Language and Computers – where to start?

- If we want to do anything with language, we need a way to represent language.
- We can interact with the computer in several ways:
  - write or read text
  - speak or listen to speech
- Computer has to have some way to represent
  - text
  - speech

Alphabetic systems

Alphabets (phonemic alphabets)

- represent all sounds, i.e., consonants and vowels
- Examples: Etruscan, Latin, Korean, Cyrillic, Runic, International Phonetic Alphabet

Abjads (consonant alphabets)

- represent consonants only (sometimes plus selected vowels; vowel diacritics generally available)
- Examples: Arabic, Aramaic, Hebrew

A note on the letter-sound correspondence

- Alphabets use letters to encode sounds (consonants, vowels).
- But the correspondence between spelling and pronunciation in many languages is quite complex, i.e., not a simple one-to-one correspondence.
- Example: English
  - same spelling – different sounds: ough:ough, cough, tough, through, though, hiccough
  - silent letters: knee, knight, knife, debt, psychology
  - one letter – multiple sounds: exit, use
  - multiple letters – one sound: the, revolution
  - alternate spellings: jail or gaol; but not possible seagh for chef (despite sure, dead, laugh)

Outline

- Writing systems
- Encoding written language
- Spoken language
- Relating written and spoken language

Alphabet example: Fraser

An alphabet used to write Liu, a Tibeto-Burmese language spoken by about 657,000 people in Myanmar, India, Thailand and in the Chinese provinces of Yunnan and Sichuan.

Consonants

| P | d | b | f | v | m | n | l | s | z | c | j | k | g | h |

Vowels

| e | i | u | a | o | w |

Tones

- high tone
- mid rising
- mid tone
- mid falling
- low tone
- low tense

(from: http://www.omniglot.com/writing/fraser.html)

More examples for non-transparent letter-sound correspondences

French

1. Versailles → [versal]
2. b. ete, etais, etait, ete → [ete]

Irish

1. Baile A' Tha Cliath (Dublin) → [blf: klj u:l]
2. b. samhradh (summer) → [sauruh]
3. scríobhaim (I write) → [sbrjhm]

What is the notation used within the []?
The International Phonetic Alphabet (IPA)

The phonetic symbols are unambiguous:
- designed so that each speech sound gets its own symbol
- eliminating the need for multiple symbols used to represent simple sounds
- one symbol being used for multiple sounds.

Interactive example chart: http://web.uvic.ca/ling/resources/ipa/charts/IPAlab/IPAlab.htm

Syllabic systems

Syllabic systems (Alphasyllabaries)
- writing systems with symbols that represent a consonant with a vowel, but the vowel can be changed by adding a diacritic (= a symbol added to the letter).
- Examples: Balinese, Javanese, Tibetan, Tamil, Thai, Tagalog

Diacritics

Diacritics are symbols used to change the pronunciation of a letter, for example, vowel diacritics around the letter k:

(from: http://www.omniglot.com/writing/lao.htm)

Logographic writing systems

Logographs (also called Logograms):
- Pictographs (Pictograms): originally pictures of things, now stylized and simplified.
- Examples: development of Chinese character horse:

(from: http://www.omniglot.com/writing/chinese.html)

Examples:
- Chinese (Zhōngwén), Japanese (Nihongo), Mayan, Vietnamese, Ancient Egyptian

Semantic-phonetic compounds

Semantic-phonetic compounds:
- Compounds: combinations of two or more ideographs or ideograms.
- Semantic-phonetic compounds: symbols with a meaning element (hints at meaning) and a phonetic element (hints at pronunciation).

Examples:
- Chinese character: 马 (horse), 马 (horse)

Two writing systems with unusual realization

Braille alphabet

Braille is a writing system that makes it possible to read and write through touch; primarily used by the (partially) blind.
- It uses patterns of raised dots arranged in cells of up to six dots in a 3 x 2 configuration.
- Each pattern represents a character, but some frequent words and letter combinations have their own pattern.

Chromatographic writing

The Benin and Edo people in southern Nigeria have developed a system of writing based on different color combinations and symbols.
There is not a simple correspondence between a writing system and a language.

