significado o que está asociada a un significado puede ser una palabra o puede ser menos que una palabra. A su vez una palabra se compone con elementos menores individualmente significativos: los morfemas. [Pickett and Elson, 1986]

La identificación de morfemas es una de las tareas más demandantes de la labor lingüística. Esto es porque los morfemas son elementos no intuitivos, los cuales son fácilmente confundibles si no se cuenta con el corpus adecuado y además exige un procedimiento bien definido desde hace años. Procedimientos lingüísticos.

Es necesario un procedimiento para identificar estas partes mínimas. En realidad, hay dos procedimientos posibles y útiles para descubrir e identificar morfemas. En el primero se recurre a la **sustitución** de diferentes formas con distintos significados en el mismo contexto [...] El segundo procedimiento envuelve el descubrimiento de elementos **parciales recurrentes** con significado constante. [Pickett and Elson, 1986]

En estos dos procedimientos anteriormente citados, el lingüista analiza los contrastes que encuentra dado un contexto común para poder decir cuales son

---

ángulo formado entre el eje polar y la recta dirigida  $\overline{OP}$  que va de  $O$  a  $P$ . El valor  $\theta$  crece en sentido antihorario y decrece en sentido horario. La distancia  $r$  ( $r \geq 0$ ) se conoce como la «coordenada radial» o «radio vector», mientras que el ángulo es la «coordenada angular» o «ángulo polar».[Wikipedia, 2017]

los posibles morfemas diferentes. Para Zellig Harris, el procedimiento en forma de un algoritmo es muy similar<sup>9</sup>, y lo explica así:

The basic procedure is to ask how many different phonemes occur (in various utterances) after the first  $n$  phonemes of some test utterance. It will be found that the number of these possible successors to the first  $n$  phonemes varies with  $n$  [...] The LIST of phonemes which occurs in any utterance after a particular utterance-initial sequence may be called the SUCCESSOR VARIETY for that sequence; while the NUMBER of phonemes in that list is the SUCCESSOR COUNT for that sequence.[Harris, 1970]<sup>10</sup>

La diferencia entre los procedimientos de Pickett y Harris es que Harris encuentra primero el contexto común. Para más detalle podemos ejemplificarlo con el paradigma del verbo *cantar* en la lengua purépecha [Chamoreau, 2009]:

- (21) a. pirikani  
          ‘Yo canto’  
      b. pirika  
          ‘Yo canto’  
      c. pirikari  
          ‘Tú cantas’  
      d. piriti  
          ‘Él/ella canta’

---

<sup>9</sup>En realidad, considero que es el mismo, aunque podría ser discutible debido a la fineza del análisis que puede realizar un ser humano.

<sup>10</sup>El procedimiento básico consiste en preguntar cuántos fonemas diferentes ocurren (en varias emisiones) después de los primeros  $n$  fonemas de algunas emisiones de prueba. Se encontrará que el número de estos posibles sucesores de los primeros  $n$  fonemas varía junto con  $n$  [...] La LISTA de fonemas que ocurre en cualquier emisión después de una secuencia inicial puede ser llamada la VARIEDAD SUCESORA para esa secuencia; Mientras el NÚMERO de fonemas en esa lista es la CUENTA SUCESIVA para esa secuencia.

- e. *pirikakxi*  
‘Nosotros cantamos (yo y él)’
- f. *pirikach’i*  
‘Nosotros cantamos (yo y tú)’

Siguiendo el procedimiento sugerido por Pickett, se deben comparar las diferencias en las formas de (21) con la diferencia de sus significados, se debe notar también que la lista presente en (21) implica ya poner los elementos en posición de sustitución y contraste. De esta manera se identifica fácilmente un contexto idéntico: *piri-*, el cual es nuestro primer elemento sugerido como posible morfema. Posteriormente se deben encontrar elementos parciales recurrentes.

Los elementos recurrentes en el caso de (21) son los siguientes:

- (22) a. *-ka/-ti*
- b. *-ni/-ri/-kxi/-ch’i/-Ø*

Como se puede ver los elementos de cada uno de los incisos de (22) están en contraste; se deben aclarar dos cosas: la primera es que el proceso de sustitución y búsqueda de contrastes son paralelos y un lingüista los realiza sin importar el orden. Lo segundo es que se ha hecho de lado el análisis sobre el cambio de significado, necesario para proponer la existencia de un morfema, priorizando la forma de encontrar lugares donde posiblemente se encuentren los lindes morféminos. En resumen, los elementos presentes en (21) se separarían como en (23).

- (23) a. *piri-ka-ni*
- b. *piri-ka*
- c. *piri-ka-ri*

#### 4. PROPUESTA

---

d. piri-ti

e. piri-ka-kxï

f. piri-ka-ch'i

Método del sucesor frecuente.

Ahora, si seguimos el procedimiento de Harris, debemos tomar en cuenta dos valores que irán cambiando: la variedad sucesora ( $V$ ) y la cuenta sucesiva ( $C$ ). El cambio de valor se lleva a cabo de la siguiente manera: cuando tomamos el sonido [p]  $C = 1$  pues hay un solo sonido,  $V = 1$  porque después de [p] solo se presenta el sonido [i]; ahora tomamos el sonido [i],  $C = 2$  porque tenemos ahora [pi], a su vez  $V = 1$  porque después de [i] solo se presenta [r]. El valor de  $V$  es constante hasta que  $C = 4$  donde podemos presenciar que luego de [i] puede estar [k] o [t], por lo que  $V = 2$ . Ahora cuando  $C = 5$ , los caminos a seguir se bifurcan en [k] y [t]. Si seguimos el camino de [k], tendremos  $C = 5$  y  $V = 1$ . Al ser un ejemplo sencillo lo mismo pasa en el camino de [t]. Sea como fuere, observamos que entre  $C = 4$  y  $C = 5$  se da un máximo valor relativo, o como Harris dice “se alcanza un pico”. Esto quiere decir que entre  $C = 4$  y  $C = 5$  se puede encontrar un linde morféxico.

Si continuáramos el proceso de Harris veríamos que la segmentación sería la misma que la presente en (23). Estos procedimientos, pueden ser también representados en nuestro lenguaje, y en caso de crear un interprete podrían también automatizarse.

Para formalizar lo explicado anteriormente en nuestro lenguaje, debemos tener en cuenta que hay que comparar parte a parte los elementos del paradigma presente en (21), empezando desde el mínimo de fonemas hasta el total de los mismos. Si declaramos en nuestro lenguaje el paradigma de (21) como una lista, necesitaremos saber cuantos elementos tiene dicha lista para poder compararlos todos uno a uno. En el código 4.5 podemos ver esta representación<sup>11</sup> a detalle.

---

<sup>11</sup>Para este ejemplo los elementos de la lista consisten en cadenas y no son sublistas de átomos



También debemos saber qué elementos de la lista vamos a comparar en cada ocasión.

```
1 (SETQ paradigma (QUOTE ("pirikani" "pirika"  
2                      "pirikari" "piriti"  
3                      "pirikakxi" "pirikachi"))))
```

Código 4.5: Declaración de (21) en forma de lista para nuestro lenguaje.

Todo lo anterior describe un procedimiento que no es necesario especificar en una descripción lingüística, es más bien, internamente procedimental para el lenguaje, por lo que podemos decir que existen (y pueden escribirse paso a paso) dos operadores del lenguaje: el primero lo llamaremos **LENGTH** e identifica cuántos elementos hay en el paradigma; el segundo lo llamaremos **NTH** y toma dos elementos específicos de la lista de elementos del paradigma, para esto basta con recurrir al código 4.6.

```
(LENGTH list)  
(NTH n list)
```

Código 4.6: Definición de dos funciones, una cuenta los elementos de una lista la otra selecciona un elemento específico.

```
1 (SETQ miLista '(3 5 9))  
2 (LENGTH miLista)  
3 3  
4 (NTH 1 miLista)  
5 5
```

Código 4.7: Ejemplos de uso de LENGTH y NTH

Una vez que hemos dicho que esos operadores existen (código 4.6) y ejemplos de cómo se usan (código 4.7), ya podemos realizar el procedimiento que nos individuales, lo anterior con el fin de mostrar que los operadores **NTH** y **LENGTH** a pesar de no ser fundamentales no son improvisados y no crean vacíos en las representaciones del lenguaje.

## 4. PROPUESTA

Representación de procedimientos.

permite encontrar posibles morfemas. Para ello debemos tener en cuenta que lo primero a hacer es comparar una parte de cada elemento de nuestra lista *paradigma* por los elementos que las conforman y que sean del mismo tamaño. Es decir si tomamos los primeros dos elementos de la lista usaríamos el operador **NTH** para seleccionarlos y el operador **SUBSTRING** para escoger la parte a comparar de cada elemento escogido para finalmente compararlos por medio del operador **STRING-EQ**, este último regresará nil si son diferentes.

```
1 | (STRING-EQ (SUBSTRING (NTH 0 paradigma) 0 1) (SUBSTRING (NTH 1 paradigma) 0 1))
```

Código 4.8: Selecciona los dos primeros elementos de una lista y compara los primeros caracteres de estos elementos.

En forma general, ese es el núcleo de nuestro procedimiento; por supuesto, debemos tomar en cuenta que debemos identificar la variedad posible, Harris nos dice que necesitamos contar la variedad sucesora ( $V$ ), la cual es realidad una lista de valores. También será requerida una variable auxiliar ( $v$ ) que al final del ciclo contendrá el valor que guardaremos en  $V$ . Finalmente veremos una variable auxiliar (*aux*) que servirá para cambiar de forma ordenada los elementos que estamos comparando. Por lo tanto necesitamos un ciclo que maneja dos condiciones: una primera condición que se encarga de cambiar el valor de  $v$  que posteriormente formará parte de  $V$  y otra para asegurarse que todos los elementos de la lista *paradigma* son comparados entre si, todo lo descrito corresponde al código 4.9.

```
1 | (LOOP  
2 |   (COND ((= aux length)  
3 |     (RETURN (SETQ V (PAIR v V)) )  
4 |   )  
5 |   (< aux length)
```

```

6      (COND ((STRING-EQ (SUBSTRING (NTH 0 list) 0 C) (
          SUBSTRING (NTH aux list) 0 C))
7          (SETQ aux (+ aux 1))
8          )
9      (t
10         (SETQ v (+ 1 v) aux (+ 1 aux))
11         ))))

```

Código 4.9: Método del sucesor frecuente para encontrar posibles lindes morfémicos.

