Introduction

"Grammar is the logic of speech,
even as logic is the grammar of reason."

~Panini

Never has the world seen a body of work so comprehensive, so groundbreaking, so profound as the *Aṣṭādhyāyī.* A body of work that has served as an inspiration for many of the world languages throughout history. A sage described as “the first descriptive linguist” and “the father of linguistics” and had revolutionized phonology, morphology, syntax, semantics, and meta-rules far before Western linguists started to understand it in the 19th century, Pāṇini stands as a cornerstone for almost all language today.

Even though the exact details of Pāṇini’s life are uncertain, the magnanimity of his *Aṣṭādhyāyī* has been solidified for millennia. With the *Aṣṭādhyāyī*, he was able to reduce a grammar that was unimaginably complicated and varied into just less than 4000 aphorisms without omission or vagueness. It was able to economically, algebraically, and logically capture the language spoken then, supplemented by texts such as *akṣarasamāmnāya*, *dhātupāṭha,* and *gaṇapāṭha*. By employing a derivational system to describe language, it was able to reach real speech from posited abstract utterances formed through affixes added to bases under certain conditions. standing on the shoulders of his predecessors like Shaakalya, Shaakataayana, Shaunaka, and Yaaska, he embodied a work far more expansive. His other works include Paataalavijaya and Jaambavatijaya.

Aṣṭādhyāyī is made of two words *aṣṭa*-, meaning 'eight' and *adhyāya*-, 'chapter', thus, eight-chaptered, or 'the book of eight chapters'.

Why We came up with the project

Pāṇini achieved with his grammar what is called “non-lossy compression” in computer science, in the sense that he was able to create the original words and forms from a very compressed set of rules. Using a simple and intuitive set of rules, we can arrive at very complex words signifying an array of meanings. These rules also make the generation of new words and their verification very easy. For example, given a root word with the “abhipray” or the intention of speaking in mind. Then run it through the *Aṣṭādhyāyī* to find a suffix that is most relevant to the “abhipray”. We have the intended word. We can employ the same mechanism while verification of new words to see if the word we have matches the one we constructed using the above mechanism. Since the entire process has an algorithmic structure and

efficiency to it, we decided to test for ourselves whether we could code the sutras in a computer in a union of two distant but somehow related disciplines.

Coding the *Aṣṭādhyāyī* with all its interdependencies and _ is a very arduous task. Yet to accomplish what we started out to do and do justice to the magnificence of the text we coded the IT Markers and the affix "*aniyar*" which means (to be) "worthy of".

Our Methodology

Since there exists a plethora of root words and their appropriate suffixes and rules, we decided to start with a small subset of the various sutras that can perform and convey one particular "abhipraay" (of being worthy).

How the *Aṣṭādhyāyī* does this is as follows:

1. Take a *Dhatu* from the *Dhatupaath* and remove the IT markers
2. Add the suffix *aniyar* to it after removing its IT markers
3. Combine the two using *sandhi* rules
4. Accommodate for certain changes after the Sandhi.

The various nuances to this are as follows

1. The *Dhatus* in the *Dhatupaath* are not listed according to their *Dhatu* but rather with IT markers for various uses.
   Hence we take this IT marked *Dhatu* and first remove the ITs
2. We take the suffix *aniyar* and remove its IT markers
3. The sandhi between the two words is interesting:
   - If the *dhatu* ends with a *Hal*, the penultimate *swar* does *Guna.*
   - If the *dhatu* ends with an *AJ*, the ultimate *swar* does a *Guna*

   Next, replace ultimate vowels with certain sets of letters

4. Under certain conditions, the *n* in *aniyar* changes to ण् and this accommodated for

Problems Faced and Solutions

There were some problems we faced in the pursuit of our objective-

Problem: Keyboard input

It was hard for us to remember the whole of the Sanskrit keyboard. On top of that, Microsoft keyboards don't support the IAST script.

Solution:

We used the Keyman: Heidelberg Input Solution and followed the IAST transliterations scheme. We could finally input both Devnagri as well as IAST scripts.