For example, English uses the Roman alphabet, but Arabic numerals (e.g., 2 instead of the Roman II).

We’ll look at three other examples:

- Japanese
- Korean
- Azeri

Japanese: logographic system kanji, syllabary katakana, syllabary hiragana

- kanji: 5,000-10,000 borrowed Chinese characters
- katakana: Used mainly for non-Chinese loan words, onomatopoeic words, foreign names, and for emphasis
- hiragana: Originally used only by women (10th century), but codified in 1946 with 48 syllables

used mainly for word endings, kids’ books, and for words with obscure kanji symbols

Romaji: Roman characters

Korean

“Korean writing is an alphabet, a syllabary and logographs all at once.”

The hangul system was developed in 1444 during King Sejong’s reign.

- There are 24 letters: 14 consonants and 10 vowels
- But the letters are grouped into syllables, i.e. the letters in a syllable are not written separately as in the English system, but together form a single character.

E.g., “Hangul” (from: http://www.omniglot.com/writing/korean.htm)

- In South Korea, hanja (logographic Chinese characters) are also used.

Azeri

A Turkish language with speakers in Azerbaijan, northwest Iran, and (former Soviet) Georgia

- 7th century until 1920s: Arabic scripts. Three different Arabic scripts used
- 1929: Latin alphabet enforced by Soviets to reduce Islamic influence.
- 1939: Cyrillic alphabet enforced by Stalin
- 1991: Back to Latin alphabet, but slightly different than before.

← Latin typewriters and computer fonts were in great demand in 1991

Comparison of writing systems

What are the pros and cons of each type of system?

- accuracy: Can every word be written down accurately?
- learnability: How long does it take to learn the system?
- cognitive ability: Are some systems unnatural? (e.g. Does dyslexia show that alphabets are unnatural?)
- language-particular differences: English has thousands of possible syllables; Japanese has very few in comparison
- connection to history/culture: Will changing a writing system have social consequences?

Encoding written language

- Information on a computer is stored in bits.
- A bit is either on (= 1, yes) or off (= 0, no).
- A list of 8 bits makes up a byte, e.g., 01001010
- Just like with the base 10 numbers we’re used to, the order of the bits in a byte matters:
  - Big Endian: most important bit is leftmost (the standard way of doing things)
    - The positions in a byte thus encode: 128 64 32 16 8 4 2 1
    - “There are 10 kinds of people in the world; those who know binary and those who don’t”
      (from: http://www.sukg.org.uk/LittleEndian)
  - Little Endian: most important bit is rightmost (only used on Intel machines)
    - The positions in a byte thus encode: 1 2 4 8 16 32 64 128

Using bytes to store characters

With 8 bits (a single byte), you can represent 256 different characters. Why would we want so many?

- If you look at a keyboard, you will find lots of non-English characters.
- With 256 possible characters, we can store every single letter used in English, plus all the things like commas, periods, space bar, percent sign (%), back space, and so on.
An encoding standard: ASCII

- **ASCII** = the American Standard Code for Information Interchange
- 7-bit code for storing English text
- 7 bits = 128 possible characters.
- The numeric order reflects alphabetic ordering.

