Character Encoding Algorithms for overlaps

Years ago we had character set detection for a mail application, and we rolled our own. The mail app was actually a WAP application, and the phone expected UTF-8. There were several steps:

Universal

We could easily detect if text was UTF-8, as there is a specific bit pattern in the top bits of bytes 2/3/etc. Once you found that pattern repeated a certain number of times you could be certain it was UTF-8.

If the file begins with a UTF-16 byte order mark, you can probably assume the rest of the text is that encoding. Otherwise, detecting UTF-16 isn't nearly as easy as UTF-8, unless you can detect the surrogate pairs pattern: but the use of surrogate pairs is rare, so that doesn't usually work. UTF-32 is similar, except there are no surrogate pairs to detect.

Regional detection

Next we would assume the reader was in a certain region. For instance, if the user was seeing the UI localized in Japanese, we could then attempt detection of the three main Japanese encodings. ISO-2022-JP is again east to detect with the escape sequences. If that fails, determining the difference between EUC-JP and Shift-JIS is not as straightforward. It's more likely that a user would receive Shift-JIS text, but there were characters in EUC-JP that didn't exist in Shift-JIS, and vice-versa, so sometimes you could get a good match.

The same procedure was used for Chinese encodings and other regions.

User's choice

If these didn't provide satisfactory results, the user must manually choose an encoding.