Reference: https://keyman.com/keyboards/heidelberginputsolution

Problem: Interpretation scheme

Panini used pure consonants and vowels in his work but UNICODE doesn't treat characters in the same way, e.g. it sees म as one character but म् as 2 characters. The computer reads म् = म + ◌्, while actually म = म् + अ. This led to issues in the interpretation of rules and comparisons.

Solution:

We came up with encoder and decoder functions to switch between the two schemes of interpretation. We also introduced some logic tricks to help in the encoding and decoding, Eg. while decoding if 2 consonants are next to each other, the first one will get ◌्

Problem: Getting the words through

Aṣṭādhyāyī introduces new words and asks us to re-run them for changes. Hence we had to come up with a to differentiate between the new words and the edited versions. We also had to keep a copy of the original root for further use, e.g. use of "anubandhas". We created a class called "upadhesh" that had all these properties.

Problem: "IT" Markers

Marking is a difficult task in an array. Adding to that removing the switches index caused logic issues.

Solution:

Made a complementary IT_MARKER array that is a bit string storing whether a particular character is IT or not.

The future scope f this i to firstly comp[lete the Aṣṭādhyāyī opfcourse but also to identify how to use modern programming lkanguages better to encode all sutrasb in amore efficient manner. Blah lah

List of Sutras

1.1.1 वृद्धिरादैच्

The वर्णs आ,ऐ,औ are called वृद्धि

1.1.2 अदेङ् गुणः

The वर्णs अ,ए,ओ are called गुण

1.1.8 मुखनासिकावचनोऽनुनासिकः

The वर्ण which needs the nose to be used in pronunciation is अनुनसिक. The list is as follows: अँ, आँ, इँ , ईँ, उँ, ऊँ, ऋँ, ॠँ, ऌँ, ॡँ, एँ, ऐँ, ओँ, औँ, ङ्, ञ्, ण्, न्, म्, यँ, वँ, लँ

1.3.2 उपदेशेऽजनुनासिक इत्

In the input (उपदेश) , all अनुनासिक स्वर are इत्

1.3.3 हलन्त्यम् (अनुवृत्तिः उपदेशे 1.3.2 , इत् 1.3.2) (Complete उपदेशे अन्त्यम् हल् इत्)

In the उपदेश, all हल् at the last of the उपदेश are इत्

1.3.4 न विभक्तौ तुस्माः (अनुवृत्तिः उपदेशे 1.3.2 , इत् 1.3.2 , हल् 1.3.3 , अन्त्यम् 1.3.3) (Complete उपदेशे विभक्तौ तुस्माः न इत्)

In उपदेश, if तुवर्ग, स्, म् come at end as विभक्ति, they are not इत्

1.3.5 आदिर्ञिटुडवः (अनुवृत्तिः उपदेशे 1.3.2 , इत् 1.3.2) (Complete उपदेशे आदिः ञिटुडवः इत्)

In उपदेश, if first letter is ञि, टु, डु, they are इत्

1.3.6 षः प्रत्ययस्य (अनुवृत्तिः उपदेशे 1.3.2 , इत् 1.3.2 , आदिः 1.3.5) (Complete उपदेशे प्रत्ययस्य आदिः षः इत्)

In प्रत्यय उपदेश, if first letter is ष्, it is इत्

1.3.7 चुटू (अनुवृत्तिः उपदेशे 1.3.2 , इत् 1.3.2 , आदिः 1.3.5 , प्रत्ययस्य 1.3.6) (Complete उपदेशे प्रत्ययस्य आदिः चुटू इत्)

In प्रत्यय उपदेश, if first letter is चवर्ग / टवर्ग, it is इत् #Is not universal need to ask exception and why

1.3.8 लशक्वतद्धिते (अनुवृत्तिः उपदेशे 1.3.2 , इत् 1.3.2 , आदिः 1.3.5 , प्रत्ययस्य 1.3.6) (Complete उपदेशे अतद्धितस्य प्रत्ययस्य आदिः ल-श-कु इत्)