Su uso completo sería tal y como se presenta en el código 4.10; en dicho código las líneas 19 a 23 son los correspondientes resultados de realizar el ciclo definido con valores de  $C = 1$  (línea 23) hasta  $C = 5$  (línea 19), que supone que la línea 5 se declaró con distintos valores de  $C$ . Nuevamente podemos ver que entre  $C = 4$  y  $C = 5$  se alcanza un pico que sugiere la presencia de un linde morfémico y será necesario un análisis más detallado para confirmar esta primera prueba.

```

1  (SETQ paradigma (QUOTE ("pirikani" "pirika" "pirikari" "piriti" "
    pirikakxi" "pirikachi")))
2  ("pirikani" "pirika" "pirikari" "piriti" "pirikakxi" "pirikachi")
3  (SETQ V ())
4  nil
5  (SETQ v 1 list paradigma length (LENGTH paradigma) aux 1 C 1)
6  1
7  (LOOP
8      (COND ((= aux length)
9          (RETURN (SETQ V (PAIR v V)))
10         )
11      ((< aux length)
12          (COND ((STRING-EQ (SUBSTRING (NTH 0 list) 0 C) (SUBSTRING (NTH
13              aux list) 0 C))
14              (SETQ aux (+ aux 1))
15              )
16          (t

```

## 4. PROPUESTA

---

```
16      (SETQ v (+ 1 v) aux (+ 1 aux))
17      ))
18  )))
19  (2 1 1 1 1)
20  (1 1 1 1)
21  (1 1 1)
22  (1 1)
23  (1)
```

Código 4.10: Ejemplo práctico del uso del Método del sucesor frecuente

Es importante decir que es posible mejorar lo escrito en el código 4.9, iniciando por crear un operador independiente, hacer que el valor de  $C$  cambie a la par de  $V$ , además de agregar casos lingüísticos como: encontrar primero sufijos y alomorfos. Sin embargo, es posible decir que el código 4.8 contiene ya la parte clave del procedimiento y que el código 4.9 describe los pasos que manualmente se deben seguir.

Algo que sería conveniente retomar es la aparición de los operadores **NTH** y **LENGTH**. Como se dijo anteriormente, en nuestro caso solo es necesario declarar su existencia ya que lingüísticamente no tienen relevancia y sería poco práctico especificar procedimientos innecesarios para los objetivos de una investigación, sin embargo, en algún momento, un elemento declarativo del lenguaje puede volverse procedimental y podemos escribir, entre otras opciones, un operador lambda que lo describa, o bien, si su uso es constante, nombrar un operador que contenga el elemento pertinente. Esto puede ser necesario para comprobar alguna situación lingüística específica. Usaremos los operadores **NTH** y **LENGTH** como ejemplo de ello, a continuación se coloca su declaración procedimental.

```
1  (DOP NTH (n list)
2  "Regresa el enésimo elemento de una lista"
3  (SETQ aux 0)
4  (LOOP
5  (COND ((= aux n) (RETURN (FST list)))))
```

```

6      (t (SETQ aux (+ aux 1) list (SND list)))
7      )))

```

Código 4.11: Operador NTH.

```

1  (DOP LENGTH (list)
2    "Regresa la longitud de una lista"
3    (SETQ length 0)
4    (LOOP
5      (COND ((AND (= length 0) (EQ nil (SND list))) (RETURN
6        length))
7        ((EQ nil (SND list)) (RETURN (+ 1 length)))
8        (t (SETQ length (+ length 1) list (SND list))))

```

Código 4.12: Operador LENGTH.

### 4.5.3. Sintaxis

El campo de la sintaxis podría considerarse uno de los más fecundos cuando se habla de métodos formales en lingüística. En lingüística computacional una de las primeras cosas por aprender es la etiquetación de elementos. Más allá de una organización de categorías, son las transformaciones una tarea común e interesante dentro de la lingüística.

Una transformación es una operación que mueve un elemento sintáctico (ya sea un elemento léxico o una frase entera) de una posición a otra dentro de una estructura[Parker and Riley, 2009].

Un ejemplo de transformación es expuesto por Parker y Riley de la siguiente manera: En inglés, el verbo *conceal* (ocultar) requiere una frase nominal que funcione como objeto directo.

#### 4. PROPUESTA

---

- (24) a. Tiny Abner has concealed the document.  
b. Tiny Abner has concealed Mary.  
c. \* Tiny Abner has concealed.  
d. \* Tiny Abner has concealed the document Mary.

Los ejemplos de (24) muestran casos válidos, donde podemos asegurar que el verbo *conceal* es un verbo transitivo y por lo tanto requiere un objeto directo. Ahora enfoquemonos en oraciones interrogativas que contengan el mismo verbo[Parker and Riley, 2009].

- (25) a. What has Tiny Abner concealed?  
b. Who has Tiny Abner concealed?  
c. \* What has Tiny Abner concealed Mary?  
d. \* Who has Tiny Abner concealed th document?

Como podemos ver en (25) las oraciones interrogativas tienen un elemento *wh*: *who*, *what*, *when*, *etc*. Además podemos saber que las respuestas a las preguntas de gramaticalmente correctas de (25) son el elemento que en (24) era el objeto directo: *the document* y *Mary* respectivamente. Entonces sabemos que el elemento *wh* sustituye en este caso el objeto directo. No obstante, el elemento *wh* no está en la posición del objeto directo, sino al inicio de la oración. Asumimos entonces que el elemento *wh* no se origina en la posición inicial de la oración, sino que una vez que es colocado en la estructura se mueve desde su posición original de objeto directo. A esto se le llama *movimiento-wh*.

Además del movimiento-*wh*, podemos observar otra transformación. La posición del primer argumento *S* y el auxiliar verbal que recibe los rasgos de tiempo, aspecto y modo cambian entre sí. Esta transformación podemos llamarla *movimiento-I*.

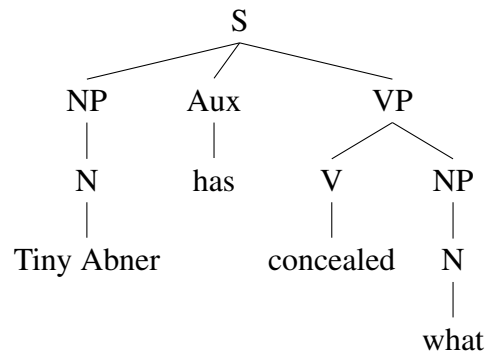


Figura 4.9: Estructura arbórea de las oraciones gramaticales de (24) una vez sustituido su objeto directo por un elemento wh

La figura 4.9 sería la representación estructural de la sustitución del objeto directo por el elemento wh. Lo primero para escribir estas transformaciones en nuestro lenguaje es tener la representación de dicho árbol, la cual esta escrita en (4.13).

```

1  (SETQ sentence (QUOTE
2      (S (
3          (NP
4              (N
5                  ("Tiny Abner"))))
6          (Aux
7              ("has"))
8          (VP (
9              (V
10                 ("concealed"))
11             (NP
12                 (N
13                     ("the document"))))))))

```

Código 4.13: Árbol equivalente a la oración (24a)

## 4. PROPUESTA

---

Se puede observar que en nuestro lenguaje se conservan con bastante claridad las jerarquías del árbol original, además de poder manejar elementos de muchas formas. Aquí las hojas del árbol son cadenas pero bien podrían ser definiciones más detalladas o bien escribir toda la definición en una sola línea y no habría ninguna diferencia. Además, también podríamos reescribir el árbol conforme al recorrido que más convenga.

Ahora es posible empezar a manejar la sustitución de un elemento de la lista, cosa que no es muy difícil en nuestro lenguaje, para modificar el primer elemento de una lista se usaría algo como el código 4.14.

```
1 (SETQ x1 (QUOTE (a b c)))
2 (a b c)
3 (SETQ x1 (PAIR (QUOTE z) (SND x1)))
4 (z b c)
```

Código 4.14: Modificación del primer elemento de una lista

Si queremos modificar otro elemento, lo único que debemos hacer es aplicar recursivamente el operador **SND** hasta el elemento que necesitamos cambiar, a este nuevo operador lo llamaremos **SUSTITUTION** y se define en los códigos 4.15 y 4.16.

```
(SUSTITUTION list n element)
```

Código 4.15: Definición de **SUSTITUTION**, que cambia un valor de una lista

```
1 (SETQ miLista '(3 5 9))
2 (SUSTITUTION miLista 1 (1 2))
3 (3 (1 2) 9)
```

Código 4.16: Ejemplo de uso de **SUSTITUTION**

Notemos que **SUSTUTION** solo opera sobre los elementos de una lista simple, por lo que en el caso de 4.13 necesitaremos encontrar primero la lista que



corresponde a la frase verbal, nada complicado, pues en la sección 4.5.2 trabajamos con **NTH**.

```
1 (SUSTITUTION (NTH 1 (NTH 2 (NTH 1 sentence))) 1 (NP (N ("
   what")))))
```

Si este procedimiento parece demasiado complejo, habrá quien prefiera redefinir la lista *sentence* cambiando un elemento y el resultado será el mismo. Esto nos muestra que en nuestro lenguaje tenemos varias alternativas para explicar procedimientos lingüísticos, todo depende de lo que se desee explicar.

Ahora vamos a referirnos a los dos movimientos que transforman el árbol. Gráficamente suelen ser representados como en 4.10. Nuevamente tenemos numerosas opciones, bien podríamos simplemente reescribir *sentence* ya con los movimientos realizados o escribir algún procedimiento más detallado.

Podríamos elaborar cada uno de los movimientos, y terminaríamos por declarar la existencia de un operador que cambie el orden de 2 elementos de una lista, sin embargo eso ya no aporta nada nuevo a esta serie de ejemplos, pues ya ha quedado claro esa capacidad del lenguaje. Abordaremos entonces, de una forma diferente dichos movimientos.

Supongamos que tenemos identificados los elementos de nuestra oración uno a uno, de la siguiente manera:

```
1 (SETQ N1 "Tiny Abner"
2   NP1 (QUOTE N1)
3   AUX "has"
4   V "concealed"
5   N2 "the document"
6   NP2 (QUOTE N2))
```

Código 4.17: Etiquetación de elementos.