The ASCII chart

<table>
<thead>
<tr>
<th>Code</th>
<th>Character</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Space</td>
</tr>
<tr>
<td>01</td>
<td>!</td>
</tr>
<tr>
<td>02</td>
<td>&quot;</td>
</tr>
<tr>
<td>03</td>
<td>#</td>
</tr>
<tr>
<td>04</td>
<td>$</td>
</tr>
<tr>
<td>05</td>
<td>%</td>
</tr>
<tr>
<td>06</td>
<td>&amp;</td>
</tr>
<tr>
<td>07</td>
<td>'</td>
</tr>
<tr>
<td>08</td>
<td>(</td>
</tr>
<tr>
<td>09</td>
<td>)</td>
</tr>
<tr>
<td>10</td>
<td>*</td>
</tr>
<tr>
<td>11</td>
<td>+</td>
</tr>
<tr>
<td>12</td>
<td>,</td>
</tr>
<tr>
<td>13</td>
<td>-</td>
</tr>
<tr>
<td>14</td>
<td>.</td>
</tr>
<tr>
<td>15</td>
<td>/</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
</tr>
<tr>
<td>19</td>
<td>3</td>
</tr>
<tr>
<td>20</td>
<td>4</td>
</tr>
<tr>
<td>21</td>
<td>5</td>
</tr>
<tr>
<td>22</td>
<td>6</td>
</tr>
<tr>
<td>23</td>
<td>7</td>
</tr>
<tr>
<td>24</td>
<td>8</td>
</tr>
<tr>
<td>25</td>
<td>9</td>
</tr>
<tr>
<td>26</td>
<td>:</td>
</tr>
<tr>
<td>27</td>
<td>;</td>
</tr>
<tr>
<td>28</td>
<td>&lt;</td>
</tr>
<tr>
<td>29</td>
<td>=</td>
</tr>
<tr>
<td>30</td>
<td>&gt;</td>
</tr>
<tr>
<td>31</td>
<td>?</td>
</tr>
</tbody>
</table>

E-mail issues

- Have you ever had something like the following at the top of an e-mail sent to you?
  
  The following text is in the "ISO-8859-1" character set.
  
  [Your display is set for the "US-ASCII" character set.]

  [Some characters may be displayed incorrectly.]

- Mail sent on the internet used to only be able to transfer 7-bit ASCII messages. But now we can detect the incoming character set and adjust the input.

- Note that this is an example of **meta-information** = information which is printed as part of the regular message, but tells us something about that message.

Multipurpose Internet Mail Extensions (MIME)

MIME provides meta-information on the text, which tells us:

- which version of MIME is being used
- what the character set is
- if that character set was altered, how it was altered

**Mime-Version: 1.0** **Content-Type: text/plain; charset=US-ASCII** **Content-Transfer-Encoding: 7bit**

Different coding systems

But wait, didn’t we want to be able to encode all languages? There are ways ...

- Extend the ASCII system with various other systems, for example:
  - ISO 8859-1: includes extra letters needed for French, German, Spanish, etc.
  - ISO 8859-7: Greek alphabet
  - ISO 8859-8: Hebrew alphabet
  - JIS X 0208: Japanese characters
- Have one system for everything → **Unicode**

Unicode

Problems with having multiple encoding systems:

- Conflicts: two encodings can use the same number for two different characters and use different numbers for the same character.
- Hassle: have to install many, many systems if you want to be able to deal with various languages

Compact encoding of Unicode characters

- **Unicode** has three versions
  - UTF-32 (32 bits): direct representation
  - UTF-16 (16 bits): 216 = 65536
  - UTF-8 (8 bits): 28 = 256
  - How is it possible to encode 256 possibilities in 8 bits (UTF-8)?
    - Several bytes are used to represent one character.
    - Use the highest bit as flag:
      - highest bit 0: single character
      - highest bit 1: part of a multi byte character
    - Nice consequence: ASCII text is in a valid UTF-8 encoding.

How do we type everything in?

- Use a keyboard tailored to your specific language... Highly noticeable how much slower your English typing is when using a Danish-designed keyboard.
- Use a processor that allows you to switch between different character systems, e.g. Type in Cyrillic characters on your English keyboard.
- Use combinations of characters. An followed by an ’è’ might result in an é.
- Pick and choose from a table of characters.

So, now we can encode every language, as long as it’s written.

How big is Unicode?

Version 3.2 has codes for 95,221 characters from alphabets, syllabaries and logographic systems.

- Uses 32 bits – meaning we can store 232 = 4,294,967,296 characters.
- 4 billion possibilities for each character? That takes a lot of space on the computer!
The need for speech

- What if we want to work with an unwritten language?
- What if we want to examine the way someone talks and don’t have time to write it down?

Many applications for encoding speech:
- Building spoken dialogue systems, i.e. speak with a computer (and have it speak back).
- Helping people sound like native speakers of a foreign language.
- Helping speech pathologists diagnose problems.

What does speech look like?