In non-तद्धित प्रत्यय उपदेश, if first letter is ल्, श्, कवर्ग, it is इत्

1.3.9 तस्य लोपः

Remove all इत्

3.1.91 धातोः

Reckoning from the present sūtra, the affixes treated of are to be understood as coming after some verbal root

This is an अधिकार सूत्र।

3.1.93 कृदतिङ् (अनुवृत्तिः तत्र 3.1.92) (Complete कृत् तत्र् अतिङ्)

All प्रत्ययs post this are called कृत् except तिङ्

3.1.95 कृत्याः (अनुवृत्तिः कृत् 3.1.93)

All प्रत्यय् post this as क्रित्य् uptil the one's mentioned in 3.1.133

3.1.96 तव्यत्तव्यानीयरः (अनुवृत्तिः कृत् 3.1.93 , कृत्याः 3.1.95)

The affixes तव्यत् , तव्य and अनीयर् come after धातु

7.3.84 सार्वधातुकार्धधातुकयोः (अनुवृत्तिः गुणः 7.3.82) (Complete अङ्गस्य गुणः सार्वधातुक-आर्धधातुकयोः)

When followed by an सार्वधातुक or an आर्धधातुक प्रत्यय, an इगन्त अङ्ग gets a गुणादेश.

7.3.86 पुगन्तलघूपधस्य च (अनुवृत्तिः गुणः 7.3.82 , सार्वधातुक-आर्धधातुकयोः 7.3.84) (Complete सार्वधातुकार्धधातुकयोः पुगन्तलघूपधस्य च गुणः)

When followed by a सार्वधातुक or an आर्धधातुक प्रत्यय, a पुगन्त अङ्ग and a लघूपध अङ्ग get a गुणादेश.

In our context this stimulates the गुणादेश for the penultimate इक् letter.

6.1.78 एचोऽयवायावः(अनुवृत्तिः अचि 6.1.77) (Complete एचः अचि अय्-अव्-आय्-आवः)

When followed by a स्वर, the letters of the एच् प्रत्याहार are respectively converted to अय्, अव्, आय् and आव् in the context of संहिता.

8.4.1 रषाभ्यां नो णः समानपदे

a नकार occurring immediately after ष् or र् in the same पद is converted to a णकार.

8.4.2 अट्कुप्वाङ्नुम्व्यवायेऽपि

a नकार occurring after ष् or र् in the same पद is converted to a णकार even when it is separated from the ष् or र् by one or more letters from the अट् प्रत्याहार, कवर्ग, पवर्ग, आङ् , or अनुस्वार.

Example Runs

1. कृ
   Enter the dhatu: डुकृञ्
   For the upadesh डुकृञ्
   डु is IT
   ञ् is IT
   After removing IT, we get कृ
   For the upadesh अनीयर्
   र् is IT
   After removing IT, we get अनीय
   After sandhi we get: करनीय
   The final word is: करणीय

2. पठ्

Enter the dhatu: पठँ
For the upadesh पठँ
अँ is IT
After removing IT, we get पठ्
For the upadesh अनीयर्
र् is IT
After removing IT, we get अनीय
After sandhi we get: पठनीय
The final word is: पठनीय

3. लिख्
Enter the dhatu: लिखँ
For the upadesh लिखँ
अँ is IT
After removing IT, we get लिख्
For the upadesh अनीयर्
र् is IT
After removing IT, we get अनीय
After sandhi we get: लेखनीय
The final word is: लेखनीय
4. मिल्
Enter the dhatu: मिलँ
For the upadesh मिलँ
अँ is IT
After removing IT, we get मिल्
For the upadesh अनीयर्
र् is IT
After removing IT, we get अनीय
After sandhi we get: मेलनीय
The final word is: मेलनीय
5. भू
Enter the dhatu: भू
For the upadesh भू
After removing IT, we get भू
For the upadesh अनीयर्
र् is IT
After removing IT, we get अनीय
After sandhi we get: भवनीय
The final word is: भवनीय