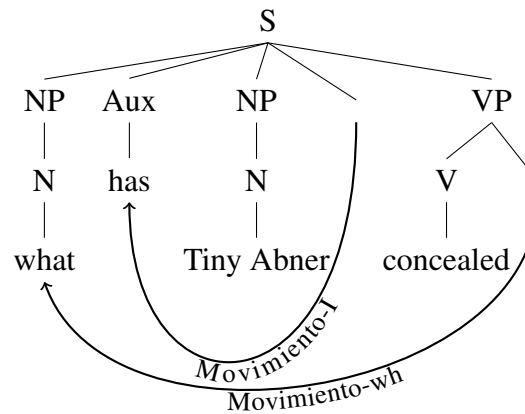


Figura 4.10: Estructura arbórea que ilustra el movimiento-I y el movimiento-wh que transforman las oraciones gramaticales de (24) en las oraciones gramaticales de (25)

Lo hecho en el código 4.17 es una etiquetación manual de elementos, dicha etiquetación responde a la oración 24a y al código 4.13. Bajo esta percepción podemos obtener los valores del árbol original, sin tener el equivalente del código 4.13. Lo importante es que podemos describir los elementos de este tipo de oraciones.

```

1 (DOP sentence-tr (v arg1 arg2 &optional aux &rest obl)
2   (PAIR arg1
3     (PAIR aux
4       (PAIR v
5         (PAIR arg2
6           (PAIR obl nil))))))

```

Código 4.18: Operador que construye oraciones transitivas.

El código 4.18 recibe las partes de una oración y las ordena, evalúa los elementos y devuelve una lista con la estructura superficial de la oración. No es un

operador inteligente, sólo respeta las pocas reglas con las que se ha estructurado, es decir, el orden básico de la lengua, identifica el verbo, sus argumentos centrales, optativamente identifica la presencia de un auxiliar verbal y si existieran más argumentos en el operador los identificará como oblicuos. Es importante hacer notar que la expresión *&optional* precede a los argumentos que son optativos y la expresión *&rest* identifica a otros argumentos. De tal forma, podemos hacer de forma similar una oración interrogativa como la que venimos examinando.

```

1 (DOP question (v arg1 arg2 aux &optional wh &rest obl)
2   (SETQ arg2 wh)
3   (PAIR arg2
4     (PAIR aux
5       (PAIR arg1
6         (PAIR v
7           (PAIR obl nil))))))

```

Código 4.19: Operador que construye oraciones interrogativas.

El código 4.19 únicamente realiza los movimientos necesarios respecto a 4.18 para conseguir la estructura de pregunta. Lo anterior pretende mostrar la posibilidad de realizar los mismos procedimientos siempre ligados a dos cosas: el detalle necesario en la cada demostración y el origen de los datos. Los dos códigos anteriores expresan lo mismo que las líneas del árbol 4.10, y si se evaluara valor por valor las declaraciones del código 4.17, obtendríamos valores muy similares a los de dicho árbol<sup>12</sup>. Estas últimas representaciones aquí mostradas pueden considerarse explicativas, pero podrían ser de utilidad si consideramos posibles aplicaciones al conocimiento lingüístico como el procesamiento del lenguaje natural, en realidad, en este lenguaje enfocado a la lingüística numerosas

<sup>12</sup>Evidentemente, no podemos decir que obtendríamos los mismos valores debido a que la estructuración de la información es distinta, sería equivocado decir que, sin mayor argumento, el simple acomodo visual de la información es suficiente para dicha equivalencia.

#### 4. PROPUESTA

---

validaciones enfocadas a los tipos de datos serán innecesarias, lo cual resulta en códigos con análisis más sencillos.

Hasta aquí consideramos que hay suficientes ejemplos que muestran las principales características de nuestro lenguaje, adicionalmente es posible encontrar algunos de los ejemplos originales al diseñar el lenguaje en el apéndice B.

A lo largo de este trabajo se han examinado diferentes representaciones formales que son utilizadas en la ciencia lingüística con el fin de encontrar una posible representación única para todas ellas. Esta representación permitiría agilizar el intercambio de conocimiento científico, incluso si se tratara de personas provenientes de otros campos del conocimiento. Puntualmente este trabajo pretendía:

- Estudiar lenguajes formales existentes en la lingüística y en otras ciencias.
- Estudiar las necesidades de la lingüística en cuanto a la representación formal de fenómenos.
- Elaborar la propuesta de lenguaje.
- Representación de algunos ejemplos.

Después de nuestro análisis podemos decir que las representaciones actuales son claras, siempre y cuando se tenga un conocimiento preciso sobre el área.

## 5. CONCLUSIÓN

---

También hay que tener a disposición un amplio abanico de representaciones para echar mano de la más adecuada en cada situación, esto multiplica los elementos a tener en cuenta al interpretarlas.

Es indiscutible la existencia de una necesidad por parte de la lingüística para crear representaciones formales; esto es visible en la constante adopción de representaciones consolidadas en otras ciencias y su adaptación que en algunas ocasiones logran alejar bastante a la representación de su forma de uso original, como es el caso de las matrices.

La innovación nunca ha sido bien vista. Todo el mundo quisiera quedarse donde está, es por eso que han llamado innovación a pequeños cambios que *mejoran* la forma de hacer ciertas actividades pero no cambian la actividad en sí, eso no es innovación. La innovación siempre será fuertemente destructiva y no *mejor*. Los primeros vehículos de combustión interna presentaban muchos inconvenientes respecto a un vehículo de tracción animal, los primeros teléfonos móviles quizás ni siquiera merecían el adjetivo de *móviles*. Todas las ventajas que hoy disfrutamos al usar dichas innovaciones son producto de un refinamiento constante llevado a cabo durante generaciones.

El lenguaje aquí presentado trae a la lingüística un beneficio que las representaciones actuales no tienen: favorece la creación de métodos de verificación y repetición experimental más efectivos. La estructura del lenguaje es pertinente para la creación de interpretes automáticos que validen los estudios lingüísticos de forma rápida y confiable siempre y cuando los lingüistas proporcionen el material de investigación completo.

El lenguaje, también tiene un primer inconveniente significativo: En este momento es demasiado mínimo. Aunque se dejó así con la intención de no perder los procedimientos teóricos que hacen efectivo al lenguaje y poder mostrar sus posibilidades, es imposible proceder día a día con un lenguaje que sólo tiene primitivas indispensables. Afortunadamente, la ciencia es acumulativa y con el tiempo, la creación de operadores básicos que eviten la anidación de los llamados

---

*operadores fundamentales* será inevitable. Por ejemplo, una lista se realizará con un operador **LIST** y no con operadores **PAIR** anidados. Operadores que seleccionen los primeros y últimos  $n$  elementos de una lista, el operador **SUSTITUTION** podrá reemplazar cualquier elemento de un par y no solo el primero; esto solo por mencionar algunos casos.

```
1 (DOP sentence-tr (v arg1 arg2 &optional aux &rest obl)
2   (LIST (arg1 aux v arg2 obl)))
```

Código 5.1: Comparese este operador con 4.18, realizan la misma función.

A pesar de ello, no hay nada que no sea solucionable mediante el refinamiento del lenguaje ya que es totalmente extensible y ampliable, su principio es básico y fácil de entender, puede expresar la estructura más compleja imaginable, es más rápido escribir en él que crear cualquier otra forma gráfica. Permite conservar orden y procedimientos lingüísticos, aunque su principal cualidad radica en ser multipropósito sin llegar a ser de propósito general. Esto significa que el conocimiento representado no requiere validaciones diferentes a las lingüísticas, no tiene elementos ajenos al uso lingüístico y tampoco se necesitan amplios conocimientos para iniciar su escritura, todo esto basado en un solo fundamento: *pares de celda*. Es decir, este lenguaje puede embotellar el conocimiento lingüístico y facilitar su uso en otras áreas, incluso en la creación de aplicaciones de uso general, como la brevemente descrita en el apéndice A.

Por supuesto, al decir que es fácil de aprender no significa descartar esfuerzo alguno. El lector lingüista atento en los ejemplos mostrados en el capítulo 4, observará que se abordan los puntos más sobresalientes del análisis de los métodos formales del capítulo 3, sin embargo también sabrá que hace falta llevar un análisis lingüístico particular para estar completamente seguro de su usabilidad. Otro lector no lingüista verá que el lenguaje podría ser mucho más útil si estuviera acompañado de un interprete de computadora, como cualquier otro lenguaje formal moderno. Esto último para algunos conforma el trabajo a un

## 5. CONCLUSIÓN

---

futuro inmediato, con vísperas a fomentar el uso del lenguaje.

Si retomamos el párrafo anterior veríamos que ambos lectores tendrían razón y a pesar de ello, tanto en el capítulo 2 como en el capítulo 4 se pueden observar dos cosas: 1) El lenguaje tiene principios lingüísticos claros, encaja muy bien con los métodos formales de la lingüística porque no se ha inventado nada nuevo, la idea de este lenguaje se basa en la misma idea que influenció a Noam Chomsky a hacer sus teorías lingüísticas, la misma que influenció a John McCarthy a crear la teoría que hoy conocemos como *inteligencia artificial* y que ha influido notablemente en la llamada lingüística computacional. 2) El lenguaje es computable, un interprete sería por supuesto un gran aporte, pero su intención no es crear un lenguaje para computadora, su intención es unificar las representaciones de la lingüística formal.

Respecto a los niveles de análisis ejemplificados en este texto, es notable no haber incluido la semántica. Mientras que los niveles anteriores son bastante consistentes entorno a sus explicaciones, en este trabajo se ha llegado a la conclusión de que en la semántica las explicaciones formales más convincentes provienen de los niveles anteriores, por ejemplo, en la sección 3.3 se busca una posible relación entre las matemáticas y la posible *doble negación* del español. En lingüística es bien sabido que esa relación semántica entre estas palabras no son nunca una doble negación sino una concordancia. La concordancia es un fenómeno propiamente sintáctico, no semántico. A opinión de este autor, los métodos representables por el lenguaje aquí expuesto no nacen en niveles de análisis superiores a la sintaxis y aunque es posible encontrar casos de uso, resultaría dudoso extender el lenguaje hasta esos niveles. Niveles como la semántica, guardan el pequeño detalle de ser mucho más cualitativos en los análisis porque es bien sabido que «el significado lingüístico no agota el significado comunicado».

