Anagrams

Anagrams

Definition. Two words are anagrams if one can be formed by permuting the letters in the other.

A 6-element anagram set:

opts pots post stop spot tops

The Problem. How would you compute all anagram sets in a dictionary of 230,000 English words? (Related problem: how to find all anagrams of an input word?)

Two Slow Algorithms

Examine All Permutations. ‘‘cholecystoduodenostomy’’ has $22! \approx 1.1 \times 10^{21}$ permutations. One picosecond each gives 1.1×10^{9} seconds, or a few decades.

(The rule of thumb that ‘‘π seconds is a nanocentury’’ is half a percent off the true value of 3.155×10^{7} seconds per year.)

Examine All Pairs of Words. Assume 230,000 words in the dictionary, 1 microsec per compare.

$$\begin{aligned} & 230{,}000 \text{ words} \times 230{,}000 \text{ comps/word} \\ & \quad \times 1 \text{ microsec/comp} \\ &= 52900 \times 10^{6} \text{ microsecs} \\ &= 52900 \text{ secs} \\ &\approx 14.7 \text{ hours} \end{aligned}$$

A Fast Algorithm

The Idea. Sign words so that those in the same anagram class have the same signature, and then collect equal signatures.

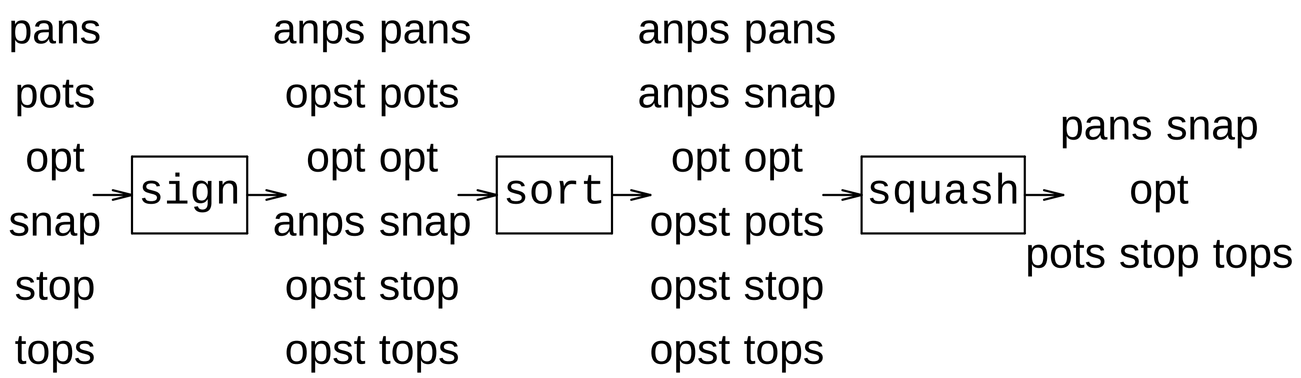
The Signature. Sort letters within a word. The signature of ‘‘deposit’’ is ‘‘deiopst’’, which is also the signature of ‘‘dopiest’’.

Collecting the Classes. Sort the words by their signatures.

A Summary. Sort this way (with a horizontal wave of the hand) then that way (a vertical wave).

Implementation with Pipes

A pipeline of three programs. Example:



sign in C

```
int charcomp(char *x, char *y)
{ return(*x - *y); }

#define WORDMAX 100
int main(void)
{   char word[WORDMAX], sig[WORDMAX];
    while (scanf("%s", word) != EOF) {
        strcpy(sig, word);
        qsort(sig, strlen(sig),
              sizeof(char), charcomp);
        printf("%s %s\n", sig, word);
    }
    return 0;
}
```

squash in C

```
int main(void)
{   char word[MAX], sig[MAX], oldsig[MAX];
    int linenum = 0;
    strcpy(oldsig, "");
    while (scanf("%s %s", sig, word) != EOF)
        if (strcmp(oldsig, sig) != 0
                && linenum > 0)
            printf("\n");
        strcpy(oldsig, sig);
        linenum++;
        printf("%s ", word);
    }
    printf("\n");
    return 0;
}
```

The Complete Program

Executed by

```
     sign <dict | sort | squash >grams
```

where `dict` contains 230,000 words.

Total run time is 18 seconds: 4 in `sign`, 11 in `sort` and 3 in `squash`.

Some Output:

```
subessential suitableness
canter creant cretan nectar recant tanrec trance
caret carte cater crate creat creta react recta trace
destain instead sainted satined
adroitly dilatory idolatry
least setal slate stale steal stela tales
reins resin rinse risen serin siren
constitutionalism misconstitutional
```