We can transcribe (write down) the speech into a phonetic alphabet.
- It is very expensive and time-consuming to have humans do all the transcription.
- To automatically transcribe, we need to know how to relate the audio file to the individual sounds that we hear.
  ⇒ We need to know:
    - some properties of speech
    - how to measure these speech properties
    - how these measurements correspond to sounds we hear.

Articulatory properties: How it’s produced

We could talk about how sounds are produced in the vocal tract, i.e. articulatory phonetics

- place of articulation (where): [t] vs. [k]
- manner of articulation (how): [t] vs. [s]
- voicing (vocal cord vibration): [t] vs. [d]

But unless the computer is modeling a vocal tract, we need to know acoustic properties of speech which we can quantify.

Acoustic properties: What it sounds like

Sound waves = “small variations in air pressure that occur very rapidly one after another” (Ladefoged, A Course in Phonetics)
  ⇒ Akin to ripples in a pond

- speech flow = rate of speaking, number and length of pauses (seconds)
- loudness (amplitude) = amount of energy (decibels)
- frequencies = how fast the sound waves are repeating (cycles per second, i.e. Hertz)
  - pitch = how high or low a sound is
    - In speech, there is a fundamental frequency, or pitch, along with higher-frequency overtones.
  - intonation = rise and fall in pitch

Oszillogram (Waveform)

Fundamental frequency (F0, pitch)

Spectrograms

Spectrogram = a graph to represent (the frequencies of) speech over time.
How did we get these measurements?

- **Sampling rate** - how many times in a given second we extract a moment of sound; measured in samples per second.
  - Sound is **continuous**, but we have to store data in a **discrete** manner.

- We store data at each discrete point, in order to capture the general pattern of the sound.

Sampling rate

- The sampling rate is often 8000 or 16,000 samples per second. The rate for CDs is 44,100 samples/second (or **Hertz** (Hz)).
- The higher the sampling rate, the better quality the recording...
- But the more space it takes.
- Speech needs at least 8000 samples/second, but most likely 16,000 or 22,050 Hz will be used nowadays.

Applications of speech encoding

- Mapping sounds to symbols (alphabet), and vice versa, isn’t all that easy.
  - **Automatic Speech Recognition (ASR)**: sounds to text
  - **Text-to-Speech Synthesis (TTS)**: texts to sounds

Automatic Speech Recognition (ASR)

Automatic speech recognition = process by which the computer maps a speech signal to text.

Uses/Applications:
- **Diction**
- **Telephone conversations**
- People with disabilities – e.g. a person hard of hearing could use an ASR system to get the text

Kinds of ASR systems

- Different kinds of systems:
  - **Speaker dependent** = work for a single speaker
  - **Speaker independent** = work for any speaker of a given variety of a language, e.g. American English
  - **Speaker adaptive** = start as independent but begin to adapt to a single speaker to improve accuracy

Steps in an ASR system

1. **Digital sampling of speech**
2. **Acoustic signal processing** = converting the speech samples into particular measurable units
3. **Recognition of sounds, groups of sounds, and words**

May or may not use more sophisticated analysis of the utterance to help.

Text-to-Speech Synthesis (TTS)

Could just record a voice saying phrases or words and then play back those words in the appropriate order.

Or can break the text down into smaller units:
1. Convert input text into phonetic alphabet
2. Synthesize phonetic characters into speech

To synthesize characters into speech, people have tried:
- Using formulas which adjust the values of the frequencies, the loudness, etc.
- Using a model of the vocal tract and trying to produce sounds based on how a human would speak
It's hard to be natural

When trying to make synthesized speech sound natural, we encounter the same problems as what makes speech encoding in general hard:

- The same sound is said differently in different contexts.
- Different sounds are sometimes said nearly the same.
- Different sentences have different intonation patterns.
- Lengths of words vary depending on where in the sentence they are spoken.

The car crashed into the tree.
It's my car.
Cars, trucks, and bikes are vehicles.

If we convert speech to text and then back to speech, it should sound the same, right?

- But at the conversion stages, there is information loss. To avoid this loss would require a lot of memory and knowledge about what exact information to store.
- The process is thus irreversible.

Text-to-Speech

- various systems and languages: http://www.ims.uni-stuttgart.de/~moehler/synthspeech/