Por otro lado, en los ejemplos presentados en este trabajo podemos observar las funciones básicas de representación de un lenguaje formal: en fonología nos enfocamos en la representación de las unidades, cómo escribir los datos lin-



---

güísticos básicos y el detalle que pueden tener; En morfología nos enfocamos en explicar las partes procedimentales de la lingüística, donde las explicaciones definidas por procesos descritos paso a paso pueden escribirse en el lenguaje, porque todo proceso formal en lingüística como en cualquier otra ciencia tiene ese adjetivo por ser recursivo y no por el lenguaje en el que está representado. Posteriormente abordamos en la sintaxis cómo los supuestos formales<sup>1</sup> de las teorías lingüísticas también son representables, aunque no sean partes procedimentales, ya que nuestro lenguaje incluye una representación declarativa basada en el cálculo lambda.

En lo que respecta a la estructura interna del lenguaje hay pocas aclaraciones que hacer, sin embargo, un lector curioso podría encontrar una contradicción entre el tratamiento de un lenguaje mínimo y la presencia de dos operadores comparativos: **EQ** y **STRING-EQ**. Las cadenas de son tratadas *tradicionalmente* en los lenguajes formales como arreglos de caracteres. En términos de estructuras de datos esto permite colocar un índice a cada elemento de la cadena. Lo anterior facilita la manipulación del dato y la implementación técnica de un interprete automático. En este trabajo, se considero pertinente tomar en cuenta el punto anterior a pesar que dista del punto de vista lingüístico, dicho esto, también ayuda en la creación de otros operadores como **SUBSTRING**.

Finalmente es importante hacer referencia al operador **LOOP** como único operador de ayuda para la definición de ciclos iterativos. Este operador únicamente abarca un ciclo general y no impide particularizar, si es necesario, otros operadores de este tipo. De tal forma, podemos definir otros operadores iterativos particulares presentes en la mayoría de los lenguajes formales como son: *while, for, repeat, etc.*

---

<sup>1</sup>Al decir *supuesto formal* pretendemos incluir aunque no limitarnos a los posibles axiomas y corolarios que necesite una teoría lingüística para funcionar.



# La lingüística en la Sociedad de la Información y el Conocimiento

---

A

En las últimas décadas del siglo XX el mundo occidental se compactó por medio de numerosos procesos económicos, políticos, sociales, culturales y tecnológicos. El conjunto de todos estos procesos se sintetizó bajo el término de Globalización (G11N). A pesar de escuchar comúnmente que estuvimos en la *era de la G11N*, debido a que en casi todos los sectores de la sociedad occidental trataron de expandir su influencia fuera de sus fronteras nacionales para estar presentes en otras partes del mundo, no es la G11N un proceso que defina como tal el paradigma por el cual se mueve la actual sociedad occidental. Globalización.

Encontramos por otro lado un concepto mucho menos difundido, no por eso menos estudiado, que si cumple con la tarea que mencionamos en el párrafo anterior: *Sociedad de la Información y el Conocimiento (SIC)*. Según Jaques Delors la SIC se define de la siguiente manera: Sociedad de la Información y el Conocimiento.

Es una forma de desarrollo económico y social en el que la adquisición, almacenamiento, procesamiento, evaluación, transmisión,

distribución y la diseminación de la información con vistas a la creación de conocimiento y a la satisfacción de las necesidades de las personas y de las organizaciones, juega un papel central en la actividad económica, en la creación de riqueza y en la definición de la calidad de vida y las prácticas culturales de los ciudadanos. [Comisión Europea, 1993]

La SIC toma en cuenta como aplicaciones y servicios fundamentales los siguientes<sup>1</sup>:

**Educación y formación a distancia** Por definición, la educación a distancia es el suministro de programas educacionales y sistemas de aprendizaje a través de medios electrónicos. La educación a distancia se basa en el uso de una computadora u otro dispositivo electrónico para proveer a las personas de material educativo.

**Comercio electrónico** Es una nueva esfera del comercio de las mercancías que atraviesan las fronteras por medios electrónicos. En términos generales, es la producción, publicidad, venta y distribución de productos a través de las redes de telecomunicaciones. Los ejemplos más evidentes de productos distribuidos electrónicamente son los libros, música y vídeos transmitidos a través de líneas telefónicas o de Internet.

**Teletrabajo** La realización de cualquier trabajo en un lugar distinto del contratante y utilizando en cualquiera de las fases, las TICs para comunicarse o desarrollar una actividad.

---

<sup>1</sup>Las aplicaciones y servicios de la SIC están basadas en alguna Tecnología de la Información y la Comunicación (TIC), que en conjunto son aquellas herramientas computacionales e informáticas que procesan almacenan, sintetizan, recuperan y presentan información representada de la más variada forma. Es un conjunto de herramientas, soportes y canales para el tratamiento y acceso a la información. Constituyen nuevos soportes y canales para dar forma, registrar, almacenar y difundir contenidos informacionales[Domínguez, 2010].

---

**Telemedicina** Es la medicina practicada a distancia, incluyendo diagnóstico y tratamientos. Es un recurso tecnológico que posibilita la optimización de los servicios de atención en salud, ahorrando tiempo, desplazamientos innecesarios y facilitando atención de especialistas en zonas distantes.

**Gobierno digital** Es la selección, implementación y uso de TICs en el gobierno para la provisión de servicios públicos, el mejoramiento de la efectividad gerencial, y la promoción de valores y mecanismos democráticos, así como el desarrollo de un marco regulador que facilite iniciativas que usan información de forma intensiva y fomente la sociedad del conocimiento.

**Entretenimiento en línea** También llamado ocio electrónico, es uno de los mercados que más ha crecido en los últimos años. El entretenimiento en línea abarca el campo de los juegos de vídeo, redes sociales, equipos y productos multimedia como los teléfonos celulares, reproductores de música entre muchos otros.

**Información en línea** La gestión de la información en línea por medio de bases de datos ha evolucionado desde una aplicación informática especializada hasta una parte esencial de las TICs en un entorno moderno y, como resultado, el conocimiento acerca de los sistemas de bases de datos se ha convertido en una parte esencial en la enseñanza de la informática.

**Servicios de control y control supervisor** Son un tipo de sistemas que utiliza una computadora convencional en una aplicación de control de procesos y donde un Controlador lógico programable (PLC) desarrolla las funciones de control pero que son vigiladas y supervisadas por una computadora.

Dentro de la SIC es importante que una comunidad tenga la capacidad para comunicarse con un entorno global y no solo de forma interna, esto es posible gracias a las TICs. Esta capacidad de comunicación es usada para la generación y difusión de contenidos que la propia comunidad y el individuo que pertenece a

ella consideren pertinente. La creación, consulta y difusión de estos contenidos así como el uso de las TICs no son el fin en si mismo, la verdadera finalidad es el fomento del desarrollo social, económico y cultural de los países así como el bienestar y la mejora en la calidad de vida de los individuos.

Bajo este marco de la SIC podemos ver que más allá de la tecnología siempre encontraremos una misma tendencia: el lenguaje humano.

Comunicación y lenguaje.

Los medios de comunicación convencionales así como las TICs tienen un proceso evolutivo común, inician con fuentes escritas que son accesibles únicamente a personas con cierto grado de alfabetización. Después agregan el lenguaje oral que los hace accesibles a toda persona que hable la lengua, esto afianza la comunicación y, completan el ciclo con la transmisión de vídeo que permite la creación de un contexto completo, lo más similar a una conversación persona a persona. Así, no es que las tecnologías actuales imiten el lenguaje humano sino que lo transmiten cada vez de forma más fiel, porque es la forma de transmisión de contenidos más efectiva para el ser humano.<sup>2</sup>

Todos estos medios de comunicación (ya sea convencionales o TIC) han sido fuertemente utilizados en el ámbito comercial por un gran número de empresas. Deseando crear una imagen global de sus productos para que sean reconocidos en cualquier lugar están facilitando (conscientemente o no) una posible solución a un problema al que se han enfrentado los países plurilingües: Las diferencias interculturales.

En esta situación, la labor del lingüista se vuelve fundamental, porque en esta disciplina es donde convergen el lenguaje, la sociedad y la información, que son tres ejes fundamentales para un comercio exitoso. La mala noticia en el terreno científico de la lingüística es que el aspecto de las lenguas se ve reducido

---

<sup>2</sup>Ya en 1996 Nicholas Negroponte hablaba de que cualquier TIC tenía como rumbo la personalización y no el lenguaje natural. Por ejemplo, una computadora no tiende a establecer una comunicación por medio del lenguaje natural con el usuario, pero si tenderá a presentarle información cada vez más personalizada, a tal punto que le será complicado diferenciar la información útil y la publicidad.[Negroponte and Plaking, 1996]

---

a la traducción publicitaria o la transferencia de un mensaje y esto acarrea que los conceptos que podrían ser útiles no funcionen enriquecer la ciencia pues tienen visperas a la creación de productos, aquí los tres conceptos en traducción libre[Fry et al., 2003]:

**Globalización (G11N)** Proceso general de integración económica, política, tecnológica y social en todo el mundo; El proceso de hacer todo lo necesario en materia técnica, financiera, de gestión, de personal, marketing, y otras decisiones de la empresa necesarios para facilitar la localización.

**Internacionalización (I18N)** Proceso para garantizar a nivel técnico y de diseño que un producto puede ser introducido en un mercado fácilmente.

**Localización (L10N)** Proceso de adaptación de un producto para satisfacer los requisitos de un mercado de destino específico.

De las definiciones anteriores podemos hacer varias observaciones; la primera es que la G11N ya no se refiere a los procesos a los que aludíamos al inicio de este apartado, la segunda observación es que estamos hablando en general de un proceso técnico y logístico y finalmente que las definiciones de I18N y G11N son necesarias para satisfacer la última, L10N.

Veamos ahora qué es L10N, cómo se lleva a cabo la parte lingüística para recuperar y aprovechar esta metodología<sup>3</sup>.

La L10N implica «la modificación de un producto para que este sea compatible y responda a las normas culturales del mercado al que se destina incluyendo la traducción o adaptación en cuanto a las equivalencias lingüísticas» [Velasco Quintana, 2014]. Siguiendo la cita anterior, existen tipos básicos de L10N que nos permiten seguir brevemente el proceso de adaptación:

---

<sup>3</sup>El proceso de localización en el ámbito de la creación de software está documentado en [Velasco Quintana, 2014], una metodología para crearlo está explicada en [Domínguez, 2010, Cap. 5.5]

**L10N cultural** Se tienen en cuenta las convenciones culturales de un lugar (historia, costumbres, etc.)

**L10N nacional** Abarca la adaptación de los alfabetos convenciones de símbolos y formatos, no limitándose solo a la traducción de la interfaz de usuario.

**L10N técnica** Engloba aspectos como la compatibilidad entre sistemas.

Más adelante, la misma Velasco Quintana resalta la importancia de un análisis lexicográfico previo y señala:

Dado que es frecuente que los productos que hay que localizar atañan a temas innovadores para los que todavía no se ha establecido la terminología en la lengua meta, la terminología debe crearse antes de iniciar el proceso de localización; esto es necesario no solo por motivos de eficiencia, puesto que en los proyectos de localización es frecuente que los plazos sean ajustados y participe un número elevado de personas, sino también porque hay que mantener la consistencia en la terminología.

“En todo proyecto, el traductor necesita un kit de localización, un paquete que (idealmente) incluye los archivos que se deben localizar, las herramientas necesarias para hacerlo, una guía de estilo, una memoria de traducción, un corpus de proyectos anteriores y glosarios específicos”. [Velasco Quintana, 2014]

En estas citas podemos ver que un lingüista fácilmente interpretaría que para hacer el trabajo de L10N de forma correcta es necesario tener previamente, una gramática y de preferencia un diccionario de la lengua destino. Además es necesario la elaboración de una etnografía para poder satisfacer las exigencias de la L10N cultural y nacional.



---

Es innegable que la L10N no es un proceso lingüístico, pero las labores lingüísticas si son indispensables para que éste se realice. La L10N es una necesidad creada por la tecnología habilitadora de la SIC, es decir el conjunto de TICs y quizás más que nunca la lingüística tenga un papel fundamental en el modelo social.

Sin embargo, los pocos alcances de la L10N pueden llevar al lingüista a preguntarse qué pasaría si este concepto se extrae del marco económico y lo usamos en lingüística con el pensamiento de que ésta ciencia es indispensable en países plurilingües para realizar entre otras cosas, planes de acción que fomenten la SIC, debido a que:

Podemos afirmar que se está creando un nuevo nivel de realidad, la realidad virtual, que nos permite realizar interacciones comunicativas equivalentes a las que tienen lugar en la realidad presencial, aunque desligadas de muchos de los condicionamientos que nos imponen los parámetros espacio-tiempo. Ahora bien, el nuevo entorno comunicativo que deriva de la aplicación de estas nuevas tecnologías plantea nuevos problemas: el acceso a la información debe ser eficaz, rápido y sencillo, y hay que prever la posibilidad de que los usuarios cometan errores. También es deseable que en el mismo entorno se integre la voz, la imagen y la lengua escrita.

La aparición de Internet como red de telecomunicaciones y de la WWW cambia el concepto de lo que se considera información: no está mejor informado quien tiene más datos, sino quien dispone de los mejores medios para obtener exclusivamente los que necesita.  
[Martí and Llisterri, 2001]

Una de las labores fundamentales para la lingüística es la clasificación de las lenguas, probar que existen nexos entre las lenguas en diferentes partes de sus

estructuras es una labor tan funcional y a que va a la par de la descripción del sistema de una lengua particular.

La concepción sintética del objeto [la lengua] se concreta en diferentes modelos de clasificación. Dichos agrupamientos de lenguas son, fundamentalmente, de naturaleza geográfica, si el criterio clasificatorio es espacial; genética, si es temporal y distribuye a las lenguas en familias originadas a partir de un ancestro común; tipológica, si el criterio es formal, fundamentado en la concordancia estructural, independientemente de los aspectos espacio-temporales; y areotipológica, si se combina el criterio geográfico con el tipológico y los rasgos compartidos se explican por el fenómeno de difusión. Así, pues, las lenguas pueden ser ordenadas por regiones geográficas, en familias, en tipos o en áreas lingüísticas, siendo estos ordenamientos esencialmente diferentes.[Puente, 1998]

Ante un mundo plurilingüe es complicado compartir información en algunas zonas de diferentes países, si bien la densidad de la penetración de TICs es necesaria, es aun más prioritario la necesidad de facilitar el acceso a la información en las diferentes lenguas. Gracias a los estudios tipológicos tenemos acceso a diferentes mapas que dejan ver la ubicación de las comunidades que necesitan estos accesos, satisfacerlos, lengua por lengua es imposible.

Según Adab «one feature of this new global culture is a tendency to destroy or at least seek to minimize, intercultural differences». Visto de esta manera, el contacto lingüístico y cultural entre las comunidades siempre ha tenido esta consecuencia y el contacto lingüístico se da entre las lenguas indígenas y la lengua mayoritaria por supuesto, pero podemos dar por sentado que se da entre lenguas indígenas también. Ese contacto entre lenguas indígenas lleva a pensar que hay ciertos eventos lingüísticos que ocurren entre lenguas de forma independiente a

---

las familias a las que perteneces, siempre y cuando tengan un contacto geográfico, por lo que una clasificación aerotipológica es un buen punto de partida.

Estas comunidades en contacto comparten lengua y cultura porque los pueblos se acercan siempre que «existe un grado de reconocimiento mutuo en cuanto a valores y principios entre las sociedades»[Ricoy, 2007].

Sin embargo, contrario a lo que la L10N propone:

Hay temas que, aunque estén en el marco de la legalidad, transgreden los límites de lo aceptable en algunas culturas. Incluso los tabúes que pueden considerarse *universales* son más admisibles para ciertas comunidades que para otras.[Ricoy, 2007]

Y tratar de ignorar esas diferencias nos llevaría a propiciar un conflicto al momento de compartir información:

...la introducción de elementos culturales ajenos tiene un efecto destabilizador en las comunidades y puede percibirse como una intrusión en las que no comparten, o que simplemente rechazan, los valores del Otro.[Ricoy, 2007]

Una vez que tenemos en cuenta estos elementos podemos decir que *dado una clasificación aerotipológica de un territorio es posible establecer regiones donde dos o más comunidades (pertenecientes al territorio) tienen elementos culturales y lingüísticos comunes que facilitarían la transferencia de información en al menos un aspecto de su realidad social (económica, política, etc).* Esto es Regionalización Lingüística (R25A). Regionalización Lingüística.

Con la R25A se busca que por medio de regiones se logren poner a la disposición de las comunidades los servicios de la SIC (desde los servicios gubernamentales básicos hasta cualquier TIC) si bien no en la lengua madre, si en una lengua que tenga mayor relevancia en el aspecto social específico de la región.

Es de esperarse que la R25A identifique centros difusores de cultura desde donde se trasladan los factores que crean la región, que las lenguas involucradas estén en muchos casos emparentadas (aunque no es necesario), y que una región se limite en muchas ocasiones a una sola actividad social. La búsqueda de regiones es entonces una consecuencia natural para la lingüística pues los estudios sobre la realidad de la lengua en una zona se limitan frecuentemente al análisis de una sola actividad pues un estudio holista es demasiado ambicioso.

Entonces para realizar la R25A se necesita un estudio aerotipológico de la zona, seguido de un fino estudio etnográfico que pueda impulsar estudios específicos sobre el uso de la lengua en circunstancias definidas tomando en cuenta los factores plurilingües, migración y prestigio de las lenguas, los cuales ayudarán a elaborar planes de acción sociales, planificación, creación de alfabetos prácticos con fines de desarrollo dentro de la SIC; estamos hablando de estudios sociolingüísticos.

## Algunos de los ejemplos originales del desarrollo del lenguaje

---

B

A continuación presento algunos de los ejemplos y estructuras originalmente pensadas para el lenguaje. Estos ejemplos corresponden a un primer borrador, donde al lenguaje se le denominaba LANPRO o Language Processor.

Este apartado es importante si se quiere tener un poco más de conocimiento de cómo fue ajustándose la idea, debido a que se podrán encontrar errores de equivalencia entre estructuras, derivados principalmente a la influencia directa de las representaciones de algunos lenguajes LISP ya existentes.

### **B.1. Equivalencia de estructuras arbóreas binarias**

Para ejemplificar el uso de LanPro para describir una lengua, nos basaremos en modelos con los que posiblemente este familiarizado el lector, primero tomaremos la estructura básica de la teoría X-Barra y en seguida mostraremos la

misma estructura representada por un árbol de pares de celda:

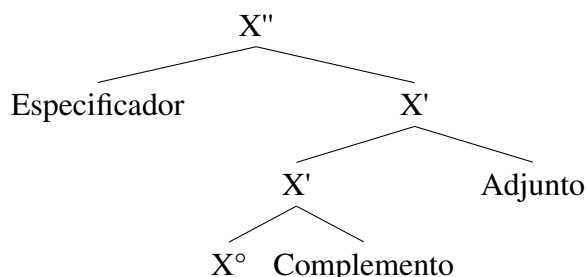


Figura B.1: Estructura arbórea de un sintagma, teoría X-barra

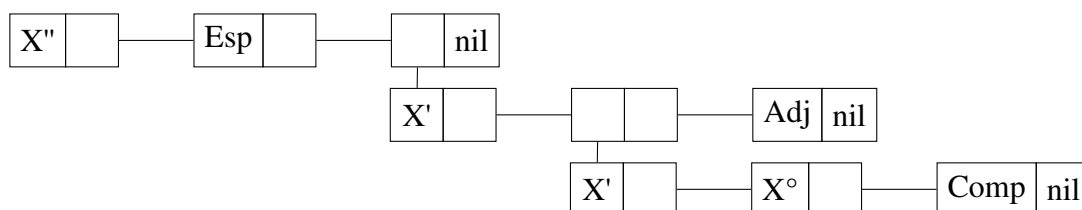


Figura B.2: Estructura arbórea de un sintagma, LanPro

La estructura arbórea de la figura B.2, ya no es propiamente un árbol binario en LanPro (si lo fuera, tendríamos que conservar únicamente las hojas del árbol), pero si es la estructura que posee los mismos elementos y características que el original<sup>1</sup>; además puede representarse con pares de celdas para obtener una secuencia plana de la siguiente manera:

(26) (X'' . (Esp . ((X' . ((X' . (X° . (Comp . nil))) . (Adj . nil))) . nil)))

Ahora por partes iremos simplificando (26) en listas para quitar los puntos y paréntesis de algunos pares de celda. La primera de ellas son los tres elementos más profundos en el árbol:

<sup>1</sup>El objetivo de este apartado no es crear una teoría nueva a través de LanPro, sino mostrar una equivalencia de nomenclatura con la teoría X-barra; por supuesto, LanPro puede formalizar cualquier teoría actual o futura que sea computable.

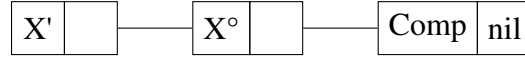


Figura B.3: Primera parte de la figura B.2

$$(27) (X' . (X^\circ . (Comp . nil))) = (X' X^\circ Comp)$$

Continuamos sustituyendo la lista obtenida en la estructura de pares de celda inicial y repetimos la misma simplificación:

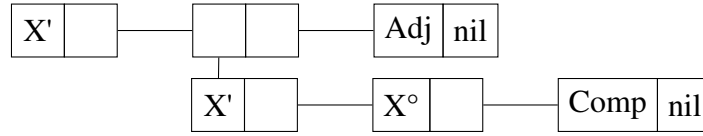


Figura B.4: Segunda parte de la figura B.2

$$(28) (X' . ((X' X^\circ Comp) . (Adj . nil))) = (X'(X' X^\circ Comp) Adj)$$

Hacemos este procedimiento por última vez para obtener una estructura simplificada de todo el árbol donde la estructura de pares anidada ya no es evidente:

$$(29) (X'' . (Esp . ((X' (X' X^\circ Comp) Adj) . nil))) = (X'' Esp (X' (X' X^\circ Comp) Adj))$$

## B.2. Estructuras arbóreas en Fonología

El modelo autosegmental-jerárquico ha usado una estructura arbórea no binaria para explicar diversos fenómenos de la lengua y el diseño de LanPro puede simplificar cualquier estructura compleja con relativa sencillez, la figura B.5 ilustra una estructura arbórea básica y en 30a la estructura plana equivalente en LanPro.

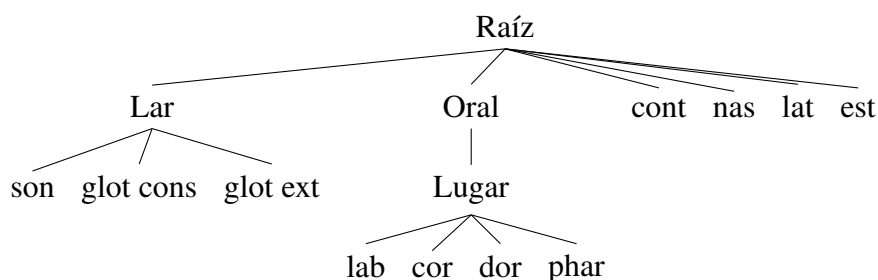


Figura B.5: Árbol de rasgos según el modelo autosegmental-jerárquico

- (30) a. (Raíz (Lar son g.cons g.ext) (Oral (Lugar lab cor dor phar)) (cont nas lat est))
- b. ((reson cons aprox) (Lar son g.cons g.ext) (Oral (Lugar lab cor dor phar)) (cont nas lat est))

Para completar al máximo esta estructura general, podemos decir que el nodo raíz de la figura B.5 está compuesto por los rasgos resonante, consonántico y aproximante, por lo que podemos sustituir claramente el símbolo “/” con una lista con estos elementos como en (30b) y podemos hacer lo mismo expandiendo las hojas para poder colocar los rasgos pertinentes.

Ahora con base en esta lista podemos representar los sonidos de una lengua escribiendo la lista ordenada de rasgos:

- (31) a. ((+cons) (Lar -son) (Oral (Lugar lab cor)) (-cont))  
Es una consonante bilabial oclusiva sorda (p)
- b. ((-reson -cons) (Lar g.ext) (+cont))  
Es la glotal fricativa sorda (h)
- c. ((-reson +cons) (Lar -son g.ext) (Oral (Lugar lab)) (+cont))  
Es una consonante bilabial oclusiva sorda glotalizada (p<sup>h</sup>)



- d. ((+reson +cons) (Lar +son) (Oral (Lugar cor)) (+cont))

Es una consonante coronal, vibrante múltiple (r)

- e. ((+reson +cons) (Lar +son) (Oral (Lugar cor)) (-cont))

Es una consonante coronal, vibrante simple (r)

### B.3. Ejemplo de descripción de un fenómeno

La descripción de un fenómeno en LanPro se ejecuta normalmente de forma interpretativa e interactivamente. En su forma más sencilla, un operador o un proceso lingüístico se presenta como una expresión completamente puesta entre paréntesis con todos los operadores necesarios en la forma prefija. Todas las variables tienen valores átomos o listas. A continuación iremos desarrollando la estructura de la descripción del *Principio de Contorno Obligatorio (PCO)* que es tan importante en Fonología porque ayuda a entender muchos comportamientos en las lenguas.

El PCO establece:

- (32) Se prohíben elementos idénticos adyacentes [Núñez et al., 1999]

Entonces el primer paso natural es saber si dos elementos adyacentes violan este principio, para eso examinaremos los efectos de el *PCO* sobre la palabra [karreta]. Estos efectos son conocidos ya que, de dos segmentos idénticos e individuados, obtendremos la unión múltiple de dos posiciones en el esqueleto con un segmento [Núñez et al., 1999].

Para saber si dos elementos adyacentes incumplen (32) necesitamos primero definir los segmentos a probar, en este caso probaremos [r] que aparece en la palabra [karreta].

Este segmento ya lo hemos definido anteriormente en la sección B.2 y la teoría explicada en la sección 4.3, por lo que procedemos a realizar un operador

## B. ALGUNOS DE LOS EJEMPLOS ORIGINALES DEL DESARROLLO DEL LENGUAJE

---

sencillo el cual tomará como argumentos dos segmentos y los comparará, si incumple el *PCO* regresará *t*, *nil* en otro caso:

```
1 (DOP pco (arg1 arg2)
2   "Prueba si se incumple el principio de contorno
   obligatorio"
3   (COND ((EQUAL arg1 arg2))))
```

En el operador definido arriba, **COND** se traduce como una sentencia if con la que disponen los lenguajes formales y **EQUAL** es la igualdad, no se usa el símbolo de igual (=) para poder hacer la distinción con las operaciones aritméticas.

La realización en nuestra descripción es la siguiente:

```
1 (PCO r r)
2 t
```

Hasta el momento, el operador **PCO** funciona solo para identificar segmentos que incumplen (32). Ahora necesitamos que el operador **PCO** no solo identifique, sino también que haga la asociación de los segmentos:

```
1 (DOP pco (arg1 arg2)
2   "Realiza el principio de contorno obligatorio"
3   (COND ((EQUAL arg1 arg2) arg1)
4   ((LIST arg1 arg2))))
```

El nuevo operador **PCO** devolverá un solo segmento si los argumentos incumplen (32), pero si no es el caso devolvera los dos argumentos.

```
1 (PCO r r)
2 ((+reson +cons) (Lar +son) (Oral (Lugar cor)) (+cont))
3 (PCO (QUOTE r) (QUOTE r))
4 r
5 (PCO r h)
```

### B.3. Ejemplo de descripción de un fenómeno

```
6 ((+reson +cons) (Lar +son) (Oral (Lugar cor)) (+cont)) ((-reson -cons  
   ) (Lar g\.ext) (+cont)))  
7 (PCO (QUOTE r) (QUOTE h))  
8 (r h)
```

Arriba evaluamos varias veces **PCO** con resultados diferentes. En la línea 1 evaluamos los fonemas y el resultado es el valor del fonema resultante. En la línea 3 aunque hacemos lo mismo, en el argumento señalamos que no evalúe el símbolo al final. En la línea 5 y 7 colocamos ejemplos de dos segmentos que no incumplen (32) para que se haga un contraste entre resultados de nuestro operador.

Además hasta ahora el operador definido cumple bien (32) en la lengua española porque evaluará los segmentos [i] y [y] como diferentes a pesar de que ambas son corales, esto permite que la palabra silla se realice como [siya] tal y como aparece en varios dialectos hispanoamericanos [Núñez et al., 1999].

Sin embargo, Nuñez también señala que en tigrinia, una lengua Eritrea<sup>2</sup> la forma /baräk-ka/ “bendito seas” se realiza como [baräx-ka], pero las formas tautosilábicas surgen inalteradas como es el caso de /fäkkärä/ que se realiza como [fäkkärä]. Nuñez señala que el motivo de la inaplicación es “la distinción fonológica que subyace estas secuencias de segmentos”. Lo que si es un hecho es que el *PCO* debe realizarse de manera diferente en tigrinia que en español y debemos modificar el operador **PCO** para que cumpla con lo que sucede en la lengua.

```
1 (DOP pco (arg1 arg2 &optional arg3)  
2   "Realiza el principio de contorno obligatorio"  
3   (COND ((NULL arg3) (COND ((EQUAL arg1 arg2) arg1)  
4     ((LIST arg1 arg2))) )
```

<sup>2</sup>El Estado de Eritrea es un país situado al noreste de África. Limita al norte y al oeste con Sudán; al sur con Etiopía y Yibuti; el este del país posee una extensa costa con el mar Rojo. Su nombre proviene del griego “eritros”, que quiere decir “rojo”. Se independizó en 1993, lo que lo convierte en uno de los estados más jóvenes del mundo. Su capital y ciudad más poblada es Asmara.

## B. ALGUNOS DE LOS EJEMPLOS ORIGINALES DEL DESARROLLO DEL LENGUAJE

```
5      ((COND ((EQUAL arg1 arg2) (LIST arg3 arg2))
6      ((LIST arg1 arg2))))))
```

Esta última versión de **PCO** agrega un argumento opcional que permite describir qué es lo que pasa en tigrinia respecto al *PCO* únicamente y no a los contextos en los que se evita el principio, porque Nuñez no da detalles suficientes como para definir las características de estos segmentos tal y como hiciéramos con los fonemas. De cualquier modo, esas definiciones no afectan **PCO** sino los contextos en los cuales podemos aplicarlos.

```
1  (pco k k x)
2  (((-reson +cons) (Lar -son) (Oral (Lugar dor)) (+cont)) ((-reson +cons
3  ) (Lar -son) (Oral (Lugar dor)) (-cont)))
4  (pco (QUOTE k) (QUOTE k) (QUOTE x))
5  (x k)
```

### B.4. Presentación del documento

Finalmente debemos resumir cómo sería la descripción limpia del *PCO*. Esta descripción consta de tres apartados, el primer apartado contendrá todas las declaraciones de los fonemas usados en la descripción, de hecho este primer apartado debe obedecer completamente a una descripción de los fonemas de la lengua con capacidad para ser reutilizado en diversos estudios.

```
1  ;; Fonemas existentes en la lengua
2  (SETQ r (QUOTE
3  ((+reson +cons)
4  (Lar +son)
5  (Oral
6  (Lugar cor))
7  (+cont))))
```

```

8
9  (SETQ h (QUOTE
10      ((-reson -cons)
11      (Lar g.ext)
12      (+cont))))
13
14
15  (SETQ k (QUOTE
16      ((-reson +cons)
17      (Lar -son)
18      (Oral
19      (Lugar dor))
20      (-cont))))
21
22  (SETQ x (QUOTE
23      ((-reson +cons)
24      (Lar -son)
25      (Oral
26      (Lugar dor))
27      (+cont))))
28
29  [...]
30
31  (PROVIDE (QUOTE fonemaEsp))

```

El segundo apartado contendrá las operaciones utilizadas en el análisis; en este caso constará únicamente de **PCO** en la última realización obtenida, ya que así podrá ser utilizada también en caso de realizar un análisis sobre la lengua tigrinia.

```

1  (DOP pco (arg1 arg2 &optional arg3)

```

## B. ALGUNOS DE LOS EJEMPLOS ORIGINALES DEL DESARROLLO DEL LENGUAJE

---

```
2  "Realiza el principio de contorno obligatorio"
3  (COND ((NULL arg3) (COND ((EQUAL arg1 arg2) arg1)
4                        ((LIST arg1 arg2))) )
5      ((COND ((EQUAL arg1 arg2) (LIST arg3 arg2))
6      ((LIST arg1 arg2))))))
7
8  (PROVIDE (QUOTE pco))
```

Estos dos primeros apartados facilitan la organización, reutilizamiento, procesamiento y lectura de la información, a su vez que hacen la función de librerías dentro de un lenguaje formal.

Finalmente esta el apartado en el que se detalla al lector el análisis de lengua. Los comentarios sobre la lengua estarán precedidos por un punto y coma (;) distinguiéndolos así de los ejemplos funcionales. La descripción comenzará mencionando el nombre de los otros apartados a los que debe recurrirse para mayor detalle aunque también pueden aparecer en otros lugares a lo largo del análisis.

```
1  ; Apartados necesarios
2  (REQUIRE (QUOTE pco))
3  (REQUIRE (QUOTE fonemaEsp))
4
5  ; El Principio de Contorno Obligatorio establece el
   impedimento de la existencia de elementos
6  ; iguales adyacentes.
7
8  ; En nuestra lengua, los dos segmentos de la palabra
   [karreta] se realizan como uno solo
9  ; [kareta] de la siguiente manera:
10
11  (PCO (QUOTE r) (QUOTE r))
```

```
12  r
13  [...]
14
15  ; Sin embargo en otras lenguas como el tigrinia se realiza
    de la siguiente manera:
16
17  (PCO (QUOTE k) (QUOTE k) (QUOTE x))
18  (x k)
19
20  [...]
```





# Bibliografía

- [Allwood et al., 1977] Allwood, J., Andersson, L., and Dahl, O. (1977). *Logic in Linguistics*. Cambridge Textbooks in Linguistics. Cambridge University Press.
- [Arbiser, 2005] Arbiser, A. (2005). *Sistemas y subsistemas de sustituciones explícitas*. Text, Facultad de Ciencias Exactas y Naturales. Universidad de Buenos Aires.
- [Arís, 2015] Arís, M. (2015). *Métodos formales en Lingüística*. Claves de la lingüística. Síntesis.
- [Armstrong et al., 1995] Armstrong, D., Stokoe, W., and Wilcox, S. (1995). *Gesture and the Nature of Language*. Cambridge University Press.
- [Arroyo and Domínguez, 2013] Arroyo, J. and Domínguez, E. (2013). Formas de tratamiento dentro de la ENAH. *Clase de Etnografía de la Comunicación*.
- [Arroyo Méndez and Marchesi, 2005] Arroyo Méndez, D. and Marchesi, J. (2005). Un lenguaje que se aprende en 10 minutos I y II. *Mundo Linux*, (84 y 85):52–55.
- [Bell, 2004] Bell, M. (2004). *Understanding English Spelling*. Pegasus Educational.

- [Bentley, 1887] Bentley, W. H. (1887). *Dictionary and grammar of the Kongo language: as spoken at San Salvador, the ancient capital of the old Kongo empire, West Africa*, volume 1. Baptist missionary society.
- [Benveniste, 1971] Benveniste, É. (1971). *Problemas de lingüística general*. Number v. 1 in Lingüística (Siglo XXI Editores). Siglo XXI.
- [Bergareche, 1992] Bergareche, B. C. (1992). Negación doble y negación simple en español moderno. *Revista de Filología Románica*, 63(9).
- [Blecua, 1999] Blecua, J. (1999). *Filología e informática: Nuevas tecnologías en los estudios filológicos*. Seminari de Filologia i Informàtica. Editorial Milenio. S.L.
- [Bosque and Gutiérrez-Rexach, 2009] Bosque, I. and Gutiérrez-Rexach, J. (2009). *Fundamentos de sintaxis formal*. Akal universitaria: Serie Lingüística. Ediciones Akal.
- [Bromley and Lamson, 1987] Bromley, H. and Lamson, R. (1987). *LISP Lore: A Guide to Programming the LISP Machine*. Springer.
- [Burgos Cárdenas, 1997] Burgos Cárdenas, R. (1997). *Manual de Referencia Rápida de LISP*. Universidad de Concepción.
- [Cairó et al., 2006] Cairó, O., Guardati, S., and Osvaldo Cairó, S. G. (2006). *Estructuras de datos*. Number Sirsi) i9789701059081. Mc Graw Hill.
- [Castro Peña, 2016] Castro Peña, J. (2016). Modelos de Computación II. [En línea; acceso 16-Abril-2016].
- [Chamorean, 2009] Chamorean, C. (2009). *Hablemos purepecha, Wanté juchari anapu*. Fondo Editorial Morevallado.
- [Chassell, 2004] Chassell, R. (2004). *An Introduction to Programming in Emacs Lisp*. GNU manual. GNU Press.

- [Chomsky, 1965] Chomsky, N. (1965). *Aspects of the Theory of Syntax*. Massachusetts Institute of Technology. M.I.T. Press.
- [Church, 1936a] Church, A. (1936a). A note on the Entscheidungsproblem. *J. Symb. Log.*, 1(1):40–41.
- [Church, 1936b] Church, A. (1936b). An unsolvable problem of elementary number theory. *American journal of mathematics*, 58(2):345–363.
- [Comisión Europea, 1993] Comisión Europea (1993). *Crecimiento, competitividad, empleo - Retos y pistas para entrar en el siglo XXI - Libro Blanco*. Oficina de Publicaciones Oficiales de la Comunidad Europea.
- [Contreras and Lleó, 1982] Contreras, H. and Lleó, C. (1982). *Aproximación a la fonología generativa: Principios teóricos y problemas*. Anagrama.
- [Crosland, 1959] Crosland, M. P. (1959). The use of diagrams as chemical ‘equations’ in the lecture notes of william cullen and joseph black. *Annals of Science*, 15(2):75–90.
- [de conectividad para las Américas, ] de conectividad para las Américas, A. Plan de acción de quito. *Edición de marzo*, 5.
- [de la Corte, 2007] de la Corte, J. (2007). *Gramática moderna de la lengua española*. Limusa.
- [de Suárez, 1992] de Suárez, Y. (1992). *Sociolingüística para Hispanoamericanos: Una Introducción*. Centro de Estudios Lingüísticos y Literarios. El Colegio de México, Centro de Estudios Lingüísticos y Literarios.
- [Domínguez, 2010] Domínguez, E. (2010). Software libre: Tecnología para la evolución a la sociedad de la información. Tesis para obtener el grado de Ingeniero en Computación.

- [Dubois, 1979] Dubois, J. (1979). *Diccionario de Lingüística*. Alianza Diccio-  
narios. Alianza Editorial.
- [Dunham, 1999] Dunham, W. (1999). *Euler: The Master of Us All*. Number v.  
22 in Dolciani Mathematical Expositions. Mathematical Association of Ame-  
rica.
- [Everett et al., 2005] Everett, D., Berlin, B., Gonalves, M., Kay, P., Levinson,  
S., Pawley, A., Surralls, A., Tomasello, M., Wierzbicka, A., and Everett,  
D. (2005). Cultural constraints on grammar and cognition in piraha: Anot-  
her look at the design features of human language. *Current anthropology*,  
46(4):621–646.
- [Expert Advisory Group on Language Engineering Standards, 1996] Expert  
Advisory Group on Language Engineering Standards (1996). The essentials  
of eagles. [En línea: revisado Diciembre 2013].
- [Feferman, 1995] Feferman, S. (1995). *Kurt Gödel, Collected Works*. Oxford  
University Press.
- [Ferreiro, 1998] Ferreiro, E. (1998). *Alfabetización: teoría y práctica*. Siglo  
XXI.
- [Fry et al., 2003] Fry, D., Lommel, A., and Association, L. I. S. (2003). *The  
Localization Industry Primer*. LISA.
- [García, 2002] García, I. (2002). *Lenguaje de señas entre niños sordos de pa-  
dres sordos y oyentes*. PhD thesis, Tesis de Grado, Licenciatura en Lingüísti-  
ca. Universidad Nacional Mayor de San Marcos, Lima, Perú.
- [Glickstein, 2010] Glickstein, B. (2010). *Writing GNU Emacs Extensions: Edi-  
tor Customizations and Creations with Lisp*. O'Reilly Media.

- [Goldsmith, 1999] Goldsmith, J. (1999). *Phonological theory: the essential readings*/[edited by] goldsmith, j. a.
- [Gurovich and de México. Facultad de Ciencias, 2008] Gurovich, E. and de México. Facultad de Ciencias, U. N. A. (2008). *Introducción a la Teoría de la Computación*. Prensas de ciencias. UNAM, Facultad de Ciencias.
- [Harris, 1970] Harris, Z. S. (1970). From phoneme to morpheme. In *Papers in Structural and Transformational Linguistics*, pages 32–67. Springer.
- [Heim and Kratzer, 1998] Heim, I. and Kratzer, A. (1998). *Semantics in Generative Grammar*. Blackwell Textbooks in Linguistics. Wiley.
- [Hines and Montgomery, 2005] Hines, W. and Montgomery, D. (2005). *Probabilidad y estadística para ingeniería*. Grupo editorial Patria.
- [Holm, 2008] Holm, N. M. (2008). *Sketchy LISP: An Introduction to Functional Programming in Scheme*. Lulu.
- [INALI, 2008] INALI (2008). *Catálogo de las lenguas indígenas nacionales: Variantes lingüísticas de México con sus autodenominaciones y referencias geoestadísticas*. Diario Oficial.
- [Jackendoff, 2011] Jackendoff, R. (2011). *Fundamentos Del Lenguaje: Mente, Significado, Gramática y Evolución*. Sección de obras de lengua y estudios literarios. Fondo De Cultura Economica USA.
- [Jakobson et al., 1969] Jakobson, R., Fant, G., and Halle, M. (1969). *Preliminaries to Speech Analysis: The Distinctive Features and Their Correlates*. M.I.T. Press.
- [Jespersen, 1917] Jespersen, O. (1917). *Negation in English and other languages*. Historisk-filologiske meddelelser. Kgl. Danske videnskabernes selskab.

- [Kay, 1979] Kay, M. (1979). Functional grammar. In *Annual Meeting of the Berkeley Linguistics Society*, volume 5, pages 142–158.
- [Lamkins, 2005] Lamkins, D. (2005). *Successful Lisp: How to Understand and Use Common Lisp*. Bookfix.com.
- [Lara, 2006] Lara, L. (2006). *Curso de Lexicología*. Estudios Lingüísticos y Literarios Series. Colegio de México.
- [Llisterri, 1999] Llisterri, J. (1999). Transcripción, etiquetado y codificación de corpus orales. *Revista española de lingüística aplicada*, (1):53–82.
- [Martí and Llisterri, 2001] Martí, M. A. and Llisterri, J. (2001). La ingeniería lingüística en la sociedad de la información.
- [Mattelart, 2007] Mattelart, A. (2007). *Historia de la sociedad de la información*. Bolsillo Paidós. Paidós.
- [McCarthy et al., 1965] McCarthy, J., of Technology. Computation Center, M. I., and of Technology. Research Laboratory of Electronics, M. I. (1965). *Lisp one five programmer's manual*. Massachusetts Institute of Technology.
- [McCarthy, 1960] McCarthy, J. L. (1960). Recursive functions of symbolic expressions and their computation by machine, part I. *Communications of the ACM*, 3(4):184–195.
- [(MR) and (Zbl), 2010] (MR), M. and (Zbl), Z. (2010). MSC2010: Mathematical Sciences Classification.
- [Negroponte and Plaking, 1996] Negroponte, N. and Plaking, D. (1996). *Ser digital*. Ojo infalible. Editorial Oceano de México.
- [Núñez et al., 1999] Núñez, R., Front, A., Vives, P., and Hualde, J. (1999). *Fonología generativa contemporánea de la lengua española*. Georgetown University Press.

- [Ortiz, 2000] Ortiz, A. M. (2000). Diseño e implementación de un lexicón computacional para lexicografía y traducción automática. *Estudios de Lingüística del Español*, (9):1.
- [Parker and Riley, 2009] Parker, F. and Riley, K. (2009). *Linguistics for Non-linguists: A Primer with Exercises*. Allyn & Bacon.
- [Partee et al., 1990] Partee, B., Meulen, A., and Wall, R. (1990). *Mathematical Methods in Linguistics*. Pallas paperbacks. Springer Netherlands.
- [Pickett and Elson, 1986] Pickett, V. B. and Elson, B. F. (1986). *Introducción a la morfología y la sintaxis*. Instituto Lingüístico de Verano.
- [Puente, 1998] Puente, F. (1998). *Los sistemas de numeración indoamericanos: un enfoque areotipológico*. Colección Lingüística indígena. Universidad Nacional Autónoma de México.
- [Real Academia Española and Asociación de Academias de la Lengua Española, 2010] Real Academia Española and Asociación de Academias de la Lengua Española (2010). *Nueva gramática de la lengua española: manual*. Number v. 22 in NUEVAS OBRAS REAL ACADEMIA. Espasa Libros.
- [Renze and Weisstein, 2014] Renze, J. and Weisstein, E. W. (2014). Discrete mathematics. From MathWorld—A Wolfram Web Resource. Last visited on 04/6/2014.
- [Renze and Weisstein, 2016] Renze, J. and Weisstein, E. W. (2016). Morgan’s laws. From MathWorld—A Wolfram Web Resource. Last visited on 16/1/2016.
- [Ricoy, 2007] Ricoy, R. D. P. (2007). Las estrategias de internacionalización en la traducción publicitaria.
- [Riggle, 2012] Riggle, J. (2012). Phonological feature chart (v12.12).

- [Rivest, 1997] Rivest, R. (1997). S-expressions (draft-rivest-sexp-00. txt). *Network Working Group, Tech. Rep., May*.
- [Saldías Uribe, 2015] Saldías Uribe, P. A. (2015). Análisis descriptivo de la categoría gramatical de aspecto en la lengua de señas chilena.
- [Schumpeter and Pascual, 2002] Schumpeter, J. and Pascual, J. (2002). *Ciclos económicos: análisis teórico, histórico y estadístico del proceso capitalista*. Clásicos (Prensas Universitarias de Zaragoza). Prensas Universitarias de Zaragoza.
- [Seibel, 2005] Seibel, P. (2005). *Practical common lisp*. Apress.
- [Simone, 2001] Simone, R. (2001). *Fundamentos de lingüística*. Ariel lingüística. Ariel.
- [Sjösten, 2012] Sjösten, T. (2012). Factorials in Haskell.
- [Tesniere et al., 2015] Tesniere, L., Osborne, T., and Kahane, S. (2015). *Elements of Structural Syntax*. John Benjamins Publishing Company.
- [Turing, 1936] Turing, A. M. (1936). On computable numbers, with an application to the Entscheidungsproblem. *J. of Math*, 58(345-363):5.
- [Tutorials Point (I) Pvt. Ltd., 2015] Tutorials Point (I) Pvt. Ltd. (2015). Learn LISP simply easy learning. Last visited on 12/2/2016.
- [Universitat Politècnica de Catalunya, 2016] Universitat Politècnica de Catalunya (2016). Introducción a las etiquetas EAGLES. Last visited on 23/2/2016.
- [Veerarajan, 2008] Veerarajan, T. (2008). *MATEMÁTICAS DISCRETAS*. McGraw-Hill Interamericana de España S.L.



- [Velasco Quintana, 2014] Velasco Quintana, E. (2014). Un acercamiento a la localización de software.
- [Wikipedia, 2016] Wikipedia (2016). Sucesión de Fibonacci — Wikipedia, the free encyclopedia. [En línea; acceso 3-Abril-2016].
- [Wikipedia, 2017] Wikipedia (2017). Coordenadas polares — Wikipedia, the free encyclopedia. [En línea; acceso 10-Septiembre-2017].
- [Zajac, 1992] Zajac, R. (1992). Inheritance and constraint-based grammar formalisms. *Computational Linguistics*, 18(2):159–182.
- [Zamudio, 2010] Zamudio, C. (2010). *Las consecuencias de la escritura alfabética en la teoría lingüística*. El Colegio de México.



# Índice alfabético

## A

Asociatividad	13, 20
Átomo	81

## C

Cálculo lambda	19, 78
Ciencias naturales	1
Ciencias sociales	1
Conectivo lógico	11, 74
condicional	11, 75
conjunción	11, 44
disyunción	11
negación	12, 44
orden de	12
Conmutatividad	13

## D

Distribucionalidad	13, 21, 39
--------------------	------------

## E

Evaluación	58, 82
Expresión S	58

## F

Fonología	8, 36, 90
-----------	-----------

## I

Iteración	86
-----------	----

## L

Lengua de señas	94
Lenguaje formal	3, 18, 25, 26, 58

## M

Matemáticas	1, 41, 74
matrices	35
Morfología	8, 97

<b>O</b>		<b>COND</b>	75
Operador	66	<b>SETQ</b>	71, 81
<b>AND</b>	75, 77	<b>SND</b>	72, 108
<b>ATOM</b>	72	<b>STRING-EQ</b>	73, 102,
<b>COND*</b>	77		117
<b>DOP</b>	74, 78	<b>SUBSTRING</b>	73, 102,
<b>EQ</b>	73, 117		117
<b>FST</b>	71	<b>SUSTITUTION</b>	108
<b>GETP</b>	80	<b>P</b>	
<b>LAMBDA</b>	78	Pragmática	36
<b>LENGTH</b>	101, 104	<b>R</b>	
<b>LET</b>	83	Rasgos distintivos	2, 95
<b>LET*</b>	85	Recurrencia	89
<b>LOOP</b>	87, 117	Recursividad	18, 22, 85, 89
<b>NOT</b>	75	<b>S</b>	
<b>NTH</b>	101, 104, 109	Secuencialidad	77, 84
<b>NULL</b>	74	Semántica	8, 36
<b>OR</b>	75	Sintaxis	8, 27, 105
<b>PAIR</b>	72	<b>T</b>	
<b>PUTP</b>	80	Teoría de grafos	9
<b>QUOTE</b>	71	árboles	2, 14
<b>REMP</b>	80	binarios	15
<b>RETURN</b>	87		

uso de	16, 31	notación prefija	15
notación infija	15	<b>v</b>	
notación posfija	16	Variable	19, 81